****

**Inteligencia Computacional**

**Práctica 1**

**Reconocimiento óptico de caracteres**

**MNIST**

Curso 2018-2019

Máster en Ingeniería Informática

Miguel de Oliveira Dias Gonçalves

**Índice**

[**1.** **Introducción** 2](#_Toc530879736)

[**2.** **Perceptron 1-capa** 3](#_Toc530879737)

# **Introducción**

Las redes neuronales consisten en sistemas de software y/o hardware basados en la operación de las neuronas humanas[[1]](#footnote-1). Una neurona artificial es un modelo matemático, dónde hay una serie de entradas, un peso por cada entrada que cambiará durante su entrenamiento, una función de activación teniendo en cuenta las entradas y sus pesos, y una salida[[2]](#footnote-2).

Una red neuronal consiste así en una combinación de neuronas organizadas por capas, dónde cada capa recibe datos, los procesa y los entrega a la capa siguiente. La primera capa es la capa de entrada de los datos, la última es la capa de salida de los datos. Su entrenamiento se realiza utilizando grandes cantidades de datos, dónde cada entrada está acompañada de la salida pretendida de modo a que la red pueda ajustar sus pesos teniendo en cuenta si las salidas obtenidas son o no las pretendidas[[3]](#footnote-3).

Un uso de las redes neuronales es en el procesamiento de imágenes. La base de datos MNIST es de uso popular en el entrenamiento de dichas redes neuronales, estando registradas las mejores tasas de error para diversos tipos de redes y preprocesamientos de datos. Consiste en 60.000 imágenes de dígitos manuscritos para entrenamiento y otras 10.000 imágenes para prueba, normalizadas de modo a que todas tengan un tamaño de 28x28 pixeles, así como sus etiquetas correspondiendo a dígitos de 0 a 9. Tasas de error tabladas varían desde 12% para clasificadores lineales de 1 capa sin preprocesamiento, hasta una tasa de 0,21% obtenida con una junción de 5 redes convolucionales[[4]](#footnote-4) [[5]](#footnote-5).

El objetivo de esta práctica es evaluar la prestación de diversos tipos de redes neuronales, enseñados en las clases prácticas de la asignatura, al clasificar los dígitos manuscritos de la base de datos MNIST. Existe la posibilidad de implementar con código los diferentes algoritmos, o de utilizar bibliotecas externas enfocadas en eficiencia que ya los implementan a cambio de una reducción en la nota máxima en la práctica. El principal criterio para evaluar los resultados obtenidos será su tasa de error sobre el conjunto de prueba de la base de datos MNIST.

# **Perceptron 1-capa**

## **Implementación y algoritmo**

Se ha empezado la práctica con una red neuronal de 1 capa con 10 neuronas binarias con umbral, cada una correspondiendo a un dígito de 0 a 9. Su algoritmo de entrenamiento es el Perceptron. Los mejores resultados se han obtenido (aunque, cómo se verá abajo, aparentemente esto no sea determinante) inicializando todos los pesos de la red con valores aleatorios entre -1 y 1, para una tasa de aprendizaje de 0.55 en el caso del conjunto de prueba y 0.95 en el caso del conjunto de entrenamiento.

Cada neurona de la red tiene 785 entradas, una por cada píxel de una imagen de la base de datos MNIST más una entrada fija igual a 1 correspondiente al umbral.

Su función de activación retorna 1 si el producto de la entrada con el peso es igual o superior a 0, y retorna 0 en caso contrario.

Los pesos durante el entrenamiento son actualizados sumando a cada peso el resultado de la fórmula



Ecuación 1 - Función de activación usada en la red neuronal de 1 capa que usa el algoritmo Perceptron

Dónde es la tasa de aprendizaje; es el valor de salida deseado de la neurona, es decir, 1 si la neurona debe reconocer el dígito introducido o 0 si no; es la salida efectiva de la neurona, 1 si la neurona cree que el dígito introducido es lo que debe reconocer y 0 si no; e es el valor de entrada correspondiente a ese peso.

Se ha conseguido una tasa de error de 23.52% sobre el conjunto de prueba y 23.44% sobre el conjunto de entrenamiento.

Se ha usado el lenguaje Java y sus bibliotecas. La implementación del algoritmo es de mi autoría, usando el fichero *MNISTDatabase.java* creado por el profesor Fernando Berzal para leer y normalizar los datos de la base de datos MNIST y sus etiquetas.

## **Experimentos realizados**

Para determinar la mejor combinación de parámetros, se han realizado 6 experimentos correspondientes a igual número de combinaciones de parámetros. En particular, se han hecho 2 experimentos con todos los pesos inicializados a 0, otros 2 con los pesos inicializados con valores aleatorios entre -0.5 y 0.5, y otros 2 con los pesos inicializados con valores aleatorios entre -1 y 1. En cada par de experimentos, en uno la salida de la neurona y el valor de salida deseado eran 1 si respectivamente la neurona creía reconocer y se debería reconocer el dígito introducido, y -1 en caso contrario. En el otro, se usaba 0 en lugar del -1. En cada experimento se ha entrenado la red 9 veces con tasas de aprendizaje entre 0.1 y 0.9, aumentándose 0.1 la tasa entre cada entrenamiento, registrándose la tasa de error sobre el conjunto de prueba después de cada entrenamiento.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Experimento** | **Pesos iniciales** | **Salidas obtenida y deseada** | **Tasa de error mínima** | **Tasa de error media** | **Tasa de error máxima** |
| 1 | 0 | 1 y -1 | 27,16 | 30,01 | 32,45 |
| 2 | 0 | 1 y 0 | 27,8 | 29,62 | 31,33 |
| 3 | [-0.5;0.5] | 1 y -1 | 27,38 | 29,47 | 32,21 |
| 4 | [-0.5;0.5] | 1 y 0 | 28,13 | 29,79 | 33,69 |
| 5 | [-1;1] | 1 y -1 | 26,72 | 30,33 | 33,89 |
| 6 | [-1;1] | 1 y 0 | 26,05 | 28,85 | 31,68 |

1. <https://searchenterpriseai.techtarget.com/definition/neural-network> [↑](#footnote-ref-1)
2. <https://en.wikipedia.org/wiki/Artificial_neuron> [↑](#footnote-ref-2)
3. <https://searchenterpriseai.techtarget.com/definition/neural-network> [↑](#footnote-ref-3)
4. <https://en.wikipedia.org/wiki/MNIST_database> [↑](#footnote-ref-4)
5. <http://yann.lecun.com/exdb/mnist/> [↑](#footnote-ref-5)