Reconocimiento de dígitos aislados mediante Redes Neuronales.

Víctor Manuel Molinos Santiago (79334548G)
Germán Vaquero Montoro (75934269E)
victormolinos@correo.ugr.es
german109@correo.ugr.es
Curso 2021/2022, Universidad de Granada

Tecnologías del Habla
Grado en Ingeniería de Tecnologías de Telecomunicación

***Abstract*—En este informe se detalla la implementación de un sistema de reconocimiento dígitos aislados mediante el uso de redes neuronales. En concreto, el sistema puede reconocer y diferenciar entre los números *"Uno"*, *"Dos"*, *"Tres"* y *"Cuatro"*. Para la caracterización de un dígito se emplean los coeficientes cepstrales correspondientes al audio de dicho dígito y que actúan como entradas de la red neuronal. El entrenamiento de la red se basa en un algoritmo de entrenamiento supervisado por descenso de gradiente conocido como *Backpropagation*.**

I. INTRODUCCIÓN

La implementación del sistema consta de 3 módulos fundamentales.

El primero tiene la función de obtener las grabaciones y coeficientes cepstrales de los diferentes dígitos, para ello se utiliza el scipt `ObtenerCoefs.m` que permite grabar una señal y extraer y almacenar los coeficientes cepstrales que caracterizan dicha grabación. Como función adicional también permite revisar los datos guardados.

El segundo se encarga de entrenar la red neuronal. El script `RedNeuronal1234.m` carga los datos almacenados para cada dígito y compone la matriz de entradas. A continuación compone la matriz de testeo correspondientes a cada entrada. Con este conjunto de datos se entrena la red y se almacena para su uso posterior.

Finalmente el último módulo permite evaluar la red entrenada. El script `tester.m` realiza una grabación con los mismos parámetros usados en el primer módulo y extrae los coeficientes cepstrales correspondientes. Se evalúa la red neuronal con este conjunto de entradas y se procesa la respuesta para visualizar el resultado, de forma que la red neuronal reconoce y diferencia el dígito pronunciado.

De esta forma el sistema es capaz de grabar, almacenar y procesar los dígitos, entrenar una red neuronal con los datos almacenados y testear el resultado.

II. MÓDULO 1. OBTENCIÓN, PROCESADO Y ALMACENAMIENTO DEL DÍGITO

Este módulo se encarga de realizar grabaciones de 1s sobre las cuales se realiza el procesado que permite obtener los coeficientes cepstrales y almacenarlos de forma adecuada para que el módulo 2 pueda conformar correctamente la matriz de entradas que entrena la red neuronal. Las grabaciones se realizan usando los parámetros descritos en la Tabla I y en este caso se realizan 100 grabaciones por dígito.

Parámetros	Valor
Frecuencia de muestreo (kHz)	8
Tiempo de grabación (s)	1
Puntos FFT	2000
Tamaño trama (muestras)	720
Desplazamiento (muestras)	576
Nº de tramas por grabación	10

TABLE I: Parámetros utilizados en `ObtenerCoefs.m`

Esta función es llevada a cabo por el programa `ObtenerCoefs.m`, que dispone de dos modos de funcionamiento.

El modo 1, encargado de grabar una señal sonora y de obtener sus coeficientes cepstrales y el modo 2, que reproduce una grabación específica y devuelve como resultado sus coeficientes cepstrales. Al seleccionar el modo 1 el programa solicita el dígito a grabar y el número de grabaciones que se quiere realizar para dicho dígito. En caso de seleccionar el modo 2, el programa pregunta que número reproducir y la grabación correspondiente, respondiendo con una representación de los coeficientes cepstrales que la caracterizan y la reproducción del audio.

El procesado para la obtención de los coeficientes sigue el mismo método que el empleado en la práctica 4. Para comenzar se diseña el banco de filtros en escala Mel, para la obtención de los coeficientes. Una vez obtenido el banco de filtros tiene lugar la grabación y posteriormente el procesado y almacenamiento del dígito. Tras la grabación se eliminan las primeras y últimas 1000 muestras de señal consideradas como silencio. El procesado se realiza sobre un entramado con ventanas tipo Hamming al que se le aplica previamente un filtro de preénfasis que presenta la forma dada por la expresión (1). Su objetivo es eliminar el nivel de continua y compensar la caída natural de la voz a altas frecuencias. Además de resaltar las altas frecuencias también es responsable de compensar el efecto de impedancia que presentan los labios a la hora de

pronunciar el dígito a analizar.

$$H(z) = 1 - z^{-0.97} \tag{1}$$

Tras el filtrado tiene lugar el entramado con ventanas de Hamming. La ventana de Hamming resulta de interés debido a que logra un efecto más suavizado en los extremos. A las tramas resultantes, tras aplicar el entramado, se les realiza la transformada de Fourier y se aplica el banco de filtros Mel.

Una vez en este punto, para conseguir los coeficientes cepstrales basta con aplicar la transformada discreta del coseno (DCT) a la salida del banco de filtros. Para la especificación de un banco de 22 filtros se obtiene 22 coeficientes cepstrales distintos. De esta forma, cada trama queda representada por 22 coeficientes cepstrales, lo que equivale a tener una matriz de 22 columnas (una por coeficiente) y 10 filas (una por cada trama).

La capacidad de cómputo de la red neuronal para procesar 22 coeficientes de entrada resultaría demasiado elevada para los equipos domésticos disponibles (laptop, pc portátil...), por lo que resulta inevitable tener que reducir el número de coeficientes cepstrales con los que se pretende caracterizar el dígito. En esta fase se toman los 5 primeros coeficientes para cada trama resultando una matriz de 5 columnas y 10 filas. Finalmente se redimensiona la matriz, convirtiéndola en un vector resultado de la concatenación ordenada de las columnas (coeficientes). De esta forma se almacena un único vector que contiene las muestras correspondientes a los cinco primeros coeficientes, tal y como muestra la figura Fig. 1, donde se diferencian claramente los 5 coeficientes.

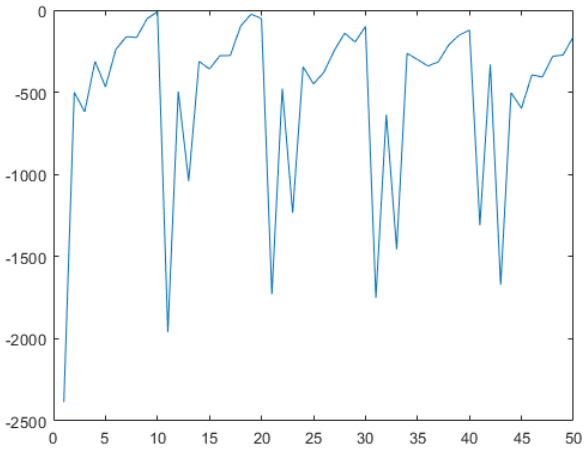


Fig. 1: Cinco primeros coeficientes cepstrales concatenados para el dígito 'UNO'.

Este vector es almacenado como columna en una matriz de forma que tras la última grabación queda una matriz con 50 filas (5 coeficientes) y tantas columnas como grabaciones se hayan hecho.

Los coeficientes cepstrales realizan una buena caracterización de las señales de voz debido a que permiten resaltar las características de la voz asociadas al tracto vocal. Por lo tanto, gracias a estos coeficientes es fácil extraer la información relevante, ignorando por otra parte toda la información menos significativa de la señal sonora. Esta información poco trascendental podría englobar características como el volumen con el que se está hablando, el tono en función de si es un hombre o una mujer, aparte de más efectos similares. Todos estos parámetros no aportan datos de gran relevancia para el reconocimiento del dígito, por lo que en los primeros coeficientes se concentran las características más importantes de la señal de voz. Gracias a esto es posible lograr una caracterización de la señal de voz con tan solo unos pocos coeficientes cepstrales, lo que resulta de interés en un proceso de codificación o reducción, como podría ser el caso de tener un menor número de entradas en la red neuronal, reduciendo la complejidad de la misma y por tanto la capacidad de cómputo.

Cabe destacar que el procesado usa tramas de tamaño grande y un desplazamiento igualmente amplio para poder minimizar la cantidad de tramas correspondientes a cada señal de voz reduciendo su número total a 10. Esto permite conseguir finalmente 10 muestras por coeficiente, de forma que, si el vector contiene la concatenación de 5 coeficientes, contiene en total 50 muestras.

III. Módulo 2. Diseño, entrenamiento y almacenamiento de la red neuronal

El cerebro humano se caracteriza por su flexibilidad y ajuste a nuevos ambientes mediante un proceso de aprendizaje, sin la necesidad de programarlo. Siguiendo este razonamiento las redes neuronales se han desarrollado con el objetivo de emular ciertas características y funcionalidades del cerebro humano. Partiendo de esta premisa se pueden definir las redes neuronales como modelos matemáticos o biológicos adaptados a un sistema computacional.

En el proceso de intentar imitar el cerebro humano surge el elemento más básico de una red neuronal, el cual es una neurona. Cada neurona está formada por lazos sinápticos (X*W), un mezclador lineal (que realiza la suma ponderada de las entradas) y una función de activación como se observa en la Fig. 2. De esta forma, cada neurona configura una regresión lineal con un conjunto de *x* entradas y *W* pesos que actúa, a su vez, como entrada a una función de activación que proporciona la no linealidad necesaria en la red. En cada iteración, la salida *y* proporcionada por la red se compara con el correspondiente conjunto de testeo asociado para establecer un criterio que permita ajustar cada neurona mediante la manipulación de sus pesos. Finalmente la red es capaz de "aprender" los patrones que caracterizan a cada salida ajustando los pesos W y puede reconocer, tras el entrenamiento, los mismos patrones en otro conjunto de entradas diferente.

El entrenamiento de una red neuronal depende de su algoritmo de entrenamiento, donde a grandes rasgos, se puede diferenciar entre entrenamiento supervisado, que es el comentado en este informe y el no supervisado, donde el entrenamiento se realiza sin un conjunto de testeo, es decir, su aprendizaje no depende de una referencia dada.

Fig. 2: Composición de una neurona.

En una red neuronal las distintas neuronas se encuentran agrupadas por capas, formando una red multicapa mediante la conexión de las distintas capas. En función de la ubicación de la capa dentro de la red pueden distinguirse varios casos:

- Capa de entrada: La primera capa donde se introducen las entradas.
- Capa oculta: Capas intermedias entre la capa de entrada y la capa de salida.
- Capa de salida: La correspondiente a la última capa que proporciona la salida de la red.

En este sistema, la topología de la red consta de 4 capas como se indica en la Tabla II. La primera contiene 30 neuronas coincidiendo con el número de entradas, aunque cada neurona recibe todas las entradas. Las capas intermedias contienen 12 y 5 neuronas siguiendo la recomendación de estructura expresada en las formulas (2) y en la Tabla III. Finalmente en la capa de salida hay solamente dos neuronas, ya que sólo se necesitan dos bits para codificar cuatro elementos diferentes, en este caso los números del 1 al 4, ambos incluidos.

Topología Red Neuronal	Neuronas	Función de activación
Capa de entrada	30	logsig
Capa oculta 1	12	logsig
Capa oculta 2	5	purelin
Capa de salida	2	purelin

TABLE II: Topología de la Red Neuronal.

Parámetros	Definición
h_1	Nº de neuronas en la capa oculta 1
h_2	Nº de neuronas en la capa oculta 2
i	Nº de neuronas en la capa de entrada
o	Nº de neuronas en la capa de salida

TABLE III: Tabla adjunta para la recomendación de estructura interna.

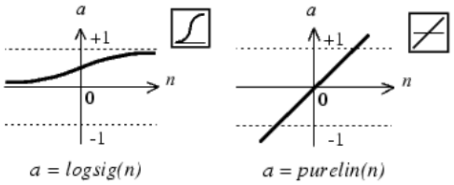
$$\begin{cases} h_1 = o \cdot r^2 \\ h_2 = o \cdot r \\ r = \frac{i}{o}^{\frac{1}{3}} \end{cases} \quad (2)$$

En relación a las conexiónes de una red, el diseño del sistema utilizado en este trabajo presenta una red feed forward caracterizada por tener las neuronas conectadas en un único sentido sin la existencia de bucles entre unas y otras. Otros tipos de red son por ejemplo las redes con conexiones en bucle en las que la información puede viajar de una capa a la anterior y de un capa a la posterior.

La función de activación se trata de una relación entrada/salida para poder proporcionar la no linealidad necesaria en la red, ya que si todas las funciones de las neuronas fuesen lineales, al final toda la red equivaldría a tener una única neurona, puesto que la suma de sumas lineales es también, a su vez, una suma lineal. Las funciones utilizadas en las capas de la red se muestran en la Tabla IV y en la Fig. 3.

Función	Nombre	Relación Entrada/Salida
purelin	Lineal	$a = n$
logsig	Sigmoidal Logaritmico	$a = \frac{1}{1+e^{-n}}$

TABLE IV: Funciones de activación.



(a) Función de activación logsig. (b) Función de activación purelin.

Fig. 3: Funciones de activación utilizadas.

Dado que el problema a resolver es un problema de clasificación, el algoritmo de entrenamiento empleado es un algoritmo de entrenamiento supervisado por descenso de gradiente conocido como *Back Propagation*. Este algoritmo permite el ajuste de los pesos de cada neurona mediante el cómputo de una función de coste que se minimiza en función de los pesos gracias al descenso de gradiente. La técnica de *Back Propagation* implica operar de forma recursiva moviendo el error desde la capa de salida hacia la de entrada y responsabilizando a cada neurona de un porcentaje del error en función de su implicación en el cómputo. Este procedimiento tiene como resultado un aumento de la precisión de la red en cada iteración.

El entrenamiento se lleva a cabo en el script `RedNeuronal1234.m` donde además se realiza una previa disposición de las entradas y conjunto de testeo correspondiente.

El programa carga los datos correspondientes a los coeficientes extraídos en el módulo 1. Para cada dígito se toman las 100 grabaciones y se eliminan el primer y último coeficiente, tomando exclusivamente los tres coeficientes intermedios. La matriz de entradas se conforma mediante la concatenación horizontal de las matrices de coeficientes de cada dígito, de forma que al final se obtiene una matriz de 400 columnas,

correspondientes a las grabaciones y 30 filas, correspondientes a las muestras de cada coeficiente. Por tanto, en cada iteración la red procesa 30 entradas. En la Fig. 4 se muestra en detalle esta matriz de entradas.

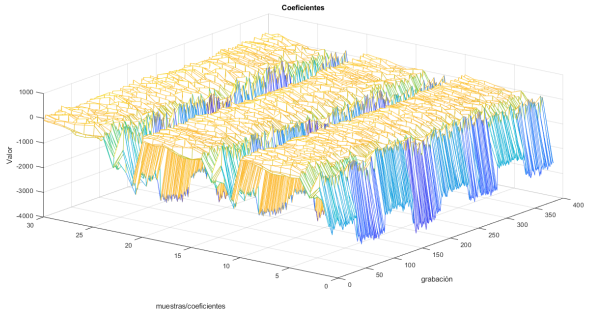


Fig. 4: Representación gráfica de los coeficientes cepstrales.

Por otra parte, el conjunto de testeo con el que se compararán las salidas de la red para computar la función de coste se disponen en una matriz de 400 columnas (una por cada conjunto de entradas en una iteración) y dos filas, ya que la red tiene dos salidas. Si se codifican los dígitos como se indica en la Tabla V y las grabaciones están dispuestas en el mismo orden lógico de los números, es decir, las 100 primeras corresponden número uno, las 100 siguientes el número dos, a continuación 100 grabaciones del número tres y finalmente las 100 del número 4; entonces la forma de la matriz de testeo toma la forma mostrada en la Fig. 5.

Dígito	Salida
1	00
2	01
3	10
4	11

TABLE V: Salidas asociadas a cada dígito.

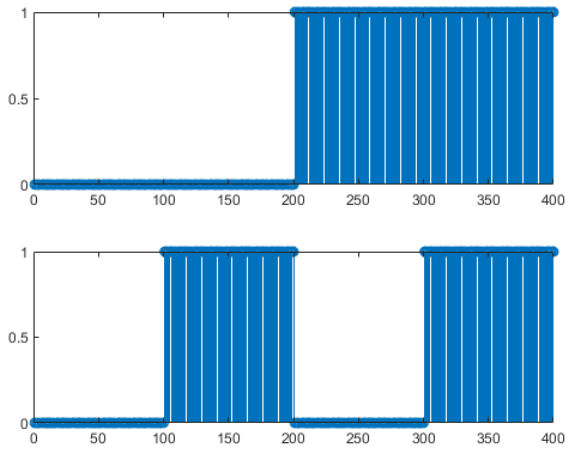


Fig. 5: Representación gráfica de las salidas asociadas a cada dígito.

Cabe destacar que el entrenamiento de la red para este volumen de datos ha requerido varias horas, terminando con los resultados expresados en Fig. 6, Fig. 7 y Fig. 8 .

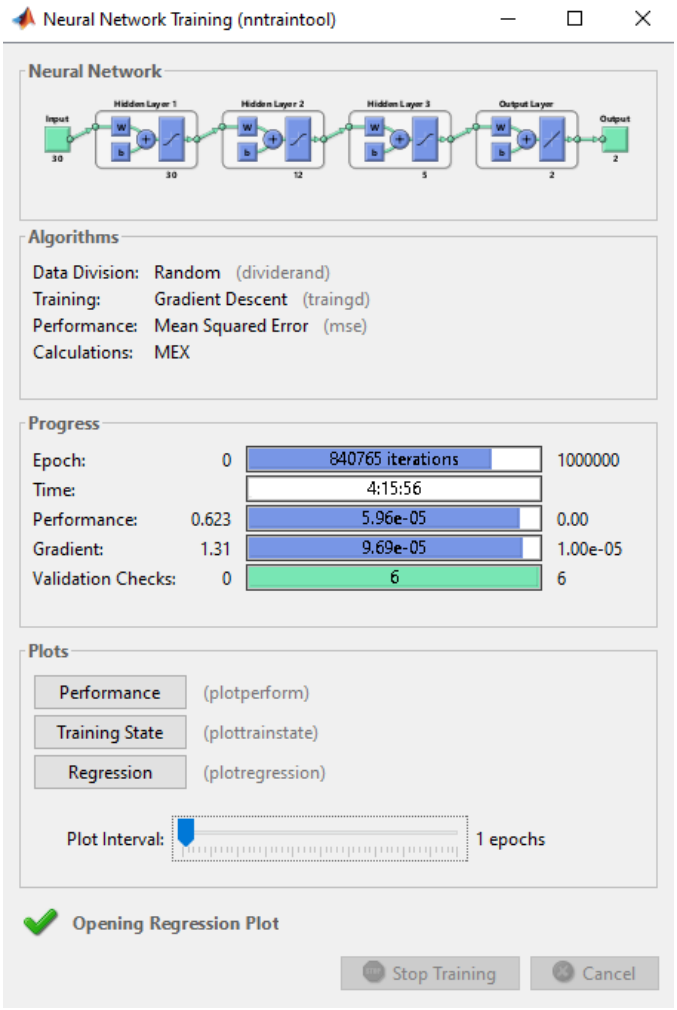


Fig. 6: Cuadro de control del entrenamiento.

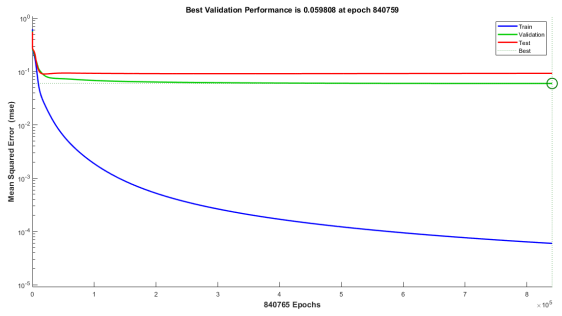


Fig. 7: Error cuadrático.

IV. Módulo 3. Evaluación de la red

Este último módulo permite evaluar la red una vez entrenada. Esto se lleva a cabo mediante el programa `tester.m`,

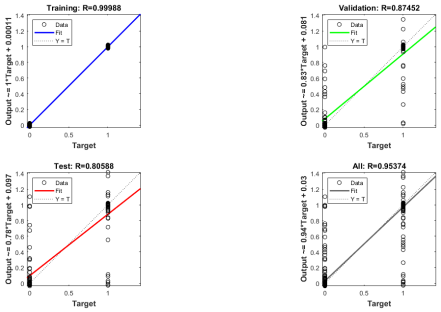


Fig. 8: Representación de las regresiones en las 4 capas.

el cual solicita al usuario un dígito del 1 al 4 reconociendo que número ha sido pronunciado. Para ello los parámetros utilizados se corresponden con los de la Tabla I.

La metodología para grabar y procesar la señal es idéntica a la usada en el módulo 1, obteniendo para cada grabación un vector de 5 coeficientes, de los cuales finalmente se consideran solamente los tres intermedios. Este vector de 30 muestras conforma la entrada a la red. La red procesa una respuesta en relación a estas entradas y devuelve un valor decimal entre 0 y 1. Dada la naturaleza de las funciones de activación, finalmente es necesario someter las salidas de la red a una capa de redondeo adicional que convertirá el valor decimal de la respuesta dada por la red a un valor binario.

La última tarea del programa consiste en traducir esta respuesta binaria a una respuesta en texto legible para un humano, identificando de esta forma el dígito pronunciado.

V. Conclusiones

El diseño se implementa para el reconocimiento de cuatro dígitos, sin embargo el sistema resulta perfectamente escalable. Esta escalabilidad hace referencia a la capacidad del sistema para procesar datos de un mayor número de dígitos a parte de los ya especificados. Para conseguir esta capacidad se requiere un cambio en la topología de la red, así como la disposición del conjunto de testeo y un aumento en las capacidades de computo de los equipos usados para el entrenamiento, el resto del sistema está preparado para realizar las grabaciones y procesado correspondientes.