UNIVERSIDAD DE SANTIAGO DE CHILE

FACULTAD DE INGENIERÍA

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA

Desempeño de redes neuronales entrenadas mediante simulated annealing frente a otros métodos

Informe Final

Propuesta de tesis

Nombre: Felipe Alberto Reyes González

Programa: Magíster en Ingeniería Informática

Profesor patrocinante: Victor Parada

Cel.: 890 26 317

email: felipe.reyesg@usach.cl

12 de mayo de 2017

Tabla de contenido

Bibliografía **17**

Descripción del problema

Las redes neuronales

El elemento básico de las NN es el nodo, que recibe un vector de entrada para producir una salida. Cada entrada tiene asociado un vector de pesos w, que se va modificando durante el proceso de aprendizaje. Cada unidad aplica una función f sobre la suma de las entradas ponderada por el vector de pesos como en la ecuacion 1.

$$y_i = \sum_j w_{ij} y_j \tag{1}$$

Donde el resultado puede servir como entrada de otras unidades.

Existen dos fases importante dentro del modelo

- Fase de entrenamiento: Se usa un conjunto de datos o patrones de entrenamiento para determinar los pesos que definen el modelo de la NN. Se calculan de manera iterativa, de acuerdo con los valores de entrenamiento, con el objeto de minimizar el error cometido entre la salida obtenida por la NN y la salida deseada.

- Fase de prueba: Durante el entrenamiento, el modelo se ajusta al conjunto de entrenamiento, perdiendo la habilidad de generalizar su aprendizaje a casos nuevos, a esta situación se le llama sobreajuste. Para evitar el sobreajuste, se utiliza un segundo grupo de datos diferentes, el conjunto de validación, que permitirá controlar el proceso de aprendizaje.

Los pesos óptimos se obtienen minimizando una función. Uno de los criterios utilizados es la minimización del error cuadrático medio entre el valor de salida y el valor real esperado.

El perceptrón multicapa

El perceptrón multicapa es una generalización del perceptrón simple, y surgió como consecuencia de las limitaciones de dicha arquitectura para resolver problemas de clasificación no lineal. Marvin Minsky (1987) mostraron que el uso de varios perceptrones simples podía resultar una solución para resolver problemas no lineales. Sin embargo, dicha propuesta no presentaba una solución al problema de la actualización de los pesos de las capas, pues la regla de aprendizaje del perceptrón simple no era aplicable. Rummelhart, Hinton y Wilians presentaron la *regla delta generalizada*, una forma de propagar el error de la red desde la capa de salida hacia las capas anteriores.

Dentro de las redes neuronales, el perceptrón multicapa es una de las arquitecturas más usadas para resolver problemas. Esto es debido a que poseen la capacidad de ser un aproximador universal. Esto no implica que sea una de las redes más potentes o con mejores resultados, el perceptrón multicapa posee una serie de limitaciones, como el proceso de aprendizaje para problemas que dependan de un gran número de variables, la dificultad para realizar un análisis teórico de la red debido a la presencia de componentes no lineales y a la alta conectividad.

Arquitectura

El perceptron multicapa posee una estructura de capas compuestas por neuronas. Cada una de las capas está formada por un conjunto de neuronas y se distinguen tres tipos de capas: la capa de entrada, las capas ocultas y la capa de salida.

Las neurones de la capa de entrada se encargan de recibir los patrónes y propagar dichas señales a las neuronas de la capa siguiente. La última capa, la capa de salida, proporciona la respuesta de la red al patrón presentado. Las neuronas de las capas ocultas realizan el procesado de las señales generadas por el patrón de entrada.

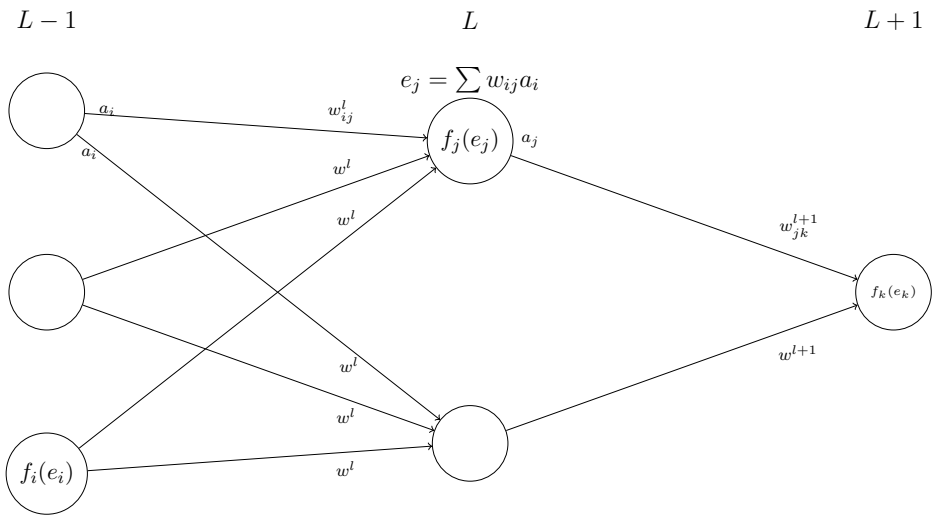

Figura 1: Neurona

En la figura 1 se observa que las conexiones van siempre hacia adelante, es decir, las neuronas de la capa l se conectan con las neuronas de la capa $l+1$. Las neuronas de una capa están conectadas a todas las neuronas de la capa siguiente.

Propagación de la entrada

El perceptrón multicapa define una relación entre la entrada y la salida. Esta relación se obtiene propagando hacia adelante los valores de las variables de entrada, es por esto que también se les llama redes *feedforward*. Cada neurona de la red procesa la entrada recibida y produce una respuesta que se propaga, mediante las conexiones, hacia las neuronas de la capa siguiente.

Si un perceptrón multicapa con C capas y n_c neuronas en la capa c, donde $W_c = (w_{ij}^c)$ es la matriz de pesos, donde w_{ij}^c representa el peso de la conexion de la neurona i de la capa c. Denotaremos a_i^c a la activación de la neurona i de la capa c que se calcula de la siguiente manera

- Activación de una neurona de la capa de entrada: Las neuronas se encargan de transmitir la entrada recibida, por lo tanto

$$a_i^1 = x_i, i = 1, 2, \cdots, n$$

donde $X = (x_1, x_2, \cdots, x_n)$ representa el vector de entrada.

- Activación de una neurona de la capa oculta: Las neuronas de una capa oculta procesa la información recibida aplicando la función de activación f a la suma de los productos de la entrada por sus pesos, es decir

$$a_i^c = f\left(\sum_{j=1}^{n_{c-1}} w_{ji}^{c-1} a_j^{c-1} + \theta_i^c\right), i = 1, 2, \cdots, n_c; c = 2, 3, \cdots, C-1$$

donde a_j^{c-1} es la salida de la capa anterior a c.

- Activación de una neurona de la capa de salida: La activación de una neurona de la capa de salida viene dada por la función de activación f aplicada a la suma de los productos de la entrada por sus pesos, es decir

$$y_i = a_i^c = f\left(\sum_{j=1}^{n_{c-1}} w_{ji}^{C-1} a_j^{C-1} + \theta_i^C\right), i = 1, \cdots, n_c$$

donde $Y = (y_1, y_2, \cdots, y_{n_c})$ es el vector de salida.

La función f es la función de activación de la neurona. Las funciones de activación mas utilizadas son la sigmoidal y la tangente hiperbólica, descritas en las escuaciones 2 y 3 respectivamente.

$$f_{sigm}(x) = \frac{1}{1 + \exp(-x)} \tag{2}$$

$$f_{tanh}(x) = \frac{1 - \exp(-x)}{1 + \exp(-x)} \tag{3}$$

Ambas funciones poseen como imagen intervalo de valores entre $[0, 1]$ y $[-1, 1]$ como se observa en la figura 2.

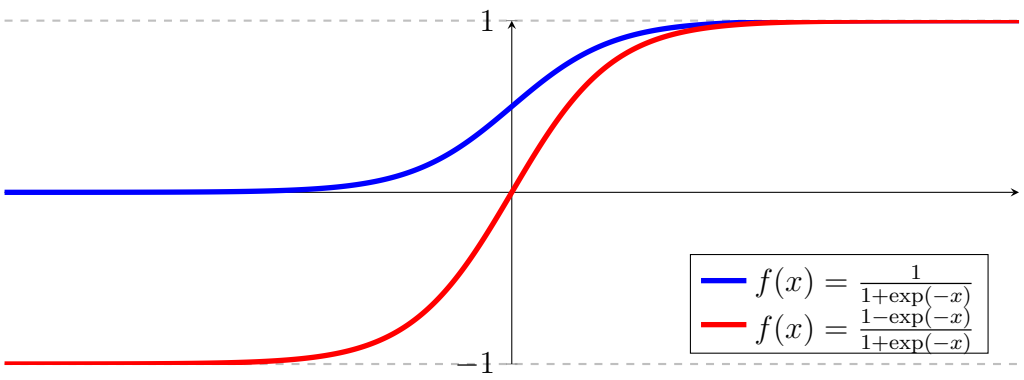

Figura 2: Funciones de activación mas utilizadas.

Algoritmo de retropropagación

Una regla de aprendizaje es el método que le permite adaptar los parámetros de la red. El perceptrón multicapa actualiza sus pesos en función de la salida obtenida de tal manera que los nuevos pesos permitan reducir el error de salida. Por tanto, para cada patrón de entrada a la red es necesario disponer de un patrón de salida deseada.

El objetivo es que la salida de la red sea lo más próxima posible a la salida deseada, debido a esto la es que el aprendizaje de la red se describe como un problema de minimización de la siguiente maner

$$\min_{W} E$$

donde W es el conjunto de parámetros de la red (pesos y umbrales) y E es una función de error que evalúa la diferencia entre las salidas de la red y las salidas deseadas. en la mayor parte de los casos, la función de error se define como:

$$E = \frac{1}{N} \sum_{i=1}^{N} e(i) \tag{4}$$

Donde N es el número de muestras y $e(n)$ es el error cometido por la red

para el patrón i, definido de la siguiente manera

$$e(i) = \frac{1}{n_C} \sum_{j=1}^{n_C} (s_j(i) - y^j(n))^2 \tag{5}$$

Siendo $Y(i) = (y_1(i), y_2(i), \cdots, y_{n_C}(i))$ y $S(i) = (s_1(i), s_2(i), \cdots, s_{n_C}(i))$ los vectores de salida y salidas deseadas para el patrón i respectivamente.

De esta manera, si W^* es un mínimo de la función de error E, en dicho punto el error será cercano a cero, y en consecuencia, la salida de la red será próxima a la salida deseada.

Así es como el aprendizaje es equivalente a encontrar un mínimo de la función de error. La presencia de funciones de activación no lineales hace que la respuesta de la red sea no lineal respecto a los parámetros ajustables, por lo que el problema de aminimización es un problema no lineal y se hace necesario el uso de técnicas de optización no lineales para su resolución.

Las técnicas utilizadas suelen basarse en la actualización de los parámetros de la red mediante la determinación de una dirección de búsqueda. En el caso de las redes neuronales multicapa, la dirección de búsqueda más utilizada se basa en la dirección contraria del gradiente de la función de error E, el método de gradiente descendente.

Si bien el aprendizaje de la red busca minimizar el error total de la red, el procedimiento está basado en métodos del agradiente estocástico, que son una sucesión de minimizaciones del error en función de cada patrón $e(i)$, en lugar de minimizar el error total E de la red. Aplicando el método del gradiente estocástico, cada parámetro w se modifica para cada patrón de entrada n según la siguiente regla de aprendizaje

$$w(i) = w(n-1) - \alpha \frac{\partial e(i)}{\partial w} \tag{6}$$

donde $e(i)$ es el error para el patrón de entrada i dado por la ecuación 5, y

α es la tasa de aprendizaje, éste último determina el desplazamiento en la superficie del error.

Como las neuronas están ordenadas por capas y en distintos niveles, es posible aplicar el método del gradiente de forma eficiente, resultando en el *algoritmo de retropropagación* (Rumelhart, Hinton, y Williams, 1986) o *regla delta generalizada*. El término retropropagación es utilizado debido a la forma de implementar el método del gradiente en las redes multicapa, pues el error cometido en la salida de la red es propagado hacia atrás, transformándolo en un error para cada una de las neuronas ocultas de la red.

El algoritmo de retropropagación es el método de entrenamiento más utilizado en redes con conexión hacia adelante. Es un método de aprendizaje supervisado de gradiente descendente, en el que se distinguen claramente dos fases:

1. Se aplica un patrón de entrada, el cual se propaga por las distintas capas que componen la red hasta producir la salida de la misma. Esta salida se compara con la salida deseada y se calcula el error cometido por cada neurona de salida.

2. Estos errores se transmiten desde la capa de salida, hacia todas neuronas de las capas anteriores (Fritsch, 1996). Cada neurona recibe un error que es proporcional a su contribución sobre el error total de la red. Basándose en el error recibido, se ajustan los errores de los pesos sinápticos de cada neurona.

El gradiente descendente

El desvanecimiento del gradiente

El problema del gradiente desvaneciente nace en las NN profundas, éstas utilizan funciones cuyo gradiente tienden a estar entre 0 y 1. Debido a que estos gradientes pequeños se multiplican durante la retropropagación, tienden a *desvanecerse* a través de las capas, evitando que la red aprenda en redes muy profundas.

Si se tiene una NN, la activación de una neurona de una capa intermedia

i con función de activación f_i y con entrada

$$net_i(t) = \sum_j w_{ij} y^j(t-1)$$

es $y^i(t) = f_i(net_i(t))$. Además w_{ij} es el peso de la conexión desde la unidad j hasta la unidad i, $d_k(t)$ será la respuesta esperada de la unidad k de la capa de salida en el tiempo t. Usando el error cuadrático medio (*Mean square error*, MSE), el error de k será

$$E_k(t) = (d_k(t) - y^k(t))^2$$

En un tiempo $\tau \leq t$ cualquiera, el error de una neurona j que no sea una neurona de entrada es la suma de los errores externos y el error propagado hacia atrás desde la neurona previa será

$$\vartheta_j(\tau) = f_j'(net_j(\tau)) \left(E_j(\tau) + \sum_i w_{ij} \vartheta_i(\tau+1) \right)$$

El peso actualizado en el tiempo τ resulta

$$w_{jl}^{new} = w_{jl}^{old} + \alpha \vartheta_j(\tau) y^l(\tau - 1)$$

donde α es la tasa de aprendizaje, y l es una unidad arbitraria conectada a la unidad j.

En un tiempo arbitrario $\tau \leq t$, el error de la señal en una unidad j, que no sea de entrada, será la suma de los errores externos y la señal propagada anterioremente como muestra la ecuación 7.

$$\vartheta(\tau) \quad = \quad f_j'(net_j(\tau)) \left(E_j(\tau) + \sum_i w_{ij} \vartheta_i(\tau+1) \right) \tag{7}$$

Entonces, los pesos actualizados serán

$$w_{jl}^{new} = w_{jl}^{old} + \alpha \vartheta_j(\tau) y^l(\tau - 1)$$

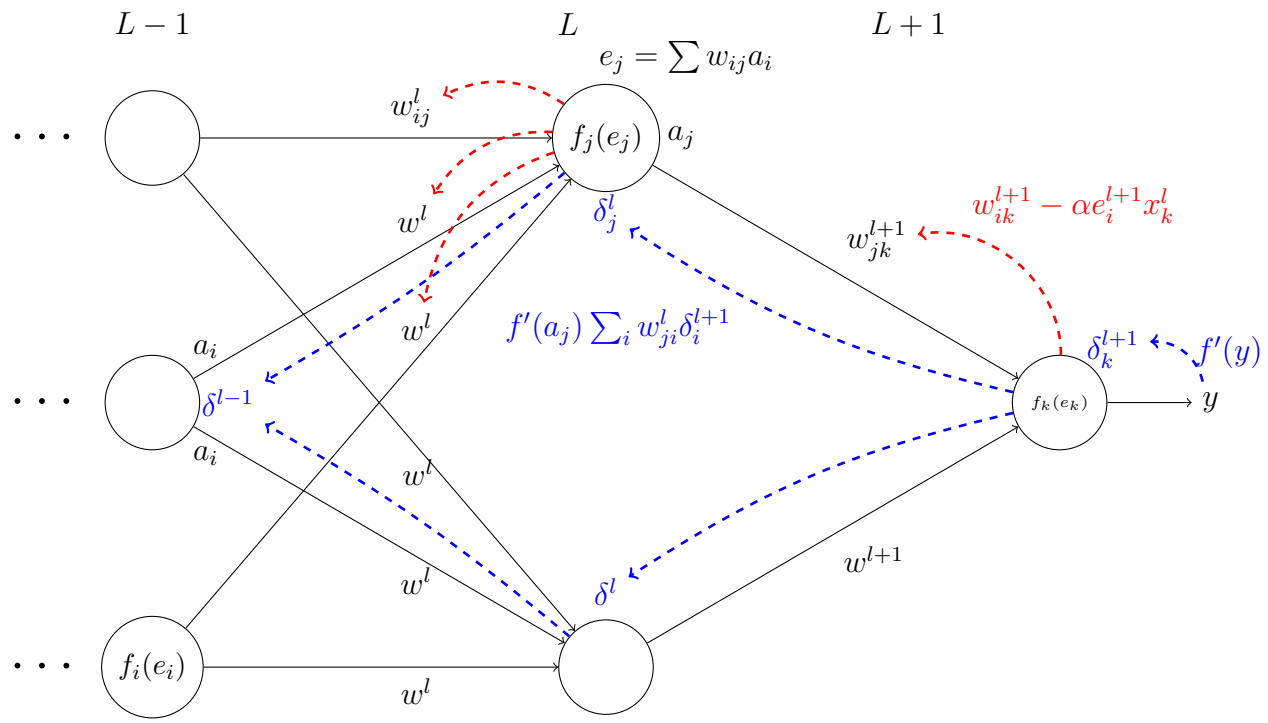

Figura 3: Gradiente descendente

Objetivos del proyecto

En esta sección se presenta el objetivo general y los objetivos específicos de la presente propuesta.

Objetivo general

Evaluar los modelos lineales y no lineales de aprendizaje de autorregulación del flujo sanguíneo cerebral basado en diferentes bandas de frecuencias para la comprensión del comportamiento de los modelos.

Objetivos específicos

Los objetivos establecidos para el presente trabajo son descritos a continuación

1. Definir los modelos lineales y no lineales a implementar.
2. Realizar el preprocesamiento de la señal de entrada.
3. Entrenar los modelos lineales y no lineales con señales en distintas bandas de frecuencias.
4. Entrenar los modelos lineales y no lineales con señales sintéticas.
5. Establecer las conclusiones del trabajo.

Descripción de la solución

Estado del arte

Características de la solución

La solución propone un análisis teórico mediante la aplicación de minería de datos, donde se estudiarán las capacidades de aprendizaje de los modelos lineales y no lineales basado en los datos que describen la autorregulación cerebral para un problema aplicado a la investigación en las ramas de la biología y medicina. Se utilizarán modelos lineales y no lineales extraídos de la literatura para determinar los modelos que se evaluarán.

Propósitos de la solución

El propósito del presente trabajo es comparar los resultados del aprendizaje de los métodos dinámicos de la autorregulación cerebral en los seres humanos, determinando las características del proceso en función de las bandas de frecuencias y ruido.

Alcances o limitaciones de la solución

Los alcances y limitaciones descritos para el trabajo son los siguientes

- El estudio se plantea desde una perspectiva teórica y no precisa de la experimentación con pacientes.
- Los datos que se utilizarán son los obtenidos por (?, ?) en su publicación, datos que fueron extraídos con el consentimiento de los pacientes y aceptados por el comité respectivo.
- Se estudiarán los modelos que describen la denominada autorregulación dinámica.
- El ruido presente en las señales será generado de forma sintética.

Metodología, herramientas y ambiente de desarrollo

A continuación se especifica la metodología, herramientas y ambientes de desarrollo que se utilizarán para llebar a cabo el proyecto.

Metodología a usar

Considerando el aspecto investigativo del trabajo, se considera la utilización del método científico. Entre las actividades que componen la metodología, (?, ?) describe los siguientes pasos para desarrollar una investigación:

- Formulación de la hipótesis: Los modelos lineales y no lineales que aprenden el comportamiento de la autorregulación del flujo sanguíneo cerebral se comportan distinto cuando se utilizan diferentes bandas de frecuencias.
- Marco teórico: Una revisión de la literatura donde se aborda el problema planteado, para situarse en el contexto actual de los problemas. Se describirán los modelos que permiten establecer relaciones entre las componentes de la autorregulación cerebral dinámica.
- Diseño de la solución: Se deberá diseñar el experimento para generar los modelos y preparar las señales para su evaluación. Diseñar y ejecutar el experimento provocando ruído en las señales utilizadas.

- Análisis y verificación de los resultados: Los resultados se analizarán considerando métodos estadísticos con un valor $p < 0{,}05$ que será considerado como diferencia significativa.
- Presentación de los resultados: Se presentarán tablas que describan los resultados obtenidos y que se consideren pertinentes.
- Conclusiones obtenidas en el desarrollo de la investigación.

Herramientas de desarrollo

Para el desarrollo y ejecución de los experimentos se utilizará un equipo con las siguientes características

Sistema Operativo	Linux Mint 17.2 Cinnamon 64-bit
Procesador	Intel Core i5-2450M CPU @ 2.50GHz x2
RAM	7.7Gb
Almacenamiento	957.2Gb

El software que se utilizará es:

- Software: Entorno para computación y gráficos estadísticos R.
- Herramienta ofimática: LaTeX.

Ambiente de desarrollo

El desarrollo de la investigación se realizará en el Departamento de Ingeniería Informática de la Universidad de Santiago de Chile, el cual cuenta con una biblioteca y un laboratorio de computación con acceso a internet, para la recopilación de información y para el desarrollo del experimento. Además, se utilizará el hogar del autor, donde se ubica el equipo de desarrollo a utilizar.

Plan de trabajo

La planificación del trabajo de investigación, se indica en una carta Gantt que se puede apreciar en la Tabla 1, donde se fija como fecha de inicio el día 12 de marzo de 2016 con fecha de término el día 30 de Junio de 2016. En un regimen de trabajo que considera periodos de 5 horas diarias de trabajo promedio, los 7 días de la semana, lo que equivale a un total de 640 horas de trabajo.

Tabla 1: Carta Gantt

NOMBRE DE LA TAREA		FECHA DE INICIO	FECHA DE TÉRMINO	DURACIÓN (DÍAS)
PROYECTO DE TESIS		07-03-16	24-06-16	
INICIO		07-03-16	08-03-16	2
	Reunión con el profesor	07-03-16	07-03-16	1
	Inscripción del tema	08-03-16	08-03-16	1
ESTADO DEL ARTE		09-03-16	29-03-16	21
	Revisión bibliográfica	09-03-16	18-03-16	10
	Reunión con el profesor	19-03-16	19-03-16	1
	Redacción del marco teórico	20-03-16	29-03-16	10
DESARROLLO DE LA SOLUCIÓN		30-03-16	24-04-16	26
	Análisis	30-03-16	02-04-16	4
	Diseño	03-04-16	06-04-16	4
	Implementación	07-04-16	16-04-16	10
	Pruebas	17-04-16	19-04-16	3
	Correcciones	20-04-16	22-04-16	3
	Reunión con el profesor	23-04-16	23-04-16	1
	Redacción del análisis, diseño e implementación	24-04-16	24-04-16	1
EXPERIMENTOS		25-04-16	25-05-16	31
	Calibración de parámetros	25-04-16	29-04-16	5
	Experimentación	30-04-16	14-05-16	15
	Reunión con el profesor	15-05-16	15-05-16	1
	Redacción de la experimentación	16-05-16	25-05-16	10
ANÁLISIS DE RESULTADOS		26-05-16	08-06-16	14
	Análisis detallado	26-05-16	29-05-16	4
	Interpretación de los resultados	30-05-16	02-06-16	4
	Reunión con el profesor	03-06-16	03-06-16	1
	Redación de los resultados	04-06-16	08-06-16	5
DOCUMENTO FINAL		09-06-16	24-06-16	16
	Redacción de la conclusión	09-06-16	10-06-16	2
	Revisión del documento	11-06-16	17-06-16	7
	Presentación al profesor	18-06-16	18-06-16	1
CORRECCIONES		19-06-16	22-06-16	4
	Versión final	23-06-16	23-06-16	1
	Entrega del documento	24-06-16	24-06-16	1

Bibliografía

Fritsch, J. (1996). *Modular neural networks for speech recognition* (Masters Thesis). KIT.

Marvin Minsky, S. A. P. (1987). *Perceptrons: An introduction to computational geometry* (Expanded ed.). The MIT Press.

Rumelhart, D. E., Hinton, G. E., y Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*, 533–536.