

# Introducción a los modelos computacionales

## Práctica 1. Implementación del perceptrón multicapa

Pedro Antonio Gutiérrez

pagutierrez@uco.es

Asignatura “Introducción a los modelos computacionales”

4º Curso Grado en Ingeniería Informática

Especialidad Computación

Escuela Politécnica Superior

(Universidad de Córdoba)

27 de septiembre de 2023



## 1 Contenidos

## 2 Notación y arquitectura



# Objetivos de la práctica

- Familiarizar al alumno con los modelos computacionales de redes neuronales, en concreto, con el perceptrón multicapa.
- Implementar el algoritmo de retropropagación básico para el perceptrón multicapa.
- Comprobar el efecto de distintos parámetros y preprocesamiento:
  - Arquitectura de la red.
  - Factor de momento.
  - Normalización de datos.
  - etc.



# Algoritmo de retropropagación

- Leer y analizar los apuntes de teoría.
- Analizar especialmente el pseudocódigo.



# Condición de parada

- **Versión estándar**, el algoritmo para si:
  - El error de entrenamiento no baja más de 0,00001 o sube, durante 50 iteraciones (bucle externo).



# Consideraciones para la normalización de datos I

- La mayoría de métodos de aprendizaje automático necesitan que los datos estén normalizados para mitigar el efecto de las diferentes magnitudes de las variables.
- Existen muchas variantes de la normalización, pero las dos más comunes son:
  - Escalado: cada variable se transforma para que esté en un intervalo  $[a, b]$ :

$$X' = a + \frac{(X - X_{\min})(b - a)}{X_{\max} - X_{\min}} \quad (1)$$

- Estandarización: normaliza cada variable típicamente produciendo una distribución de media cero y desviación típica 1:

$$X' = \frac{X - \mu}{\sigma} \quad (2)$$



# Consideraciones para la normalización de datos II

- En ambos casos es importante que los parámetros de escalado se calculen **sobre el conjunto de entrenamiento**, y luego se apliquen las transformaciones en el conjunto de entrenamiento y en el test.
- Un **error común** es aplicar la normalización sobre el conjunto de test calculando los valores mínimos y máximos ( $X_{\text{máx}}$  y  $X_{\text{mín}}$ ), o la media y desviación ( $\mu$  y  $\sigma$ ), sobre este conjunto o antes de realizar la partición en conjuntos de entrenamiento y test. En ambos casos se está utilizando directa o indirectamente información de test para la construcción del modelo.
- En nuestro caso, vamos a escalar las variables de entrada en el rango  $[-1, 1]$  y las variables de salida en  $[0, 1]$  ya que la última capa de salida es una sigmoide.



# Algoritmo de retropropagación

## Algoritmo de retropropagación *on-line*

### Inicio

①  $w_{ji}^h \leftarrow U[-1, 1]$  // Aleatorios entre  $-1$  y  $+1$

### ② Repetir

① Para cada patrón con entradas  $\mathbf{x}$ , y salidas  $\mathbf{d}$

①  $\Delta w_{ji}^h \leftarrow 0$  // Se aplicarán cambios por cada patrón

②  $out_j^0 \leftarrow x_j$  // Alimentar entradas

③ forwardPropagation() // Propagar las entradas ( $\Rightarrow \Rightarrow$ )

④ backPropagation() // Retropropagar el error ( $\Leftarrow \Leftarrow$ )

⑤ accumulateChange() // Calcular ajuste de pesos

⑥ weightAdjustment() // Aplicar el ajuste calculado

### Fin Para

Hasta (CondicionParada)

③ Devolver matrices de pesos.

### Fin





# Algoritmo de retropropagación

## Algoritmo de retropropagación *off-line*

### Inicio

①  $w_{ji}^h \leftarrow U[-1, 1]$  // Aleatorios entre  $-1$  y  $+1$

### ② Repetir

①  $\Delta w_{ji}^h \leftarrow 0$  // Se aplicarán cambios al final

② Para cada patrón con entradas  $\mathbf{x}$ , y salidas  $\mathbf{d}$

①  $out_j^0 \leftarrow x_j$  // Alimentar entradas

② forwardPropagation() // Propagar las entradas ( $\Rightarrow \Rightarrow$ )

③ backPropagation() // Retropropagar el error ( $\Leftarrow \Leftarrow$ )

④ accumulateChange() // Acumular ajuste de pesos

### Fin Para

③ weightAdjustment() // Aplicar el ajuste calculado

### Hasta (CondicionParada)

③ Devolver matrices de pesos.

### Fin



# Algoritmo de retropropagación: funciones

forwardPropagation()

**Inicio**

- ① **Para**  $h$  de 1 a  $H$  // *Para cada capa ( $\Rightarrow \Rightarrow$ )*
  - ① **Para**  $j$  de 1 a  $n_h$  // *Para cada neurona de la capa  $h$* 
    - ①  $net_j^h \leftarrow w_{j0}^h + \sum_{i=1}^{n_{h-1}} w_{ji}^h out_i^{h-1}$
    - ②  $out_j^h \leftarrow \frac{1}{1 + \exp(-net_j^h)}$

**Fin Para**

**Fin Para**

**Fin**



# Algoritmo de retropropagación: funciones

backPropagation()

**Inicio**

- ① **Para**  $j$  de 1 a  $n_H$  *// Para cada neurona de salida*
  - ①  $\delta_j^H \leftarrow -(d_j - out_j^H) \cdot g'(net_j^H)$  *// Hemos obviado la constante (2), ya que el resultado será el mismo*

**Fin Para**

- ② **Para**  $h$  de  $H - 1$  a 1 *// Para cada capa ( $\Leftarrow \Leftarrow$ )*
  - ① **Para**  $j$  de 1 a  $n_h$  *// Para cada neurona de la capa  $h$* 
    - ①  $\delta_j^h \leftarrow (\sum_{i=1}^{n_{h+1}} w_{ij}^{h+1} \delta_i^{h+1}) \cdot out_j^h \cdot (1 - out_j^h)$  *// Pasa por todas las neuronas de la capa  $h + 1$  conectadas con  $j$*

**Fin Para**

**Fin Para**

**Fin**



# Algoritmo de retropropagación: funciones

acummmulateChange()

**Inicio**

- ① **Para**  $h$  de 1 a  $H$  // *Para cada capa ( $\Rightarrow \Rightarrow$ )*
  - ① **Para**  $j$  de 1 a  $n_h$  // *Para cada neurona de la capa  $h$* 
    - ① **Para**  $i$  de 1 a  $n_{h-1}$  // *Para cada neurona de la capa  $h - 1$* 

$$\Delta w_{ji}^h \leftarrow \Delta w_{ji}^h + \delta_j^h \cdot out_i^{h-1}$$

**Fin Para**

- ②  $\Delta w_{j0}^h \leftarrow \Delta w_{j0}^h + \delta_j^h \cdot 1$  // *Sesgo*

**Fin Para**

**Fin Para**

**Fin**



# Algoritmo de retropropagación: funciones

weightAdjustment()

**Inicio**

- ① **Para**  $h$  de 1 a  $H$  // *Para cada capa ( $\Rightarrow \Rightarrow$ )*
    - ① **Para**  $j$  de 1 a  $n_h$  // *Para cada neurona de la capa  $h$* 
      - ① **Para**  $i$  de 1 a  $n_{h-1}$  // *Para cada neurona de la capa  $h - 1$* 

$$w_{ji}^h \leftarrow w_{ji}^h - \eta \Delta w_{ji}^h - \mu (\eta \Delta w_{ji}^h (t - 1))$$

**Fin Para**
      - ②  $w_{j0}^h \leftarrow w_{j0}^h - \eta \Delta w_{j0}^h - \mu (\eta \Delta w_{j0}^h (t - 1))$  // *Sesgo*

**Fin Para**

**Fin Para**
- Fin**



# Introducción a los modelos computacionales

## Práctica 1. Implementación del perceptrón multicapa

Pedro Antonio Gutiérrez

`pagutierrez@uco.es`

Asignatura “Introducción a los modelos computacionales”

4º Curso Grado en Ingeniería Informática

Especialidad Computación

Escuela Politécnica Superior

(Universidad de Córdoba)

27 de septiembre de 2023

