Aprendizaje profundo

Ian GooDfellow

Yoshua Bengio

Aaron Courville

# Contenidos

# Website

1. Website

[www.deeplearningbook.org](http://www.deeplearningbook.org)

Este libro está acompañado por el sitio web anterior. El sitio web proporciona una variedad de material complementario, que incluye ejercicios, diapositivas de conferencias, correcciones de errores y otros recursos que deberían ser útiles tanto para los lectores como para los instructores.

# Agradecimientos

Este libro no hubiera sido posible sin las contribuciones de muchas personas.

Quisiéramos agradecer a quienes comentaron sobre nuestra propuesta para el libro y ayudaron a planificar su contenido y organización: Guillaume Alain, Kyunghyun Cho, Çağlar Gülçehre, David Krueger, Hugo Larochelle, Razvan Pascanu and Thomas Rohée.

Nos gustaría agradecer a las personas que recibieron comentarios sobre el contenido del libro en sí. Algunos de los comentarios sobre muchos capítulos: Martín Abadi, Ishaq Aden-Ali,Guillaume Alain, Ion Androutsopoulos, Laura Ball, Fred Bertsch, Olexa Bilaniuk,Ufuk Can Biçici, Matko Bošnjak, John Boersma, François Brault, Greg Brockman,Alexandre de Brébisson, Pierre Luc Carrier, Sarath Chandar, Pawel Chilinski,Mark Daoust, Oleg Dashevskii, Laurent Dinh, Stephan Dreseitl, GudmundurEinarsson, Hannes von Essen, Jim Fan, Miao Fan, Meire Fortunato, FrédéricFrancis, Nando de Freitas, Çağlar Gülçehre, Jurgen Van Gael, Yaroslav Ganin,Javier Alonso García, Aydin Gerek, Stefan Heil, Jonathan Hunt, Gopi Jeyaram,Chingiz Kabytayev, Lukasz Kaiser, Varun Kanade, Asifullah Khan, Akiel Khan,John King, Diederik P. Kingma, Dominik Laupheimer, Yann LeCun, Minh Lê, MaxMarion, Rudolf Mathey, Matías Mattamala, Abhinav Maurya, Vincent Michalski,Kevin Murphy, Oleg Mürk, Hung Ngo, Roman Novak, Augustus Q. Odena, SimonPavlik, Karl Pichotta, Eddie Pierce, Kari Pulli, Roussel Rahman, Tapani Raiko,Anurag Ranjan, Johannes Roith, Mihaela Rosca, Halis Sak, César Salgado, GrigorySapunov, Yoshinori Sasaki, Mike Schuster, Julian Serban, Nir Shabat, Ken Shirriﬀ,Andre Simpelo, Scott Stanley, David Sussillo, Ilya Sutskever, Carles Gelada Sáez,Graham Taylor, Valentin Tolmer, Massimiliano Tomassoli, An Tran, ShubhenduTrivedi, Alexey Umnov, Vincent Vanhoucke, Robert Viragh, Marco Visentini-Scarzanella, Martin Vita, David Warde-Farley, Dustin Webb, Shan-Conrad Wolf,Kelvin Xu, Wei Xue, Ke Yang, Li Yao, Zygmunt Zając and Ozan Çağlayan.

También nos gustaría agradecer a quienes nos brindaron comentarios útiles sobre capítulos individuales:

* Notation: Zhang Yuanhang.
* Chapter 1, Introduction: Yusuf Akgul, Sebastien Bratieres, Samira Ebrahimi,Charlie Gorichanaz, Benned Hedegaard, Brendan Loudermilk, Petros Mani-atis, Eric Morris, Cosmin Pârvulescu, Muriel Rambeloarison, Alfredo Solanoand Timothy Whelan.
* Chapter 2, Linear Algebra: Amjad Almahairi, Nikola Banić, Kevin Bennett,Philippe Castonguay, Oscar Chang, Eric Fosler-Lussier, Andrey Khalyavin,Sergey Oreshkov, István Petrás, Dennis Prangle, Thomas Rohée, GitanjaliGulve Sehgal, Colby Toland, Alessandro Vitale and Bob Welland.
* Chapter 3, Probability and Information Theory: John Philip Anderson, KaiArulkumaran, Ana-Maria Cretu, Vincent Dumoulin, Rui Fa, Stephan Gouws,Artem Oboturov, Patrick Pan, Antti Rasmus, Alexey Surkov and VolkerTresp.
* Chapter 4, Numerical Computation: Tran Lam An, Ian Fischer, WilliamGandler, Mahendra Kariya and Hu Yuhuang.
* Chapter 5, Machine Learning Basics: Dzmitry Bahdanau, Mark Cramer,Eric Dolores, Justin Domingue, Ron Fedkiw, Nikhil Garg, Guillaume deLaboulaye, Jon McKay, Makoto Otsuka, Bob Pepin, Philip Popien, KlausRadke, Emmanuel Rayner, Eric Sabo, Imran Saleh, Peter Shepard, Kee-BongSong, Zheng Sun, Alexandre Torres and Andy Wu.
* Chapter 6, Deep Feedforward Networks: Uriel Berdugo, Fabrizio Bottarel,Elizabeth Burl, Ishan Durugkar, Jeﬀ Hlywa, Jong Wook Kim, David Kruegerand Aditya Kumar Praharaj.
* Chapter 7, Regularization for Deep Learning: Brian Bartoldson, MortenKolbæk, Kshitij Lauria, Inkyu Lee, Sunil Mohan, Hai Phong Phan andJoshua Salisbury.
* Chapter 8, Optimization for Training Deep Models: Marcel Ackermann,Tushar Agarwal, Peter Armitage, Rowel Atienza, Andrew Brock, Max HaydenChiz, Gregory Galperin, Aaron Golden, Russell Howes, Hill Ma, TeganMaharaj, James Martens, Kashif Rasul, Thomas Stanley, Klaus Strobl,Nicholas Turner and David Zhang.
* Chapter 9, Convolutional Networks: Martín Arjovsky, Eugene Brevdo, JaneBromley, Konstantin Divilov, Eric Jensen, Mehdi Mirza, Alex Paino, Guil-laume Rochette, Marjorie Sayer, Ryan Stout and Wentao Wu.x
* Chapter 10, Sequence Modeling: Recurrent and Recursive Nets: GökçenEraslan, Nasos Evangelou, Steven Hickson, Christoph Kamann, MartinKrasser, Razvan Pascanu, Diogo Pernes, Ryan Pilgrim, Lorenzo von Ritter,Rui Rodrigues, Dmitriy Serdyuk, Dongyu Shi, Kaiyu Yang and Ruiqing Yin.
* Chapter 11, Practical Methodology: Daniel Beckstein and Kenji Kaneda.
* Chapter 12, Applications: George Dahl, Vladimir Nekrasov and RibanaRoscher.
* Chapter 13, Linear Factor Models: Jayanth Koushik.
* Chapter 14, Autoencoders: Hassan Masum.
* Chapter 15, Representation Learning: Mateo Torres-Ruiz , Kunal Ghosh andRodney Melchers.
* Chapter 16, Structured Probabilistic Models for Deep Learning: Deng Qingyu, Harry Braviner, Timothy Cogan, Diego Marez, Anton Varfolom and VictorXie.
* Chapter 18, Confronting the Partition Function: Sam Bowman and Jin Kim.
* Chapter 19, Approximate Inference: Yujia Bao.
* Chapter 20, Deep Generative Models: Nicolas Chapados, Daniel Galvez,Wenming Ma, Fady Medhat, Shakir Mohamed and Grégoire Montavon.
* Bibliography: Lukas Michelbacher, Leslie N. Smith and Max

También queremos agradecer a quienes nos permitieron reproducir imágenes, cifras o datos de sus publicaciones. Indicamos sus contribuciones en las leyendas de la figura a lo largo del texto.

Nos gustaría agradecer a Lu Wang por escribir pdf2htmlEX, que utilizamos para hacer la versión web del libro, y por ofrecer soporte para mejorar la calidad del HTML resultante. También agradecemos a Simon Lefrançois por la incorporación de las ediciones de MIT Press a nuestro manuscrito en la edición web, y por ayudar a los comentarios de lectores incorporados de la web.

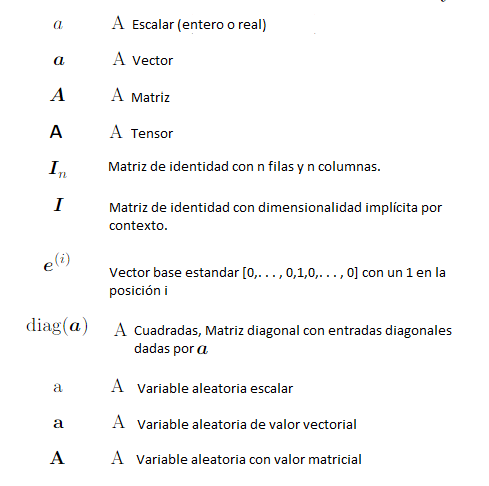
Nos gustaría agradecer a la esposa de Ian, Daniela Flori Goodfellow, por su apoyo paciente a Ian durante la redacción del libro, así como por su ayuda en la revisión.

Nos gustaría agradecer al equipo Google Brain por proporcionar un entorno intelectual en el que Ian podría dedicar una gran cantidad de tiempo a escribir este libro y recibir comentarios y orientación de sus colegas. Nos gustaría agradecer especialmente al ex gerente de Ian, Greg Corrado, y a su gerente actual, Samy Bengio, por su apoyo a este proyecto. Finalmente, nos gustaría agradecer a GeoﬀreyHinton por el estímulo cuando la escritura fue difícil.

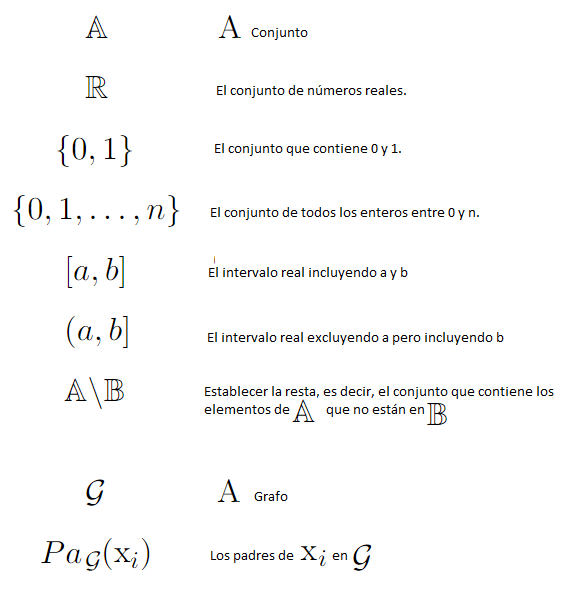
# Notación

Esta sección proporciona una referencia concisa que describe la notación utilizada en este libro. Si no está familiarizado con alguno de los conceptos matemáticos correspondientes, describimos la mayoría de estas ideas en los capítulos 2–4.

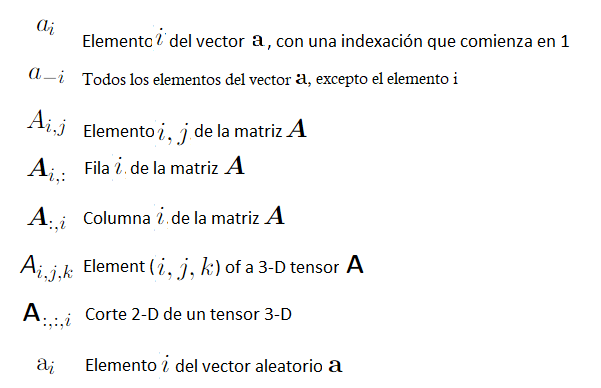
**Números y matrices**



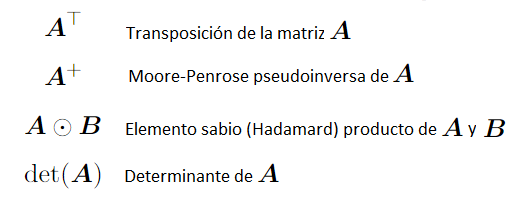
**Conjuntos y Grafos**



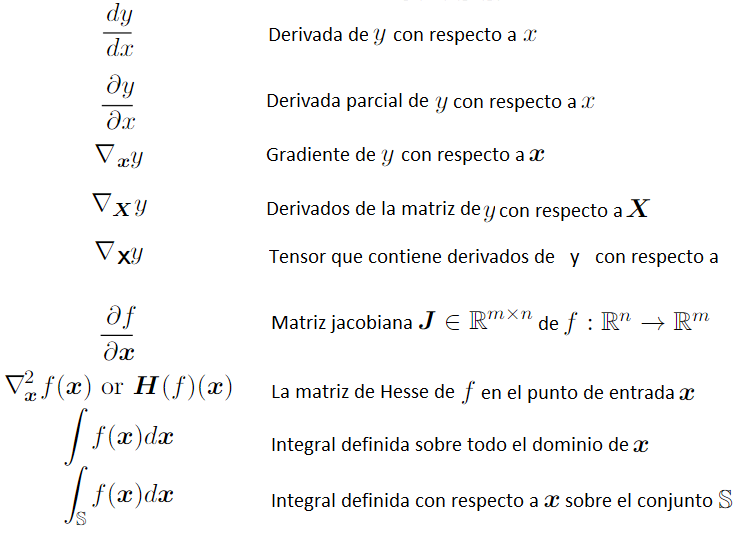
**Indexación**



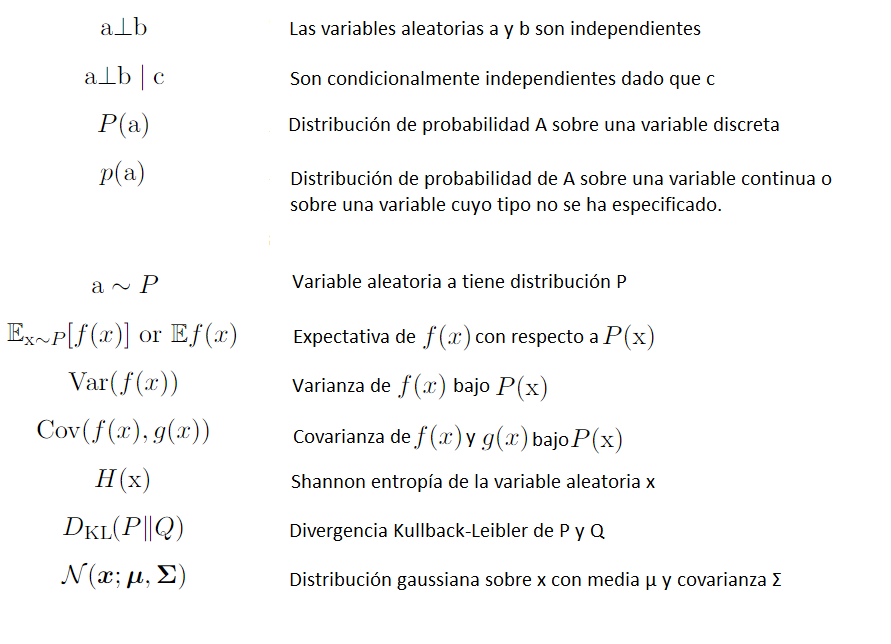
**Operaciones de álgebra lineal**



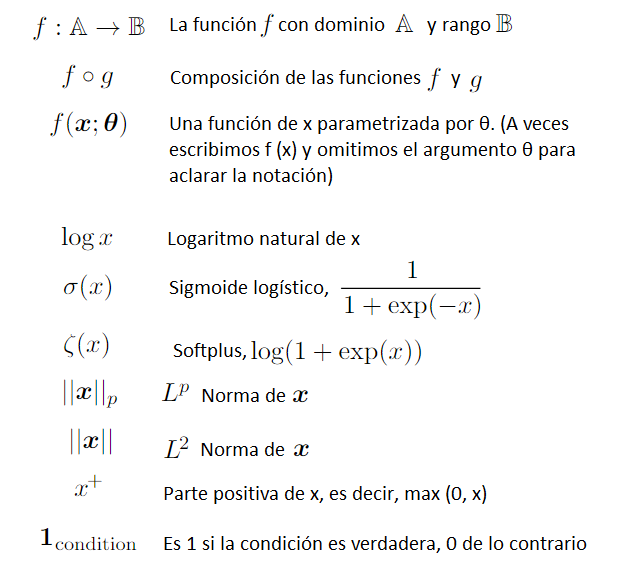
**Cálculo**



**Probabilidad y teoría de la información**

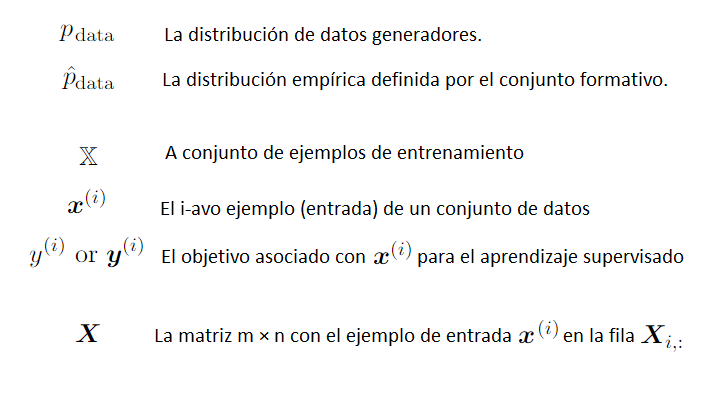


**Funciones**



Algunas veces usamos una función  cuyo argumento es un escalar pero lo aplicamos a un vector, una matriz o un tensor:  o . Esto denota la aplicación de  a la matriz de elementos. Por ejemplo, si , entonces  para todos los valores válidos de  y 

**Conjuntos de datos y distribuciones**



# Introducción

Los inventores siempre han soñado con crear máquinas que piensen. Este deseo se remonta al menos a la época de la antigua Grecia. Las figuras míticas Pygmalion, Daedalus y Hephaestus pueden interpretarse como inventores legendarios, y Galatea, Talos y Pandora pueden considerarse vida artificial (Ovid y Martin, 2004; Sparkes, 1996; Tandy, 1997).

Cuando las computadoras programables fueron concebidas por primera vez, la gente se preguntaba si tales máquinas podrían volverse inteligentes, más de cien años antes de que se construyera una (Lovelace, 1842). Hoy en día, la **inteligencia** **artificial** (IA) es un campo próspero con muchas aplicaciones prácticas y temas de investigación activa. Buscamos software inteligente para automatizar el trabajo de rutina, entender el habla o las imágenes, hacer diagnósticos en medicina y apoyar la investigación científica básica.

En los primeros días de la inteligencia artificial, el campo abordó y resolvió rápidamente problemas que son intelectualmente difíciles para los seres humanos pero relativamente sencillos para las computadoras, problemas que pueden describirse mediante una lista de reglas formales matemáticas. El verdadero desafío para la inteligencia artificial demostró estar resolviendo las tareas que son fáciles de realizar para las personas, pero difíciles de describir formalmente: problemas que resolvemos intuitivamente, que se sienten automáticos, como reconocer palabras o caras en las imágenes.

Este libro trata sobre una solución a estos problemas más intuitivos. Esta solución es permitir que las computadoras aprendan de la experiencia y entiendan el mundo en términos de una jerarquía de conceptos, con cada concepto definido a través de su relación con conceptos más simples. Al reunir el conocimiento de la experiencia, este enfoque evita la necesidad de que los operadores humanos especifiquen formalmente todo el conocimiento que la computadora necesita. La jerarquía de conceptos permite a la computadora aprender conceptos complicados al construirlos de los más simples. Si dibujamos un grafo que muestra cómo estos conceptos se construyen uno encima del otro, el grafo es profundo, con muchas capas. Por esta razón, llamamos a este enfoque del **aprendizaje** **profundo** de la IA.

Muchos de los primeros éxitos de la IA tuvieron lugar en ambientes relativamente estériles y formales y no requerían que las computadoras tuvieran mucho conocimiento sobre el mundo. Por ejemplo, el sistema de ajedrez Deep Blue de IBM derrotó al campeón mundial Garry Kasparov en 1997 (Hsu, 2002). El ajedrez es, por supuesto, un mundo muy simple, que contiene solo sesenta y cuatro lugares y treinta y dos piezas que pueden moverse en formas rígidamente circunscritas. Lograr una estrategia de ajedrez exitosa es un logro extraordinario, pero el desafío no se debe a la dificultad de describir el conjunto de piezas de ajedrez y los movimientos permitidos en la computadora. El ajedrez se puede describir completamente mediante una lista muy breve de reglas completamente formales, que el programador puede proporcionar con anticipación.

Irónicamente, las tareas abstractas y formales que se encuentran entre las deficiencias mentales más difíciles para un ser humano se encuentran entre las más fáciles para una computadora. Durante mucho tiempo, las computadoras han sido capaces de derrotar incluso al mejor jugador de ajedrez humano, pero solo recientemente han comenzado a igualar algunas de las habilidades de los seres humanos promedio para reconocer objetos o hablar. La vida cotidiana de una persona requiere un inmenso conocimiento sobre el mundo. Gran parte de este conocimiento es subjetivo e intuitivo y, por lo tanto, difícil de articular de manera formal. Las computadoras necesitan capturar este mismo conocimiento para comportarse de manera inteligente. Uno de los desafíos clave en la inteligencia artificial es cómo obtener este conocimiento informal en una computadora.

Varios proyectos de inteligencia artificial han buscado el conocimiento duro sobre el mundo en lenguajes formales. Una computadora puede razonar automáticamente sobre las declaraciones en estos lenguajes formales usando reglas de inferencia lógica. Esto se conoce como un **acercamiento básico** a la inteligencia artificial. Ninguno de estos proyectos ha llevado a un gran éxito. Uno de los proyectos más famosos es Cyc (Lenat y Guha, 1989). Cyc es un motor de inferencia y una base de datos de sentencias en un CycL denominado. Estas declaraciones son introducidas por un personal de supervisores humanos. Es un proceso difícil de manejar. La gente lucha por diseñar reglas formales con suficiente complejidad para describir con precisión el mundo. Por ejemplo, Cyc no pudo comprender una historia sobre una persona llamada Fred que se afeitaba en la mañana (Linde, 1992). Su motor de inferencia detectó una inconsistencia en la historia: sabía que las personas no tienen partes eléctricas, pero debido a que Fred sostenía una máquina de afeitar eléctrica, creía que la entidad "FredWhileShaving" contenía partes eléctricas. Por lo tanto, preguntó si Fred todavía era una persona mientras se afeitaba.

Las dificultades a las que se enfrentan los sistemas que se basan en un conocimiento codificado de forma rígida sugieren que los sistemas de inteligencia artificial necesitan la capacidad de adquirir su propio conocimiento mediante la extracción de patrones a partir de datos sin procesar. Esta capacidad se conoce como **aprendizaje** **automático**. La introducción del aprendizaje automático permitió a las computadoras abordar problemas que involucran el conocimiento del mundo real y tomar decisiones que parecen subjetivas. Un simple algoritmo de aprendizaje automático denominado regresión logística puede determinar si recomendar el parto por cesárea (Mor-Yosef et al., 1990). Un simple algoritmo de aprendizaje automático llamado **Bayes** **ingenuo** puede separar el correo electrónico legítimo del correo no deseado.

El rendimiento de estos simples algoritmos de aprendizaje automático depende en gran medida de la **representación** de los datos que se proporcionan. Por ejemplo, cuando se utiliza la regresión logística para recomendar el parto por cesárea, el sistema de AI no examina al paciente directamente. En cambio, el médico le dice al sistema varias piezas de información relevante, como la presencia o ausencia de una cicatriz uterina. Cada pieza de información incluida en la representación del paciente se conoce como una **característica**. La regresión logística aprende cómo cada una de estas características del paciente se correlaciona con varios resultados. Sin embargo, no puede influir en cómo se definen las características de ninguna manera. Si a la regresión logística se le realizara una resonancia magnética del paciente, en lugar del informe formal del médico, no podría hacer predicciones útiles. Los píxeles individuales en una exploración de MRI tienen una correlación insignificante con cualquier complicación que pueda ocurrir durante el parto.

Esta dependencia de las representaciones es un fenómeno general que aparece en las ciencias de la computación e incluso en la vida cotidiana. En informática, las operaciones como la búsqueda de una recopilación de datos pueden realizarse de forma exponencial más rápida si la recopilación está estructurada e indexada de forma inteligente. Las personas pueden realizar aritmética fácilmente en los números arábigos, pero la aritmética en los números romanos requiere mucho más tiempo. No es sorprendente que la elección de representación tenga un efecto enorme en el rendimiento de los algoritmos de aprendizaje automático. Para un ejemplo visual simple, vea la figura 1.1.

Muchas tareas de inteligencia artificial se pueden resolver diseñando el conjunto correcto de características para extraer para esa tarea, y luego proporcionándolas a un algoritmo simple de aprendizaje automático. Por ejemplo, una característica útil para la identificación del hablante a partir del sonido es una estimación del tamaño del tracto vocal del hablante. Esta característica da una idea clara de si el orador es un hombre, una mujer o un niño.

Para muchas tareas, sin embargo, es difícil saber qué características deben extraerse. Por ejemplo, supongamos que nos gustaría escribir un programa para detectar automóviles en fotografías. Sabemos que los autos tienen ruedas, por lo que nos gustaría usar la presencia de una rueda como característica. Desafortunadamente, es difícil describir exactamente cómo se ve una rueda en términos de valores de píxeles. Una rueda tiene una forma geométrica simple, pero su imagen puede complicarse por las sombras que caen en la rueda, el sol deslumbra sobre las partes metálicas de la rueda, el guardabarros del automóvil o un objeto en el primer plano que oculta parte de la rueda, y luego.

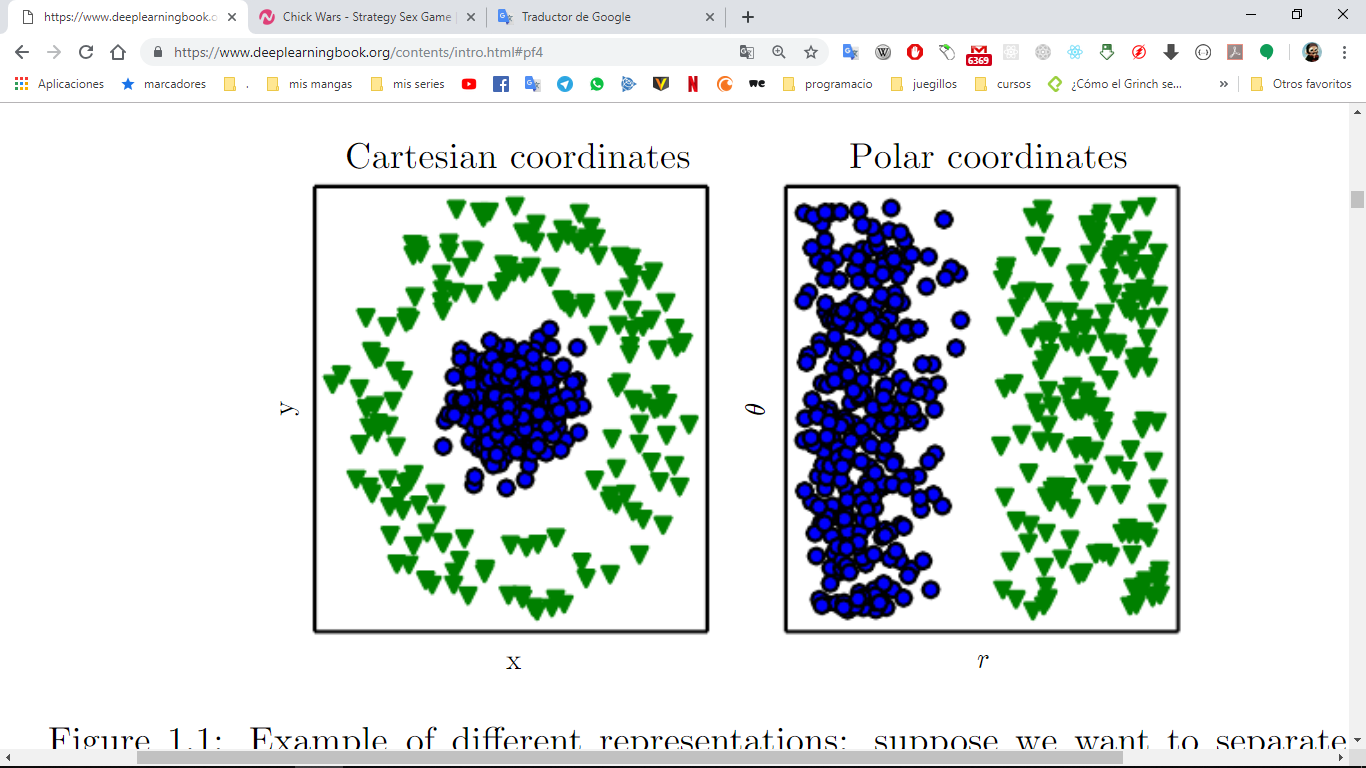


Ilustración 1‑1: Ejemplo de diferentes representaciones: supongamos que queremos separar dos categorías de datos dibujando una línea entre ellas en un diagrama de dispersión. En la gráfica de la izquierda, representamos algunos datos utilizando coordenadas cartesianas, y la tarea es imposible. En la gráfica de la derecha, representamos los datos con coordenadas polares y la tarea se vuelve fácil de resolver con una línea vertical. (Figura producida en colaboración con David Warde-Farley.)

Una solución a este problema es utilizar el aprendizaje automático para descubrir no solo la asignación de la representación a la salida, sino también la representación misma. Este enfoque se conoce como **aprendizaje de representación**. Las representaciones aprendidas a menudo dan como resultado un rendimiento mucho mejor que el que se puede obtener con representaciones diseñadas a mano. También permiten que los sistemas de inteligencia artificial se adapten rápidamente a nuevas tareas, con una intervención humana mínima. Un algoritmo de aprendizaje de representación puede descubrir un buen conjunto de características para una tarea simple en minutos o para una tarea compleja en horas o meses. El diseño manual de características para una tarea compleja requiere mucho tiempo y esfuerzo por parte de los humanos; puede llevar décadas para toda una comunidad de investigadores.

El ejemplo por excelencia de un algoritmo de aprendizaje de representación es el **autoencoder**. Un codificador automático es la combinación de una función de **codificador**, que convierte los datos de entrada en una representación diferente, y una función de **decodificador**, que convierte la nueva representación de nuevo al formato original. Los codificadores automáticos están entrenados para conservar la mayor cantidad de información posible cuando una entrada se ejecuta a través del codificador y luego el decodificador, pero también están entrenados para hacer que la nueva representación tenga varias propiedades interesantes. Diferentes tipos de autoencodificadores tienen como objetivo lograr diferentes tipos de propiedades.

Al diseñar características o algoritmos para las características de aprendizaje, nuestro objetivo generalmente es separar los **factores** **de** **variación** que explican los datos observados. En este contexto, usamos la palabra “factores” simplemente para referirnos a fuentes separadas de influencia; Los factores no suelen combinarse por multiplicación. Tales factores a menudo no son cantidades que se observan directamente. En cambio, pueden existir como objetos no observados o como fuerzas no observadas en el mundo físico que afectan cantidades observables. También pueden existir como constructos en la mente humana que proporcionan explicaciones simplificadas útiles o causas inferidas de los datos observados. Se pueden considerar conceptos o abstracciones que nos ayudan a comprender la variada variabilidad de los datos. Al analizar una grabación de un discurso, los factores de variación incluyen la edad del hablante, su sexo, su acento y las palabras que están hablando. Al analizar una imagen de un automóvil, los factores de variación incluyen la posición del automóvil, su color y el ángulo y brillo del sol.

Una de las principales fuentes de dificultades en muchas aplicaciones de inteligencia artificial en el mundo real es que muchos de los factores de variación influyen en cada pieza de datos que podemos observar. Los píxeles individuales en una imagen de un automóvil rojo podrían estar muy cerca del negro en la noche. La forma de la silueta del coche depende del ángulo de visión. La mayoría de las aplicaciones nos obligan a desentrañar los factores de variación y descartar los que no nos interesan.

Por supuesto, puede ser muy difícil extraer características abstractas de alto nivel de datos sin procesar. Muchos de estos factores de variación, como el acento de un hablante, pueden identificarse solo utilizando una comprensión sofisticada, casi a nivel humano de los datos. Cuando es casi tan difícil obtener una representación como resolver el problema original, el aprendizaje por representación no parece, a primera vista, ayudarnos.

El **aprendizaje** **profundo** resuelve este problema central en el aprendizaje por representación introduciendo representaciones que se expresan en términos de otras representaciones más simples. El aprendizaje profundo permite que la computadora construya conceptos complejos a partir de conceptos más simples. La Figura 1.2 muestra cómo un sistema de aprendizaje profundo puede representar el concepto de una imagen de una persona al combinar conceptos más simples, como esquinas y contornos, que a su vez se definen en términos de bordes.

El ejemplo por excelencia de un modelo de aprendizaje profundo es la red profunda de avance, o **perceptrón** **multicapa** (MLP). Un perceptrón multicapa es solo una función matemática que mapea un conjunto de valores de entrada a valores de salida. La función se forma al componer muchas funciones más simples. Podemos pensar que cada aplicación de una función matemática diferente proporciona una nueva representación de la entrada.

La idea de aprender la representación correcta para los datos proporciona una perspectiva sobre el aprendizaje profundo. Otra perspectiva sobre el aprendizaje profundo es que la profundidad le permite a la computadora aprender un programa de computadora de varios pasos. Cada capa de la representación se puede considerar como el estado de la memoria de la computadora después de ejecutar otro conjunto de instrucciones en paralelo. Las redes con mayor profundidad pueden ejecutar más instrucciones en secuencia. Las instrucciones secuenciales ofrecen un gran poder porque las instrucciones posteriores pueden referirse a los resultados de las instrucciones anteriores. De acuerdo con esta visión del aprendizaje profundo, no toda la información en las activaciones de una capa necesariamente codifica factores de variación que explican la entrada. La representación también almacena información de estado que ayuda a ejecutar un programa que puede dar sentido a la entrada. Esta información de estado podría ser análoga a un contador o puntero en un programa de computadora tradicional. No tiene nada que ver con el contenido de la entrada específicamente, pero ayuda al modelo a organizar su procesamiento.

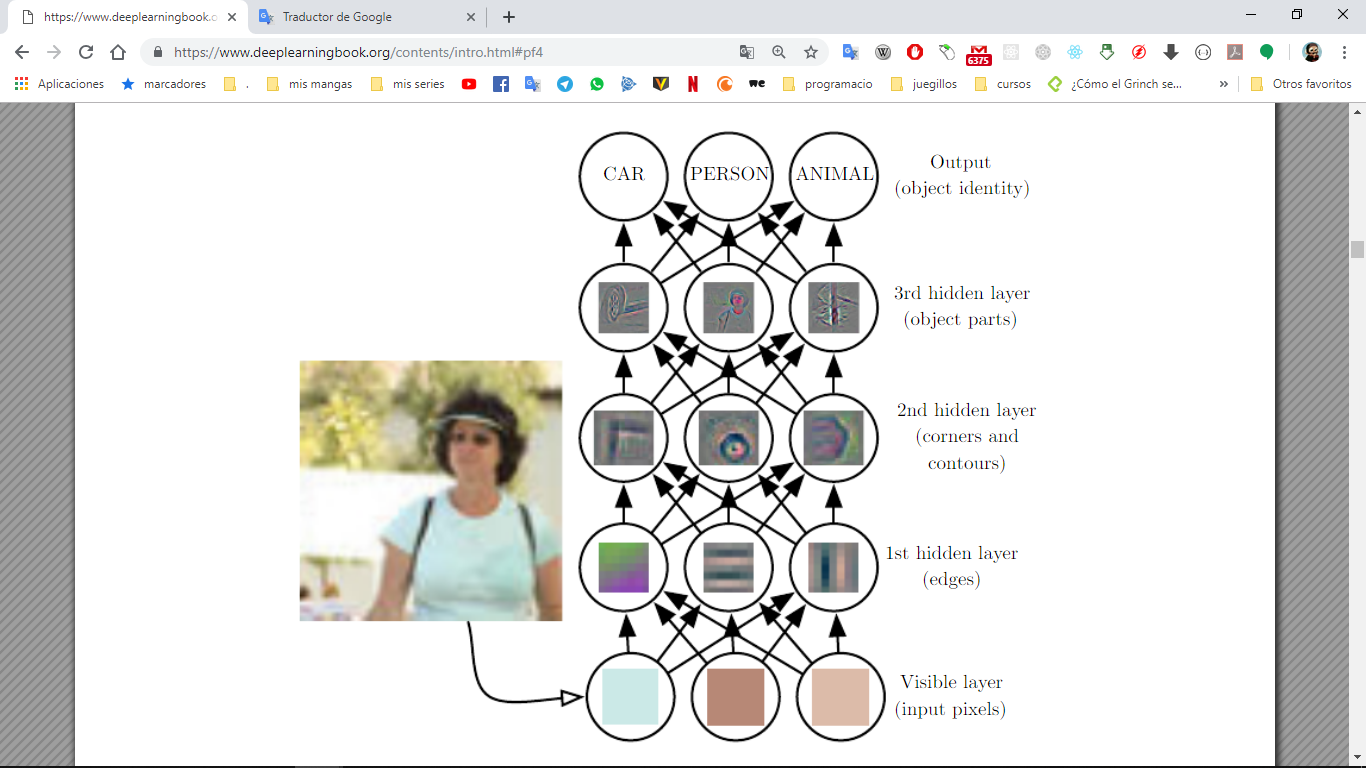


Ilustración 1‑2: Ilustración de un modelo de aprendizaje profundo. Es difícil para una computadora entender el significado de los datos de entrada sensoriales sin procesar, como esta imagen representada como una colección de valores de píxeles. La asignación de funciones de un conjunto de píxeles a una identidad de objeto es muy complicada. Aprender o evaluar este mapeo parece insuperable si se aborda directamente. El aprendizaje profundo resuelve esta dificultad dividiendo la complicada asignación deseada en una serie de asignaciones simples anidadas, cada una descrita por una capa diferente del modelo. La entrada se presenta en la capa visible, llamada así porque contiene las variables que podemos observar. Luego, una serie de capas ocultas extrae características cada vez más abstractas de la imagen. Estas capas se llaman "ocultas" porque sus valores no se dan en los datos; en cambio, el modelo debe determinar qué conceptos son útiles para explicar las relaciones en los datos observados. Las imágenes aquí son visualizaciones del tipo de característica representada por cada unidad oculta. Dados los píxeles, la primera capa puede identificar fácilmente los bordes, comparando el brillo de los píxeles vecinos. Dada la descripción de los bordes de la primera capa oculta, la segunda capa oculta puede buscar fácilmente esquinas y contornos extendidos, que son reconocibles como colecciones de bordes. Dada la descripción de la segunda capa oculta de la imagen en términos de esquinas y contornos, la tercera capa oculta puede detectar partes enteras de objetos específicos, al encontrar colecciones específicas de contornos y esquinas. Finalmente, esta descripción de la imagen en términos de las partes del objeto que contiene puede usarse para reconocer los objetos presentes en la imagen. Imágenes reproducidas con permiso de Zeiler y Fergus (2014).

Hay dos formas principales de medir la profundidad de un modelo. La primera vista se basa en el número de instrucciones secuenciales que deben ejecutarse para evaluar la arquitectura. Podemos pensar en esto como la longitud del camino más largo a través de un gráfico de flujo que describe cómo calcular cada una de las salidas del modelo dadas sus entradas. Del mismo modo que dos programas de computadora equivalentes tendrán diferentes longitudes dependiendo del idioma en el que esté escrito el programa, la misma función se puede dibujar como un diagrama de flujo con diferentes profundidades dependiendo de qué funciones permitimos que se usen como pasos individuales en el diagrama de flujo. La Figura 1.3 ilustra cómo esta elección de idioma puede dar dos medidas diferentes para la misma arquitectura.

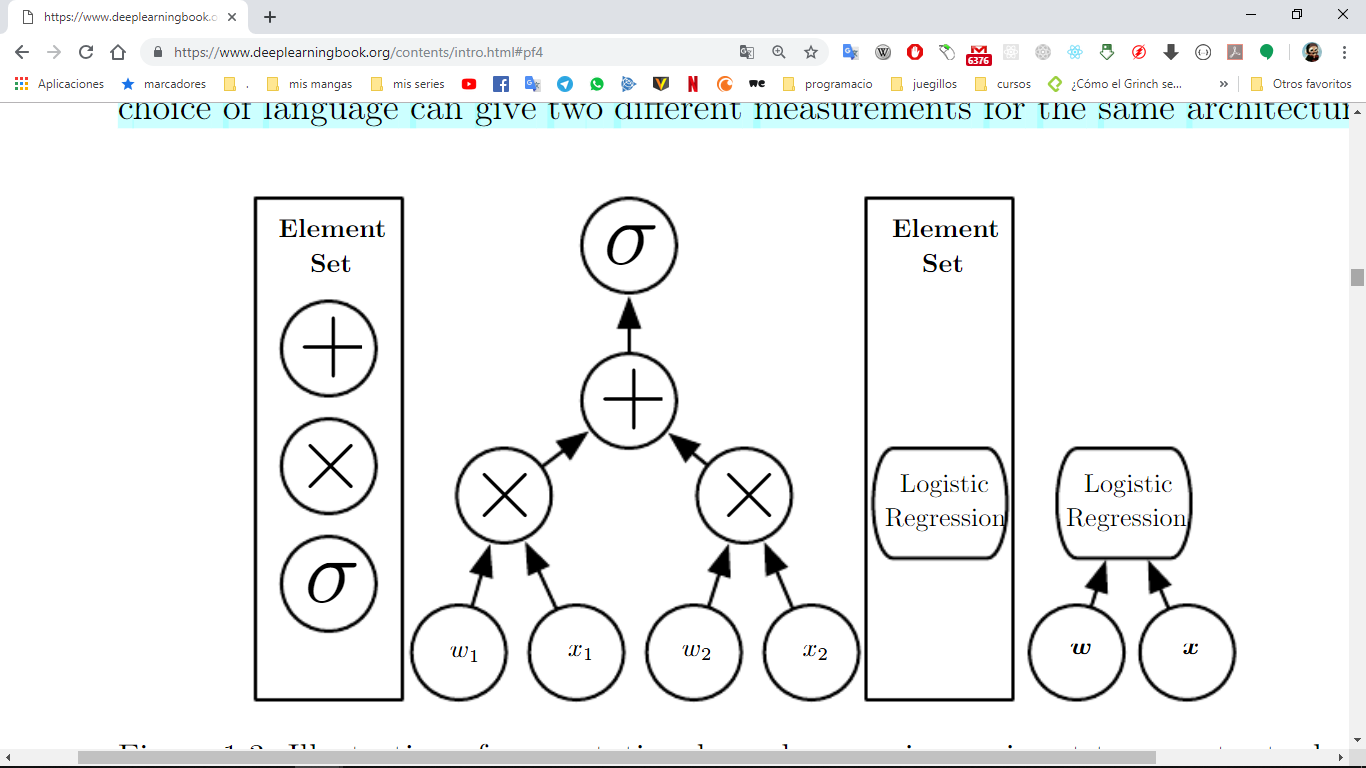


Ilustración 1‑3: Ilustración de gráficos computacionales que asignan una entrada a una salida donde cada nodo realiza una operación. Profundidad es la longitud del camino más largo desde la entrada hasta la salida, pero depende de la definición de lo que constituye un posible paso computacional. El cálculo que se muestra en estos gráficos es la salida de un modelo de regresión logística, σ (wTx), donde está la función sigmoide logística. Si usamos sigmoides de suma, multiplicación y logística como elementos de nuestro lenguaje informático, entonces este modelo tiene la profundidad tres. Si vemos la regresión logística como un elemento en sí mismo, entonces este modelo tiene uno de profundidad.

Otro enfoque, utilizado por los modelos probabilísticos profundos, considera que la profundidad de un modelo no es la profundidad del grafo computacional, sino la profundidad del grafo que describe cómo se relacionan los conceptos entre sí. En este caso, la profundidad del diagrama de flujo de los cálculos necesarios para calcular la representación de cada concepto puede ser mucho más profunda que la gráfica de los conceptos mismos. Esto se debe a que la comprensión del sistema de los conceptos más simples se puede refinar dada la información sobre los conceptos más complejos. Por ejemplo, un sistema de IA que observa una imagen de una cara con un ojo en la sombra puede ver inicialmente solo un ojo. Después de detectar que una cara está presente, el sistema puede inferir que probablemente también esté presente un segundo ojo. En este caso, la gráfica de conceptos incluye solo dos capas, una capa para ojos y una capa para caras, pero el grafo de cómputos incluye 2n capas si refinamos nuestra estimación de cada concepto dados los otros tiempos.

Debido a que no siempre está claro cuál de estas dos vistas (la profundidad del grafo computacional o la profundidad del grafo de modelado probabilístico) es más relevante, y debido a que diferentes personas eligen diferentes conjuntos de elementos más pequeños a partir de los cuales construir sus grafos, no existe un solo valor correcto para la profundidad de una arquitectura, al igual que no hay un solo valor correcto para la longitud de un programa de computadora. Tampoco existe un consenso acerca de cuánta profundidad requiere un modelo para calificar como "profundo". Sin embargo, el aprendizaje profundo puede considerarse de manera segura como el estudio de modelos que involucran una mayor cantidad de composición de funciones aprendidas o conceptos aprendidos que el aprendizaje automático tradicional hace.

Para resumir, el aprendizaje profundo, el tema de este libro, es un enfoque de la IA. Específicamente, es un tipo de aprendizaje automático, una técnica que permite a los sistemas informáticos mejorar con la experiencia y los datos. Sostenemos que el aprendizaje automático es el único enfoque viable para construir sistemas de inteligencia artificial que pueden funcionar en entornos complejos del mundo real. El aprendizaje profundo es un tipo particular de aprendizaje automático que logra un gran poder y flexibilidad al representar el mundo como una jerarquía anidada de conceptos, con cada concepto definido en relación con conceptos más simples y representaciones más abstractas computadas en términos de conceptos menos abstractos. La Figura 1.4 ilustra la relación entre estas diferentes disciplinas de IA. La figura 1.5 muestra un esquema de alto nivel de cómo funciona cada uno.

## ¿Quién debería leer este libro?

Este libro puede ser útil para una variedad de lectores, pero lo escribimos con dos audiencias objetivo en mente. Una de estas audiencias objetivo son los estudiantes universitarios (pregrado o posgrado) que aprenden sobre el aprendizaje automático, incluidos aquellos que están comenzando una carrera en el aprendizaje profundo y la investigación de inteligencia artificial. El otro público objetivo son los ingenieros de software que no tienen conocimientos de aprendizaje automático o estadísticas pero desean adquirirlas rápidamente y comenzar a utilizar el aprendizaje profundo en su producto o plataforma. El aprendizaje profundo ya ha demostrado ser útil en muchas disciplinas de software, que incluyen visión artificial, procesamiento del habla y audio, procesamiento de lenguaje natural, robótica, bioinformática y química, videojuegos, motores de búsqueda, publicidad en línea y finanzas.

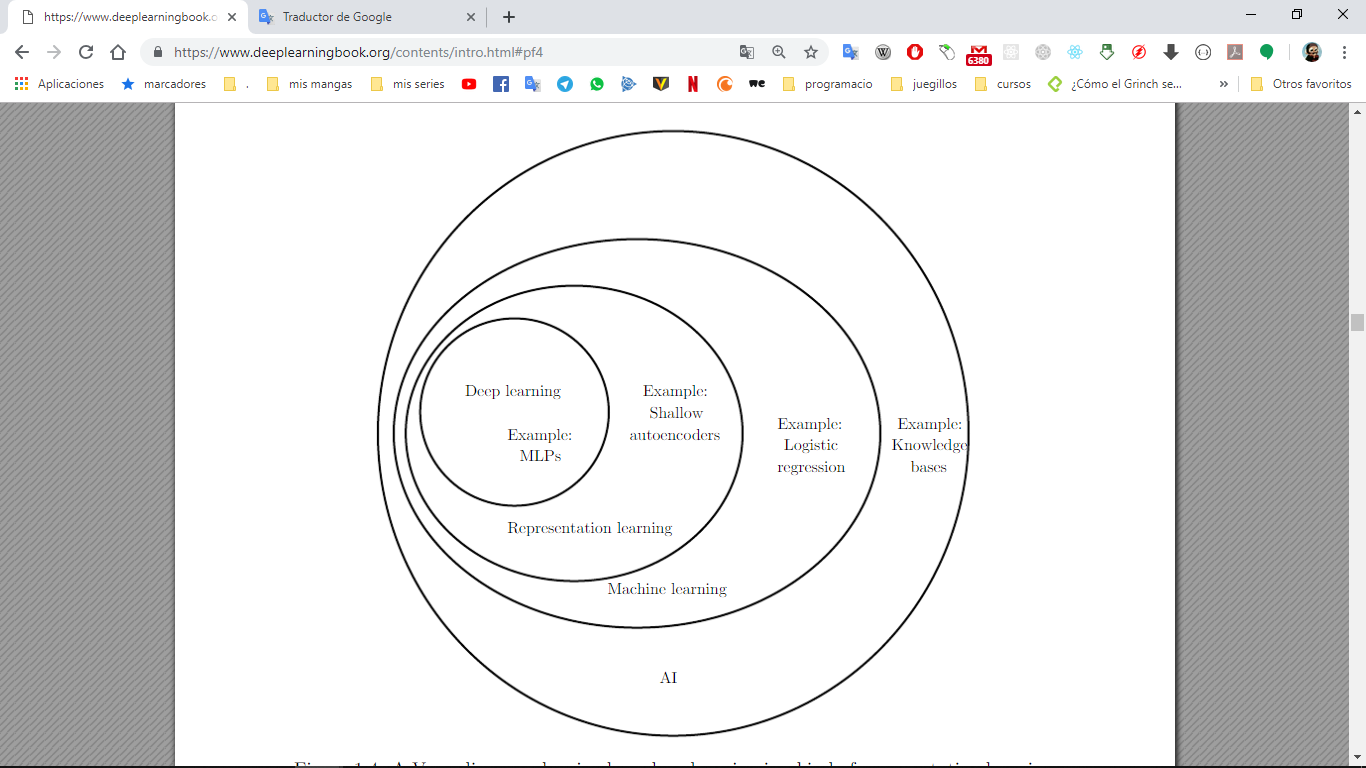


Ilustración 1‑4: Un diagrama de Venn que muestra cómo el aprendizaje profundo es un tipo de aprendizaje de representación, que a su vez es un tipo de aprendizaje automático, que se utiliza para muchos, pero no para todos los enfoques de la IA. Cada sección del diagrama de Venn incluye un ejemplo de una tecnología AI.

Este libro se ha organizado en tres partes para adaptarse mejor a una variedad de lectores. La parte I introduce herramientas matemáticas básicas y conceptos de aprendizaje automático. La Parte II describe los algoritmos de aprendizaje profundo más establecidos, que son esencialmente tecnologías resueltas. La Parte III describe ideas más especulativas que se cree que son importantes para futuras investigaciones en el aprendizaje profundo.

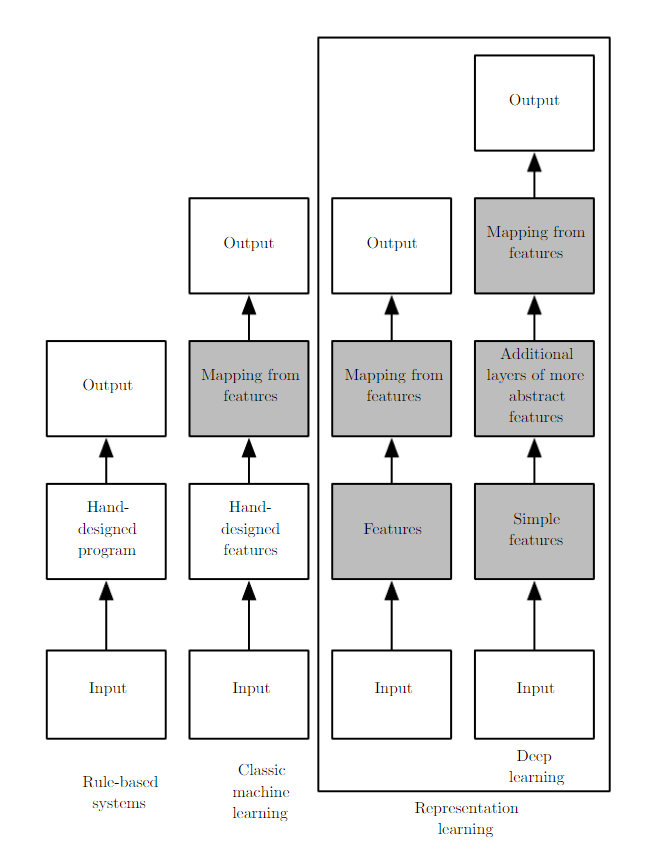


Ilustración 1‑5: Los diagramas de flujo muestran cómo las diferentes partes de un sistema de AI se relacionan entre sí dentro de diferentes disciplinas de AI. Los cuadros sombreados indican componentes que pueden aprender de los datos.

Los lectores deben sentirse libres de saltear partes que no son relevantes dados sus intereses o antecedentes. Los lectores familiarizados con el álgebra lineal, la probabilidad y los conceptos fundamentales de aprendizaje automático pueden omitir la parte I, por ejemplo, mientras que aquellos que solo quieren implementar un sistema operativo no necesitan leer más allá de la parte II. Para ayudar a elegir qué capítulos leer, la figura 1.6 proporciona un diagrama de flujo que muestra el alto nivel de organización del libro.

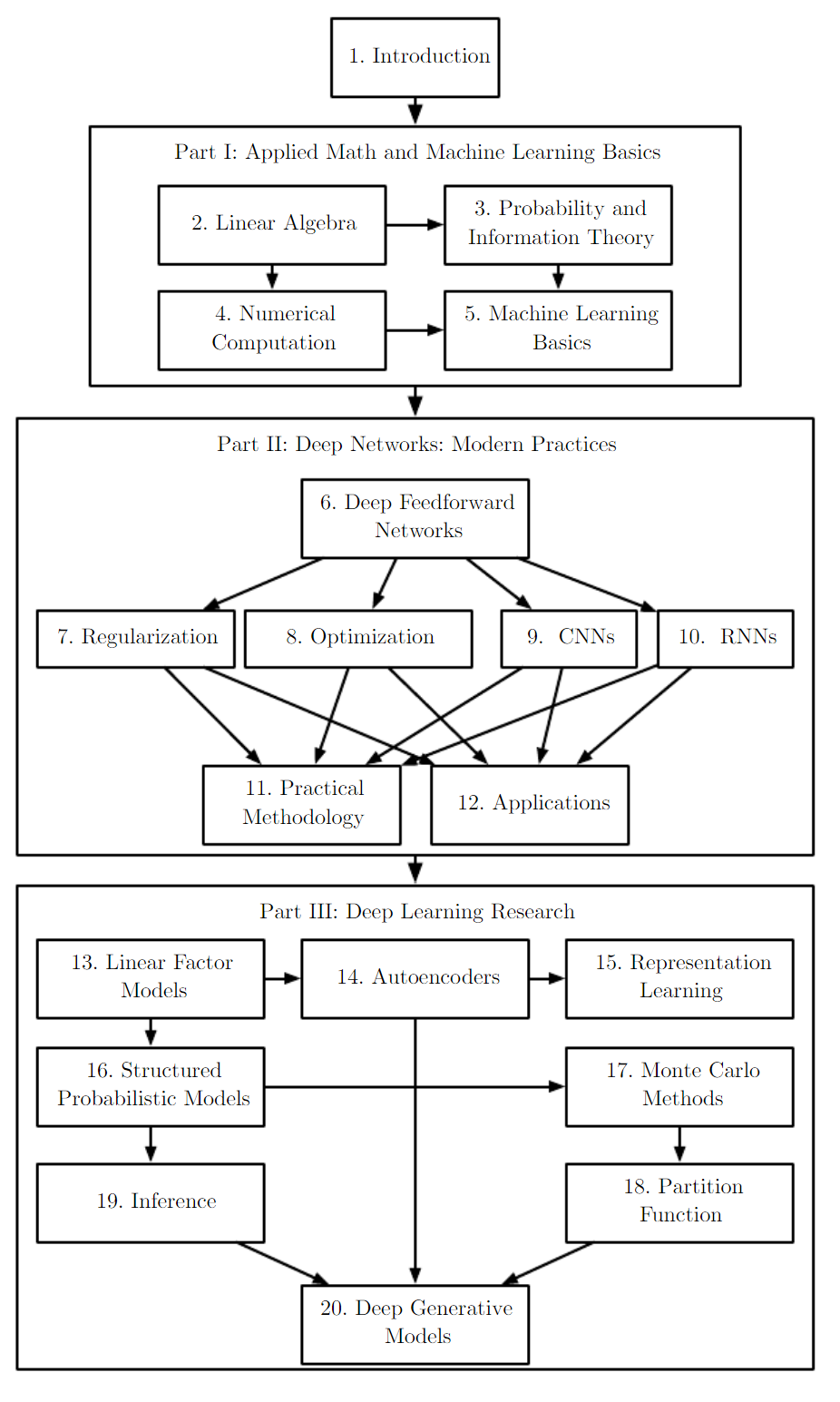


Ilustración 1‑6: La organización de alto nivel del libro. Una flecha de un capítulo a otro indica que el capítulo anterior es un requisito previo para comprender este último.

Suponemos que todos los lectores provienen de una formación informática. Asumimos familiaridad con la programación, una comprensión básica de los problemas de rendimiento computacional, teoría de la complejidad, cálculo de nivel introductorio y parte de la terminología de la teoría de grafos.

## Tendencias históricas en el aprendizaje profundo

Es más fácil entender el aprendizaje profundo con algún contexto histórico. En lugar de proporcionar una historia detallada del aprendizaje profundo, identificamos algunas tendencias clave:

* El aprendizaje profundo ha tenido una larga y rica historia, pero ha tenido muchos nombres, reflejando diferentes puntos de vista filosóficos, y ha ido creciendo y disminuyendo en popularidad.
* El aprendizaje profundo se ha vuelto más útil a medida que aumenta la cantidad de datos de capacitación disponibles.
* Los modelos de aprendizaje profundo han crecido en tamaño con el tiempo a medida que la infraestructura informática (tanto hardware como software) para que el aprendizaje profundo mejore.
* El aprendizaje profundo ha resuelto aplicaciones cada vez más complicadas con mayor precisión a lo largo del tiempo.

### Los muchos nombres y fortuitos cambios de Redes Neuronales

Esperamos que muchos lectores de este libro hayan oído hablar del aprendizaje profundo como una tecnología nueva y emocionante, y estamos sorprendidos de ver una mención de "historia" en un libro sobre un campo emergente. De hecho, el aprendizaje profundo se remonta a la década de 1940. El aprendizaje profundo solo parece ser nuevo, ya que fue relativamente impopular durante varios años antes de su popularidad actual, y debido a que ha recibido muchos nombres diferentes, solo recientemente se ha denominado “aprendizaje profundo”. El campo ha sido renombrado muchas veces, reflejando la Influencia de diferentes investigadores y diferentes perspectivas.

Una historia completa del aprendizaje profundo está más allá del alcance de este libro de texto. Sin embargo, algún contexto básico es útil para comprender el aprendizaje profundo. En términos generales, ha habido tres oleadas de desarrollo: el aprendizaje profundo conocido como **cibernética** en la década de 1940 a 1960, el aprendizaje profundo conocido como **conexionismo** en la década de 1980 a 1990 y el resurgimiento actual bajo el nombre de aprendizaje profundo a partir de 2006. Esto se ilustra cuantitativamente Figura 1.7.

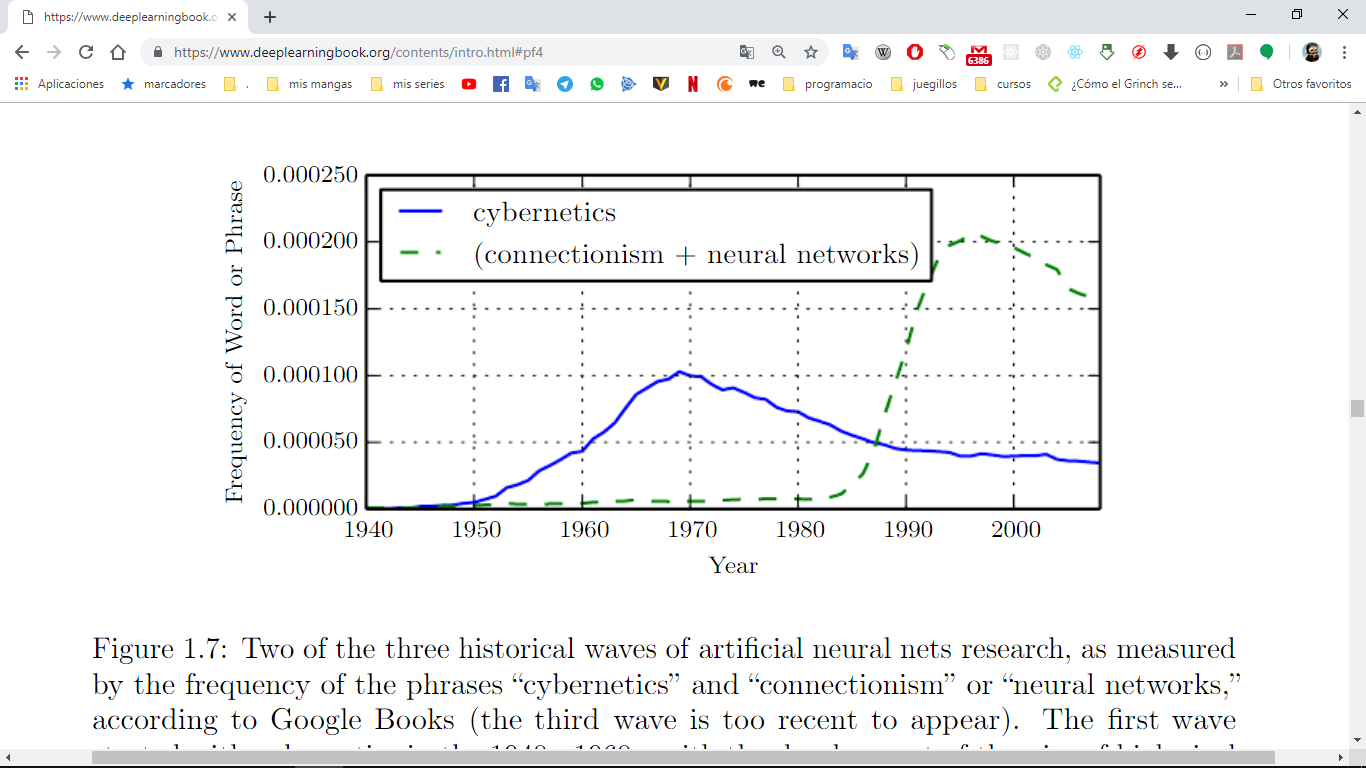


Ilustración 1‑7: Dos de las tres oleadas históricas de redes neuronales artificiales investigan, según la frecuencia de las frases "cibernética" y "conexionismo" o "redes neuronales", según Google Books (la tercera oleada es demasiado reciente para aparecer). La primera ola comenzó con la cibernética en la década de 1940 y 1960, con el desarrollo de teorías del aprendizaje biológico (McCulloch y Pitts, 1943; Hebb, 1949) y las implementaciones de los primeros modelos, como el perceptrón (Rosenblatt, 1958), que permitió el entrenamiento de una sola neurona. La segunda ola comenzó con el enfoque conexionista del período 1980-1995, con propagación hacia atrás (Rumelhart et al., 1986a) para entrenar una red neuronal con una o dos capas ocultas. La actual y tercera ola, el aprendizaje profundo, comenzó alrededor de 2006 (Hinton et al., 2006; Bengio et al., 2007; Ranzato et al., 2007a) y está apareciendo ahora en forma de libro a partir de 2016. Las otras dos olas apareció de forma similar en forma de libro mucho más tarde de lo que ocurrió la actividad científica correspondiente.

Algunos de los algoritmos de aprendizaje más tempranos que reconocemos hoy pretendían ser modelos computacionales de aprendizaje biológico, es decir, modelos de cómo el aprendizaje ocurre o podría ocurrir en el cerebro. Como resultado, uno de los nombres por los que ha pasado el aprendizaje profundo es **redes** **neuronales** **artificiales** (ANN). La perspectiva correspondiente de los modelos de aprendizaje profundo es que se trata de sistemas de ingeniería inspirados en el cerebro biológico (ya sea el cerebro humano o el cerebro de otro animal). Si bien los tipos de redes neuronales utilizadas para el aprendizaje automático a veces se han utilizado para entender la función cerebral (Hinton y Shallice, 1991), en general no están diseñadas para ser modelos realistas de función biológica. La perspectiva neuronal del aprendizaje profundo está motivada por dos ideas principales. Una idea es que el cerebro proporciona una prueba con el ejemplo de que el comportamiento inteligente es posible, y un camino conceptual hacia la construcción de la inteligencia es la ingeniería inversa de los principios computacionales detrás del cerebro y duplicar su funcionalidad. Otra perspectiva es que sería profundamente interesante comprender el cerebro y los principios que subyacen a la inteligencia humana, por lo que los modelos de aprendizaje automático que arrojan luz sobre estas preguntas científicas básicas son útiles además de su capacidad para resolver aplicaciones de ingeniería.

El término moderno “aprendizaje profundo” va más allá de la perspectiva neurocientífica sobre la generación actual de modelos de aprendizaje automático. Apela a un principio más general de aprendizaje de múltiples niveles de composición, que se puede aplicar en marcos de aprendizaje automático que no son necesariamente de inspiración neuronal.

La neurona de McCulloch-Pitts (McCulloch y Pitts, 1943) fue un modelo temprano de la función cerebral. Este modelo lineal podría reconocer dos categorías diferentes de entradas al probar si f (x, w) es positivo o negativo. Por supuesto, para que el modelo se corresponda con la definición deseada de las categorías, los pesos deben establecerse correctamente. Estos pesos pueden ser establecidos por el operador humano. En la década de 1950, el perceptrón (Rosenblatt, 1958, 1962) se convirtió en el primer modelo que podía aprender los pesos que definían las categorías a través de ejemplos de entradas de cada categoría. El **elemento** **lineal** **adaptativo** (ADALINE), que data de aproximadamente el mismo tiempo, simplemente devolvió el valor de f (x) para predecir un número real (Widrow y Hoff, 1960) y también podría aprender a predecir estos números a partir de los datos.

Estos simples algoritmos de aprendizaje afectaron en gran medida el panorama moderno del aprendizaje automático. El algoritmo de entrenamiento utilizado para adaptar los pesos de ADALINE fue un caso especial de un algoritmo llamado **descenso** **de** **gradiente** **estocástico**. Las versiones ligeramente modificadas del algoritmo de descenso de gradiente estocástico siguen siendo los algoritmos de entrenamiento dominantes para los modelos de aprendizaje profundo en la actualidad.

Los modelos basados en la f (x, w) utilizados por el perceptrón y ADALINE se denominan **modelos** **lineales**. Estos modelos siguen siendo algunos de los modelos de aprendizaje automático más utilizados, aunque en muchos casos se capacitan de manera diferente a la de los modelos originales.

Los modelos lineales tienen muchas limitaciones. Lo más famoso es que no pueden aprender la función XOR, donde f ([0,1], w) = 1 y f ([1,0], w) = 1 pero f ([1,1], w) = 0 y f ([0,0], w) = 0. Los críticos que observaron estos errores en modelos lineales causaron una reacción en contra del aprendizaje de inspiración biológica en general (Minsky y Papert, 1969). Esta fue la primera caída importante en la popularidad de las redes neuronales.

Hoy en día, la neurociencia es considerada como una importante fuente de inspiración para los investigadores de aprendizaje profundo, pero ya no es la guía predominante para el campo.

La razón principal del papel disminuido de la neurociencia en la investigación de aprendizaje profundo hoy en día es que simplemente no tenemos suficiente información sobre el cerebro para usarla como guía. Para obtener una comprensión profunda de los algoritmos reales utilizados por el cerebro, deberíamos poder monitorear la actividad de (al menos) miles de neuronas interconectadas simultáneamente. Como no podemos hacerlo, estamos lejos de comprender incluso algunas de las partes más simples y mejor estudiadas del cerebro (Olshausen y Field, 2005).

La neurociencia nos ha dado una razón para esperar que un solo algoritmo de aprendizaje profundo pueda resolver muchas tareas diferentes. Los neurocientíficos han descubierto que los hurones pueden aprender a "ver" con la región de procesamiento auditivo de su cerebro si sus cerebros están cableados para enviar señales visuales a esa área (Von Melchner et al., 2000). Esto sugiere que gran parte del cerebro de los mamíferos podría usar un solo algoritmo para resolver la mayoría de las diferentes tareas que resuelve el cerebro. Antes de esta hipótesis, la investigación del aprendizaje automático estaba más fragmentada, con diferentes comunidades de investigadores que estudiaban el procesamiento del lenguaje natural, la visión, la planificación del movimiento y el reconocimiento de voz. Hoy en día, estas comunidades de aplicaciones aún están separadas, pero es común que los grupos de investigación de aprendizaje profundo estudien muchas o incluso todas estas áreas de aplicación simultáneamente.

Podemos dibujar algunas pautas aproximadas de la neurociencia. La idea básica de tener muchas unidades computacionales que se vuelven inteligentes solo a través de sus interacciones entre sí está inspirada en el cerebro. El neocognitron (Fukushima, 1980) introdujo una poderosa arquitectura de modelo para procesar imágenes que se inspiró en la estructura del sistema visual de los mamíferos y luego se convirtió en la base de la moderna red convolucional (LeCun et al., 1998b), como veremos en sección 9.10. La mayoría de las redes neuronales actuales se basan en un modelo de neurona llamada **unidad lineal rectificada**. El cognitron original (Fukushima, 1975) introdujo una versión más complicada que fue altamente inspirada por nuestro conocimiento de la función cerebral. La versión moderna simplificada fue desarrollada incorporando ideas desde muchos puntos de vista, con Nair y Hinton (2010) y Glorot et al. (2011a) citando a la neurociencia como una influencia, y Jarrett et al. (2009) citando más influencias orientadas a la ingeniería. Si bien la neurociencia es una fuente importante de inspiración, no debe tomarse como una guía rígida. Sabemos que las neuronas reales computan funciones muy diferentes a las unidades lineales rectificadas modernas, pero un mayor realismo neuronal aún no ha conducido a una mejora en el rendimiento del aprendizaje automático. Además, si bien la neurociencia ha inspirado con éxito varias arquitecturas de redes neuronales, todavía no sabemos lo suficiente sobre el aprendizaje biológico para que la neurociencia ofrezca mucha orientación para los algoritmos de aprendizaje que usamos para entrenar estas arquitecturas.

Las cuentas de los medios a menudo enfatizan la similitud del aprendizaje profundo con el cerebro. Si bien es cierto que los investigadores de aprendizaje profundo tienen más probabilidades de citar al cerebro como una influencia que los investigadores que trabajan en otros campos de aprendizaje automático, como las máquinas de núcleo o las estadísticas bayesianas, no se debe considerar el aprendizaje profundo como un intento de simular el cerebro. El aprendizaje profundo moderno se inspira en muchos campos, especialmente los fundamentos matemáticos aplicados como el álgebra lineal, la probabilidad, la teoría de la información y la optimización numérica. Mientras que algunos investigadores de aprendizaje profundo citan la neurociencia como una importante fuente de inspiración, otros no están interesados en la neurociencia en absoluto.

Vale la pena señalar que el esfuerzo por comprender cómo funciona el cerebro en un nivel algorítmico está vivo y bien. Este esfuerzo se conoce principalmente como "neurociencia computacional" y es un campo de estudio separado del aprendizaje profundo. Es común que los investigadores se muevan de un lado a otro entre ambos campos. El campo del aprendizaje profundo se ocupa principalmente de cómo construir sistemas informáticos capaces de resolver con éxito tareas que requieren inteligencia, mientras que el campo de la neurociencia computacional se ocupa principalmente de construir modelos más precisos de cómo funciona realmente el cerebro.

En la década de 1980, la segunda ola de investigación de redes neuronales surgió en gran parte a través de un movimiento llamado **conexionismo** o **procesamiento** **distribuido** **en** **paralelo** (Rumelhart et al., 1986c; McClelland et al., 1995). El conexionismo surgió en el contexto de la ciencia cognitiva. La ciencia cognitiva es un enfoque interdisciplinario para comprender la mente, combinando múltiples niveles de análisis diferentes. A principios de la década de 1980, la mayoría de los científicos cognitivos estudiaron modelos de razonamiento simbólico. A pesar de su popularidad, los modelos simbólicos eran difíciles de explicar en términos de cómo el cerebro podía implementarlos usando neuronas. Los conexionistas comenzaron a estudiar modelos de cognición que en realidad podrían basarse en implementaciones neuronales (Touretzky y Minton, 1985), reviviendo muchas ideas que se remontan al trabajo del psicólogo Donald Hebb en la década de 1940 (Hebb, 1949).

En la década de 1980, la segunda ola de investigación de redes neuronales surgió en gran parte a través de un movimiento llamado conexionismo o procesamiento distribuido en paralelo (Rumelhart et al., 1986c; McClelland et al., 1995). El conexionismo surgió en el contexto de la ciencia cognitiva. La ciencia cognitiva es un enfoque interdisciplinario para comprender la mente, combinando múltiples niveles de análisis diferentes. A principios de la década de 1980, la mayoría de los científicos cognitivos estudiaron modelos de razonamiento simbólico. A pesar de su popularidad, los modelos simbólicos eran difíciles de explicar en términos de cómo el cerebro podía implementarlos usando neuronas. Los conexionistas comenzaron a estudiar modelos de cognición que en realidad podrían basarse en implementaciones neuronales (Touretzky y Minton, 1985), reviviendo muchas ideas que se remontan al trabajo del psicólogo Donald Hebb en la década de 1940 (Hebb, 1949).

La idea central en el conexionismo es que una gran cantidad de unidades computacionales simples pueden lograr un comportamiento inteligente cuando están conectadas en red. Esta visión se aplica igualmente a las neuronas en los sistemas nerviosos biológicos, como lo hace a las unidades ocultas en los modelos computacionales.

Varios conceptos clave surgieron durante el movimiento de conexionismo de la década de 1980 que sigue siendo fundamental para el aprendizaje profundo de hoy.

Uno de estos conceptos es el de la **representación** **distribuida** (Hinton et al., 1986). Esta es la idea de que cada entrada a un sistema debe estar representada por muchas características, y cada función debe participar en la representación de muchas entradas posibles. Por ejemplo, supongamos que tenemos un sistema de visión que puede reconocer automóviles, camiones y aves, y que estos objetos pueden ser rojos, verdes o azules. Una forma de representar estas entradas sería tener una neurona separada o una unidad oculta que se active para cada una de las nueve combinaciones posibles: camión rojo, coche rojo, pájaro rojo, camión verde, etc. Esto requiere nueve neuronas diferentes, y cada neurona debe aprender de forma independiente el concepto de color y la identidad del objeto. Una forma de mejorar esta situación es usar una representación distribuida, con tres neuronas que describen el color y tres neuronas que describen la identidad del objeto. Esto requiere solo un total de seis neuronas en lugar de nueve, y la neurona que describe el enrojecimiento puede aprender sobre el enrojecimiento de imágenes de automóviles, camiones y aves, no solo de imágenes de una categoría específica de objetos. El concepto de representación distribuida es fundamental para este libro y se describe con mayor detalle en el capítulo 15.

Otro logro importante del movimiento conexionista fue el uso exitoso de la back-propagation para entrenar redes neuronales profundas con representaciones internas y la popularización del algoritmo de back-propagation (Rumelhart et al., 1986a; LeCun, 1987). Este algoritmo ha crecido y disminuido en popularidad pero, a partir de este escrito, es el enfoque dominante para entrenar modelos profundos.

Durante la década de 1990, los investigadores hicieron avances importantes en el modelado de secuencias con redes neuronales. Hochreiter (1991) y Bengio et al. (1994) identificaron algunas de las dificultades matemáticas fundamentales para modelar secuencias largas, que se describen en la sección 10.7. Hochreiter y Schmidhuber (1997) introdujeron la red de memoria a corto plazo (LSTM) para resolver algunas de estas dificultades. Hoy en día, el LSTM se usa ampliamente para muchas tareas de modelado de secuencias, incluidas muchas tareas de procesamiento de lenguaje natural en Google.

La segunda ola de investigación de redes neuronales duró hasta mediados de los años noventa. Las empresas basadas en redes neuronales y otras tecnologías de inteligencia artificial empezaron a hacer reclamos ambiciosos irreales mientras buscaban inversiones. Cuando la investigación de AI no cumplió con estas expectativas irrazonables, los inversores se sintieron decepcionados. Simultáneamente, otros campos de aprendizaje automático hicieron avances. Las máquinas Kernel (Boser et al., 1992; Cortes y Vapnik, 1995; Schölkopf et al., 1999) y los modelos gráficos (Jordan, 1998) lograron buenos resultados en muchas tareas importantes. Estos dos factores llevaron a una disminución en la popularidad de las redes neuronales que duró hasta 2007.

Durante este tiempo, las redes neuronales continuaron obteniendo un rendimiento impresionante en algunas tareas (LeCun et al., 1998b; Bengio et al., 2001). El Instituto Canadiense de Investigación Avanzada (CIFAR) ayudó a mantener viva la investigación de redes neuronales a través de su iniciativa de investigación de Computación Neural y Percepción Adaptativa (NCAP). Este programa reunió grupos de investigación de aprendizaje automático liderados por Geoffrey Hinton en la Universidad de Toronto, Yoshua Bengio en la Universidad de Montreal y Yann LeCun en la Universidad de Nueva York. La iniciativa de investigación multidisciplinaria CIFAR NCAP también incluyó neurocientíficos y expertos en visión humana y computacional.

En este punto, en general se creía que las redes profundas eran muy difíciles de entrenar. Ahora sabemos que los algoritmos que han existido desde la década de 1980 funcionan bastante bien, pero esto no fue aparente asta alrededor del 2006. El problema quizás sea simplemente que estos algoritmos eran demasiado costosos computacionalmente para permitir mucha experimentación con el hardware disponible en ese momento.

La tercera ola de investigación de redes neuronales comenzó con un gran avance en 2006. Geoffrey Hinton demostró que un tipo de red neuronal llamada red de creencias profundas podría ser entrenada de manera eficiente utilizando una estrategia llamada pre-entrenamiento codicioso (Hinton et al., 2006). que describimos con más detalle en la sección 15.1. Los otros grupos de investigación afiliados a CIFAR demostraron rápidamente que la misma estrategia podría usarse para entrenar muchos otros tipos de redes profundas (Bengio et al., 2007; Ranzato et al., 2007a) y ayudaron sistemáticamente a mejorar la generalización en los ejemplos de pruebas. Esta ola de investigación de redes neuronales popularizó el uso del término "aprendizaje profundo" para enfatizar que los investigadores ahora podían entrenar redes neuronales más profundas de lo que había sido posible antes, y para centrar la atención en la importancia teórica de la profundidad (Bengio y LeCun, 2007; Delalleau y Bengio, 2011; Pascanu et al., 2014a; Montufar et al., 2014). En este momento, las redes neuronales profundas superaron a los sistemas de inteligencia artificial de la competencia basados en otras tecnologías de aprendizaje automático, así como en la funcionalidad diseñada a mano. Esta tercera ola de popularidad de las redes neuronales continúa hasta el momento de este escrito, aunque el enfoque de la investigación de aprendizaje profundo ha cambiado dramáticamente en el tiempo de esta ola. La tercera ola comenzó con un enfoque en las nuevas técnicas de aprendizaje no supervisadas y la capacidad de los modelos profundos para generalizar bien a partir de conjuntos de datos pequeños, pero hoy en día hay más interés en los algoritmos de aprendizaje supervisado mucho más antiguos y la capacidad de los modelos profundos para aprovechar grandes conjuntos de datos etiquetados.

### Aumento del tamaño de los conjuntos de datos

Uno puede preguntarse por qué el aprendizaje en profundidad se ha reconocido recientemente como una tecnología crucial, aunque los primeros experimentos con redes neuronales artificiales se llevaron a cabo en la década de 1950. El aprendizaje profundo se ha utilizado con éxito en aplicaciones comerciales desde la década de 1990, pero a menudo se lo consideraba más un arte que una tecnología y algo que solo un experto podía usar, hasta hace poco. Es cierto que se requiere cierta habilidad para obtener un buen rendimiento de un algoritmo de aprendizaje profundo. Afortunadamente, la cantidad de habilidad requerida se reduce a medida que aumenta la cantidad de datos de entrenamiento. Los algoritmos de aprendizaje que alcanzan el desempeño humano en tareas complejas hoy en día son casi idénticos a los algoritmos de aprendizaje que lucharon por resolver problemas de juguetes en la década de 1980, aunque los modelos que entrenamos con estos algoritmos han sufrido cambios que simplifican la capacitación de arquitecturas muy profundas. El nuevo desarrollo más importante es que hoy podemos proporcionar estos algoritmos con los recursos que necesitan para tener éxito. La Figura 1.8 muestra cómo el tamaño de los conjuntos de datos de referencia se ha ampliado notablemente con el tiempo. esta tendencia es impulsada por la creciente digitalización de la sociedad. A medida que más y más de nuestras actividades tienen lugar en las computadoras, más y más de lo que hacemos se registra. A medida que nuestras computadoras están cada vez más conectadas en red, se vuelve más fácil centralizar estos registros y agruparlos en un conjunto de datos adecuado para las aplicaciones de aprendizaje automático. La era del "Big Data" ha facilitado mucho el aprendizaje automático porque la carga clave de la estimación estadística, que se generaliza bien a los nuevos datos después de observar solo una pequeña cantidad de datos, se ha aligerado considerablemente. A partir de 2016, una regla general es que un algoritmo de aprendizaje profundo supervisado generalmente logrará un rendimiento aceptable con alrededor de 5,000 ejemplos etiquetados por categoría y coincidirá o superará el desempeño humano cuando se entrene con un conjunto de datos que contenga al menos 10 millones de ejemplos etiquetados. Trabajar con éxito con conjuntos de datos más pequeños es un área de investigación importante, que se centra en particular en cómo podemos aprovechar las grandes cantidades de ejemplos sin etiqueta, con aprendizaje sin supervisión o semi-supervisado.

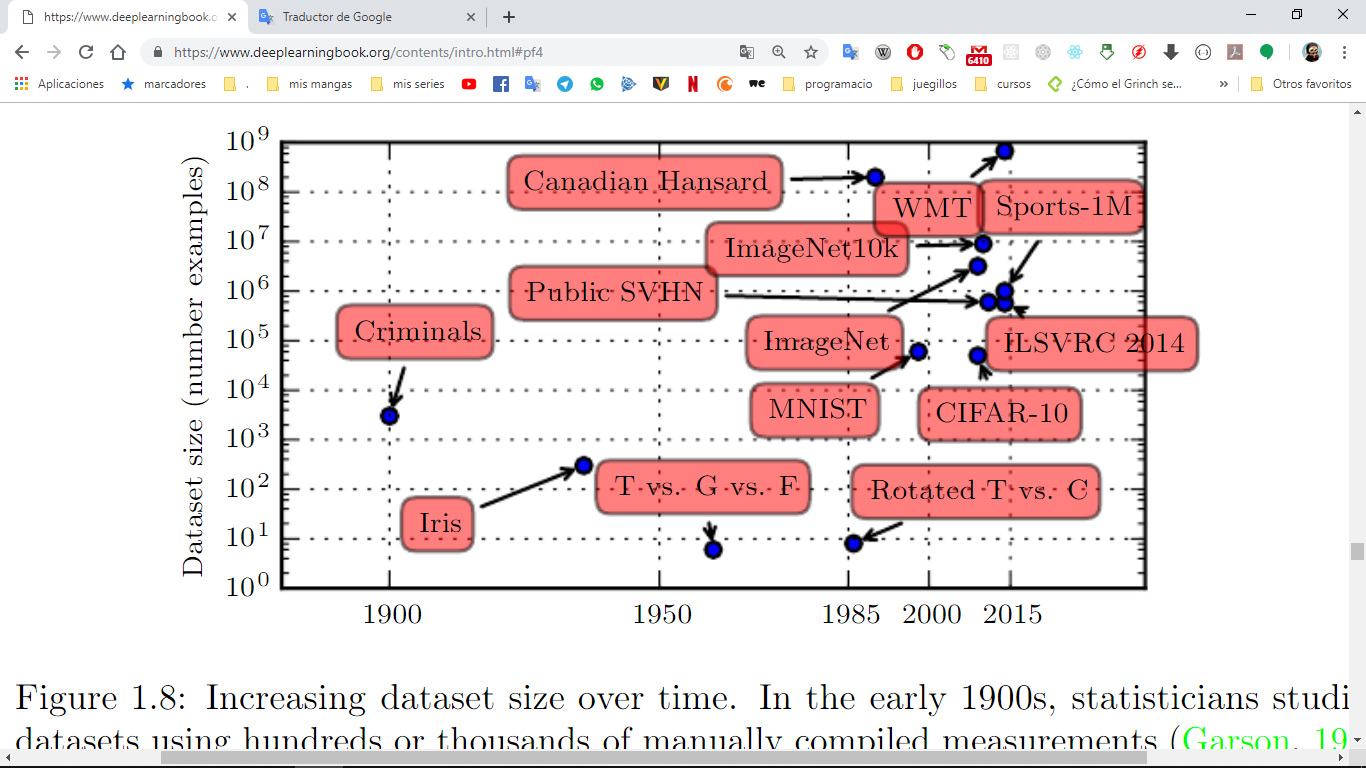


Ilustración 1‑8: Aumentar el tamaño del conjunto de datos con el tiempo. A principios de la década de 1900, los estadísticos estudiaron conjuntos de datos utilizando cientos o miles de mediciones compiladas manualmente (Garson, 1900; Gosset, 1908; Anderson, 1935; Fisher, 1936). Desde la década de 1950 hasta la década de 1980, los pioneros del aprendizaje automático de inspiración biológica trabajaron a menudo con pequeños conjuntos de datos sintéticos, como mapas de bits de baja resolución, que fueron diseñados para incurrir en un bajo costo computacional y demostrar que las redes neuronales podían aprender tipos específicos de funciones (Widrow y Hoff, 1960; Rumelhart et al., 1986b). En los años 80 y 90, el aprendizaje automático se hizo más estadístico y comenzó a aprovechar conjuntos de datos más grandes que contenían decenas de miles de ejemplos, como el conjunto de datos MNIST (que se muestra en la figura 1.9) de escaneos de números escritos a mano (LeCun et al., 1998b). En la primera década del 2000, se continuaron produciendo conjuntos de datos más sofisticados de este mismo tamaño, como el conjunto de datos CIFAR-10 (Krizhevsky y Hinton, 2009). Hacia el final de esa década y durante la primera mitad de la década de 2010, conjuntos de datos significativamente más grandes, que contenían de cientos de miles a decenas de millones de ejemplos, cambiaron por completo lo que era posible con el aprendizaje profundo. Estos conjuntos de datos incluyeron el conjunto de datos de Street View House Numbers públicos (Netzer et al., 2011), varias versiones del conjunto de datos ImageNet (Deng et al., 2009, 2010a; Russakovsky et al., 2014a), y el conjunto de datos Sports-1M (Karpathy et al., 2014). En la parte superior del gráfico, vemos que los conjuntos de datos de oraciones traducidas, como el conjunto de datos de IBM construido a partir del Hansard canadiense (Brown et al., 1990) y el conjunto de datos de inglés a francés de WMT 2014 (Schwenk, 2014), suelen estar muy por delante de otros tamaños de conjuntos de datos.

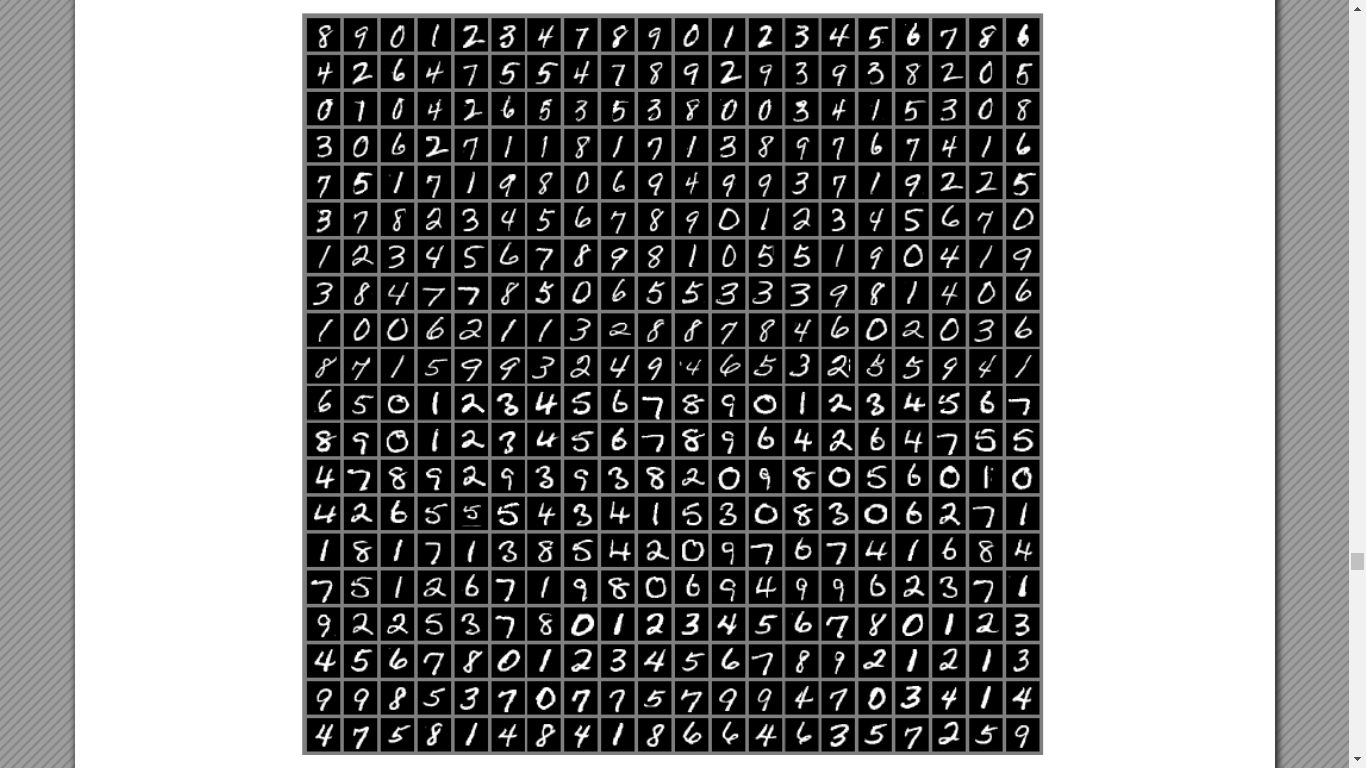


Ilustración 1‑9: : Ejemplo de entradas del conjunto de datos MNIST. El "NIST" significa Instituto Nacional de Estándares y Tecnología, la agencia que originalmente recopiló estos datos. La "M" significa "modificado", ya que los datos han sido preprocesados para un uso más sencillo con algoritmos de aprendizaje automático. El conjunto de datos MNIST consiste en escaneos de dígitos escritos a mano y etiquetas asociadas que describen qué dígito 0–9 está contenido en cada imagen. Este simple problema de clasificación es una de las pruebas más simples y más utilizadas en la investigación de aprendizaje profundo. Sigue siendo popular a pesar de ser bastante fácil de resolver para las técnicas modernas. Geoffrey Hinton lo ha descrito como "la drosofila del aprendizaje automático", lo que significa que permite a los investigadores del aprendizaje automático estudiar sus algoritmos en condiciones de laboratorio controladas, al igual que los biólogos a menudo estudian los huertos de la fruta.

### Aumento del tamaño de los modelos

Otra razón clave por la que las redes neuronales tienen un gran éxito hoy después de haber tenido un éxito relativamente pequeño desde la década de 1980 es que tenemos los recursos computacionales para ejecutar modelos mucho más grandes en la actualidad. Una de las principales ideas del conexionismo es que los animales se vuelven inteligentes cuando muchas de sus neuronas trabajan juntas. Una neurona individual o una pequeña colección de neuronas no es particularmente útil.

Las neuronas biológicas no están especialmente densamente conectadas. Como se ve en la figura 1.10, nuestros modelos de aprendizaje automático han tenido un número de conexiones por neurona dentro de un orden de magnitud de incluso cerebros de mamíferos durante décadas.

En términos del número total de neuronas, las redes neuronales han sido sorprendentemente pequeñas hasta hace muy poco, como se muestra en la figura 1.11. Desde la introducción de unidades ocultas, las redes neuronales artificiales han duplicado su tamaño aproximadamente cada 2.4 años. Este crecimiento es impulsado por computadoras más rápidas con mayor memoria y por la disponibilidad de conjuntos de datos más grandes. Las redes más grandes pueden lograr una mayor precisión en tareas más complejas. Esta tendencia parece que va a continuar durante décadas. A menos que las nuevas tecnologías permitan una escala más rápida, las redes neuronales artificiales no tendrán el mismo número de neuronas que el cerebro humano hasta al menos la década de 2050. Las neuronas biológicas pueden representar funciones más complicadas que las neuronas artificiales actuales, por lo que las redes neuronales biológicas pueden ser incluso más grandes de lo que esta gráfica representa.

En retrospectiva, no es particularmente sorprendente que las redes neuronales con menos neuronas que una sanguijuela no pudieran resolver problemas sofisticados de inteligencia artificial. Incluso las redes actuales, que consideramos bastante grandes desde el punto de vista de los sistemas computacionales, son más pequeñas que el sistema nervioso de animales vertebrados relativamente primitivos, como las ranas.

El aumento en el tamaño del modelo a lo largo del tiempo, debido a la disponibilidad de CPU más rápidas, la llegada de las GPU de propósito general (descritas en la sección 12.1.2), una conectividad de red más rápida y una mejor infraestructura de software para computación distribuida, es una de las tendencias más importantes en La historia del aprendizaje profundo. En general, se espera que esta tendencia continúe en el futuro.

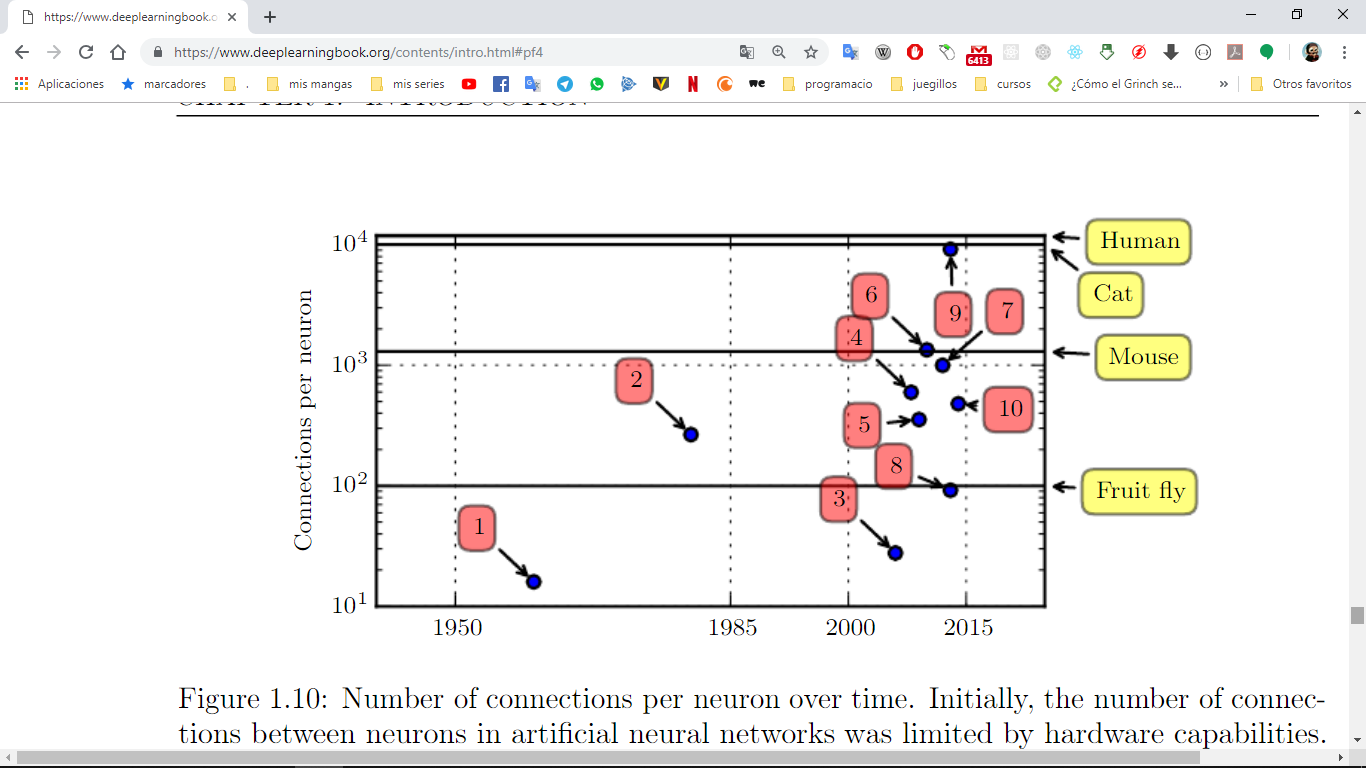


Ilustración 1‑10: Número de conexiones por neurona en el tiempo. Inicialmente, el número de conexiones entre neuronas en redes neuronales artificiales estaba limitado por las capacidades del hardware. Hoy en día, el número de conexiones entre neuronas es principalmente una consideración de diseño. Algunas redes neuronales artificiales tienen casi tantas conexiones por neurona como un gato, y es bastante común que otras redes neuronales tengan tantas conexiones por neurona como mamíferos más pequeños como los ratones. Incluso el cerebro humano no tiene una cantidad exorbitante de conexiones por neurona. Tamaños de redes neuronales biológicas de Wikipedia (2015).

1. Elemento lineal adaptativo (Widrow y Ho ﬀ, 1960)
2. Neocognitron (Fukushima, 1980)
3. Red convolucional acelerada por GPU (Chellapilla et al., 2006)
4. Máquina de Boltzmann profunda (Salakhutdinov y Hinton, 2009a)
5. Red convolucional no supervisada (Jarrett et al., 2009)
6. Perceptrón multicapa acelerado por GPU (Ciresan et al., 2010)
7. Autoencoder distribuido (Le et al., 2012)
8. Red convolucional multi-GPU (Krizhevsky et al., 2012)
9. Red convolucional no supervisada COTS HPC (Coates et al., 2013)
10. GoogLeNet (Szegedy et al., 2014a)

### Mayor precisión, complejidad e impacto en el mundo real

Desde la década de 1980, el aprendizaje profundo ha mejorado constantemente en su capacidad para proporcionar un reconocimiento y una predicción precisa. Además, el aprendizaje profundo se ha aplicado de manera consistente con éxito a conjuntos de aplicaciones más amplios y más amplios.

Los primeros modelos profundos se utilizaron para reconocer objetos individuales en imágenes extremadamente pequeñas y recortadas (Rumelhart et al., 1986a). Desde entonces, ha habido un aumento gradual en el tamaño de las imágenes que las redes neuronales podrían procesar. Las redes modernas de reconocimiento de objetos procesan ricas fotografías de alta resolución y no tienen el requisito de que la foto se recorte cerca del objeto a reconocer (Krizhevsky et al., 2012). De manera similar, las redes más antiguas podrían reconocer solo dos tipos de objetos (o, en algunos casos, la ausencia o presencia de un solo tipo de objeto), mientras que estas redes modernas generalmente reconocen al menos 1,000 categorías diferentes de objetos. El concurso más grande en reconocimiento de objetos es el desafío de reconocimiento visual a gran escala de ImageNet (ILSVRC, por sus siglas en inglés) que se realiza cada año. Un momento dramático en el meteórico ascenso del aprendizaje profundo llegó cuando una red convolucional ganó este desafío por primera vez y por un amplio margen, lo que redujo la tasa de error de los 5 principales del 26,1 por ciento al 15,3 por ciento ( Krizhevsky et al., 2012), lo que significa que la red convolucional produce una lista clasificada de categorías posibles para cada imagen, y la categoría correcta apareció en las primeras cinco entradas de esta lista para todos menos el 15.3 por ciento de los ejemplos de prueba. Desde entonces, estas competiciones se ganan constantemente con redes de convolución profundas y, en el momento de redactar este informe, los avances en el aprendizaje profundo han reducido la tasa de error de los 5 primeros en este concurso a un 3,6 por ciento, como se muestra en la figura 1.12.

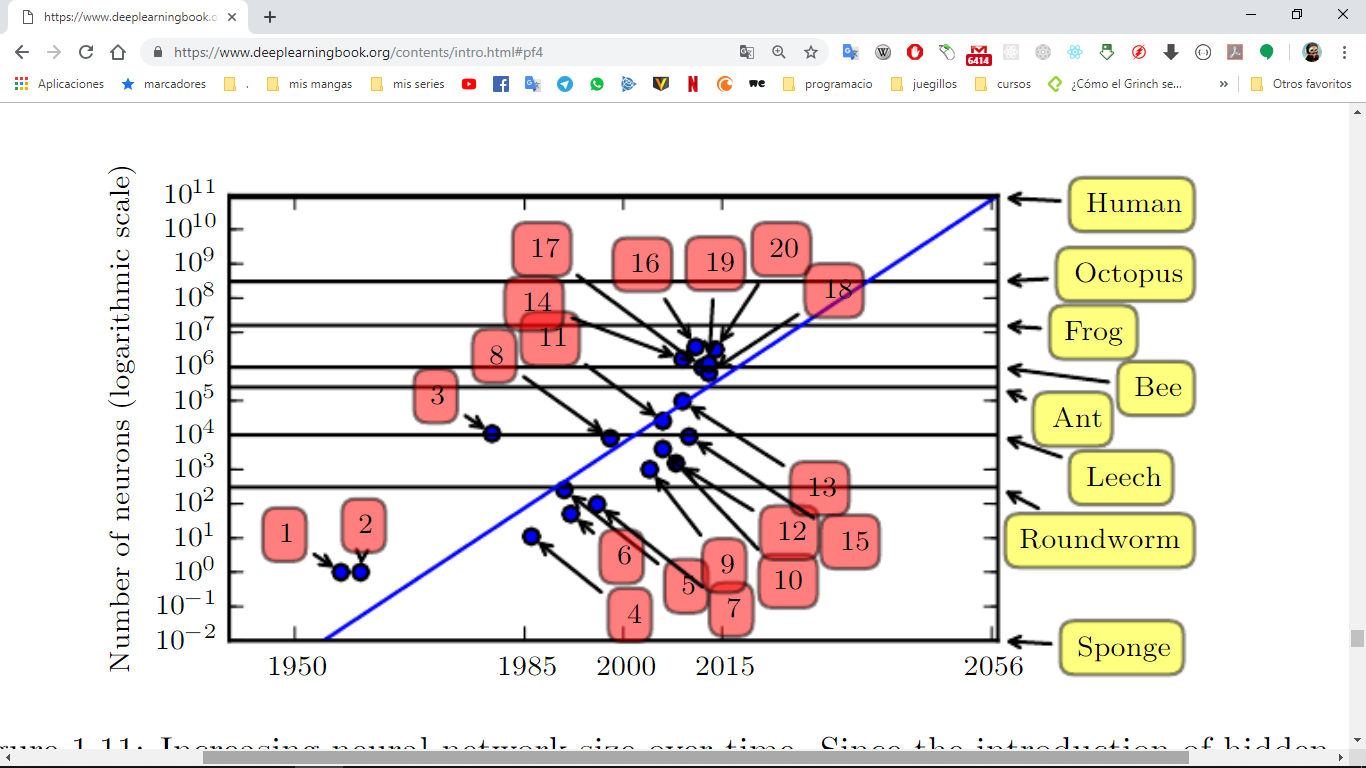


Ilustración 1‑11: Aumento del tamaño de la red neuronal con el tiempo. Desde la introducción de unidades ocultas, las redes neuronales artificiales han duplicado su tamaño aproximadamente cada 2.4 años. Tamaños de redes neuronales biológicas de Wikipedia (2015).

1. Perceptrón (Rosenblatt, 1958, 1962)
2. Elemento lineal adaptativo (Widrow y Hoff, 1960)
3. Neocognitron (Fukushima, 1980)
4. Red temprana de propagación inversa (Rumelhart et al., 1986b)
5. Red neuronal recurrente para el reconocimiento de voz (Robinson y Fallside, 1991)
6. Perceptrón multicapa para el reconocimiento de voz (Bengio et al., 1991)
7. Red de creencias sigmoideas de campo medio (Saul et al., 1996)
8. LeNet-5 (LeCun et al., 1998b)
9. Red estatal de eco (Jaeger y Haas, 2004).
10. Red de creencias profundas (Hinton et al., 2006)
11. Red convolucional acelerada por GPU (Chellapilla et al., 2006)
12. Máquina de Boltzmann profunda (Salakhutdinov y Hinton, 2009a)
13. Red de creencias profundas acelerada por GPU (Raina et al., 2009)
14. Red convolucional no supervisada (Jarrett et al., 2009)
15. Perceptrón multicapa acelerado por GPU (Ciresan et al., 2010)
16. Red OMP-1 (Coates y Ng, 2011)
17. Autoencoder distribuido (Le et al., 2012)
18. Red convolucional multi-GPU (Krizhevsky et al., 2012)
19. Red convolucional no supervisada COTS HPC (Coates et al., 2013)
20. GoogLeNet (Szegedy et al., 2014a)

El aprendizaje profundo también ha tenido un impacto dramático en el reconocimiento de voz. Después de mejorar a lo largo de la década de 1990, las tasas de error para el reconocimiento de voz se estancaron a partir aproximadamente de 2000. La introducción del aprendizaje profundo (Dahl et al., 2010; Deng et al., 2010b; Seide et al., 2011; Hinton et al., 2012a) al reconocimiento de voz resultó en una repentina caída en las tasas de error, con algunas tasas de error reducidas a la mitad. Exploramos esta historia con más detalle en la sección 12.3.

Las redes profundas también han tenido éxitos espectaculares para la detección de peatones y la segmentación de imágenes (Sermanet et al., 2013; Farabet et al., 2013; Couprie et al., 2013) y obtuvieron un rendimiento sobrehumano en la clasificación de señales de tráfico (Ciresan et al., 2012).

Al mismo tiempo que la escala y la precisión de las redes profundas han aumentado, también lo ha hecho la complejidad de las tareas que pueden resolver. Goodfellow et al. (2014d) mostró que las redes neuronales podrían aprender a generar una secuencia completa de caracteres transcritos de una imagen, en lugar de solo identificar un solo objeto. Anteriormente, se creía que este tipo de aprendizaje requería el etiquetado de los elementos individuales de la secuencia (Gülçehre y Bengio, 2013). Las redes neuronales recurrentes, como el modelo de secuencia LSTM mencionado anteriormente, ahora se utilizan para modelar relaciones entre secuencias y otras secuencias en lugar de solo entradas fijas. Este aprendizaje secuencia a secuencia parece estar a punto de revolucionar otra aplicación: la traducción automática (Sutskever et al., 2014; Bahdanau et al., 2015).

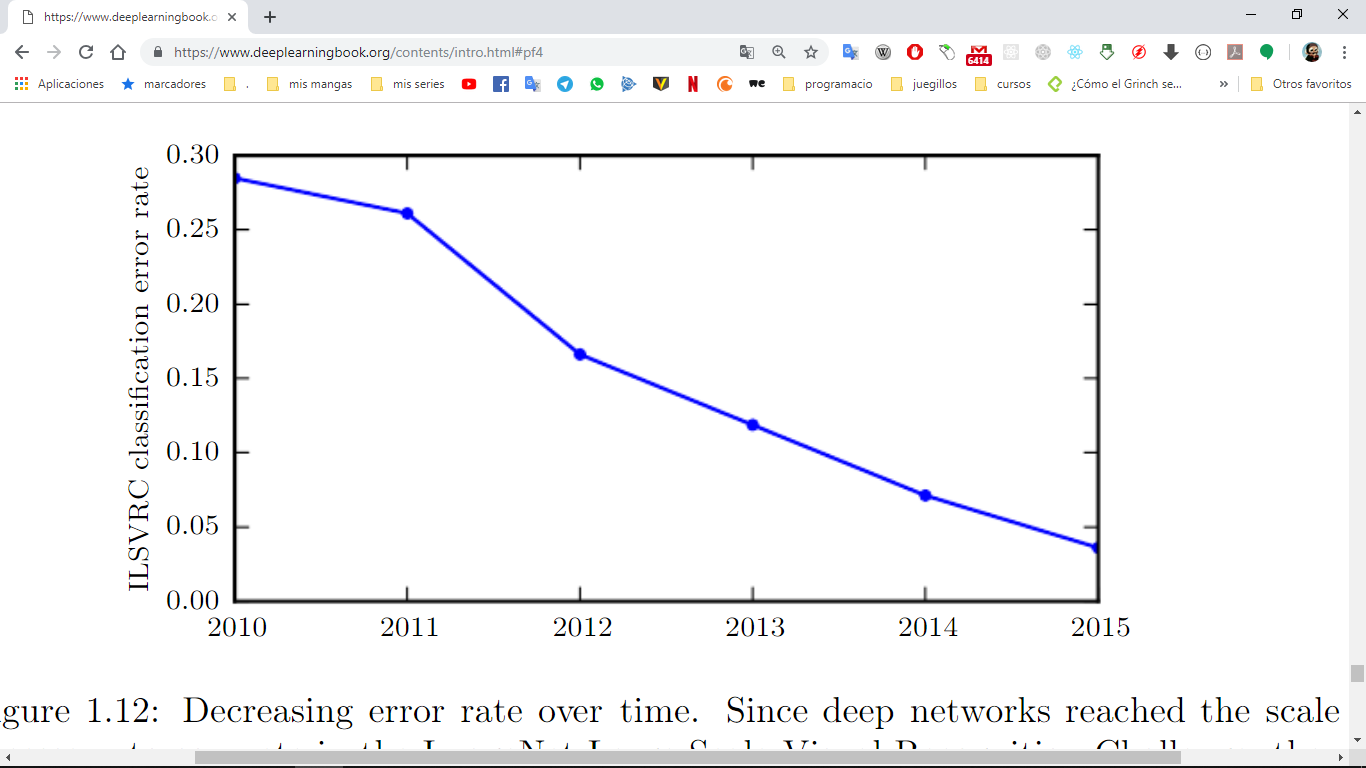


Ilustración 1‑12: Disminuyendo la tasa de error en el tiempo. Desde que las redes profundas alcanzaron la escala necesaria para competir en el Reto de Reconocimiento Visual de Gran Escala de ImageNet, han ganado la competencia constantemente todos los años, lo que genera tasas de error cada vez más bajas. Los datos de Russakovsky et al. (2014b) y He et al. (2015).

Esta tendencia de complejidad creciente se ha llevado a su conclusión lógica con la introducción de las máquinas neuronales de Turing (Graves et al., 2014) que aprenden a leer desde celdas de memoria y escriben contenido arbitrario en las celdas de memoria. Dichas redes neuronales pueden aprender programas simples de ejemplos de comportamiento deseado. Por ejemplo, pueden aprender a ordenar listas de números dados ejemplos de secuencias codificadas y ordenadas. Esta tecnología de autoprogramación está en su infancia, pero en el futuro, en principio, podría aplicarse a casi cualquier tarea.

Otro logro culminante del aprendizaje profundo es su extensión al dominio del **aprendizaje** **por** **refuerzo**. En el contexto del aprendizaje por refuerzo, un agente autónomo debe aprender a realizar una tarea mediante prueba y error, sin la guía del operador humano. DeepMind demostró que un sistema de aprendizaje de refuerzo basado en el aprendizaje profundo es capaz de aprender a jugar videojuegos Atari, alcanzando el desempeño a nivel humano en muchas tareas (Mnih et al., 2015). El aprendizaje profundo también ha mejorado significativamente el rendimiento del aprendizaje de refuerzo para robótica (Finn et al., 2015).

Muchas de estas aplicaciones de aprendizaje profundo son altamente rentables. El aprendizaje profundo ahora lo utilizan muchas de las principales empresas de tecnología, como Google, Microsoft, Facebook, IBM, Baidu, Apple, Adobe, Netflix, NVIDIA y NEC.

Los avances en el aprendizaje profundo también han dependido en gran medida de los avances en la infraestructura de software. Bibliotecas de software como Theano (Bergstra et al., 2010; Bastien et al., 2012), PyLearn2 (Goodfellow et al., 2013c), Torch (Collobert et al., 2011b), DistBelief (Dean et al., 2012) , Caffe (Jia, 2013), MXNet (Chen et al., 2015) y TensorFlow (Abadi et al., 2015) han apoyado importantes proyectos de investigación o productos comerciales.

El aprendizaje profundo también ha hecho contribuciones a otras ciencias. Las modernas redes convolucionales para el reconocimiento de objetos proporcionan un modelo de procesamiento visual que los neurocientíficos pueden estudiar (DiCarlo, 2013). El aprendizaje profundo también proporciona herramientas útiles para procesar grandes cantidades de datos y hacer predicciones útiles en campos científicos. Se ha utilizado con éxito para predecir cómo interactúan las moléculas para ayudar a las compañías farmacéuticas a diseñar nuevos fármacos (Dahl et al., 2014), a buscar partículas subatómicas (Baldi et al., 2014) y a analizar automáticamente las imágenes de microscopio utilizadas. para construir un mapa 3D del cerebro humano (Knowles-Barley et al., 2014). Esperamos que el aprendizaje profundo aparezca en campos cada vez más científicos en el futuro.

En resumen, el aprendizaje profundo es un enfoque del aprendizaje automático que se ha basado en gran medida en nuestro conocimiento del cerebro humano, las estadísticas y las matemáticas aplicadas a medida que se desarrollaban durante las últimas décadas. En los últimos años, el aprendizaje profundo ha experimentado un enorme crecimiento en su popularidad y utilidad, en gran parte como resultado de computadoras más poderosas, conjuntos de datos más grandes y técnicas para entrenar redes más profundas. Los próximos años están llenos de desafíos y oportunidades para mejorar aún más el aprendizaje profundo y llevarlo a nuevas fronteras.

# Parte I: Matemática aplicada y conceptos básicos de aprendizaje automático.

Esta parte del libro presenta los conceptos matemáticos básicos necesarios para comprender el aprendizaje profundo. Comenzamos con ideas generales de matemáticas aplicadas que nos permiten definir funciones de muchas variables, encontrar los puntos más altos y más bajos en estas funciones y cuantificar los grados de creencia.

A continuación, describimos los objetivos fundamentales del aprendizaje automático. Describimos cómo lograr estos objetivos especificando un modelo que represente ciertas creencias, diseñando una función de costo que mida qué tan bien esas creencias se corresponden con la realidad y usando un algoritmo de entrenamiento para minimizar esa función de costo.

Este marco elemental es la base de una amplia variedad de algoritmos de aprendizaje automático, incluidos los enfoques de aprendizaje automático que no son profundos. En las partes posteriores del libro, desarrollamos algoritmos de aprendizaje profundo dentro de este marco.

# Álgebra lineal

El álgebra lineal es una rama de las matemáticas que se usa ampliamente en toda la ciencia y la ingeniería. Sin embargo, dado que el álgebra lineal es una forma de matemática continua en lugar de discreta, muchos científicos informáticos tienen poca experiencia con ella. Una buena comprensión del álgebra lineal es esencial para comprender y trabajar con muchos algoritmos de aprendizaje automático, especialmente los algoritmos de aprendizaje profundo. Por lo tanto, precede nuestra introducción al aprendizaje profundo con una presentación enfocada de los requisitos previos clave de álgebra lineal.

Si ya está familiarizado con el álgebra lineal, no dude en omitir este capítulo. Si tiene experiencia previa con estos conceptos pero necesita una hoja de referencia detallada para revisar las fórmulas clave, le recomendamos The Matrix Cookbook (Petersen and Pedersen, 2006). Si no ha tenido ninguna exposición al álgebra lineal, este capítulo le enseñará lo suficiente como para leer este libro, pero le recomendamos que consulte también otro recurso centrado exclusivamente en la enseñanza del álgebra lineal, como Shilov (1977). Este capítulo omite completamente muchos temas importantes de álgebra lineal que no son esenciales para comprender el aprendizaje profundo.

## Scalars, Vectores, Matrices y Tensores

El estudio del álgebra lineal involucra varios tipos de objetos matemáticos:

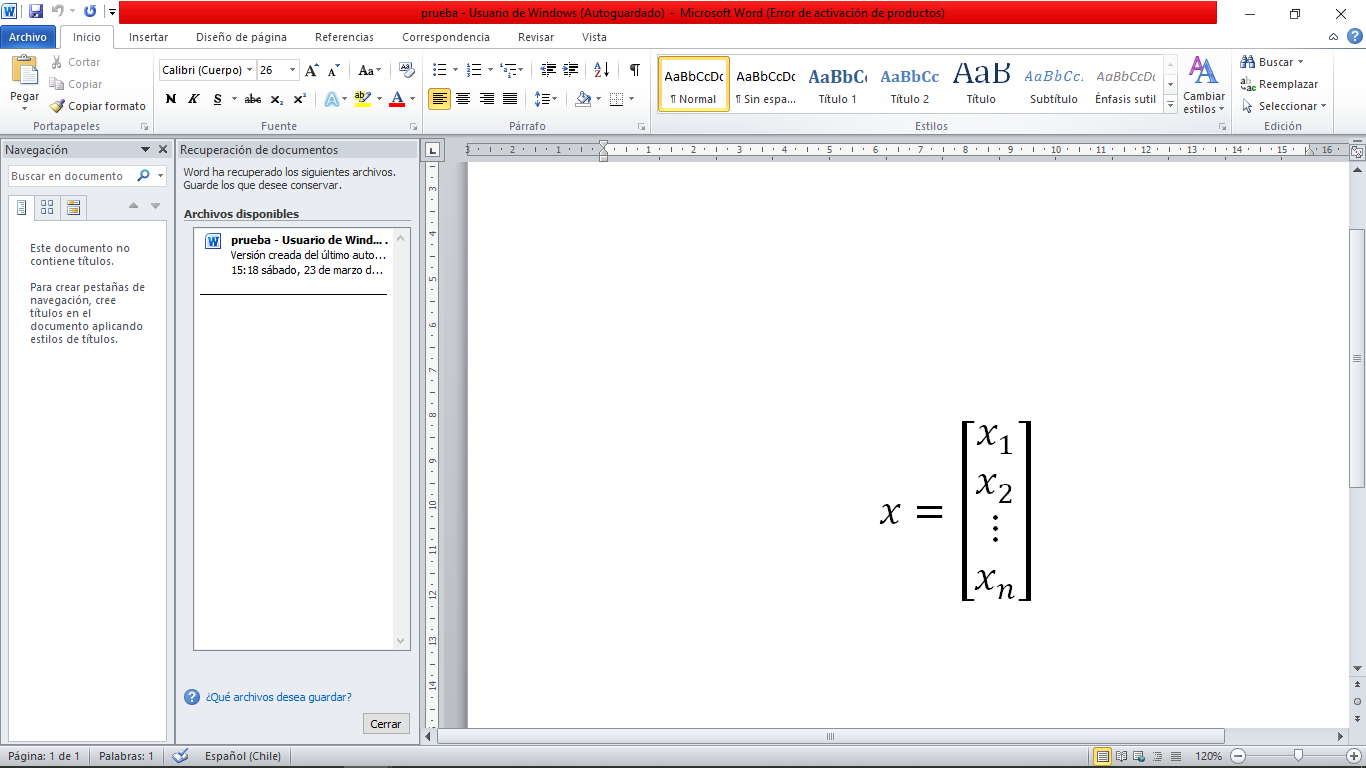
* **Scalars**: Un escalar es solo un número único, en contraste con la mayoría de los otros objetos estudiados en álgebra lineal, que generalmente son matrices de múltiples números. Escribimos los escalares en cursiva. Por lo general, damos nombres de variables en minúsculas a los escalares. Cuando los introducimos, especificamos qué tipo de número son. Por ejemplo, podríamos decir "Sea s ∈ R la pendiente de la línea", mientras definimos un escalar de valor real, o " Sea n ∈ N es el número de unidades", mientras definimos un número natural escalar.
* **Vectores**: Un vector es una matriz de números. Los números se ordenan en orden. Podemos identificar cada número individual por su índice en ese orden. Normalmente, damos a los vectores los nombres en minúscula en negrita, como x. Los elementos del vector se identifican escribiendo su nombre en letra cursiva, con un subíndice. El primer elemento de x es x1, el segundo elemento es x2, y así sucesivamente. También debemos decir qué tipo de números se almacenan en el vector. Si cada elemento está en R, y el vector tiene n elementos, entonces el vector se encuentra en el conjunto formado al tomar el producto cartesiano de R n veces, denotado como Rn. Cuando necesitamos identificar explícitamente los elementos de un vector, los escribimos como una columna encerrada entre corchetes:
* 

Ilustración 2‑3‑1

Podemos pensar en los vectores como puntos de identificación en el espacio, con cada elemento dada la coordenada a lo largo de un eje diferente.

A veces necesitamos indexar un conjunto de elementos de un vector. En este caso, definimos un conjunto que contiene los índices y escribimos el conjunto como un subíndice. Por ejemplo, para acceder a x1, x3 y x6, definimos el conjunto S = {1, 3, 6} y escribimos xS. Usamos el signo − para indexar el complemento de un conjunto. Por ejemplo, x−1 es el vector que contiene todos los elementos de x excepto x1, y x−S es el vector que contiene todos los elementos de x, excepto x1, x3 y x6

**Matrices**: una matriz es una matriz de números en 2-D, por lo que cada elemento está identificado por dos índices en lugar de uno solo. Por lo general, damos a las matrices nombres de variables en mayúsculas con letra en negrita, como *A*. Si una matriz de valores reales *A* tiene una altura de m y un ancho de n, entonces decimos que *A* ∈ Rm×n. Por lo general, identificamos los elementos de una matriz usando su nombre en cursiva pero no en negrita, y los índices se enumeran con comas separadoras. Por ejemplo, *A*1,1 es la entrada superior izquierda de *A* y *A*m,n es la entrada inferior derecha. Podemos identificar todos los números con la coordenada vertical i escribiendo un ":" para la coordenada horizontal. Por ejemplo, *A*i,: denota la sección transversal horizontal de *A* con la coordenada vertical i. Esto se conoce como la i-ésima fila de *A*. Del mismo modo, *A*:, i es la i-ésima columna de *A*. Cuando necesitamos identificar explícitamente los elementos de una matriz, los escribimos como una matriz entre corchetes:

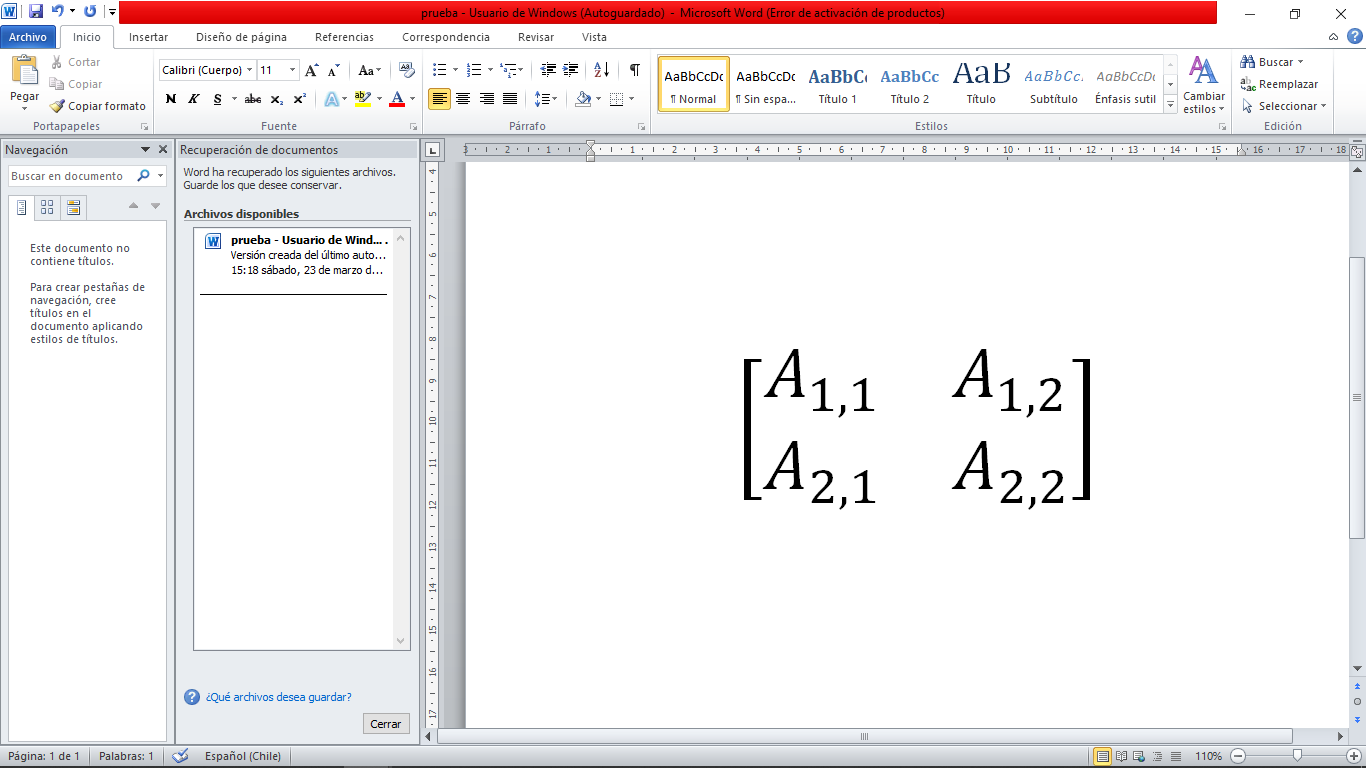


Ilustración 2‑3‑2

A veces es posible que tengamos que indexar expresiones con valores matriciales que no sean una sola letra. En este caso, usamos subíndices después de la expresión pero no convertimos nada a minúsculas. Por ejemplo, f (*A*)i,j da el elemento (i,j) de la matriz calculada aplicando la función f a *A*.

**Tensores**: en algunos casos necesitaremos una matriz con más de dos ejes. En el caso general, una matriz de números organizados en una cuadrícula regular con un número variable de ejes se conoce como tensor. Denotamos un tensor llamado "A" con este tipo de letra: **A**. Identificamos el elemento de **A** en las coordenadas (i, j, k) escribiendo *A*i,j,k.

Una operación importante sobre matrices es la transposición. La transposición de una matriz es la imagen reflejada de la matriz a través de una línea diagonal, llamada la diagonal principal, descendiendo hacia la derecha, comenzando desde su esquina superior izquierda. Vea la figura 2.1 para una representación gráfica de esta operación. Denotamos la transposición de una matriz ***A*** como ***A***ᴛ, y se define de tal manera que

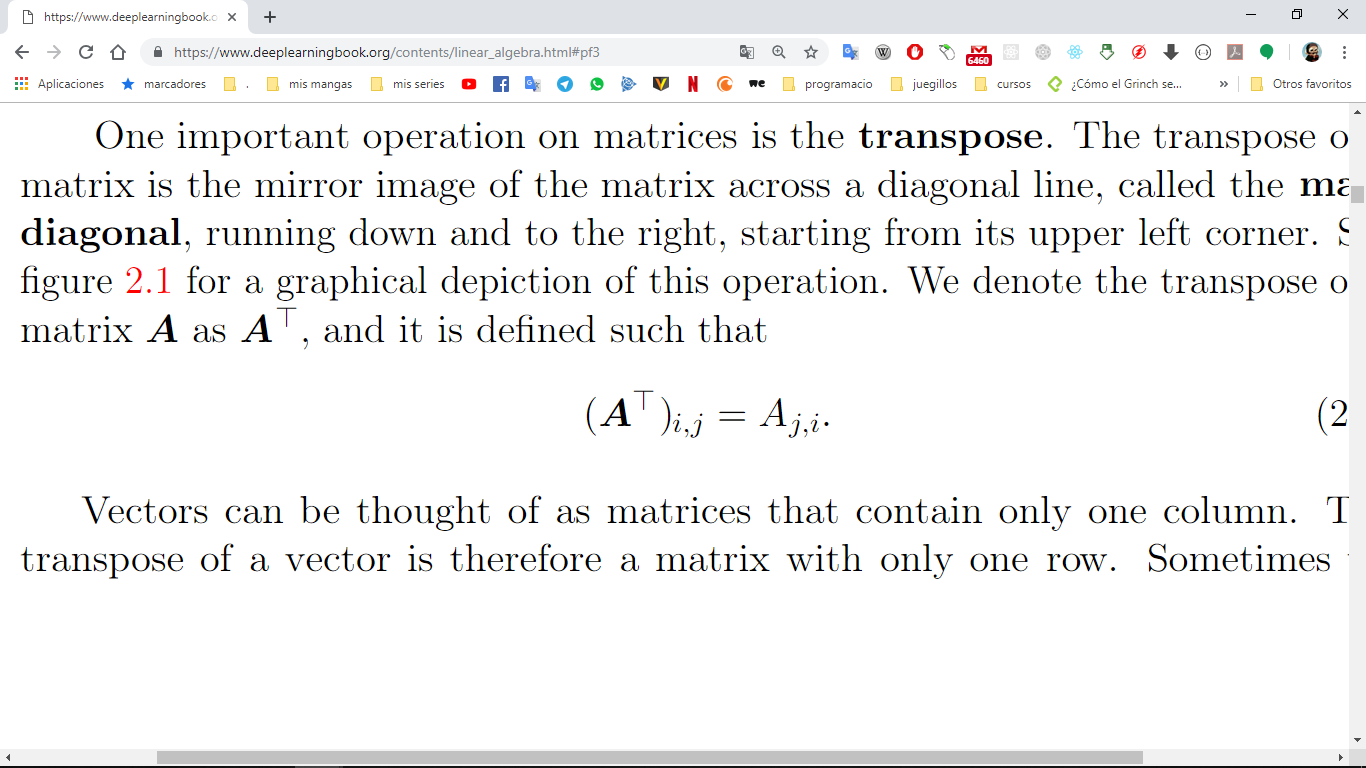


Ilustración 2‑3

## Multiplicando matrices y vectores

## Identidad y matrices inversas.

## Linear Dependence and Span

## Normas

## Tipos especiales de matrices y vectores

## Descomposición espectral

## Valor singular de descomposición

## La Pseudoinversa de Moore-Penrose

## El operador traza

## El determinante

## Ejemplo: Análisis de componentes principales

# Probabilidad y teoría de la información

## ¿Por qué probabilidad?

## Random Variables

## Distribuciones de probabilidad

## Probabilidad marginal

## La probabilidad condicional

## La regla de la cadena de probabilidades condicionales

## Independence and Conditional Independence

## Expectativa, varianza y covarianza

## Distribuciones de probabilidad comunes

## Propiedades útiles de funciones comunes

## Regla de Bayes

## Detalles técnicos de las variables continuas

## Teoría de la información

## Modelos Probabilísticos Estructurados

# Computación Numérica

## Sobreflujo y subflujo

## Mal acondicionamiento

## Optimización basada en gradiente

## Optimización con restricciones

## Ejemplo: cuadrados mínimos lineales

# Conceptos básicos del aprendizaje automático

## Algoritmos de aprendizaje

## Capacidad, sobreajuste y subequipo

## Hiperparámetros y Conjuntos de Validación

## Estimadores, sesgo y varianza

## Estimación de máxima verosimilitud

## Estadística Bayesiana

## Algoritmos de aprendizaje supervisado

## Algoritmos de aprendizaje no supervisados

## Descenso de gradiente estocástico

## Construyendo un algoritmo de aprendizaje automático

## Desafíos que motivan el aprendizaje profundo

# Redes profundas: Prácticas modernas

# Redes de avance profundo

## Ejemplo: Aprendiendo XOR

## Gradient-Based Learning

## Hidden Unidos

## Diseño de arquitectura

## Propagación hacia atrás y otros algoritmos de diferenciación.

## Notas históricas

# Regularización para el Aprendizaje Profundo

## Parámetros Norma de Sanciones

## Las sanciones de la norma como optimización restringida

## Regularización y problemas sub-limitados

## Aumento de Conjunto de datos

## Robustez del ruido

## Aprendizaje semi-supervisado

## Aprendizaje multitarea

## Detención temprana

## Vinculación de parámetros y uso compartido de parámetros

## Representaciones dispersas

## Empaquetamiento y otros métodos de conjunto

## Abandonar

## Entrenamiento adversario

## Distancia tangente, prop tangente y clasificador de tangente múltiple (Clasificador de colector de tangente)

# Optimización para entrenamiento de modelos profundos

## Cómo aprende el aprendizaje de Optimización pura

## Desafíos en la optimización de redes neuronales

## Algoritmos básicos

## Parámetros de inicialización de estrategias

## Algoritmos con tasas de aprendizaje adaptativas

## Métodos aproximados de segundo orden

## Optimización de estrategias y meta-algoritmos

# Redes convolucionales

## La Operación de Convolución

## Motivación

## Agrupación

## Convolución y Agrupación como un Inﬁnitely Strong Prior

## Variantes de la función de convolución básica

## Salidas Estructuradas

## Tipos de datos

## Algoritmos de convolución eficientes

## Características aleatorias o no supervisadas

## Las bases neurocientíficas para redes convolucionales

## Redes convolucionales y la historia del aprendizaje profundo

# Modelado de secuencias: redes recurrentes y recursivas

## Despliegue de grafos computacionales

## Redes neuronales recurrentes

## NRNs(RNNs) bidireccionales

## Arquitecturas de secuencia a secuencia de codificador-decodificador

## Redes Profundas Recurrentes

## Redes neuronales recursivas

## El desafío de las dependencias a largo plazo

## Redes de estado de eco

## Unidades con fugas y otras estrategias para escalas de tiempo múltiple

## La larga memoria a corto plazo (LSTM) y otras NRNs (RNNs) cerradas

## Optimización para Dependencias a Largo Plazo

## Memoria explícita

# Metodología práctica

## Métricas de rendimiento

## Modelos de línea base predeterminados

## Determinar si recopilar más datos

## Seleccionando los hiperparámetros

## Estrategias de depuración

## Ejemplo: Reconocimiento de números de varios dígitos

# Aplicaciones

## Aprendizaje profundo a gran escala

## Visión por ordenador

## Reconocimiento de voz

## Procesamiento natural del lenguaje

## Otras aplicaciones

# Investigación de Aprendizaje Profundo

# Modelos de factor lineal

## PCA probabilística y análisis factorial

## Análisis de componentes independientes (ICA)

## Análisis de características lentas

## Codificación dispersa

## Interpretación múltiple de PCA

# Autoencoders

## Auto-codificadores incompletos

## Autoencoders Regularizados

## Poder de representación, tamaño de capa y profundidad

## Codificadores y decodificadores estocásticos

## Eliminación de ruido (Denoising) Autoencoders

## Manifolds de aprendizaje con Autoencoders

## Autoencoders contractivos

## Descomposición dispersa predictiva

## Aplicaciones de Autoencoders

# Representación de aprendizaje

## Pretratamiento codicioso de capa sabia no supervisado

## Transferencia de aprendizaje y adaptación de dominio

## Desenredo semi-supervisado de factores causales

## Representación distribuida

## Ganancias exponenciales de la profundidad

## Proporcionando pistas para descubrir causas subyacentes

# Modelos probabilísticos estructurados para el aprendizaje profundo

## El desafío del modelado no estructurado

## Uso de grafos para describir la estructura del modelo

## Muestreo a partir de modelos grafos

## Ventajas del modelado estructurado

## Aprender acerca de las dependencias

## Inferencia e Inferencia aproximada

## El Enfoque de Aprendizaje Profundo de Modelos Probabilísticos Estructurados

# Métodos Monte Carlo

## Muestreo y métodos de Monte Carlo

## Muestreo de importancia

## Cadena de Markov métodos de Monte Carlo

## Muestreo de Gibbs

## El desafío de mezclar entre separados

## Modos

# Confrontando la función de partición

## El gradiente de Función de verosimilitud

## Probabilidad máxima estocástica y divergencia contrastiva

## Pseudo probabilidad

## Coincidencia de puntuación y Coincidencia de relación

## Eliminación de ruido de Igualar puntaje

## Estimación de contraste de ruido

## Estimando la función de partición

# Inferencia aproximada

## Inferencia como optimización

## Maximización de la expectativa

## MAP Inferencia y codificación dispersa

## Inferencia y aprendizaje variacional

## Inferencia aproximada aprendida

# Modelos generativos profundos

## Máquinas de Boltzmann

## Máquinas de Boltzmann restringidas

## Redes de creencias profundas

## Máquinas de boltzmann profundas

## Máquinas Boltzmann para datos de valor real

## Máquinas de Boltzmann convolucionales

## Máquinas Boltzmann para salidas estructuradas o secuenciales

## Otras máquinas de Boltzmann

## Propagación hacia atrás a través de operaciones aleatorias

## Redes Generativas Dirigidas

## Dibujo de muestras de Autoencoders. . . . . . . . . . . . . . . . 707

## Redes estocásticas generativas. . . . . . . . . . . . . . . . . . . 710

## Otros esquemas de generación. . . . . . . . . . . . . . . . . . . . . . 712

## Evaluando modelos generativos. . . . . . . . . . . . . . . . . . . . 713

## Conclusión . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 716

# Bibliografía

# Indice 773