Bloque 4: Electromiograma

Redes Neuronales Artificiales (RNA)

Luis Bote Curiel
Francisco Manuel Melgarejo Meseguer

DTSC

Curso 24-25

Índice

Introducción

Redes Neuronales

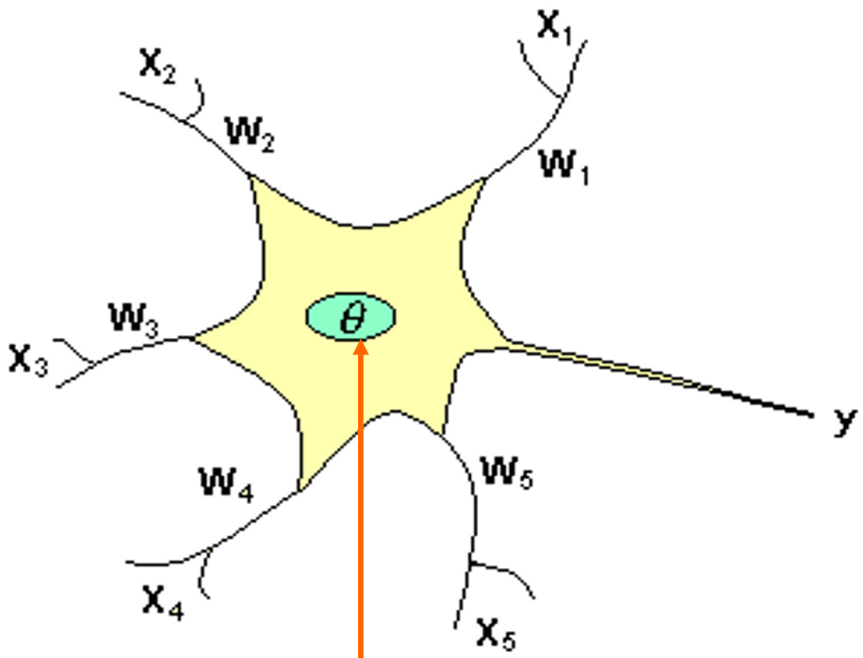
Surgimiento

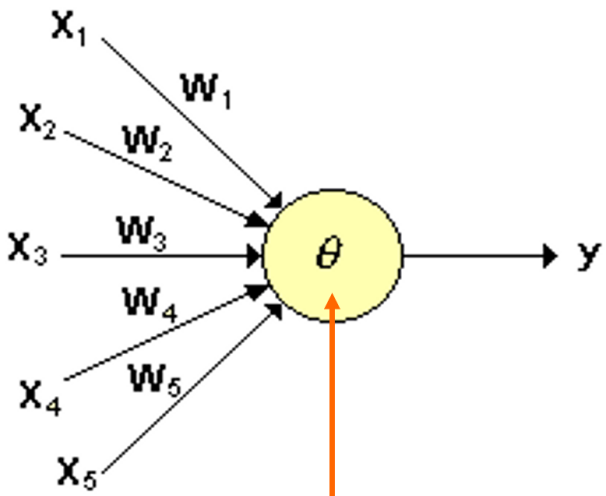
- Nacieron con el objetivo de emular el funcionamiento de los sistemas neuronales biológicos
- Intentando emular a muy bajo nivel la estructura del cerebro
- Formadas por un conjunto de unidades, llamadas nodos o neuronas, conectadas unas con otras.

Neuronas

- El cerebro consta de un gran número de elementos (aprox. 10^{11}) altamente interconectados (aprox.10^4 conexiones por elemento), llamados neuronas.
- El cerebro humano
  - Procesa información imprecisa rápidamente.
  - Aprende sin instrucciones explícitas.
  - Crea representaciones internas que permiten estas habilidades

Inspiración





θ es la función umbral que la neurona debe sobrepasar para activarse; este proceso ocurre biológicamente en el cuerpo de la célula.

Inspiración

Pero no mucha

- El conocimiento que se posee sobre el sistema nervioso en general no es completo.
- Debemos definir funcionalidades y estructuras de conexiones que no tengan que guardar similitud con los sistemas biológicos.
- Son arquitecturas ligadas a las necesidades de las aplicaciones para las que se diseñan.

Definición

Las redes neuronales artificiales son modelos matemáticos multiparamétricos no lineales, capaces de inducir una correspondencia entre conjuntos de patrones de información (la relación estímulo - respuesta). El procesamiento de la información se realiza a la manera de los sistemas neuronales biológicos.

Resumen

Resumen

- Modelos simplificados de las redes neuronales biológicas: grupo interconectado de neuronas artificiales.
- Las neuronas generalmente se distribuyen en capas de distintos niveles, con conexiones que unen las neuronas de distintas capas y/o neuronas de una misma capa.
- Usan un modelo matemático para procesar la información.
- Son un enfoque de sistema adaptativo.
- El comportamiento de la red está determinado por:
  - su topología,
  - los pesos de las conexiones.
  - la función de activación de las neuronas.

Características

- **Aprendizaje Adaptativo**: Las RNA aprenden a realizar tareas a partir de un conjunto de datos dados en el proceso de aprendizaje.
- **Auto-organización**: Pueden crear su propia organización o representación de la información recibida.
- **Operación en tiempo real**: Las operaciones realizadas pueden ser llevadas a cabo por computadores paralelos, o dispositivos de hardware especiales que aprovechan esta capacidad.
- **Tolerancia a fallos parciales**: La destrucción parcial de una red daña parcialmente el funcionamiento de la misma, pero no la destruye completamente. Esto es debido a la redundancia de la información contenida.
- **Capacidad para resolver problemas no lineales**

Perceptrón Simple

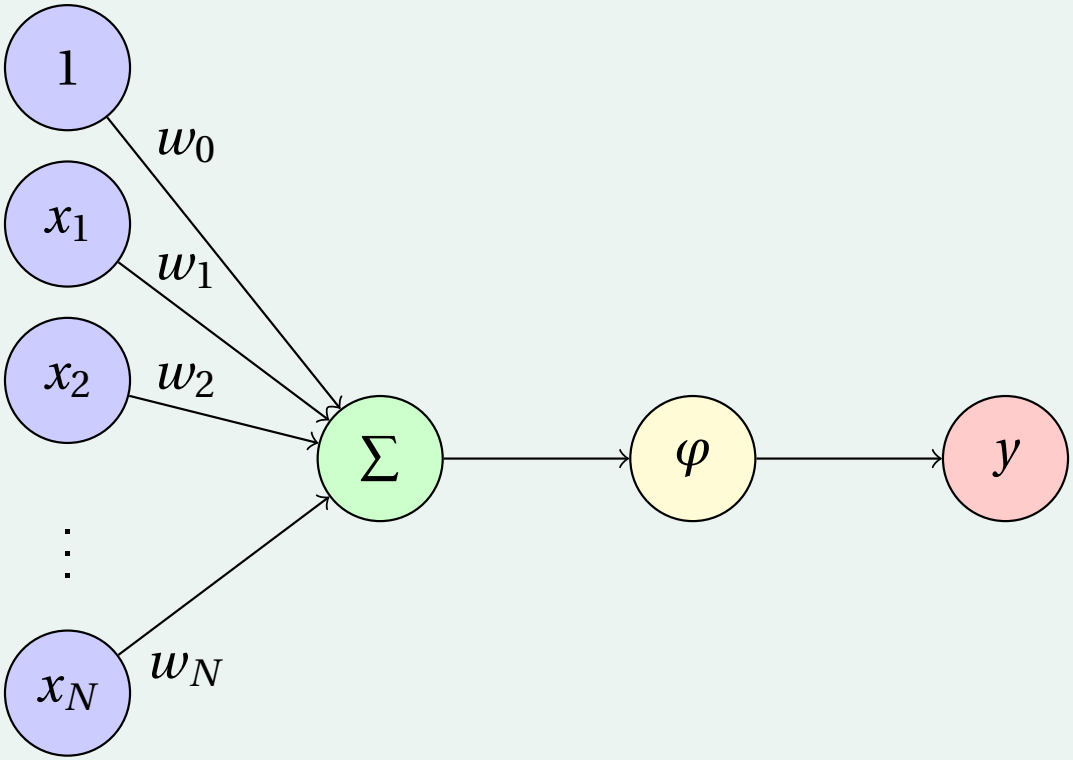
Neurona Artificial

Perceptrón simple

- Es la unidad o nodo de proceso de información fundamental en una red neuronal.
- Recibe un conjunto de señales de entradas procedentes del mundo exterior o de otras neuronas.
- Las señales de entrada se reciben a través de unas conexiones, que tienen un número real asociado llamado peso.
- Procesa la información recibida, mediante una serie de operaciones simples.
- Emite una señal de salida como respuesta a las señales de entrada.

Esquema

Esquema



- Pesos sinápticos.
- Sumador.
- Función de activación.
- Sesgo.

Perceptrón Simple

Notación

- De acuerdo con la representación anterior, la salida del perceptrón simple puede escribirse como

$$y = \varphi(\boldsymbol{w}^T \boldsymbol{x})$$

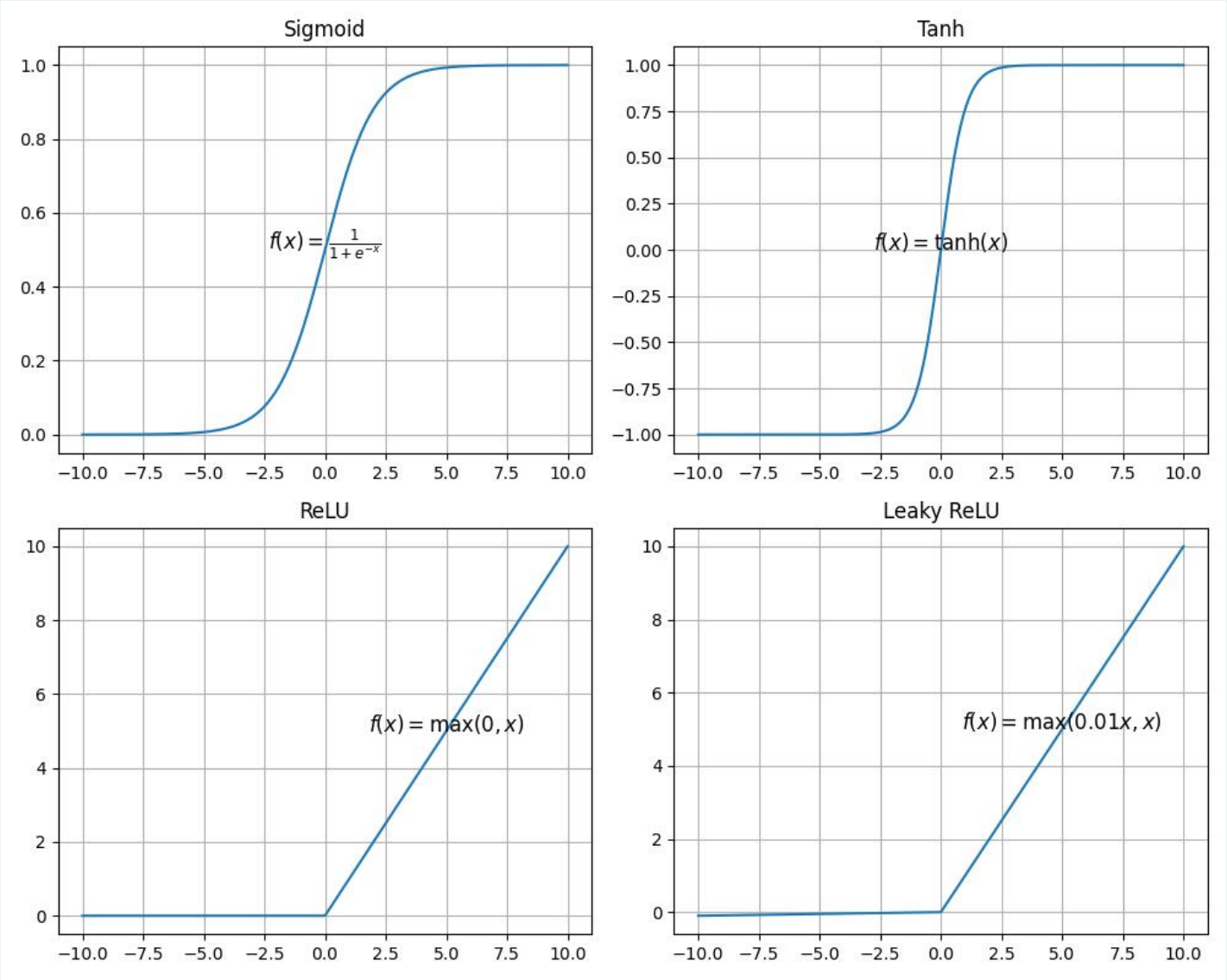
- Si φ es la función lineal, tiene la forma de un hiperplano.
- De esta forma, sabemos que podemos realizar tareas de clasificación en problemas lineales.

Funciones de activación

- Dado que no solo de problemas lineales vive el ingeniero, es necesario aplicar algunos cambios para poder trabajar en esos escenarios.
- El enfoque seguido es algo distinto al utilizado en SVM. Aquí usaremos funciones no lineales que mapeen las salidas de los percetrones.
- Estas funcionen deben ser obligatoriamente **derivables**.

Perceptrón Simple

Funciones de activación



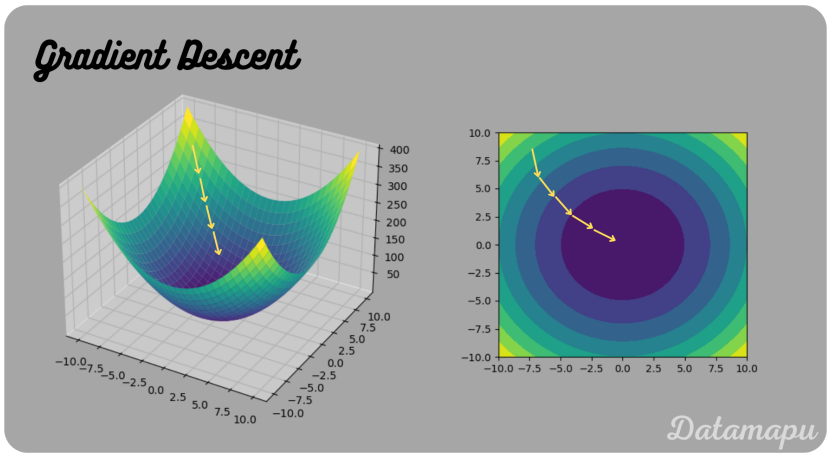
Perceptrón Simple

Entrenamiento

- Supongamos que tenemos un conjunto de entrenamiento formado por D tuplas de $\{\boldsymbol{x}_1, y_1; \boldsymbol{x}_2, y_2; \ldots; \boldsymbol{x}_D, y_D\}$.
- Podemos definir el error de nuestro perceptrón como

$$E(\boldsymbol{w}) = \frac{1}{2} \sum_{d}^{D} (y_d - \hat{y}_d)^2$$

- Idealmente, $E(\boldsymbol{w})$ es convexo con un único mínimo.
- Nuestro objetivo será *descender* por la curva de error.



Descenso del gradiente

- Queremos ir actualizando nuestros pesos de manera que lleguemos al valor mínimo.

$$\boldsymbol{w} = \boldsymbol{w} + \Delta \boldsymbol{w}$$

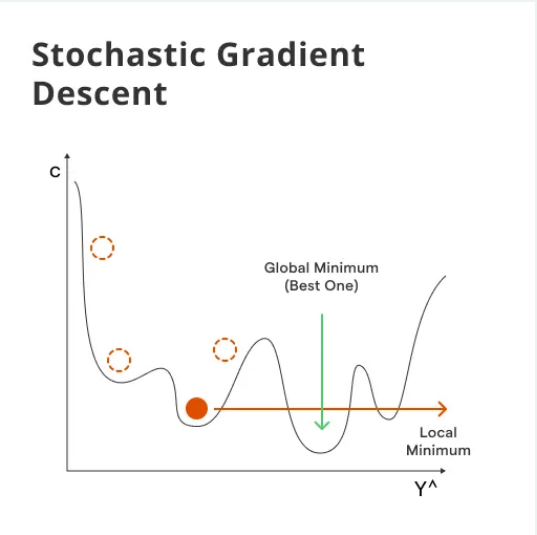
- Sabemos que las tasas de cambio de las funciones vienen dadas por las derivadas.
- Queremos *movernos* para que nuestro error se minimice rápido.
- De manera que nos moveremos en la dirección de las derivadas pero en sentido contrario,

$$\Delta \boldsymbol{w} = -\gamma \frac{\partial E}{\partial \boldsymbol{w}} \rightarrow \frac{\partial}{\partial w_i} \frac{1}{2} \sum_d^D (y_d - \hat{y}_d)^2 = \sum_d^D (y_d - \hat{y}_d) x_{id}$$

donde x_{id} es la componente i-ésima del elemento d-ésimo del conjunto de entrenamiento y γ es la tasa de aprendizaje, para evitar los pasos sean muy grandes y no caigamos en el mínimo.

Descenso estocástico del gradiente

- El descenso por gradiente es una técnica bastante poderosa que nos permite minimizar funciones.
- Solo tenemos un problema, la definición del error de forma global, hay una suma sobre todos los elementos puede acarrear problemas de mínimos locales en caso de que la función los tenga.



- Por ello se propone hacer variaciones por cada uno de los elementos en vez de hacer las variaciones de golpe.

$$\Delta \boldsymbol{w}_D = -\gamma(y_d - \hat{y}_d)\boldsymbol{x}_d$$

Perceptrón Multicapa

Perceptrón Multicapa

Definición

Un perceptrón multicapa (MLP) es una red neuronal artificial que consiste en al menos tres capas de nodos: una capa de entrada, una capa oculta y una capa de salida.

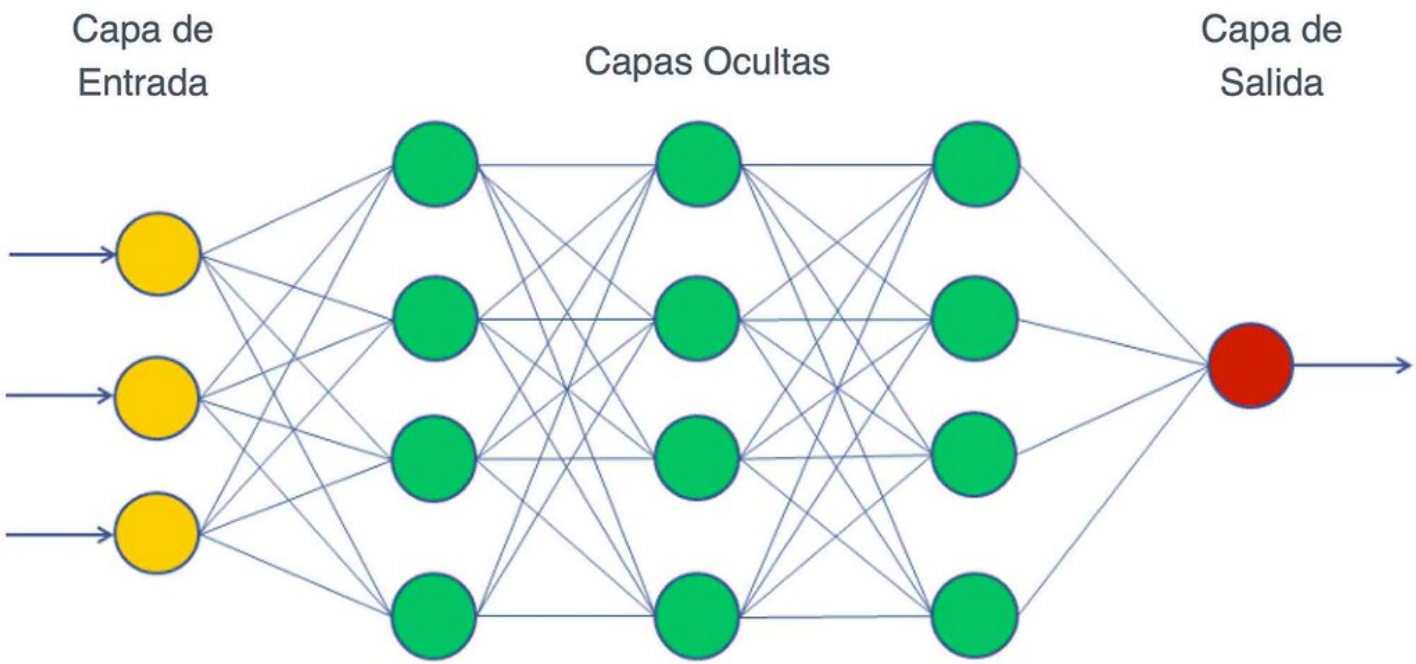
Arquitectura

- Capa de entrada: Recibe las señales de entrada.
- Capas ocultas: Procesan las señales recibidas de la capa de entrada.
- Capa de salida: Produce la salida final de la red.

Entrenamiento

El entrenamiento de un MLP implica ajustar los pesos de las conexiones utilizando algoritmos como el descenso de gradiente y la retropropagación.

Perceptrón Multicapa



Perceptrón Multicapa

Perceptrones Simples

- Limitados a problemas linealmente separables
- No pueden resolver problemas no lineales

Perceptrones Multicapa

- Pueden resolver problemas no lineales
- Capaces de aprender representaciones más complejas
- Utilizan funciones de activación no lineales
- Mayor capacidad de generalización

Perceptrón Multicapa

Entrenamiento

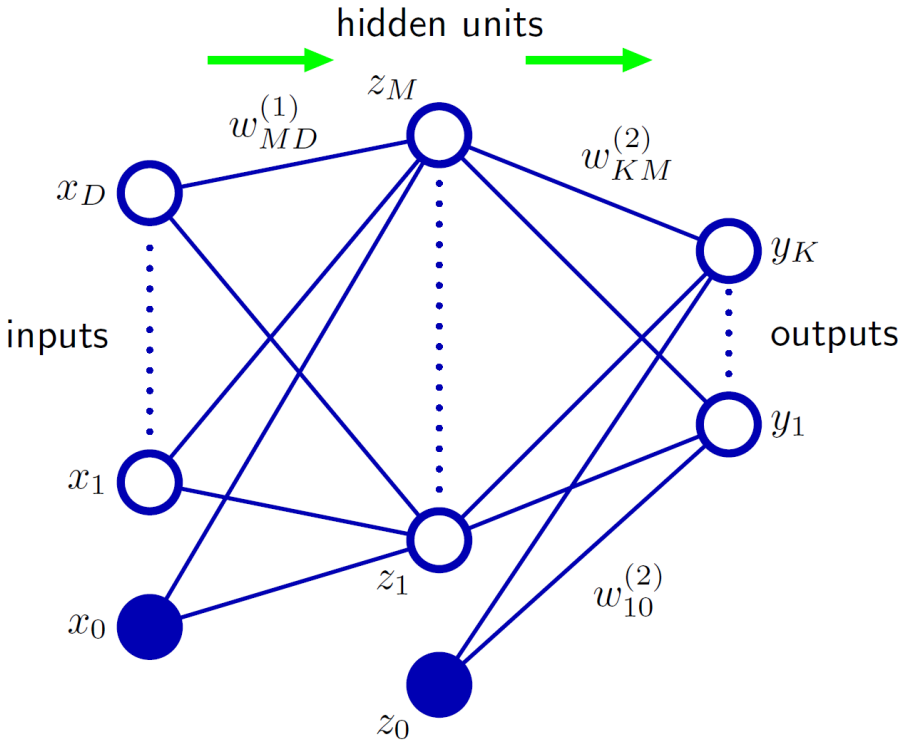
- **Inicialización de Pesos**: Pesos inicializados aleatoriamente $\left(U(-0{,}05, 0{,}05)\right)$.
- **Propagación hacia Adelante**: Cálculo de salidas capa por capa.
- **Cálculo del Error**: Comparación con la salida deseada.
- **Propagación hacia Atrás**: Cálculo de gradientes del error.
- **Actualización de Pesos**: Ajuste de pesos usando optimización.
- **Iteración**: Repetición hasta minimizar el error.

Perceptrón Multicapa

Formulación

- Llamaremos $a_j = \sum_i w_{ji} z_i$ a la salida de la neurona j-ésima.
- Llamaremos $z_j = \varphi(a_j)$ a la aplicación de la función de activación sobre la salida a_j.
- El error definido anteriormente $\Delta w_{ji} = \frac{\partial E}{\partial a_j} \frac{\partial a_j}{\partial w_{ij}}$ por la regla de la cadena.
- Resolviendo, obtenemos que $\frac{\partial a_j}{\partial w_{ij}} = z_i$
- Sustituyendo y llamando $\delta_j = \frac{\partial E}{\partial a_j}$ obtenemos que

$$\Delta w_{ji} = \frac{\partial E}{\partial w_{ij}} = \delta_j z_i$$



Derivación de las Ecuaciones

Para evaluar los δ de las unidades ocultas, utilizamos la regla de la cadena para derivadas parciales:

$$\delta_j \equiv \frac{\partial E_n}{\partial a_j} = \sum_k \frac{\partial E_n}{\partial a_k} \frac{\partial a_k}{\partial a_j}$$

Donde la suma se realiza sobre todas las unidades k a las que la unidad j envía conexiones. Las unidades etiquetadas como k pueden incluir otras unidades ocultas y/o unidades de salida.

Sustitución

Al sustituir la definición de δ y hacer uso de las ecuaciones anteriores, obtenemos la siguiente fórmula de retropropagación:

$$\delta_j = \varphi'(a_j) \sum_k w_{kj} \delta_k$$

Esto nos dice que el valor de δ para una unidad oculta particular se puede obtener propagando los δ hacia atrás desde las unidades superiores en la red.

Sustitución y Fórmula de Retropropagación

Proceso Recursivo

Como ya conocemos los valores de los δ para las unidades de salida, al aplicar recursivamente esta fórmula podemos evaluar los δ para todas las unidades ocultas en una red feed-forward, independientemente de su topología.

Ejemplo

- Considere una red con una capa oculta y una de salida.
- La función de activación será $\varphi(a) = \tanh(a) \xrightarrow{\partial a} (1 - \varphi(a)^2)$
- Definimos el error como $E = \frac{1}{2}\sum_k^K (\hat{y}_k - y_k)^2$
- Definimos la salidas de la capa oculta como $a_j = \sum_i^D w_{ji}^1 x_i$ y $z_i = \tanh(a_i)$
- Definimos la salida de la red como $\hat{y}_k = \sum_j^M w_{kj}^2 z_j$
- Calculamos el error hacia adelante $\delta_k = \hat{y}_k - y_k$
- Retropropagamos: $\delta_j = (1 - z_j^2)\sum_k^K w_{kj}\delta_k$
- Obtenemos las Δw para cada capa $\Delta w_{ji}^1 = \delta_j x_i$ y $\Delta w_{kj}^2 = \delta_k z_j$

Bibliografía

Bibliografía

- Agradecimiento a los profesores Javier Gimeno Blanes y Alberto Rodríguez Martínez.

L. Sörnmo and P. Laguna, *Bioelectrical Signal Processing in Cardiac and Neurological Applications*.
Academic Press series in biomedical engineering, Elsevier Science, 2005.