UNIVERSIDAD DE SANTIAGO DE CHILE
FACULTAD DE INGENIERIA
DEPARTAMENTO DE INGENIERIA INFORMÁTICA

PROPUESTA DE TESIS

Felipe Alberto Reyes González

Profesor Guía: Victor Parada

Memoria para obtener el título de Analista en Computación Científica

Santiago, Chile

2017

Tabla de contenido

Índice de tablas

Índice de figuras

Capítulo 1

Introducción

W. S. McCulloch y Pitts (1943) presentaron el primer modelo de redes neuronales artificiales (*Neural networks*, NN) en términos de un modelo computacional de actividad nerviosa. Las NN se han inspirado en las redes neuronales biológicas y las conexiones que construyen. Este modelo era un modelo binario, donde cada neurona posee un umbral, y sirvió de base para los modelos posteriores.

Las características principales de las NN son las siguientes:

1. Auto-organización y adaptabilidad: utilizan algoritmos de aprendizaje adaptativo y auto-organizativo, por lo que ofrecen mejores posibilidades de procesado robusto y adaptativo.
2. Procesado no lineal: aumenta la capacidad de la red para aproximar funciones, clasificar patrones y aumenta su inmunidad frente al ruido.
3. Procesado paralelo: normalmente se usa un gran número de nodos de procesado, con alto nivel de interconectividad.

1.1 Las neuronas biológicas

El sistema nervioso y hormonal permite la comunicación de los distintos órganos que componen el cuerpo humano, esto a través de las neuronas, que permiten la realización de potenciales eléctricos.

El sistema de comunicación se compone de tres partes:

1. Los *receptores*, que recogen la información en forma de estímulo.
2. El *sistema nervioso*, que recibe y envía al resto de los órganos.
3. Los *órganos diana* o *órganos efectores* que reciben la información y la interpretan en forma de acciones.

La estructura base del sistema de comunicación son las neuronas. Éstas utilizan reacciones químicas para la transmisión de información en forma de potenciales eléctricos, esta información se envía a través de prolongaciones de las mismas neuronas, formando una red neuronal. Las neuronas se componen de *dendritas*, que propagan la información al interior de la neurona, el *cuerpo*, que procesa la información hasta generar una respuesta, y el *axón*, para la salida de la información mediante la propagación de impulsos electro-químicos (ver figura 1.1).

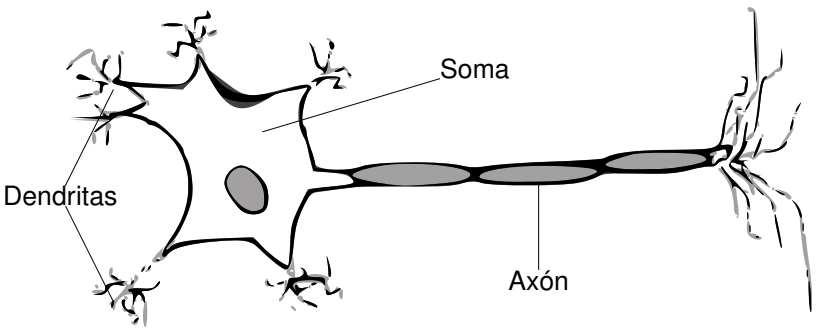

Figura 1.1: Neurona biológica.

Éstas realizan una serie de funciones, dentro de las cuales se destacan las siguientes

- Recogen información que viene en forma de impulso desde otras neuronas.
- Integran la información antes de la activación.
- Transmiten la información en forma de impulsos a través de su axón.
- A través de las ramificaciones el axón distribuye la información.
- Transmite la información a las neuronas adyacentes o células efectoras.

1.2 Redes neuronales

Los primeros trabajos sobre redes neuronales artificiales datan del siglo XIX con los trabajos de Sigmund Freud, pero es en la década de los 40 que de la mano de W. McCulloch y Pitts (1943) realizaron el primer modelo matemático de una Red Neuronal Artificial. Este modelo está basado en la idea de la utilización de los impulsos binarios. Introducen la utilización de una función de paso por umbral para la activación.

Posteriormente, Donald Hebb desarrolla un procedimiento matemático de aprendizaje descrito en su libro *Organization of Behavior* (Hebb, 2002), describiendo un paradigma de aprendizaje que lleva su nombre.

Con los trabajos de Rosenblatt (1957) y Rosenblatt y Laboratory (1958) se generalizó el modelo de W. McCulloch y Pitts (1943), agregándole el proceso de aprendizaje, llamandolo *perceptrón*. El modelo ajustaba los pesos de las conexiones de las entradas y salidas en forma proporcional al error. Dos años después, Bernard Widrow diseñó un perceptrón llamado *ADALINE* (Adaptative Linear Element), este ajusta los pesos entre los niveles de entrada y salida en función del error entre el valor esperado y el valor obtenido.

Kohonen (1972, 1974), desarrolló estudios entorno a la memoria asociativa y matrices de correlación, similar a los trabajos desarrollados por Anderson (1968, 1970) y Steinbuch (1961); Steinbuch y Piske (1963).

Hopfield (1982) describe un método de análisis del estado estable de una red autoasociativa. Demostró que se puede construir una ecuación de energía capaz de describir la actividad de una Red Neuronal Monocapa, y que el sistema puede converger a un mínimo local, este resultado hizo resurgir el interés de aplicar las redes neuronales para la resolución de problemas difíciles que los computadores no pudiesen resolver.

1.3 El perceptrón

El perceptrón fue concebido como un sistema que permite la clasificación de forma automática. Busca disponer de un sistema que fuera capaz de determinar la frontera de un grupo de clases diferentes. Los diversos ejemplos de cada clase existente conformaban los patrones, los que aportan la información necesaria para lograr la discriminación de las clases existentes. Finalmente, el sistema determinaba a que clase correspondía cada uno de los ejemplos presentados, incluso de ejemplos nuevos.

1.3.1 Arquitectura del perceptrón

Es una arquitectura monocapa conformada por una capa de entrada y una capa de salida, donde cada capa contiene tantas neuronas como sean necesarias. Cada una de las neuronas de entrada está conectada con todas las neuronas de salida y son estas conexiones las encargadas de determinar las superficies de discriminación del sistema.

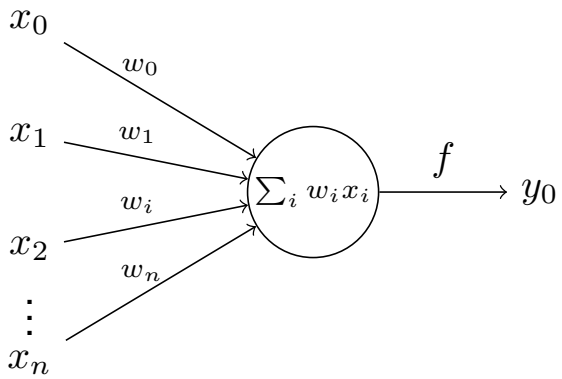

Figura 1.2: Perceptrón simple

Se entenderá que un perceptrón simple es una neurona artificial, que es la unidad básica de inferencia en forma de discriminador lineal (Cruz, 2011). Consiste en la suma ponderada

del vector de características que, en la fase inicial, permite a la red ser entrenada mediante el uso de un patrón definido, logrando determinar la salida de la red (ver figura 1.2). Si la suma ponderada del vector de entrada es mayor o menor que el patrón definido la salida de la red será uno (1), y en caso contrario la salida será cero (0). Dada la naturaleza del perceptrón, este no es capaz de distinguir patrones complejos.

Capítulo 2

Redes neuronales

2.1 Las redes neuronales

El elemento básico de las NN es el nodo, que recibe un vector de entrada para producir una salida como muestra en la figura 1.2. 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 2.1.

$$y_i = \sum_j w_{ij} y_j \tag{2.1}$$

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.

2.2 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.

2.2.1 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.

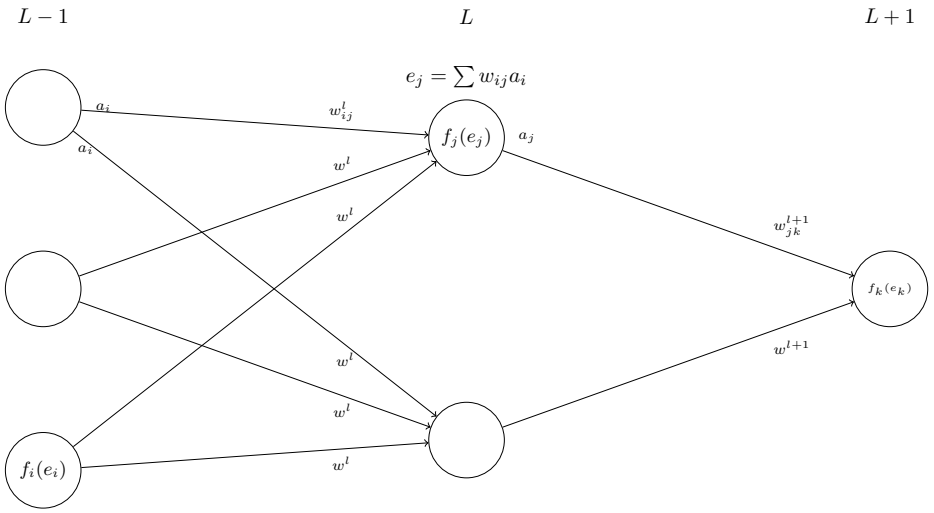

Figura 2.1: Neurona

En la figura 2.1 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.

2.2.2 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 2.2 y 2.3

respectivamente.

$$f_{sigm}(x) = \frac{1}{1+\exp(-x)} \tag{2.2}$$

$$f_{tanh}(x) = \frac{1-\exp(-x)}{1+\exp(-x)} \tag{2.3}$$

Ambas funciones poseen como imagen intervalo de valores entre $[0, 1]$ y $[-1, 1]$ como se observa en la figura 2.2.

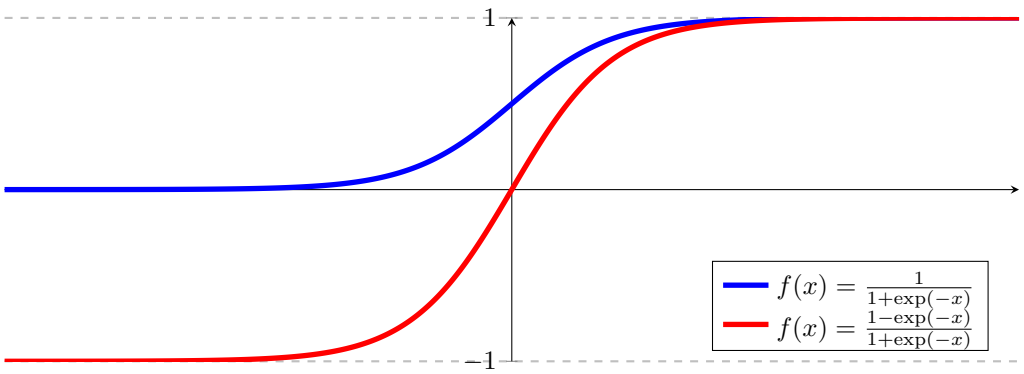

Figura 2.2: Funciones de activación mas utilizadas.

2.3 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:

$$E = \frac{1}{N}\sum_{i=1}^{N} e(i) \tag{2.4}$$

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

$$e(i) = \frac{1}{n_C} \sum_{j=1}^{n_C} (s_j(i) - y^j(n))^2 \tag{2.5}$$

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

$$w(i) = w(n-1) - \alpha \frac{\partial e(i)}{\partial w} \tag{2.6}$$

donde $e(i)$ es el error para el patrón de entrada i dado por la ecuación 2.5, 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 des-

cendente, 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.

2.4 El gradiente descendente

2.4.1 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.

Error en el factor de escala. Véase también las contribuciones anteriroes al análisis del desvanecimiento del gradiente (Hochreiter y Schmidhuber, 1997; Bengio, Simard, y Frasconi, 1994; Hochreiter, 1991). Propagando el error de una unidad u en el timpo t hacia una unidad v

for q time steps, el error escala como muestra la ecuación 2.7

$$\frac{\partial \vartheta_v(t-q)}{\partial \vartheta_u(t)} = \begin{cases} f'_v(net_v(t-1))w_{uv} & q = 1 \\ f'_v(net_v(t-q)) \sum_{l=1}^{n} \frac{\partial \vartheta_l(t-q+1)}{\partial \vartheta_u(t)} w_{lv} & q > 1 \end{cases} \tag{2.7}$$

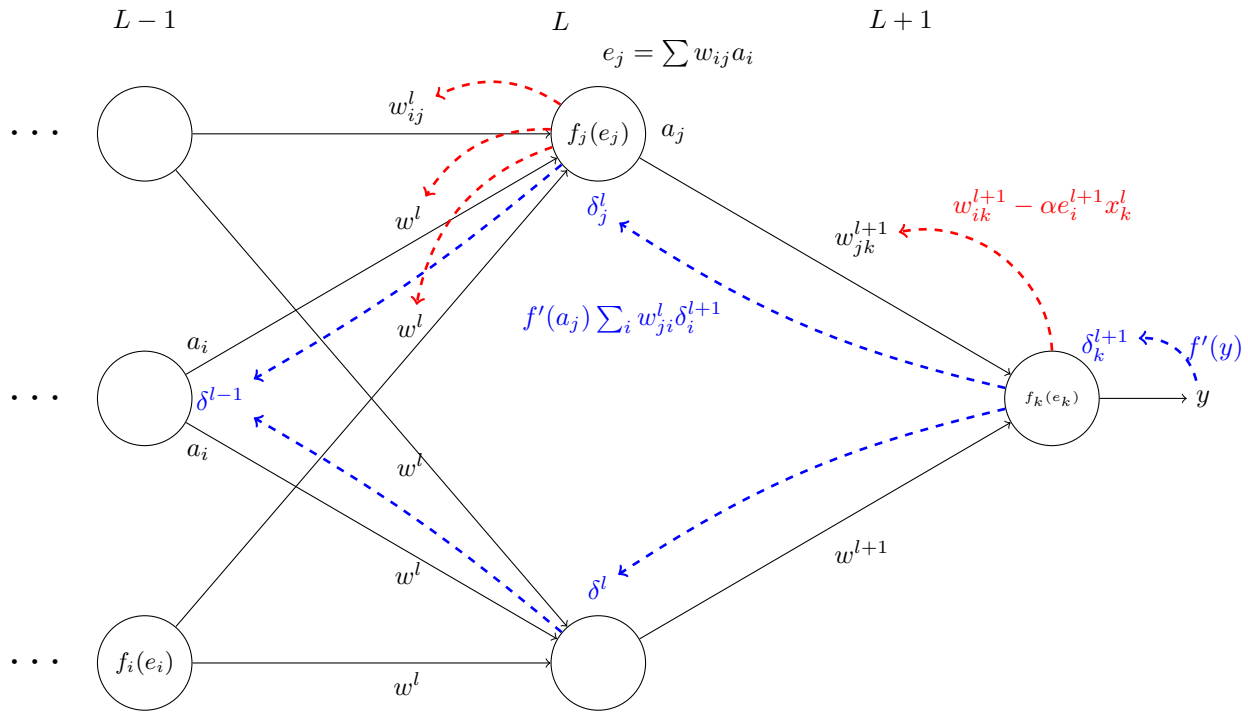

Figura 2.3: Gradiente descendente

Bibliografía

Anderson, J. A. (1968). A memory storage model utilizing spatial correlation functions. *Kybernetik*, *5*, 113–119.

Anderson, J. A. (1970). *Two models for memory organization using interacting traces.* doi: 10 .1016/0025-5564(70)90147-1

Bengio, Y., Simard, P., y Frasconi, P. (1994, Mar). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, *5*(2), 157-166. doi: 10.1109/ 72.279181

Cruz, P. (2011). *Inteligencia artificial con aplicaciones a la ingeniería*. Barcelona: Marcombo.

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

Hebb, D. (2002). *The organization of behavior: A neuropsychological theory*. Taylor & Francis.

Hochreiter, S. (1991). *Untersuchungen zu dynamischen neuronalen netzen* (Diploma thesis). Institut f. Informatik, Technische Univ. Munich.

Hochreiter, S., y Schmidhuber, J. (1997, noviembre). Long short-term memory. *Neural Comput.*, *9*(8), 1735–1780. Descargado de `http://dx.doi.org/10.1162/neco.1997.9.8.1735` doi: 10.1162/neco.1997.9.8.1735

Hopfield, J. J. (1982, 1 de abril). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the United States of America*, *79*(8), 2554–2558.

Kohonen, T. (1972, April). Correlation matrix memories. *Computers, IEEE Transactions on*, *C-21*(4), 353-359. doi: 10.1109/TC.1972.5008975

Kohonen, T. (1974, April). An adaptive associative memory principle. *Computers, IEEE Transactions on*, *C-23*(4), 444-445. doi: 10.1109/T-C.1974.223960

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

McCulloch, W., y Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, *5*(4), 115-133. doi: 10.1007/BF02478259

McCulloch, W. S., y Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, *5*(4), 115–133. doi: 10.1007/BF02478259

Rosenblatt, F. (1957). *The perceptron—a perceiving and recognizing automaton* (Report n.º 85-460-1). Cornell Aeronautical Laboratory.

Rosenblatt, F., y Laboratory, C. A. (1958). *The perceptron: a theory of statistical separability in cognitive systems (project para)*. Cornell Aeronautical Laboratory.

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

Steinbuch, K. (1961). Die lernmatrix. *Kybernetic*, *1*, 36–45.

Steinbuch, K., y Piske, U. A. W. (1963, Dec). Learning matrices and their applications. *Electronic Computers, IEEE Transactions on*, *EC-12*(6), 846-862. doi: 10.1109/PGEC.1963.263588