Universidad de Alcalá
Escuela Politécnica Superior

Grado en Ingeniería Informática

Trabajo Fin de Grado

Estudio de técnicas de Deep Learning aplicadas a la clasificación de señales electroencefalográficas

Autor: Javier Madrid González

Tutor: José Luis Martín Sánchez

2023

UNIVERSIDAD DE ALCALÁ

ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería

InformáticaTrabajo Fin de

Grado

Estudio de técnicas de Deep Learning aplicadas a la clasificación de señales electroencefalográficas

Autor: Javier Madrid González

Tutor: José Luis Martín Sánchez

Tribunal:

Presidente: Alfredo Gardel Vicente

Vocal 1º: Almudena López Dorado

Vocal 2º: José Luis Martín Sánchez

Suplente: Fº Javier Dongil Moreno

Fecha de depósito: 22 de marzo de 2023

Agradecimientos.

A José Luis Martín Sánchez, mi tutor, por haberme guiado y marcado el camino a seguir durante la realización de este proyecto, a mi familia y amigos por haberme apoyado durante este periodo de mi vida y a las personas que me han proporcionado los distintos registros que he utilizado.

Gracias

Resumen.

Este proyecto consiste en el entrenamiento de una red neuronal convolucional con la actividad cerebral de distintas personas, para posteriormente descomponer y clasificar dicha actividad cerebral en tipos de tareas mentales que el usuario está realizando en ese momento, como mover los distintos miembros del cuerpo humano, ya sean manos, pies, brazos, etc. [1]. Los registros que contienen dicha actividad cerebral serán interpretados y decodificados por un programa realizado en MATLAB cuya función es la comentada anteriormente, el entrenamiento de una red neuronal y la posterior clasificación de los datos que pertenecen a los registros electroencefalográficos.

Palabras clave: Red neuronal convolucional, MATLAB.

Abstract.

This project consists of training a convolutional neural network with the brain activity of different people, to later break down and classify said brain activity into types of mental tasks that the user is performing at that moment, such as moving the different limbs of the human body, be it hands, feet, arms, etc. The records containing said brain activity will be interpreted and decoded by a program made in MATLAB whose function is the one mentioned above, the training of a neural network and the subsequent classification of the data belonging to the electroencephalographic records.

Keywords: Convolutional neural network, MATLAB.

Resumen Extendido.

Este proyecto es una propuesta para una posible forma de clasificación de los distintos pensamientos que el cerebro humano puede desarrollar ya sea motriz, intelectual, artístico, etc. Este proyecto se centra específicamente en el movimiento de las partes del cuerpo humano. Para ello se necesitarán dos componentes: los registros electroencefalográficos provenientes de bancos de señales y la una red neuronal convolucional.

Los registros que se van a utilizar en este proyecto provienen de dos bancos de datos distintos: uno de estos bancos de datos proviene de la Universidad de Alcalá y el segundo banco de datos que se va a utilizar va a ser del sitio web de PhysioNet. El primero de estos, contiene señales EEG que corresponden a la imaginación del movimiento de las manos derecha e izquierda y en el segundo banco de datos se usa también el movimiento de los pies.

En cuanto a la red neuronal convolucional que se va a utilizar es la que se conoce con el nombre de AlexNet. Esta red ya se encuentra previamente entrenada. Se va a utilizar una red preentrenada para reducir el tiempo de entrenamiento.

Como esta red está diseñada para el análisis y clasificación de imágenes, en este proyecto, se ha tenido que modificar la estructura y las capas de la red para que sea capaz de analizar señales EEG y para su posterior clasificación.

Tabla de Contenido.

Índice de Figuras.

Glosario de Acrónimos y Abreviaturas.

RAM	Random access memory	Memoria de acceso aleatorio
BCI	Brain-computer interface	Interfaz cerebro-computador
EEG	Electroencephalography	Electroencefalografía
FFT	Fast Fourier Transformation	Transformada de Fourier Rápida
CNN	Convolutional Neural Network	Redes Neuronales Convolucionales

Capítulo 1.

1.Introducción.

1.1 Presentación.

Actualmente una gran parte de las personas que se dedican a la medicina hacen uso de las interfaces cerebro-computador ya que son muy útiles a la hora de comunicarse con los distintos dispositivos electrónicos y también son de gran ayuda a la hora de analizar el comportamiento del cerebro humano. Gracias a los distintos avances en la tecnología, los dispositivos han evolucionado, lo que hace que estos dispositivos sean más intuitivos de utilizar y las personas se han adaptado a estos avances tecnológicos.

Este proyecto utiliza las interfaces BCI para realizar un clasificador de señales EEG, por lo tanto, este proyecto se centra en el uso de las redes neuronales para la extracción de características de los EEG.

1.2 Motivación.

La principal motivación de la realización de este proyecto es la de evaluar la aplicación de las redes neuronales convolucionales junto a la aplicación de técnicas de Deep Learning para el diseño de interfaces BCI.

Seguidamente, otra de las motivaciones más importantes de este proyecto, no solo viene dada por la extracción de las principales características de un EEG para identificar las distintas tareas mentales que el cerebro humano puede realizar, si no que, si este trabajo se le añade más potencia de cálculo y más datos de sujetos, este trabajo se puede aplicar a la detección de características de determinadas enfermedades que las personas pueden llegar a padecer.

Otra de las motivaciones que se podría plantear en este trabajo es la posible identificación de cuál de los hemisferios del cerebro una persona tiene más desarrollado y así poder distinguir si una persona posee una mentalidad más creativa (ya que la creatividad de una persona está más ligado al hemisferio derecho) o si esa persona tiene una mentalidad más analítica o científica (la capacidad de análisis está localizado en el hemisferio izquierdo del cerebro).

1.3 Objetivos.

En cuanto al principal objetivo de este trabajo es el siguiente: A partir de un banco de datos de señales EEG, cuyos datos pertenecen al movimiento de ambas manos y ambos pies, entrenar una red neuronal convolucional capaz de clasificar las distintas tareas haciendo uso de técnicas de Deep Learning. Para la clasificación de estas tareas, el primer paso es la extracción de las características de las señales EEG. Con la extracción de las características, se puede entrenar la red neuronal con los datos necesarios para la correcta clasificación de las distintas tareas.

Como segundo objetivo de la implementación de este trabajo es que en un futuro sea capaz de clasificar otras tareas sencillas.

1.4 Aplicaciones.

Como se ha comentado anteriormente una de las posibles aplicaciones de este trabajo es en área de la medicina con la detección de patrones anormales en los EEG como la detección de enfermedades cerebrales.

1.5 Estructuración del proyecto.

En este apartado de la memoria se mostrará el flujo de trabajo que se ha realizado para el desarrollo del proyecto. Las etapas son las siguientes:

- **Etapa 1**. En esta primera etapa del proyecto se abordó la instalación del software necesario. En este caso se ha descargado el software de MATLAB.
- **Etapa 2**. Durante esta etapa se estudiaron los dos bancos de muestras que se han utilizado en este proyecto (la base de datos de la UAH y la base de datos de PhysioNet). A su vez se ha realizado un estudio acerca de los patrones cerebrales de las señales EEG y las características de las tareas simples que se van a procesar.

  Además, se han estudiado los fundamentos del Deep Learning para aplicarlos en el proyecto.
- **Etapa 3**. En esta fase se realizó un estudio sobre las distintas tareas que componen a los bancos de registros. Estas tareas son las siguientes:
  - **Movimiento de la mano izquierda.**
  - **Movimiento de la mano derecha.**
  - **Movimiento de ambos pies.**
- **Etapa 4**. En esta etapa se estudió y se diseñó la red neuronal encargada de extraer las características de las señales EEG, para su posterior clasificación.
- **Etapa 5**. En esta última etapa del proyecto se redactó la memoria con todo lo que se ha realizado, junto a las conclusiones y las posibles mejoras futuras.

1.6 Estructuración de la memoria.

En este apartado, se explicará la estructuración y organización de la memoria para ofrecer un mayor entendimiento y comprensión acerca del funcionamiento de este proyecto.

- **Bloque de teoría**
  En este bloque se explicará el funcionamiento del cerebro y las características que se pueden observar en las señales EEG. Seguidamente se explicará el funcionamiento de las CNN y del Deep Learning para la extracción de las características de las señales EEG que permitirán obtener un modelo para la clasificación de señales.

- **Bloque práctico**
  Seguidamente en este bloque se explicarán los resultados obtenidos al procesar los distintos datos pertenecientes a las señales EEG. Este bloque finalizará con una explicación de los resultados obtenidos.

- **Conclusión**
  Finalmente, se presentará un conclusión sobre el proyecto, y el impacto que tendría en la sociedad.

Capítulo 2.

2.Interfaces BCI.

2.1 Introducción.

Durante esta parte del proyecto, se hablará sobre las interfaces cerebro-computador, el cerebro y las señales que se producen en él, y la forma en la que se pueden visualizar estas señales junto a sus características. Seguidamente se explicará cómo se pueden usar dichas características para interpretar tareas sencillas, como la imaginación del movimiento de las manos derecha e izquierda y de los pies.

2.2 BCI.

Una BCI es un sistema que permite una comunicación directa entre el cerebro humano y un ordenador. Cuando las neuronas se comunican entre sí, estas generan actividad eléctrica. Esta actividad eléctrica se procesa y se decodifica en forma de comandos que son enviados a un dispositivo para llevar a cabo la tarea deseada.

La forma de adquisición de la actividad cerebral se puede realizar de dos formas:

- **Dispositivos invasivos**: estos dispositivos realizan la toma de datos directamente desde el cerebro del usuario, por lo que el usuario debe ser intervenido quirúrgicamente.
- **Dispositivos no invasivos**: son aquellos dispositivos que se colocan en el cuerpo humano sin realizar ninguna intrusión. Estos dispositivos son los que más se utilizan en la tecnología BCI y los más seguros.

2.3 El Cerebro.

El cerebro se puede definir como un órgano complejo, ubicado dentro del cráneo, que gestiona la actividad del sistema nervioso. Forma parte del Sistema Nervioso Central (SNC) y constituye la parte más voluminosa y conocida del encéfalo.

El cerebro se centra sobre todo en el control y la regulación de la gran mayoría del cuerpo y de la mente, como: el control de las funciones vitales controla los movimientos, procesa la información que se obtiene a través de los sentidos, etc.

En el cerebro se pueden distinguir dos hemisferios, separados por un surco, el lado izquierdo, o el hemisferio izquierdo, del cerebro predomina frente al lado derecho cuando se trata del razonamiento y del lenguaje, así como el cálculo matemático y las habilidades científicas. Por otro lado, el hemisferio derecho es más influyente a la hora de la imaginación, el arte, la música, etc. Es decir, el hemisferio derecho está más ligado al pensamiento creativo.

En cuanto a las capacidades motrices, el hemisferio izquierdo, se encarga de controlar el movimiento de todos los miembros del cuerpo de la parte derecha, y en cuanto a la parte derecha del cerebro, se encarga de la parte izquierda del cuerpo. Esto quiere decir que el hemisferio derecho controla la parte izquierda del cuerpo y la parte izquierda del cerebro, la parte derecha del cuerpo.

Estos hemisferios están compuestos por cuatro lóbulos: frontal, parietal, temporal y occipital.

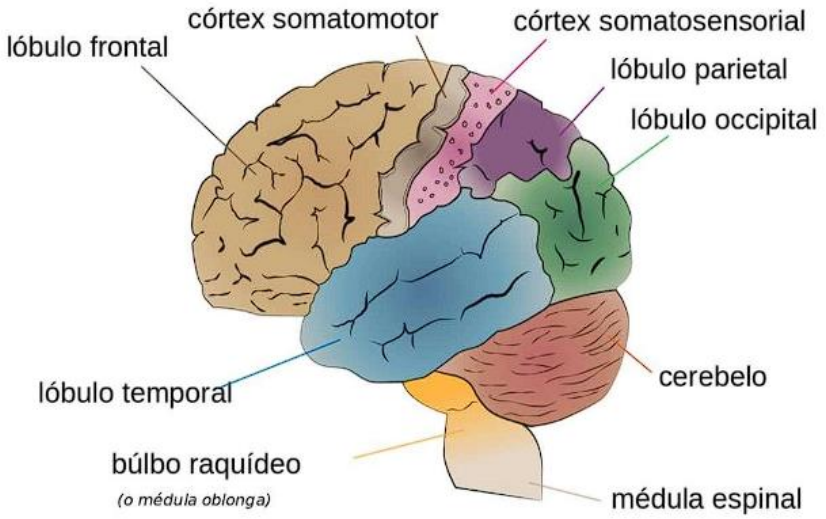


Figura 1. Cerebro. Fuente ¿Cuáles son las partes del cerebro?

Las principales funciones de los distintos lóbulos son las siguientes:

- **Lóbulo frontal**: se encarga de los movimientos voluntarios y es donde se moldean las reacciones emocionales y las expresiones.
- **Lóbulo parietal**: el centro del gusto está ubicado en esta zona.
- **Lóbulo temporal**: es la parte del cerebro donde se procesan los sonidos que escuchamos. Esta zona también es fundamental para el aprendizaje, la memoria y para sentir emociones.
- **Lóbulo occipital**: su función es la de analizar la información visual de la retina. Si estos lóbulos se dañan, la persona podría quedarse ciega.

La estructura propia del cerebro está conformada por varios millones de células llamadas neuronas, estas células son la unidad anatómica y funcional independiente. La interconexión de las neuronas nos permite razonar para experimentar sentimientos y para comprender el mundo que nos rodea. Las neuronas se componen de dos partes: un cuerpo celular, del que le salen unas ramificaciones, conocidas como dendritas, que son las encargadas de recibir información de otras neuronas o de otras células nerviosas, y el axón, que es la encargada de conducir la información a otras neuronas.

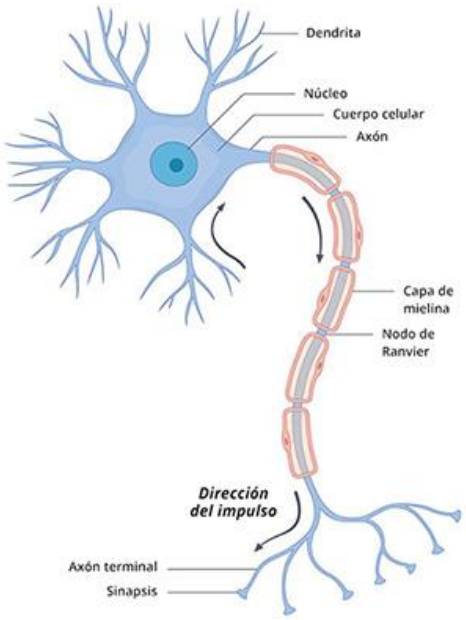


Figura 2. La neurona. Fuente ¿Cuáles son las partes del sistema nervioso?

La actividad cerebral se puede representar en forma de señales denominadas ritmos y la forma de distinguirlos e identificarlos se hace en función de la amplitud y de la frecuencia durante distintas actividades mentales. Los ritmos más significativos son los siguientes [2]:

- **Ondas Delta** (0.2-4 Hz): son ondas muy lentas, pero son las que mayor amplitud presentan. Estas señales son características cuando el individuo se encuentra en el sueño profundo, pero sin soñar. También se pueden observar estas ondas en estados de meditación.

Figura 3. Ondas de frecuencia de los ritmos Delta.

- **Ondas Theta** (4-8 Hz): estas ondas se pueden visualizar cuando los sentidos están procesando información. Son muy importantes durante el aprendizaje y memoria. También se asocian a capacidades imaginativas. Estas ondas muestran una gran actividad cuando se experimentan emociones muy profundas. Aparecen en la zona parietal o temporal.

Figura 4. Ondas de frecuencia de los ritmos Theta.

- **Ondas Alfa** (8-12 Hz): predominan cuando el Sistema Nervioso Central está en reposo, en estado de relajación, pero consciente, es decir, cuando hay actividad cerebral escasa. Estas ondas pueden llegar a desaparecer dependiendo de los sujetos, con la actividad mental o con la propia apertura de los parpados. Están ondas se originan en la parte posterior de la corteza cerebral.

Figura 5. Ondas de frecuencia de los ritmos Alfa.

- **Ondas Mu** (8-12 Hz): estas ondas se encuentran en el espectro de frecuencia de las ondas Alfa, pero son independientes a esta debido a su topografía, ya que es menos sinusoidal que los ritmos alfa. Estas ondas están vinculadas a los sistemas sensoriales y motrices. [3]

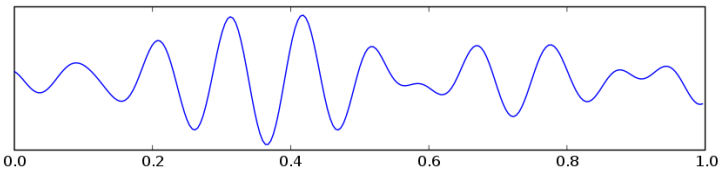


Figura 6. Ondas de frecuencia de los ritmos Mu.

- **Ondas Beta** (12-30 Hz): predominan durante el periodo de vigilia, cuando el cerebro está implicado en actividades mentales. Un exceso de estas ondas o una sobre-activación neuronal, puede derivar en un estado de ansiedad o estrés capaz de perjudicarnos. Cuando las personas ejecutan tareas que suponen el uso de la memoria, el cerebro usa estas ondas para ir cambiando entre las distintas partes de la información, es decir, el cerebro es capaz de decidir cuándo se lee o se descarta la información.

Figura 7. Ondas de frecuencia de los ritmos Beta

- **Ondas Gamma** (30-90 Hz): estas ondas son las más rápidas ya que se producen en ráfagas cortas. Se puede observar una mayor actividad de estas ondas cuando el cerebro está en estado de alta resolución o cuando se está en un momento de extrema atención o en momentos muy estresantes.

Figura 8. Ondas de frecuencia de los ritmos Gamma.

La forma de obtención y la visualización de todos estos ritmos, se realiza mediante técnicas no invasivas, como los electroencefalogramas, o EEG. Los EEG son una serie de estudios que miden la actividad eléctrica del cerebro mediante electrodos colocados sobre el cuero cabelludo.

Esta señal eléctrica que se ha recogido se amplifica y se representa en forma de líneas, interpretando la actividad de las distintas áreas del cerebro a lo largo del tiempo.

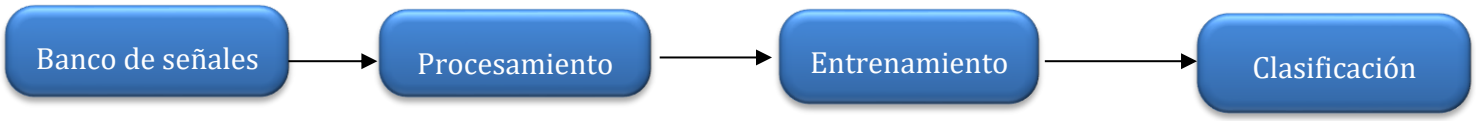
Como se ha indicado anteriormente, se quiere entrenar una red neuronal para que sea capaz de clasificar la intención motora de ambas manos. Dichas tareas se representan a través del lóbulo frontal. Dependiendo de la acción a realizar, se produce una cierta desincronización en ciertas partes del lóbulo frontal que serán utilizadas para entrenar a la red neuronal y posteriormente será capaz de detectar las siguientes tareas:

- **Primera tarea**: esta tarea consiste en imaginar el movimiento de la mano derecha que produce, en teoría, una desincronización de las ondas cerebrales en el lóbulo frontal izquierdo.
- **Segunda tarea**: la siguiente tarea consiste en imaginar el movimiento de la mano izquierda. La desincronización que se produce se puede visualizar en el lóbulo frontal derecho.
- **Tercera tarea**: esta última tarea consiste en la imaginación del movimiento de ambos pies. Cuando el sujeto sometido a la prueba tiene que realizar dicha tarea, se produce una desincronización de las ondas cerebrales en el lóbulo frontal central.

Capítulo 3.

3.Arquitectura del clasificador.

En este apartado de la memoria se explicará los distintos bloques que componen la arquitectura del clasificador.



3.1 Bloque de Banco de señales.

Durante el transcurso de este proyecto, se han utilizado dos bancos de señales ya citados anteriormente: uno que pertenece a la Universidad de Alcalá y el segundo banco de señales que publicado en el sitio web de PhysioNet. En ambos bancos de señales se analizarán la imaginación del movimiento de ambas manos y en el caso del banco de señales de PhysioNet se le incluirá el movimiento imaginario de ambos pies.

3.1.1 Banco de señales de la Universidad de Alcalá.

Para la adquisición de las señales EEG de los 12 sujetos que componen este banco de señales se ha utilizado un equipo de la compañía de g.tec. Este dispositivo consiste en un electroencefalógrafo de ocho canales que cuenta con electrodos amplificadores, el modelo más concretamente es el "Bsamp 8 Channels".

Estas señales han sido tomadas por el Dr. José Luis Martín Sánchez en la Universidad de Alcalá.

3.1.2 Banco de datos de PhysioNet.

El segundo banco de datos proviene de un repositorio del sitio web de medicina llamado PhysioNet. Este sitio web es un repositorio que almacena una gran cantidad de señales, ya sean EEG, electrocardiogramas o ECG y demás señales que se pueden obtener del cuerpo humano.

La forma en la que se han grabado las señales de los distintos paciente que componen a este banco, es mediante el sistema BCI2000 [4], de 64 canales. La forma de colocación de los electrodos sigue el sistema 10-10.

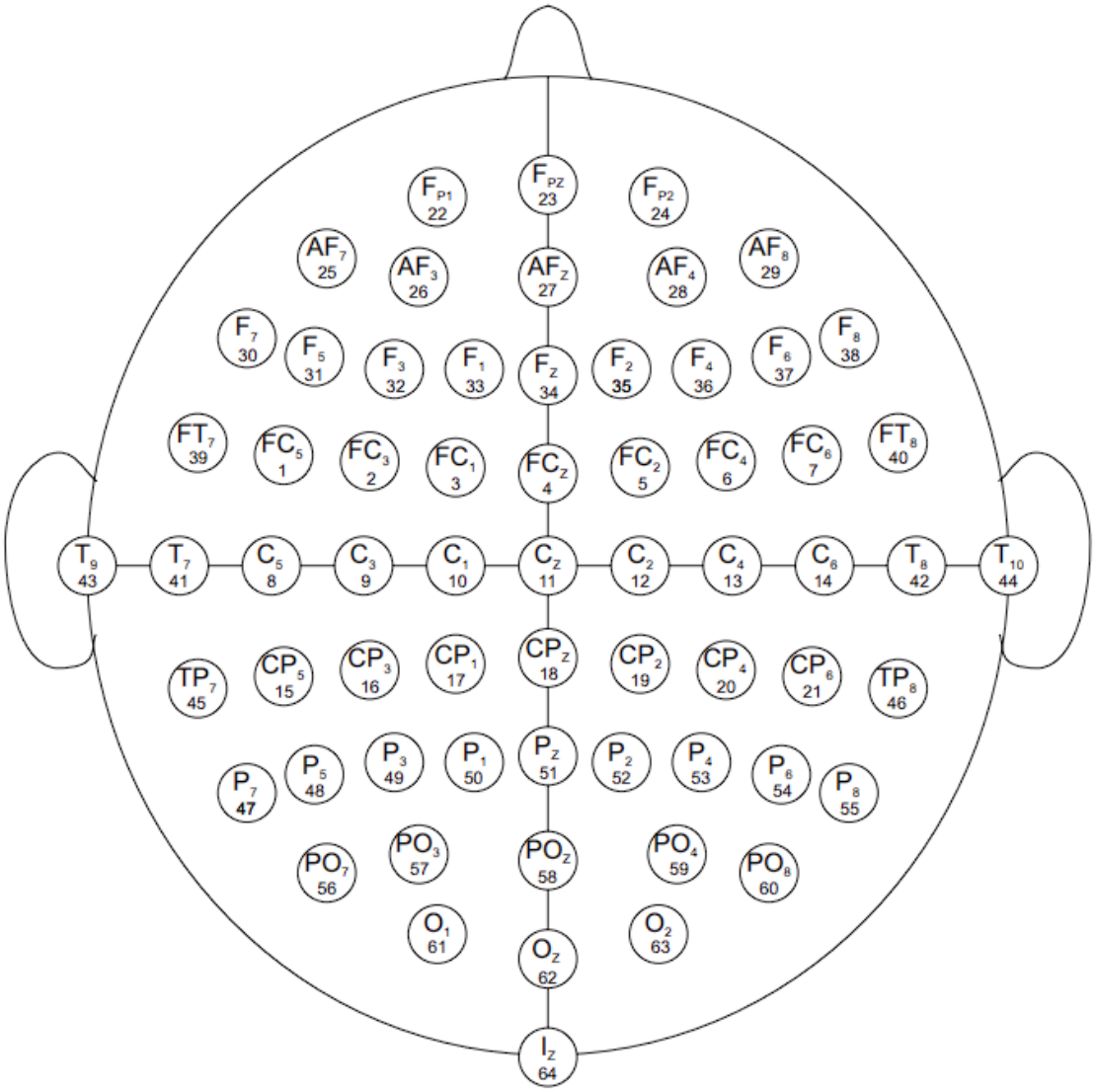


Figura 9. Colocación de los electrodos siguiendo el sistema 10-10

3.2 Bloque de Procesamiento de las señales.

Este bloque es el encargado de procesar las señales y permitirá extraer las características necesarias para que el clasificador sea capaz de elegir a qué acción se corresponden los datos procesados.

Primeramente, se utiliza un filtro paso banda para observar las señales en las frecuencias que se desea. La banda de frecuencia que queremos analizar está entre 8 y 30 Hz que corresponde a los ritmos Alfa, Beta y Mu, que son aquellos ritmos que aportan más información sobre la imaginación motriz.

Como se ha comentado anteriormente, cuando el sujeto imagina el movimiento de uno de los miembros del cuerpo que se van a analizar en este proyecto, produce la aparición o desaparición de componentes frecuenciales, sobre todo en los ritmos Mu y Beta, por lo que, como primera opción de procesamiento de las señales, se va a utilizar la transformada de Fourier para la extracción de las características iniciales.

Debido a las limitaciones de la transformada de Fourier, se va a utilizar otro método para extraer más información frecuencial y temporal de las distintas señales que se están analizando. Para ello se hará uso de la combinación de la transformada Wavelet junto a la transformada de Fourier. Esta combinación es muy interesante ya que potencia las componentes frecuenciales más relevantes y, al mismo tiempo, reduce el número de coeficientes empleados.

- **Transformada Rápida de Fourier**: es un algoritmo que permite calcular la Transformada de Fourier Discreta y su inversa. Es de suma importancia en el análisis, diseño y realización de algoritmos y sistemas de procesamiento de señales. Su expresión y la forma de calcularla es la siguiente:

$$X(k) = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} nk} \quad con\ k = 0,1,\dots,N-1$$

Figura 10. Transformada rápida de Fourier

  Si se utiliza con un enfoque médico, la Transformada Rápida de Fourier, nos permitirá obtener la distribución de la amplitud de espectro de las señales EEG y extraer el pico del espectro para reflejar las diferentes tareas del cerebro [5]. Mediante el uso de la FFT, la señal de EEG puede mapearse desde el dominio del tiempo al dominio de la frecuencia. Según el estudio de [6] la transformada de Fourier puede ser utilizada para la extracción de características en la clasificación de tareas motoras.

  Para extraer más características se debe hacer un análisis más detenido y exhaustivo. Mediante un estudio simultaneo en tiempo y frecuencia se conseguirán más características para la posterior clasificación, por ello se hará uso de la transformada Wavelet.

- **Transformada Wavelet**: es una herramienta matemática que tiene múltiples aplicaciones en el procesamiento de señales, también es utilizada en la detección de anomalías sintomáticas en medicina e ingeniería.

  La aplicación de la transformada Wavelet se realiza a través de la función llamada wavelet madre, con la que se descompone una señal en diferentes componentes de frecuencia. Estas modificaciones que sufre la wavelet madre están a cargo de los parámetros de escalamiento y desplazamiento. La adecuada selección de una wavelet madre y el número de niveles de descomposición son muy importantes en el análisis de señales EEG para encontrar niveles de clasificación promisorios [7].

  El escalamiento hace referencia a cuanto se alarga o se comprime la wavelet, lo que permite ver los detalles y los componentes de la señal. Mientras que el desplazamiento indica el

recorrido de la wavelet a lo largo de la señal.

La wavelet madre se define de la siguiente forma:

$$\psi_{\tau,a} = \frac{1}{\sqrt{a}}\psi\left(\frac{t-\tau}{a}\right)$$

Figura 11. Transformada Wavelet madre.

Donde a es el escalamiento y τ es el desplazamiento.

Resultados en EEG [8] han demostrado que el método propuesto con precisión podría extraer características sustanciales EEG y proporcionar un medio eficaz para clasificar las tareas mentales motoras.

Existen distintos tipos de transformada wavelet:

- **Transformada wavelet continua** (**CWT**), que se define como la suma de la multiplicación de una señal continua y la wavelet madre en su forma desplazada y escalada. Se define de la siguiente manera:

$$Wf(u,s) = \langle f, \psi_{u,s} \rangle = \int_{-\infty}^{\infty} f(t)\frac{1}{\sqrt{s}}\psi^*\left(\frac{t-u}{s}\right)dt$$

Figura 12. Transformada Wavelet continua.

- **Transformada wavelet discreta** (**DWT**), se obtiene a la hora de darle valores discretos a los parámetros de desplazamiento y escalamiento dentro de la CWT. Viene definida por la siguiente ecuación:

$$W_{m,n}f = s_0^{-\frac{m}{2}} \int_{-\infty}^{\infty} f(t)\psi\left(s_0^{-m}t - nu_0\right)dt$$

Figura 13. Transformada Wavelet discreta.

Después de explicar el funcionamiento de la transformada wavelet, se puede deducir que la transformada wavelet se comporta como un conjunto de filtros paso bajo y paso alto.

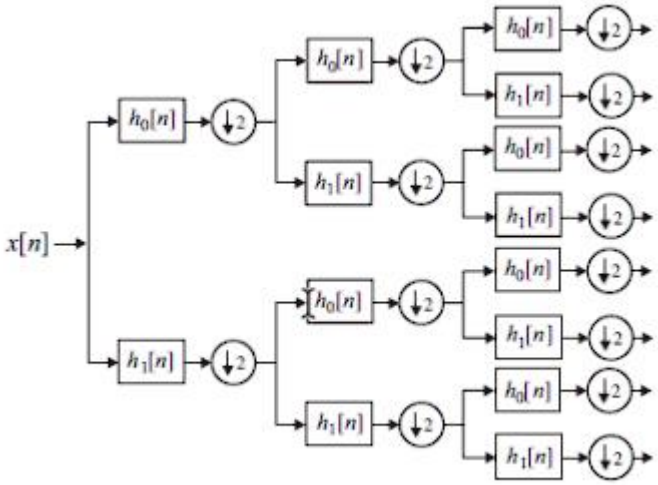


Figura 14. Wavelet Packet.

En nuestro caso se usará la descomposición wavelet para poder observar las señales en el dominio temporal en las bandas de frecuencia que se han definido y al mismo tiempo reducir el número de coeficientes a la hora de realizar la extracción de las características.

3.3 Bloque de Entrenamiento de la red neuronal.

Las redes neuronales son modelos simples que simulan la forma en la que el cerebro humano procesa la información. La unidad básica de estos modelos son las neuronas y estas neuronas se organizan en capas. Existen tres tipos de capas:

- **Capa de entrada**: esta es la capa en la que se indican los campos o valores de entrada para la red neuronal.
- **Capas ocultas**: puede haber una o varias capas de este tipo, estas son las que se encargan de procesar y tratar los datos.
- **Capa de salida**: esta capa contiene una o varias unidades que representan el campo o los campos de salida.

Las redes neuronales, suelen seguir el siguiente esquema:

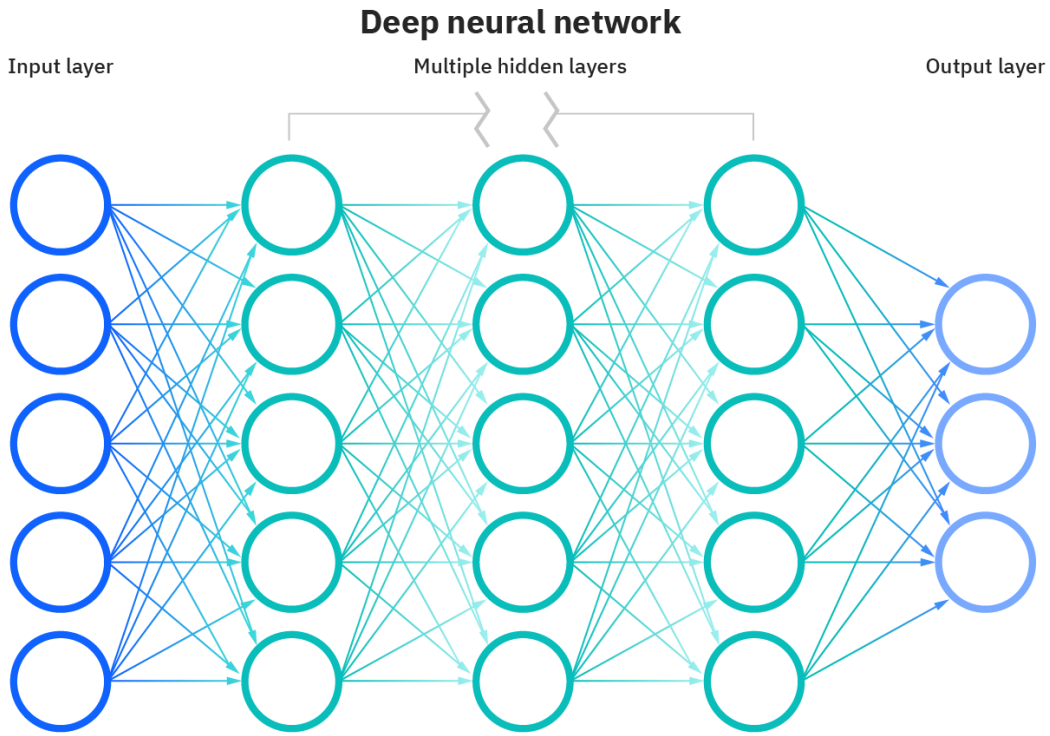


Figura 15. Estructura básica de las redes neuronales.

La red neuronal aprende a base de examinar los distintos registros individuales de un conjunto de entrenamiento, generando una predicción para cada registro y realizando una serie de ajustes en las ponderaciones de los pesos de las distintas capas cuando se realiza una predicción incorrecta.

El aspecto con el que más cuidado se debe tener a la hora de entrenar una red neuronal es el posible sobreajuste. El sobreajuste es el hecho de hacer un modelo tan ajustado a los datos de entrenamiento que haga que no generalice bien los datos de prueba ya que el principal objetivo de los modelos de aprendizaje autónomo es el de obtener patrones de los datos de entrenamiento disponibles de cara a predecir o inferir correctamente datos nuevos [9].

Hay varios tipos de redes neuronales como: el perceptrón, el perceptrón multicapa, las redes neuronales convolucionales y las redes neuronales recurrentes. En este trabajo, se hará uso de las redes neuronales convolucionales.

3.3.1 Redes Neuronales Convolucionales.

Estas redes son una variación del perceptrón multicapa. Las redes neuronales convolucionales utilizan el aprendizaje supervisado para identificar características en los datos de entrada. [10]

Estas redes son muy útiles para la visión artificial, como por ejemplo, en la clasificación y segmentación de imágenes.

La estructura de las redes neuronales convolucionales es la siguiente: constan de múltiples filtros convolucionales de una o más dimensiones que permiten la extracción de las características de la imagen o de la sucesión de datos que componen los datos de entrenamiento. Entre estos filtros suele haber una función de activación.

Para el correcto entrenamiento de estas redes se necesita una gran cantidad de datos, ya que estas redes neuronales necesitan encontrar un patrón que distinga a los distintos datos y así poder agrupar los datos que pertenecen al mismo conjunto de datos.

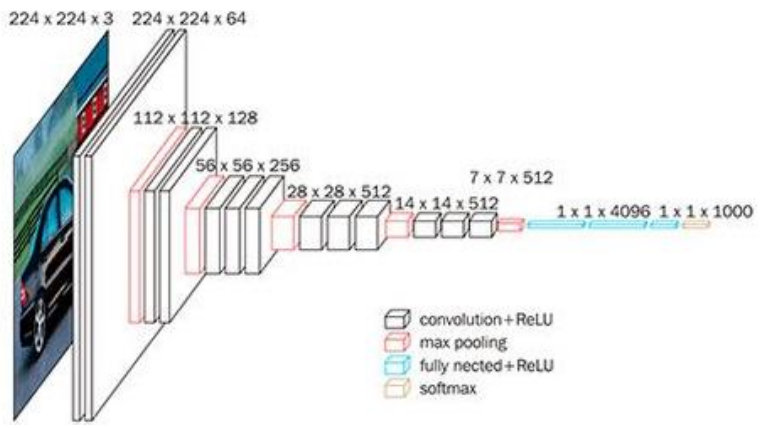


Figura 16. Estructura de las redes neuronales convolucionales.

Las redes neuronales convolucionales constan de dos partes fundamentales:

- **Kernel**: en las redes convolucionales se les consideran como los filtros que se aplican cuando se procesa una imagen, esto se hace para obtener ciertas características importantes o patrones de esta. Las características más importantes que se extraen al aplicar el kernel a la imagen son: detectar bordes, enfoque, desenfoque, etc.

Un ejemplo de aplicar un filtro a una imagen es la siguiente:

Figura 17. Imagen antes de aplicar el kernel

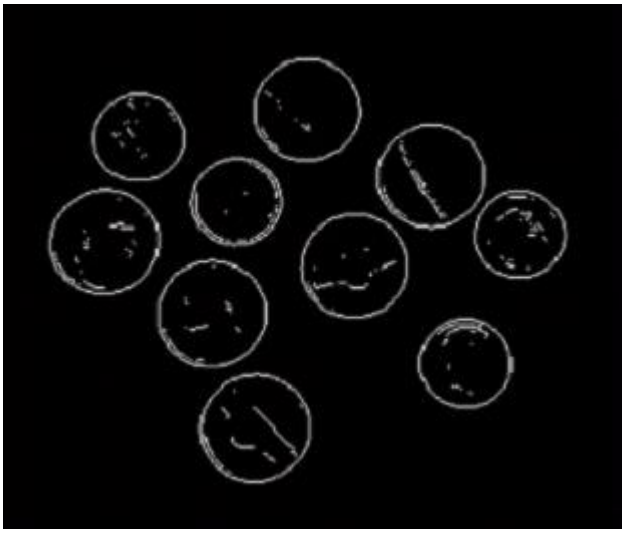


Figura 18. Imagen despues de aplicar el kernel

- **Convolución**: es una de las operaciones más importantes y distintivas de las redes neuronales convolucionales. El proceso consiste en tomar un conjunto de pixeles de la imagen de entrada y a continuación se realiza el producto escalar con un kernel. El kernel se encargará de recorrer todas la neuronas de entrada y se obtendrá una nueva matriz que se convertirá en una de las capas ocultas.

Después del entrenamiento de la red neuronal, esta, será el clasificador que se va a usar en el proyecto.

Este clasificador está compuesto por un modelo creado en el ámbito de la inteligencia artificial basado en técnicas de clasificación de Deep Learning. Seguidamente se explicará en qué consiste el Deep Learning.

Deep Learning o aprendizaje profundo es una técnica de aprendizaje automático basada en el modelo de red neuronal, en el que se apilan decenas o incluso cientos de capas de neuronas que aportan mayor complejidad al establecimiento de reglas. Los algoritmos que se utilizan en estas técnicas son capaces de imitar las acciones del cerebro humano, y son capaces de aprender y extraer características sin intervención humana. Gracias a esto, los sistemas basados en Deep Learning son capaces de procesar datos incluso sin que estén estructurados. [11]

Las distintas capas de la red neuronal que componen a estos sistemas analizan las nuevas entradas en busca de otras características, que el sistema utiliza para decidir como clasificar las entradas.

A diferencia del Machine Learning, o aprendizaje automático, que le basta con una base de datos manejable para funcionar, los sistemas que hacen uso de la tecnología de Deep Learning requieren de una gran cantidad de datos.

Después del pre-procesamiento, de la extracción de las características y del entrenamiento de la red neuronal, el clasificador de Deep Learnig es capaz de clasificar los datos de las señales que no han sido utilizados para el entrenamiento de la red neuronal.

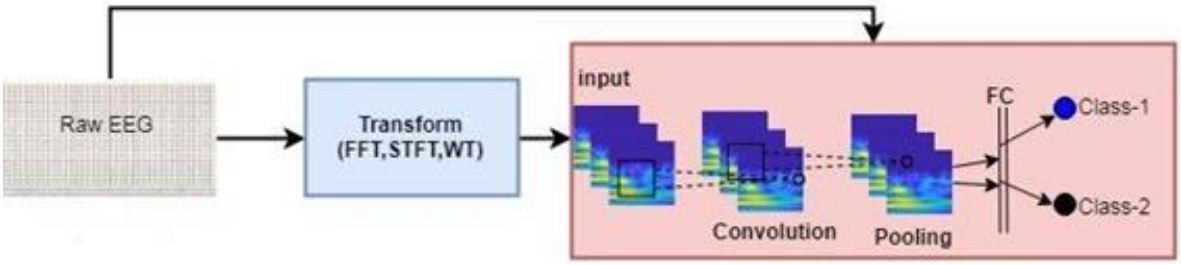


Figura 19. Secuencia de la clasificación

Capítulo 4.

4.Desarrollo del clasificador.

4.1 Etapas de desarrollo del clasificador.

Para poder desarrollar un clasificador con técnicas de Deep Learning adecuado, se ha seguido un flujo de trabajo dividido en varias etapas.

4.1.1 Etapa de calibración y de entrenamiento.

Una de las mayores dificultades que hay a la hora de trabajar con las señales EEG es la gran complejidad de estas señales, ya que las señales son distintas dependiendo de la persona y del momento en el que se realiza la adquisición de las muestras. Para la realización del clasificador, se debe calibrar la herramienta encargada de obtener las señales EEG, y a su vez los sujetos sometidos a las distintas tareas previamente deben de estar entrenados.

En el caso de este proyecto, no ha hecho falta entrenar a los sujetos, ya que los datos de las señales EEG provenían de sujetos ya entrenados.

4.1.2 Etapa de diseño.

Durante esta etapa se han recolectado las muestras que se han adquirido en la anterior etapa. Seguidamente, se ha realizado un filtrado. Esto es importante y muy útil para quedarnos únicamente con las muestras de las tareas que queremos utilizar para que nuestro clasificador sea capaz de clasificar de la forma más precisa las tareas, ya que, si no se hace este filtrado, el clasificador utilizaría datos de tareas que no nos interesan y no sería del todo preciso.

4.2 Herramientas de trabajo para el diseño del clasificador.

En este apartado se realizará una breve descripción acerca del Software y del Hardware que se ha utilizado para el desarrollo del clasificador.

4.2.1 Hardware.

Ordenador. Para la realización del clasificador y del entrenamiento de la red neuronal se ha hecho uso de un ordenador sobremesa, con las siguientes especificaciones:

- Ryzen 5 5500. 3,6 GHz.
- RAM: 16GB
- Disco duro M.2: 500 GB.
- Disco duro: 2 TB.

4.2.2 Software.

MATLAB: para el desarrollo de todo este proyecto se ha hecho uso del entorno de desarrollo de MATLAB. Se ha usado para el procesamiento de las señales, diseño de las capas de la red neuronal, entrenamiento de esta y finalmente la clasificación de las señales. Para realizar todas estas tareas se han descargado las siguientes Toolbox: Deep Learning Toolbox, Deep Learning Toolbox Model for AlexNet Network y una Toolbox que se encarga de leer los archivos con extensión ".edf".

Para la customización de la red AlexNet para que sea capaz de trabajar con secuencias de datos, se ha utilizado la API Deep Network Designer.

4.3 Diseño e implementación del clasificador.

4.3.1 Señales.

Tal y como se ha comentado en el capítulo x, este proyecto va a utilizar las señales EEG provenientes de dos bancos de señales. En este apartado se va a describir detalladamente la forma en la que se han adquirido los datos de las señales y las tareas que han tenido que realizar los usuarios.

4.3.1.1 Banco de datos de la UAH.

Este banco de datos está compuesto por señales doce personas diestras: diez hombres y dos mujeres, en total se compone de sesenta muestras.

La forma en la que se han adquirido los datos ha sido de la siguiente forma: el sujeto es sometido a una secuencia temporal: durante los dos primeros segundos el ordenador permanece con la pantalla en negro y el paciente en reposo. En el instante t = 2s, suena un pitido y se dibuja una cruz en la pantalla del ordenador, esto quiere decir que la prueba está a punto de empezar. Para finalizar en el instante t = 3s, se muestra en la pantalla una flecha apuntando a la izquierda o a la derecha de la pantalla, indicando que el sujeto tiene que imaginar el movimiento de esa mano desde ese instante hasta el instante t = 9s.

En nuestro caso, se tomará como datos validos los datos capturados del instante t = 4s hasta el instante t = 9s. La frecuencia de adquisición de los datos que componen a las señales es de 128 Hz.

Las tareas realizadas durante este experimento han sido las siguientes:

- **Tarea 1**: cuando la flecha que aparece y apunta a la derecha, le indica al sujeto que tiene que imaginar el movimiento de la mano derecha.
- **Tarea 2**: si la flecha al aparecer apunta hacia la izquierda, el sujeto debe imaginar el movimiento de la mano izquierda.

En cuanto a los canales de adquisición en este banco de datos, únicamente están disponibles los canales C3, Cz y C4.

Para la mano izquierda, el canal que más información nos puede llegar a aportar es el canal C4, mientras que para la mano derecha el canal más representativo es el canal C3. Finalmente, el canal Cz únicamente nos va a aportar información poco relevante para la imaginación del movimiento de ambas manos.

4.3.1.2 Banco de datos de PhysioNet.

El banco de señales de physionet está compuesta por 1500 señales de 109 sujetos, y cada uno de estos sujetos tiene catorce muestras. La frecuencia de adquisición de los datos de las señales es de 160 Hz.

De todos los canales de adquisición únicamente se utilizarán los canales C3, Cz y C4 ya que estos tres canales son los que más información aportan sobre la intención motora.

Las tareas que se han realizado durante la adquisición de los datos son las siguientes:

- **Primera tarea**: aparece un objetivo a la derecha o a la izquierda de la pantalla. El sujeto abre y cierra el puño correspondiente hasta que el objetivo desaparece, seguidamente el sujeto se relaja.
- **Segunda tarea**: es lo mismo que la primera tarea, pero en vez de abrir y cerrar el puño, el sujeto imagina que está realizando esas acciones. Cuando el objetivo desaparece, el sujeto se relaja.
- **Tercera tarea**: en esta tarea, aparece un objetivo en la parte de arriba o en la parte de debajo de la pantalla. Entonces, el sujeto abre y cierra ambos puños (si el objetivo está en la parte de arriba) y en el caso de que aparezca el objetivo en la parte de abajo, el sujeto abre y cierra ambos pies. Cuando el objetivo desaparece, el sujeto se relaja.
- **Cuarta tarea**: es muy parecida a la tercera tarea, pero está vez el sujeto se imagina las acciones de abrir y cerrar los puños y los pies. Cuando el objetivo desaparece, el sujeto se relaja.

Las muestras contienen anotaciones que determinan que tareas se están desarrollado en ese momento, esas anotaciones constan de tres códigos (T0, T1 o T2):

- **T0**: corresponde a cuando el sujeto se encuentra en reposo.
- **T1**: corresponde al movimiento del puño izquierdo, ya sea real o imaginario, en las sesiones: 3, 4, 7, 8, 11 y 12. También puede denotar el movimiento de ambas manos en las sesiones: 5, 6, 9, 10, 13 y 14.
- **T2**: corresponde al movimiento del puño derecho, ya sea real o imaginario, en las sesiones: 3, 4, 7, 8, 11 y 12. A su vez, significa el movimiento de ambos pies en las sesiones: 5, 6, 9, 10, 13 y 14.

En el diseño del clasificador de este proyecto, solo se harán uso de los eventos del movimiento imaginario de las manos derecha e izquierda y el movimiento imaginario de los pies, ya que no se pretende entrenar a la red neuronal con datos del estado de relajación ni del movimiento de ambos puños al mismo tiempo.

Como se ha comentado al inicio de este subapartado, de los 64 canales únicamente utilizaremos tres, C3, Cz y C4.

Al igual que en anterior banco de datos, el canal C3, nos va a aportar más información sobre la imaginación del movimiento de la mano derecha, el canal C4 para la mano izquierda y el canal Cz, nos va a aportar información poco relevante para el movimiento de las manos.

En cuanto a la imaginación del movimiento de los pies, el canal que más información nos puede aportar es el Cz, mientras que los canales C3 y C4 no nos aportan información.

4.3.2 Preprocesamiento y extracción de características.

La etapa de preprocesamiento se basa en el filtrado de las señales en las bandas de frecuencia correspondientes a los ritmos Alfa, Mu y Beta que son los que se quieren observar. Una vez se han filtrado las señales, se les aplica la transformada wavelet para así reducir el número de coeficientes, y descomponer la señal y finalmente, a la señal resultante se le aplica la transformada de Fourier.

Este preprocesamiento se va a aplicar parciamente en los dos bancos de datos, es decir, en el primer banco de datos, el perteneciente a la UAH, únicamente se le va a aplicar la transformada de Fourier, mientras que a la base de datos de PhysioNet se le va a aplicar la transformada wavelet junto a la transformada de Fourier. Esto se debe a que este banco de señales tiene un mayor número de muestras y de sujetos que la otra base de datos. El código que se encarga del procesamiento y la extracción de las distintas características se ha realizado en MATLAB y se adjunta en el anexo.

4.3.3 Red neuronal.

A su vez, tal y como se ha comentado en el capítulo 3, este proyecto va a hacer uso de una red neuronal convolucional. Al principio se pensó crear desde cero una red neuronal convolucional, pero debido al tiempo de construcción de una red neuronal consistente y al tiempo de entrenamiento que suponía al configurar una nueva red neuronal, se decidió seleccionar una red neuronal ya previamente configurada y entrenada.

La red neuronal que se va a utilizar en este proyecto se le conoce como AlexNet. Esta red neuronal ganó el desafío de acreditación visual a gran escala de Imaginet en 2012, este modelo de red neuronal fue inventada por Alex Krizhevsky. [12]

Esta red está compuesta por ocho capas capaces de aprender. Consta de cinco capas con una combinación de tres capas absolutamente conectadas y usan la activación Relu en cada una de estas capas, excepto en la capa de salida.

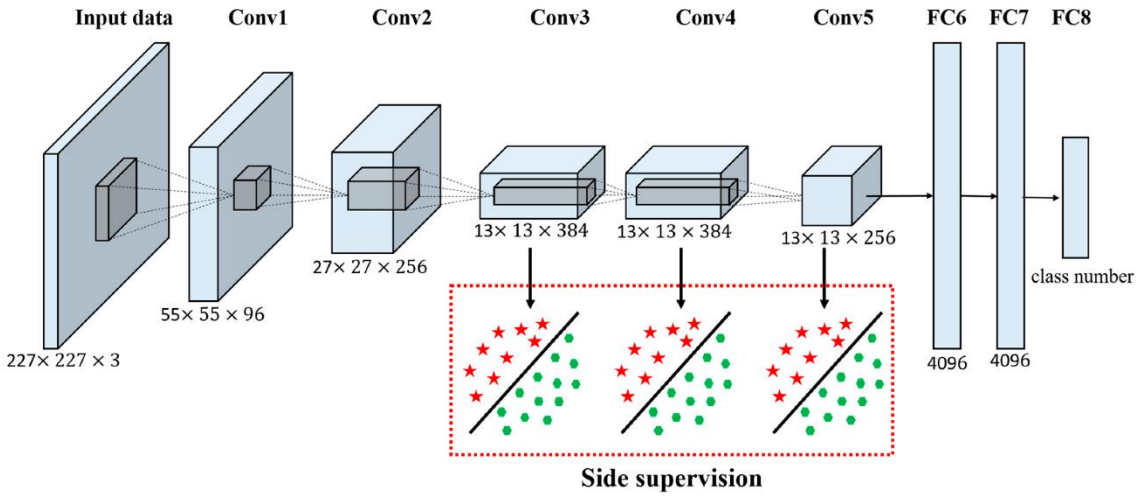


Figura 20. Estructura de la red neuronal AlexNet.

Todas las redes neuronales constan de al menos una función de activación, esto depende del número de capas que posea la red neuronal. AlexNet hace uso de la función de activación Relu. La función Relu, es una función lineal cuya salida es directamente los datos de entrada si estos son positivos y sino la salida de la función es cero.

Se descubrió que el uso de esa función de activación hizo que acelerase la velocidad de entrenamiento casi seis veces. Además, se utilizaron las capas de abandono que impidieron que su modelo se sobreajuste.

Tal y como se ha comentado al inicio de este subapartado, AlexNet es una red neuronal que solo permite como datos de entrada imágenes, y estas imágenes son procesadas en las distintas capas ocultas. El problema en nuestro caso es que los datos de entrada no corresponden a imágenes, sino que son datos numéricos pertenecientes a señales EEG. Por lo tanto, se ha tenido que modificar la estructura de AlexNet, cambiando una serie de bloques que utilizan dos dimensiones, que vienen dadas por el número de pixeles del ancho y largo de las imágenes, por bloques que únicamente utilizan una única dimensión. Esto ha sido posible gracias a una API de MATLAB para poder construir redes neuronales a partir de un modelo o estructura ya definidos.

La arquitectura de AlexNet en MATLAB y la que se ha implementado para este proyecto son las siguientes:

Figura 21. Estructura de bloques de AlexNet en MATLAB.

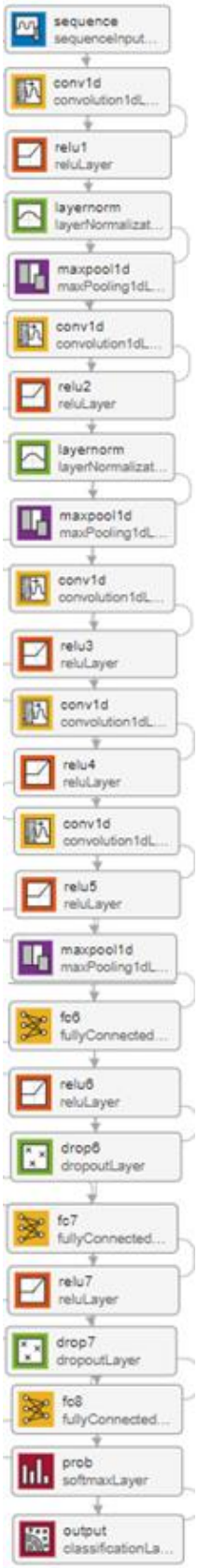


Figura 22. Estructura de bloques de la red neuronal utilizada en el proyecto en MATLAB.

Para el entrenamiento de la red neuronal con los datos de ambos bancos de señales se han creado dos ficheros con la misma red neuronal lo único que cambia es el peso o las ponderaciones de las distintas capas que componen a la red neuronal. Tras el preprocesamiento de las señales, se ha conseguido extraer las características propias de cada evento.

- Para la base de datos de PhysioNet, ya sea la imaginación del movimiento de la mano izquierda, de la mano izquierda o de ambos pies, estos eventos, están identificados por las siguientes etiquetas:
    - **0**: esta etiqueta representa el movimiento imaginario de ambos pies.
    - **1**: representa el movimiento imaginario de la mano izquierda.
    - **2**: indica el movimiento imaginario de la mano derecha.

  Para el entrenamiento de la red neuronal de ese fichero, se han utilizado una parte de los datos extraídos, es decir, una parte de los datos de la mano derecha, de la mano izquierda y de los pies.

- En cuanto a la base de datos de la UAH, para la imaginación del movimiento la mano derecha o el de la mano izquierda, cada uno de estos eventos toma un valor dentro de la variable *clase* que se encuentra al procesar los ficheros que componen a esta base de datos:
    - **1**: es la etiqueta que representa el movimiento de la mano derecha.
    - **2**: representa la etiqueta del movimiento de la mano izquierda.

  En este caso, los datos que forman parte del conjunto de entrenamiento únicamente son una parte de los datos recogidos de la mano derecha y otra parte de los datos de la mano izquierda

Una vez extraídos todos los datos de entrenamiento de la red, posteriormente se los introduciremos a la red neuronal construida. Una vez que la red ha sido entrenada, está lista para la clasificación de los datos que no se han utilizado en el entrenamiento de la red.

Capítulo 5.

5.Resultados.

Para comparar los resultados de los dos bancos de datos de señales que se poseen, se han creado dos scripts diferentes en MATLAB. Cada uno de estos dos ficheros tratará uno de estos bancos de datos.

Lo que se busca en este apartado es ver la precisión que tiene la red neuronal a la hora de clasificar las tareas. Para ello se ha creado una función que cuenta los aciertos y los fallos de la red neuronal.

5.1 Fichero del banco de datos de la UAH.

Para el tratamiento de las señales procedentes del banco de datos de la UAH, tal y como se ha comentado al inicio del apartado, se ha creado un script en MATLAB.

Las características de este fichero es que de los 12 sujetos que componen el banco, únicamente se van a utilizar los primeros experimentos de cada sujeto, ya que hay algunos sujetos que tienen más de 3 sesiones mientras que algunos únicamente poseen una sesión. Otra de las características de este fichero es que solo va a haber una única red neuronal a entrenar y esta va a ser la encargada de clasificar las dos tareas posibles.

El funcionamiento de este script es el siguiente:

La trama de ejecución de este script viene definida por un bucle que es el encargado de ir seleccionando de cada uno de los doce pacientes su primer experimento, a continuación, se utilizarán los datos recogidos por los canales C3 y C4, y se compone una matriz con estos datos.

Seguidamente, a los datos de la matriz anterior, le aplicamos la FFT y en la última fila le añadimos el significado de estos datos, es decir si esa columna pertenece a la mano derecha o a la mano izquierda y seguidamente se divide esa matriz en función del significado de esas columnas. Por lo tanto, se obtienen dos nuevas matrices: una con los datos pertenecientes a la mano derecha y otra de la mano izquierda.

Siguiendo con la trama de ejecución, se componen los datos de entrenamiento para la red neuronal, se define las capas de la red neuronal y se entrena la red con los datos de entrenamiento.

Para finalizar, los datos que no se han utilizado para el entrenamiento de la red neuronal son utilizados para la clasificación. Para la clasificación de estos datos se ha creado una función que recibe los siguientes parámetros: la red neuronal, los datos no usados en el entrenamiento y por último una etiqueta. Esta etiqueta toma el valor de **uno** si es la mano derecha o **dos** si es la mano izquierda. El objetivo de esta función es la siguiente: clasifica los datos, y cuenta la tasa de acierto y fallo de la red en función de la etiqueta y, por último, se guarda la tasa de acierto y fallo en una matriz.

Al finalizar el bucle, la matriz resultante tiene la misma estructura que los resultados que se muestran a continuación:

Sujeto	Tasa acierto Mano Derecha	Tasa fallo Mano Derecha	Tasa acierto Mano Izquierda	Tasa fallo Mano Izquierda
1	81.82%	18.18%	9.09%	90.91%
2	54.55%	45.45%	0%	100%
3	90.91%	9.09%	45.45%	54.55%
4	27.27%	72.73%	63.64%	36.36%
5	100%	0%	45.45%	54.55%
6	100%	0%	45.45%	54.55%
7	81.82%	18.18%	0%	100%
8	100%	0%	9.09%	90.91%
9	27.27%	72.73%	0%	100%
10	72.73%	27.27%	45.45%	54.55%
11	36.36%	63.64%	100%	0%
12	45.45%	54.55%	100%	0%

Tabla 1. Resultado 1 de la clasificación de las tareas del banco de datos de la UAH

Número de pacientes	Tasa media acierto Mano Derecha	Tasa media fallo Mano Derecha	Tasa media acierto Mano Izquierda	Tasa media fallo Mano Izquierda
12	68.18%	31.82%	38.64%	61.37%

Tabla 2. Resultado medio de la Tabla 1.

Como se ha comentado anteriormente, la red neuronal ha sido entrenada con una serie de datos pertenecientes a los datos de las manos derecha y la mano izquierda de cada sujeto. Como se puede observar en cuanto a la capacidad de clasificación, el clasificador, hace un gran trabajo a la hora de clasificar la mano derecha, en cambio, el porcentaje de la mano izquierda es algo inferior.

Pero en otros intentos los porcentajes están más igualados o próximos entre sí, como en este ejemplo:

Sujeto	Tasa acierto Mano Derecha	Tasa fallo Mano Derecha	Tasa acierto Mano Izquierda	Tasa fallo Mano Izquierda
1	100%	0%	27.27%	72.73%
2	63.64%	36.36%	27.275	72.73%
3	18.18%	81.82%	27.27%	72.73%
4	0%	100%	100%	0%
5	100%	0%	0%	100%
6	90.91%	9.09%	45.45%	54.55%
7	0	100%	100%	0%
8	45.45%	54.55%	100%	0%
9	0%	100%	90.91%	9.09%
10	45.45%	54.55%	100%	0%
11	90.91%	9.09%	0%	100%
12	100%	0%	27.27%	72.73%

Tabla 3. Resultado 2 de la clasificación de las tareas del banco de datos de la UAH

Número de pacientes	Tasa media acierto Mano Derecha	Tasa media fallo Mano Derecha	Tasa media acierto Mano Izquierda	Tasa media fallo Mano Izquierda
12	54.55%	45.46%	53.79%	46.21%

Tabla 4. Resultado medio de la Tabla 3.

En este caso, los valores de acierto medio son más equilibrados y además el clasificador ha hecho un gran trabajo a la hora de clasificar las dos tareas.

En ocasiones, como en el siguiente ejemplo, las tasas de acierto medio concuerdan con la tasa media de fallo de la otra tarea y viceversa.

Sujeto	Tasa acierto Mano Derecha	Tasa fallo Mano Derecha	Tasa acierto Mano Izquierda	Tasa fallo Mano Izquierda
1	27.27%	72.73%	0%	100%
2	9.09%	90.91%	81.82%	18.18%
3	45.45%	54.55%	81.82%	18.18%
4	18.18%	81.82%	81.82%	18.18%
5	18.18%	81.82%	100%	0%
6	63.64%	36.36%	90.91%	9.09%
7	81.82%	18.18%	63.64%	36.36%
8	9.09%	90.91%	0%	100%
9	9.09%	90.91%	81.82%	18.18%
10	100%	0%	36.36%	63.64%
11	0%	100%	100%	0%
12	100%	0%	0%	100%

Tabla 5. Resultado 3 de la clasificación de las tareas del banco de datos de la UAH.

Número de pacientes	Tasa media acierto Mano Derecha	Tasa media fallo Mano Derecha	Tasa media acierto Mano Izquierda	Tasa media fallo Mano Izquierda
12	40.15%	59.85%	59.85%	40.15%

Tabla 6. Resultado medio de la Tabla 5.

Esto quiere decir que cuando el clasificador ha fallado a la hora de clasificar, por ejemplo, la mano derecha, ha acertado a la hora de clasificar la mano izquierda. Esto es posible debido a que en este banco de datos solo existen dos tareas.

5.2 Fichero del banco de datos de PhysioNet.

Para el fichero del banco de datos de PhysioNet se han utilizado dos modelos. El primero de ellos tiene que ver con el uso de una única red neuronal haciendo uso de la FFT, mientras que el segundo modelo hace uso de forma conjunta la FFT con la transformada Wavelet y la segunda modificación con respecto al primer modelo, es el uso de tres redes neuronales.

En ambos modelos, al igual que en el anterior fichero, se han utilizado doce sujetos, pero en este caso estos doce sujetos son elegidos aleatoriamente de los 109 sujetos que componen a este banco de datos. Los sujetos elegidos no se repiten así que no hay posibilidad de que a la hora de entrenar la red neuronal se entrene con los mismos datos de entrenamiento.

En este caso, aparte de imaginar el movimiento de las manos derecha e izquierda, el clasificador también tiene que ser capaz de clasificar la imaginación del movimiento de los pies.

Para este fichero, se han creado numerosas funciones. A continuación, se comentará el nombre de estas funciones, su funcionalidad y como se utilizan de forma conjunta.

- **lecturaFichero**. Esta función recibe por parámetro: la sesión y el sujeto y devuelve los datos que componen al fichero y las anotaciones del fichero. El objetivo de esta función es la de leer el fichero de datos del sujeto que se le pasa por parámetro.
- **rellenarMatriz**. Los parámetros que recibe esta función son los siguientes: los datos del fichero de datos, la tarea que se está desarrollando, un contador que nos indica cuantas veces se desarrolla esa tarea, una etiqueta (toma los valores “T1” o “T2”) y los segundos válidos. Este último parámetro indica las posiciones con datos efectivos, es decir, son los datos que corresponden a las tareas de movimiento de la mano derecha, mano izquierda o de ambos pies.

  Esta función devuelve una matriz con los datos correspondientes a la tarea que se está desarrollando. El funcionamiento de esta función es la siguiente: únicamente se añaden los datos en la matriz cuando **etiqueta** y **tarea** coinciden.
- **rellenarMatrizManos**. Los parámetros de esta función son los siguientes: los datos que componen al fichero de datos y las anotaciones de este. Devuelve dos matrices: una matriz con los datos correspondientes a la mano derecha llamada **manoDerecha** y los datos de la mano izquierda llamada **manoIzquierda**. Esta función utiliza la función anterior para poder rellenar las matrices que se van a devolver.

  El funcionamiento de la función es la siguiente: se definen las variables **segundosValidos**, **tarea**, **t1**, **t2**. Seguidamente se realiza un bucle para ir incrementando las variables **t1** y **t2**, en función de la variable **tarea**. La variable **tarea** es un vector con los valores de las anotaciones que se pasan por parámetro, es decir, es un vector con valores “T1” y “T2”. No se utilizará el valor “T0”. Una vez finalizado este bucle se llama a la función anterior y devuelve dos matrices ***canalC4*** y ***canalC3***.

  Esta es la única función que cambia, ya que para el punto 5.2.1, al ***canalC4*** y ***canalC3*** se les aplica la FFT. Para el punto 5.2.2 al ***canalC4*** y ***canalC3*** se les aplica un filtro de paso banda para quedarnos con la banda de frecuencia de entre 8 y 30 Hz, seguidamente le aplicamos la transformada Wavelet en una dimensión y finalmente le aplicamos la FFT.

  En ambos puntos al ***canalC4*** en su última fila para todas las columnas se le añade el valor **1**, mientras que al ***canalC3*** se le añade el valor **2**. Por lo tanto, el ***canalC4*** pasa a ser

manoIzquierda y ***canalC3*** pasa a ser **manoDerecha**.

- **rellenarMatrizPies**. Esta función es igual a la anterior función, lo único en lo que se diferencia es que solo se devuelve una única matriz llamada **pies**, con los datos correspondientes al movimiento de los pies. Otra diferencia es que en la última fila para todas las columnas de la matriz se le añade el valor **0**.
- **obtenerDatosSesiones**. El parámetro de esta función es el sujeto que se está analizando. Esta función utiliza las funciones: **lecturaFichero**, **rellenarMatrizManos** y **rellenarMatrizPies**. Esta función devuelve las matrices de las manos derecha, manos izquierda y de los pies de todas las sesiones del sujeto que se pasa por parámetro.

  El funcionamiento de esta función es la siguiente: se llama a la función **lecturaFichero** y así se obtienen los datos y las anotaciones del fichero de datos del sujeto pasado por parámetro. Seguidamente se realiza un bucle que va desde 1 hasta 14 (son el número de sesiones de un sujeto), las dos primeras sesiones no se usan ya que no aportan información, por lo tanto, se usará de la sesión 3 hasta la sesión 14. Cuando la sesión sea: 3, 4, 7, 8, 11, 12 se llamará a la función de **rellenarMatrizManos,** mientras que para las sesiones: 5, 6, 9, 10, 13, 14 se usará la función de **rellenarMatrizPies**.
- **datosEntrenamientoRed**. Esta función recibe como parámetros: todos los datos de la mano derecha, todos los datos de la mano izquierda y todos los datos de los pies de un sujeto. Devuelve una matriz con una serie de datos que se usarán para entrenar la red neuronal.
- **clasificadorDatos**. Esta función recibe como parámetros: los datos de entrenamiento de la red neuronal, los datos a clasificar (son los que no se han usado para el entrenamiento de la red neuronal) y una etiqueta (toma los valores: **1**, **2** o **0**). Devuelve la tasa de acierto y fallo de la tarea que se está clasificando.

  El funcionamiento de esta función es la siguiente: se definen las capas y las opciones de entrenamiento de la red neuronal, seguidamente se entrena la red neuronal con los datos de entrenamiento de los parámetros. Una vez se ha entrenado la red neuronal, se procede a la clasificación de los datos para clasificar pasados por parámetro y finalmente se calcula la tasa de acierto y fallo.

La secuencia de ejecución de este fichero es el siguiente:

Se comienza definiendo las distintas etiquetas de las tareas y una matriz que guarda las tasas de acierto y fallo medios de las distintas tareas. Seguidamente se realiza un bucle (de 1 a 20, esto se ha usado para obtener varios resultados) que es el que comienza con el fujo de ejecución del programa, en él se define una matriz, llamada ***datosSujetos***, que almacenará la tasa de acierto y fallo de las distintas tareas para 12 sujetos y la última fila de esta matriz, corresponde a la media de las tasas de acierto y fallo de las tareas.

A continuación, se eligen 12 sujetos aleatoriamente sin repetición y se guardan los 12 sujetos en un vector, seguidamente se recorre ese vector y para cada sujeto de ese vector se llama a la función de **obtenerDatosSesiones** y así se obtienen todos los valores de las manos derecha, manos izquierda y de los pies de ese sujeto. A continuación, se agrupan los datos de la manos derecha, de las manos izquierda y de los pies en tres matrices distintas llamadas: ***pacienteManoDerecha***, ***pacienteManoIzquierda*** y ***pacientePies***. Posteriormente se llama a la función **datosEntrenamientoRed** y se le pasa por parámetro las matrices: ***pacienteManoDerecha***, ***pacienteManoIzquierda*** y ***pacientePies***.

Tras componer los datos de entrenamiento, se llama a la función **clasificadorDatos** pero se hará de

la siguiente forma. Se llamará a la función **clasificadorDatos** con los siguientes parámetros:

- Para la **mano derecha**: el primer parámetro será los datos de entrenamiento, el segundo parámetro es ***pacienteManoDerecha*** y el ultimo parámetro es ***etiquetaManoDerecha*** (toma el valor de 2).
- Para la **mano izquierda**: el primer parámetro será los datos de entrenamiento, el segundo parámetro es ***pacienteManoIxquierda*** y el ultimo parámetro es ***etiquetaManoIzquierda*** (toma el valor de 1).
- Para los **pies**: el primer parámetro será los datos de entrenamiento, el segundo parámetro es ***pacientePies*** y el ultimo parámetro es ***etiquetaPies*** (toma el valor de 0).

Una vez finalizado la clasificación, se guardan las tasas de acierto y fallo de las tareas en la matriz ***datosSujetos***. Al finalizar este bucle que recorre todos los sujetos seleccionados aleatoriamente, se calcula la tasa media de acierto y fallo de las tareas, y se añaden a la última fila de esta matriz.

Cada matriz de ***datosSujetos*** se exporta a un documento Excel.

5.2.1 Resultados usando únicamente la FFT.

Sujeto	Tasa acierto Mano Derecha	Tasa fallo Mano Derecha	Tasa acierto Mano Izquierda	Tasa fallo Mano Izquierda	Tasa acierto Pies	Tasa fallo Pies
1	84.61%	15.38%	0%	100%	26.92%	73.07%
2	66.66%	33.33%	64%	36%	0%	100%
3	75%	25%	32.14%	67.85%	23.07%	76.92%
4	92.30%	7.69%	0%	100%	25%	75%
5	7.40%	92.59%	0%	100%	100%	0%
6	26.92%	73.07%	0%	100%	100%	0%
7	29.16%	70.83%	80.76%	19.23%	3.84%	96.15%
8	100%	0%	0%	100%	0%	100%
9	0%	100%	7.69%	92.30%	100%	0%
10	0%	100%	0%	100%	100%	0%

11	62.96%	37.03%	48%	52%	18.51%	81.48%
12	0%	100%	50%	50%	44.44%	55.55%

Tabla 7. Resultado de la clasificación de las tareas del banco de datos de PhysioNet usando la FFT.

Número de pacientes	Tasa media acierto Mano Derecha	Tasa media fallo Mano Derecha	Tasa media acierto Mano Izquierda	Tasa media fallo Mano Izquierda	Tasa media acierto Pies	Tasa media fallo Pies
12	41.92%	50.38%	21.73%	70.56%	41.67%	50.63%

Tabla 8. Resultado medio de la Tabla 7.

Tras hacer varios intentos, este es uno de los intentos donde se ha obtenido mejores resultados. Como se puede observar, en comparación con el anterior banco de datos, las tasas de acierto de las distintas tareas son algo más bajos ya que en este banco de datos existen tres tareas a realizar, así que la red neuronal tiene más probabilidades de fallar a la hora de clasificar las tareas.

Cabe mencionar que, con tal de mejorar los resultados en este modelo, se implementó una serie de líneas de código para seleccionar a otros 12 sujetos. Los datos de los primeros 12 sujetos servían para entrenar la red neuronal, mientras que los 12 adicionales serían para la posterior clasificación. Pero los resultados no eran los esperados, porque para que las redes neuronales creen un modelo de clasificación aceptable, necesitan una gran cantidad de datos. En el caso de este proyecto, la principal dificultad y complejidad es a la hora de trabajar con datos provenientes de EEG ya que, para una misma tarea en dos sujetos diferentes, los valores captados por los canales de adquisición varían mucho entre sujetos.

5.2.2 Resultados usando la FFT y la transformada Wavelet.

Con tal de mejorar los resultados de la clasificación de las distintas tareas, se ha propuesto el siguiente modelo de clasificación. Para ello, tal y como se indica en el subtítulo de este apartado, se ha optado por aplicar de forma conjunta la FFT y la transformada Wavelet. Al aplicar este cambio, los resultados mejoraron significativamente, pero como se buscaba una mejora aun mayor, se experimentó con usar tres redes neuronales para el entrenamiento y la clasificación de las tres distintas tareas, en vez de usar una única red neuronal como se ha utilizado en el anterior ejemplo. Este cambio significa que cada una de las tres redes neuronales se especialice en una única tarea, una red neuronal solo para la tarea de la mano derecha, una para la mano izquierda y la última para los pies.

Para obtener mejores resultados se ha aplicado un filtro de paso banda para quedarnos con las frecuencias de los distintos ritmos cerebrales que aportan información cuando se desempeña una

tarea motriz. Estas frecuencias van de los 8-30 Hz. Otra modificación que se ha aplicado para mejorar la tasa de acierto de las tareas ha sido cambiar los datos de entrenamiento de la red. A su vez, también se han modificado algunos de los parámetros de las capas de la red neuronal y las opciones de esta.

Sujeto	Tasa acierto Mano Derecha	Tasa fallo Mano Derecha	Tasa acierto Mano Izquierda	Tasa fallo Mano Izquierda	Tasa acierto Pies	Tasa fallo Pies
1	0%	100%	41.67%	58.33%	100%	0%
2	0%	100%	0%	100%	100%	0%
3	0%	100%	55.56%	44.44%	37.04%	62.96%
4	92.59%	7.41%	52%	48%	100%	0%
5	0%	100%	36%	64%	100%	0%
6	0%	100%	20%	80%	0%	100%
7	73.08%	26.92%	0%	100%	100%	0%
8	73.08%	26.92%	100%	0%	0%	100%
9	100%	0%	57.69%	42.31%	100%	0%
10	11.54%	88.46%	100%	0%	100%	0%
11	100%	0%	92%	8%	37.04%	62.96%
12	34.62%	65.38%	0%	100%	7.69%	92.31%

Tabla 9. Resultado 1 del banco de datos de PhysioNet usando 3 redes neuronales y la combinación de la FFT y la transformada Wavelet.

Número de pacientes	Tasa media acierto Mano Derecha	Tasa media fallo Mano Derecha	Tasa media acierto Mano Izquierda	Tasa media fallo Mano Izquierda	Tasa media acierto Pies	Tasa media fallo Pies
12	40.41%	59.59%	46.24%	53.76%	65.15%	34.85%

Tabla 10. Resultado medio de la Tabla 9.

Gracias a estas mejoras y modificaciones en el código se pueden obtener resultados como el anterior.

Cabe mencionar que siempre una de las tareas va a tener un porcentaje de acierto menor que las otras dos, esto se debe a que, al ser varias tareas, conforme el número de tareas aumente, al clasificador lo va a costar más distinguir una tarea de otra. Por ejemplo:

Sujeto	Tasa acierto Mano Derecha	Tasa fallo Mano Derecha	Tasa acierto Mano Izquierda	Tasa fallo Mano Izquierda	Tasa acierto Pies	Tasa fallo Pies
1	85.71%	14.29%	0%	100%	100%	0%
2	15.38%	84.62%	100%	0%	92.59%	7.41%
3	0%	100%	100%	0%	30.77%	69.23%
4	0%	100%	100%	0%	22.22%	77.78%
5	100%	0%	83.33%	16.67%	0%	100%
6	0%	100%	7.69%	92.31%	0%	100%
7	0%	100%	70.83%	29.17%	30.77%	69.23%
8	40.74%	59.26%	100%	0%	37.04%	62.96%
9	100%	0%	100%	0%	0%	100%
10	0%	100%	100%	0%	0%	100%
11	14.29%	85.71%	20.83%	79.17%	0%	100%
12	100%	0%	100%	0%	0%	100%

Tabla 11. Resultado 2 del banco de datos de PhysioNet usando 3 redes neuronales y la combinación de la FFT y la transformada Wavelet.

Número de pacientes	Tasa media acierto Mano Derecha	Tasa media fallo Mano Derecha	Tasa media acierto Mano Izquierda	Tasa media fallo Mano Izquierda	Tasa media acierto Pies	Tasa media fallo Pies
12	38.01%	61.99%	73.56%	26.44%	26.12%	73.88%

Tabla 12. Resultado medio de la Tabla 11.

O también está la posibilidad de que los valores de la tasa de acierto de las tareas tengan valores muy similares como en este ejemplo:

Sujeto	Tasa acierto Mano Derecha	Tasa fallo Mano Derecha	Tasa acierto Mano Izquierda	Tasa fallo Mano Izquierda	Tasa acierto Pies	Tasa fallo Pies
1	100%	0%	100%	0%	56%	44%
2	0%	100%	100%	0%	34.62%	65.38%
3	100%	0%	3.7%	96.3%	0%	100%
4	0%	100%	0%	100%	0%	100%
5	100%	0%	100%	0%	0%	100%
6	100%	0%	0%	100%	100%	0%
7	0%	100%	24%	76%	25.93%	74.07%
8	76%	24%	62.96%	37.04%	33.33%	66.67%
9	57.14%	42.86%	0%	100%	80%	20%
10	0%	100%	0%	100%	100%	0%
11	73.08%	26.92%	23.08%	76.92%	26.92%	73.08%
12	100%	0%	100%	0%	0%	100%

Tabla 13. Resultado 3 del banco de datos de PhysioNet usando 3 redes neuronales y la combinación de la FFT y la transformada Wavelet.

Número de pacientes	Tasa media acierto Mano Derecha	Tasa media fallo Mano Derecha	Tasa media acierto Mano Izquierda	Tasa media fallo Mano Izquierda	Tasa media acierto Pies	Tasa media fallo Pies
12	58.85%	41.15%	42.81%	57.19%	38.07%	61.93%

Tabla 14. Resultado medio de la Tabla 13.

Cabe resaltar que antes de llegar a la idea de combinar la FFT y la transformada Wavelet, también se propuso cambiar de solucionador en las opciones de entrenamiento de la red neuronal, pero no resultó ser de gran ayuda ya que la red neuronal ya no sería una variación de AlexNet, sino que sería otra red neuronal y en este trabajo no se buscaba eso, además los resultados proporcionados por los diferentes solucionadores no aportaron mejoría apenas.

En el script empleado para desarrollar la clasificación de los datos que provienen de la base de datos de PhysioNet, se ha modificado varias veces los pesos de las capas de la red, así como el tamaño de los filtros, el peso de las "bias" y el factor de aprendizaje de estas. Por lo tanto, la configuración de las capas de la red neuronal utilizada para este script es la siguiente:

```
layers = [
    sequenceInputLayer(length(datosEntrenamiento)-1,"Name","sequence")
    convolution1dLayer(7,32,"Name","conv1d","Padding","same")
    reluLayer("Name","relu1")
    layerNormalizationLayer("Name","layernorm")
    maxPooling1dLayer(5,"Name","maxpool1d","Padding","same")
    convolution1dLayer(7,32,"Name","conv1d_1","Padding","same")
    reluLayer("Name","relu2")
    layerNormalizationLayer("Name","layernorm_2")
    maxPooling1dLayer(5,"Name","maxpool1d_2","Padding","same")
    convolution1dLayer(7,32,"Name","conv1d_2","Padding","same")
    reluLayer("Name","relu3")
    convolution1dLayer(7,32,"Name","conv1d_3","Padding","same")
    reluLayer("Name","relu4")
    convolution1dLayer(5,32,"Name","conv1d_4","Padding","same")
    reluLayer("Name","relu5")
    maxPooling1dLayer(7,"Name","maxpool1d_3","Padding","same")
    fullyConnectedLayer(4000,"Name","fc6","BiasLearnRateFactor",60)
    reluLayer("Name","relu6")
    dropoutLayer(0.7,"Name","drop6")
    fullyConnectedLayer(4000,"Name","fc7","BiasLearnRateFactor",65)
    reluLayer("Name","relu7")
    dropoutLayer(0.15,"Name","drop7")
    fullyConnectedLayer(3,"Name","fc8","BiasLearnRateFactor",65)
    softmaxLayer("Name","prob")
    classificationLayer("Name","output")];
```

Figura 23. Configuración de las capas de la red neuronal

Capítulo 6.

Presupuesto.

Para la realización del presupuesto de este proyecto se ha tenido en cuenta una serie de aspectos. En este caso se ha tenido en cuenta el coste de la licencia de MATLAB y el coste de la mano de obra a la hora de realizar el proyecto. En este caso el coste de un ingeniero junior es de 75€/h, sin incluir gastos por seguridad social y otros impuestos por contratación. En total este proyecto ha durado 200 horas.

Seguidamente se muestra el presupuesto total del proyecto:

Coste Software	Coste (€)
Licencia MATLAB	800€
Coste Hardware	**Coste (€)**
Coste ordenador	1.300€
Coste electroencefalógrafo	13.090€
Coste Personal	**Coste total (€)**
Coste de Ingeniero	1.5000 €
Total	**16.690€**

Tabla 15. Presupuesto.

Capítulo 7.

Conclusiones y trabajos futuros.

Al finalizar este proyecto, se ha conseguido crear un clasificador mediante una interfaz BCI capaz de clasificar una serie de tareas mentales sencillas.

Podemos llegar a la conclusión de que al especializar una red neuronal a una única tarea los resultados en cuanto a la tasa de acierto de dicha tarea a la hora de clasificarla, se obtienen muy buenos resultados incluso llegando al 73%. También cabe resaltar que en función del número de tareas que el clasificador tiene que procesar y clasificar, el porcentaje de acierto se irá reduciendo esto se debe a la complejidad de trabajar con EEG, pero esto se puede arreglar durante la fase de extracción de características, ya que si se realiza una serie de filtrados para se queden las bandas de frecuencia con las que queremos trabajar, se suprime el ruido que puedan llegar a detectar los electrodos, y si se amplifican los canales de adquisición con los que se van a trabajar y por último, los sujetos que realizan estos ensayos están bien entrenados para realizar las distintas tareas, se obtendrán muy buenos resultados a la hora de la clasificación.

A esto le podemos sumar que, aumentando la capacidad de cálculo del ordenador, se podrá entrenar a la red neuronal con una cantidad masiva de datos para que a la hora de la clasificación las tasas de acierto sean mucho mayores. A su vez, se necesitarán una gran cantidad de datos para poder realizar lo que se acaba de comentar.

También se ha podido visualizar que las técnicas de Deep Learning han sido de gran ayuda ya que no se ha hecho falta la extracción manual de las características que configuran a las distintas tareas. Los trabajos futuros dentro del ámbito en el que se encuentra este proyecto pueden partir por aumentar la cantidad de registros y la capacidad de cálculo y procesamiento de los ordenadores, ya que a través del Deep Learning se pueden crear unas interfaces BCI más robustas y al mismo tiempo crear redes neuronales que se asemejen lo más posible al cerebro humano, pero a día de hoy, debido a la falta de computación, no se ha podido llegar a la profundidad necesaria para descifrar todos los patrones cerebrales cuando re está realizando una tarea mental ya sea simple o compleja.

Bibliografía.

[1] Cómo el Deep Learning está Revolucionando el Procesamiento de Señal de EEG. [En línea]. Disponible en: https://www.bitbrain.com/es/blog/deep-learning-y-se%C3%B1al-eeg

[2] ¿Qué son las ondas cerebrales? [En línea]. Disponible en: https://neuroscenter.com/neurofeedback/ondas-cerebrales/

[3] Ritmo mu. [En línea]. Disponible en: https://es.wikipedia.org/wiki/Ritmo_mu

[4] Sistema BCI2000. [En línea]. Disponible en: http://www.bci2000.org

[5] Kai Keng Ang, Zhang Yang Chin, Haihong Zhang, & Cuntai Guan. (2008). Filter Bank Common Spatial Pattern (FBCSP) in Brain-Computer Interface. In 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence) (pp. 2390–2397). IEEE. [En línea]. Disponible en: https://doi.org/10.1109/IJCNN.2008.4634130

[6] Huang, S., & Wu, X. (2010). Feature extraction and classification of EEG for imagery movement based on mu/beta rhythms. In 2010 3rd International Conference on Biomedical Engineering and Informatics (pp. 891–894). IEEE. [En línea]. Disponible en: https://doi.org/10.1109/BMEI.2010.5639888

[7] Hu, D., Li, W., & Chen, X. (2011). Feature extraction of motor imagery EEG signals based on wavelet packet decomposition. In The 2011 IEEE/ICME International Conference on Complex Medical Engineering (pp. 694–697). IEEE. [En línea]. Disponible en: https://doi.org/10.1109/ICCME.2011.5876829

[8] Qiao, X., Wang, Y., Li, D., & Lifeng Tian. (2010). Feature extraction and classifier evaluation of EEG for imaginary hand movements. In 2010 Sixth International Conference on Natural Computation (pp.2112–2116). IEEE. [En línea]. Disponible en: https://doi.org/10.1109/ICNC.2010.5582453

[9] Qué es overfitting y underfitting y cómo solucionarlo [En línea]. Disponible en: https://www.aprendemachinelearning.com/que-es-overfitting-y-underfitting-y-como-solucionarlo/

[10] Qué son las Redes Neuronales Convolucionales. [En línea]. Disponible en: https://www.iebschool.com/blog/redes-neuronales-convolucionales-big-data/

[11] Deep Learning o Aprendizaje profundo: ¿qué es? [En línea]. Disponible en:

https://datascientest.com/es/deep-learning-definicion

[12] AlexNet y clasificación de imágenes. [En línea]. Disponible en: https://lamaquinaoraculo.com/computacion/alexnet/

Anexos.

A. Fichero banco de datos de la UAH.

A.1 Función que calcula la tasa de acierto y fallo a la hora de la clasificación de las tareas.

```
function [tasaAcierto,tasaFallo] = clasificadorDatosUAH(netTransfer, datosClasificar, etiqueta)

% Función encargada de contar los aciertos y los fallos a la hora de la
% clasificación en el fichero del banco de datos de la UAH.

numAciertos = 0;
numFallos = 0;

class = classify(netTransfer, datosClasificar(1:end-1, 20:end));

for j=1:length(class)
    if(class(j) == categorical(etiqueta))
        numAciertos = numAciertos + 1;
    else
        numFallos = numFallos + 1;
    end
end

tasaAcierto = round((numAciertos/(numAciertos + numFallos)) * 100, 2);
tasaFallo = round((numFallos/(numAciertos + numFallos)) * 100, 2);

end
```

A.2 Código principal para el banco de datos de la UAH.

```
clc; close all; clear

% Se definen las variables globales.

etiquetaManoDerecha = 1;
etiquetaManoIzquierda = 2;
datosPaciente = zeros(13, 5);
f = 1;

% Extraemos del banco de datos todas las primeras sesiones de todos los usuarios.

for sujetos = 1:12
    if(sujetos<10)
        paciente = strcat('0', string(sujetos));
    else
        paciente = string(sujetos);
    end

    fileName = strcat('UAH_BCI_database_description\A',paciente, '_1.mat');

    muestra = importdata(fileName);

    datosCanal = muestra.canal([1,3],:); % Del fichero anterior únicamente nos quedamos con los valores que proporciona C3 y C4 que están en las filas 1 y 3

    contadorMuestra = 1;
    datosValidos = 511 + muestra.muestra(contadorMuestra); % Inicializamos esta variable a 512 porque es a partir de este indice donde los datos son válidos, a partir del segundo t = 4
    columnas = 60;
    filas = 641; % Esto sale de restar 512 al segundo indice de muestra.muestra. Es decir 1153-512 = 641. Esto se cumple con los demás indisces de esa variable
    datos = zeros(filas,columnas);


    % Creamos uma matriz de 641x60, que se rellena con los 641 datos que aporta
    % cada experimento.

    for i=1:columnas
        for j=1:filas
            datos(j,i) = datosCanal(datosValidos);
            datosValidos = datosValidos + 1;
        end
        contadorMuestra = contadorMuestra + 1;
    end

    % A partir de la matriz anterior realizamos la transformada de Fourier

    datosTransformadaFourier = real(fft(datos));

    % Creamos una matriz con la transformada de Fourier y el significado de
    % cada muestra

    tablaDatos = [datosTransformadaFourier ; muestra.clase];

    [numRows, numCols] = size(tablaDatos);

    manoDerecha = zeros(numRows, numCols/2);
    manoIzquierda = zeros(numRows, numCols/2);

    n=1;
    r = 1;
    l = 1;

    % Separamos los datos de la mano derecha de los de la mano izquierda

    while n <= numCols
        if(tablaDatos(end,n) == 1)
            manoDerecha(:,r) = tablaDatos(:,n);
            r = r + 1;
        else
            manoIzquierda(:,l) = tablaDatos(:,n);
            l = l + 1;
        end
        n = n + 1;
    end

    % Componemos una nueva matriz con una serie de datos para entrenar la red

    datosEntrenamientoRed = [manoDerecha(:,1:5:20) manoIzquierda(:,1:5:20) manoDerecha(:,1:2:20) manoIzquierda(:,1:2:20) manoDerecha(:,1:3:20) manoIzquierda(:,1:3:20)];

    % Indicamos las capas de la red nueronal AlexNet

    layers = [
        sequenceInputLayer(length(datosEntrenamientoRed)-1,"Name","sequence")
        convolution1dLayer(3,32,"Name","conv1d","Padding","same")
        reluLayer("Name","relu1")
        layerNormalizationLayer("Name","layernorm")
        maxPooling1dLayer(1,"Name","maxpool1d","Padding","same")
        convolution1dLayer(3,32,"Name","conv1d_1","Padding","same")
        reluLayer("Name","relu2")
        layerNormalizationLayer("Name","layernorm_2")
        maxPooling1dLayer(5,"Name","maxpool1d_2","Padding","same")
        convolution1dLayer(3,32,"Name","conv1d_2","Padding","same")
        reluLayer("Name","relu3")
```

```
        convolution1dLayer(3,32,"Name","conv1d_3","Padding","same")
        reluLayer("Name","relu4")
        convolution1dLayer(3,32,"Name","conv1d_4","Padding","same")
        reluLayer("Name","relu5")
        maxPooling1dLayer(5,"Name","maxpool1d_3","Padding","same")
        fullyConnectedLayer(4000,"Name","fc6","BiasLearnRateFactor",10)
        reluLayer("Name","relu6")
        dropoutLayer(0.2,"Name","drop6")
        fullyConnectedLayer(4000,"Name","fc7","BiasLearnRateFactor",10)
        reluLayer("Name","relu7")
        dropoutLayer(0.2,"Name","drop7")
        fullyConnectedLayer(2,"Name","fc8","BiasLearnRateFactor",10)
        softmaxLayer("Name","prob")
        classificationLayer("Name","output")];

    options = trainingOptions('sgdm', ...
        'MiniBatchSize',10, ...
        'MaxEpochs',12, ...
        'InitialLearnRate',1e-4, ...
        'Shuffle','every-epoch', ...
        'ValidationFrequency',3, ...
        'Verbose',false, ...
        'Plots','none');

    % Entrenamos la red con los datos mencionados anteriormente

    netTransfer = trainNetwork(datosEntrenamientoRed(1:end-1,:),categorical(datosEntrenamientoRed(end,:)),layers,options);

    % Utilizamos los datos que no se han utilizado en el entrenamiento para ver si se clasifican
    % correctamente

    [tasaAciertoMD, tasaFalloMD] = clasificadorDatosUAH(netTransfer, manoDerecha, etiquetaManoDerecha);
    [tasaAciertoMI, tasaFalloMI] = clasificadorDatosUAH(netTransfer, manoIzquierda, etiquetaManoIzquierda);

    datosPaciente(f,:) = [sujetos tasaAciertoMD tasaFalloMD tasaAciertoMI tasaFalloMI];
    f = f + 1;
end

datosPaciente(end, :) = [max(sujetos) round(mean(datosPaciente(1:end-1,2)),2) round(mean(datosPaciente(1:end-1,3)),2) round(mean(datosPaciente(1:end-1,4)),2) round(mean(datosPaciente(1:end-1,5)),2)];
```

B. Fichero banco de datos de PhysioNet.

B.1 Función para la lectura de los ficheros de datos.

```
function [data,anotaciones] = lecturaFichero(sesion, sujetos)

% Función destinada a la lectura de todas las sesiones de un sujeto

directorio = 'Physionet_Database\Muestras\';
separador = '\';
extension = '.edf';

if(sesion<10)
    record = 'R0';
else
    record = 'R';
end

if(sujetos < 10)
    paciente = strcat('00', string(sujetos));
elseif(9 < sujetos) && (sujetos < 100)
    paciente = strcat('0', string(sujetos));
else
    paciente = string(sujetos);
end

filename = strcat(directorio,'S',paciente,separador,'S',paciente,record,string(sesion),extension);

[data, anotaciones] = edfread(filename);

end
```

B.2 Función para rellenar matrices con datos.

```
function [canal] = rellenarMatriz(data, tarea, t, etiqueta, segundosValidos)

% Funcion para rellenar matrices con los datos de las sesiones

n = 1;
cont = 1;
contadorSegundosValidos = 1;
canal = zeros(length(data{1,1})*4,t);

while n <= t
    k = 1;
    if(tarea(cont) == etiqueta)
        for i=0:3
            for j=1:length(data{1,1})
                canal(k,n) = data{segundosValidos(contadorSegundosValidos)+i,1}(j);
                k = k + 1;
            end
        end
        n = n + 1;
        contadorSegundosValidos = contadorSegundosValidos+1;
    end
    cont = cont + 1;
end
end
```

B.3 Función para rellenar las matrices de las manos.

```
function [manoIzquierda, manoDerecha] = rellenarMatrizDatosManos(data,anotaciones)

% Función encargada de rellenar las matrices con datos de las manos

segundosValidos = 1 + floor(seconds(anotaciones.Onset(2:2:end))); % Esto nos da las posiciones efectivas de data con datos
tarea = anotaciones.Annotations(2:2:end); % Va de 2 en 2 porque entre tarea y tarea hay un descanso
t1 = 0;
t2 = 0;

% Contabilizamos que tarea pertenece a la mano derecha y cual a la mano
% izquierda

for i=1:length(tarea)
    if(tarea(i) == "T1")
        t1 = t1+1;
    elseif(tarea(i) == "T2")
        t2 = t2+1;
    end
end
```

```
20
21      % Creamos la matriz C4
22
23      canalC4 = rellenarMatriz(data.C4__, tarea, t1, "T1", segundosValidos);
24
25      % Aplicamos un filtro de paso banda para quedarnos con las bandas de
26      % frecuencia de 8 a 30 Hz
27
28      canalC4Filtro = bandpass(canalC4,[8,30],160);
29      [~,numCols] = size(canalC4Filtro);
30
31      % Aplicamos la transformada Wavelet en una dimensión.
32
33      for i=1:numCols
34          [cC4, lC4] = wavedec(canalC4Filtro(:,i),1,'db2');
35          c4Band = appcoef(cC4, lC4,'db2');
36          c4(:,i) = c4Band;
37      end
38
39      % Al resultado, le aplicamos la FFT
40
41      canalC4 = abs(fft(c4));
42      canalC4(end+1,:) = 1;
43      manoIzquierda = canalC4;
44
45      % Creamos la matriz C3
46
47      canalC3 = rellenarMatriz(data.C3__, tarea, t2, "T2", segundosValidos);
48
49      % Aplicamos un filtro de paso banda para quedarnos con las bandas de
50      % frecuencia de 8 a 30 Hz
51
52      canalC3Filtro = bandpass(canalC3,[8,30],160);
53      [~,numCols] = size(canalC3Filtro);
54
55      % Aplicamos la transformada Wavelet en una dimensión.
56
57      for i=1:numCols
58          [cC3, lC3] = wavedec(canalC3Filtro(:,i),1,'db2');
59          c3Band = appcoef(cC3, lC3,'db2');
60          c3(:,i) = c3Band;
61      end
62
63      % Al resultado, le aplicamos la FFT
64
65      canalC3 = abs(fft(c3));
66      canalC3(end+1,:) = 2;
67      manoDerecha = canalC3;
68
69      end
```

B.4 Función para rellenar las matrices de los pies.

```
function [pies] = rellenarMatrizDatosPies(data,anotaciones)

% Función encargada de rellenar las matrices con datos de los pies

segundosValidos = 1 + floor(seconds(anotaciones.Onset(2:2:end))); % Esto nos da las posiciones efectivas de data con datos
tarea = anotaciones.Annotations(2:2:end); % Va de 2 en 2 porque entre tarea y tarea hay un descanso
t2 = 0;

% Contabilizamos que tarea pertenece a los pies

for i=1:length(tarea)
    if(tarea(i) == "T2")
        t2 = t2+1;
    end
end
```

```
  % Creamos la matriz Cz

  canalCz = rellenarMatriz(data.Cz__, tarea, t2, "T2", segundosValidos);

  % Aplicamos un filtro de paso banda para quedarnos con las bandas de
  % frecuencia de 8 a 30 Hz

  canalCzFiltro = bandpass(canalCz,[8,30],160);
  [~,numCols] = size(canalCzFiltro);

  % Aplicamos la transformada Wavelet en una dimensión.

  for i=1:numCols
      [cCz, lCz] = wavedec(canalCzFiltro(:,i),1,'db2');
      czBand = appcoef(cCz, lCz,'db2');
      cz(:,i) = czBand;
  end

  % Al resultado, le aplicamos la FFT

  canalCz = abs(fft(cz));
  canalCz(end+1,:) = 0;
  pies = canalCz;
  end
```

B.5 Función para obtener los datos de todas las sesiones de un sujeto.

```
function [mDS3,mIS3,mDS4,mIS4,pS5,pS6,mDS7,mIS7,mDS8,mIS8,pS9,pS10,mDS11,mIS11,mDS12,mIS12,pS13,pS14] = obtenerDatosSesiones(sujetos)

% Obtenemos todos los datos de las manos y pies de todas las sesiones de un
% sujeto

for sesion=1:14

    [data, anotaciones] = lecturaFichero(sesion, sujetos);

    switch sesion
        case 3
            [mDS3, mIS3] = rellenarMatrizDatosManos(data, anotaciones);
        case 4
            [mDS4, mIS4] = rellenarMatrizDatosManos(data, anotaciones);
        case 5
            pS5 = rellenarMatrizDatosPies(data, anotaciones);
        case 6
            pS6 = rellenarMatrizDatosPies(data, anotaciones);
        case 7
            [mDS7, mIS7] = rellenarMatrizDatosManos(data, anotaciones);
        case 8
            [mDS8, mIS8] = rellenarMatrizDatosManos(data, anotaciones);
        case 9
            pS9 = rellenarMatrizDatosPies(data, anotaciones);
        case 10
            pS10 = rellenarMatrizDatosPies(data, anotaciones);
        case 11
            [mDS11, mIS11] = rellenarMatrizDatosManos(data, anotaciones);
        case 12
            [mDS12, mIS12] = rellenarMatrizDatosManos(data, anotaciones);
        case 13
            pS13 = rellenarMatrizDatosPies(data, anotaciones);
        case 14
            pS14 = rellenarMatrizDatosPies(data, anotaciones);
    end

end
end
```

B.6 Función para el entrenamiento de la red neuronal.

```
function [datosEntrenamiento] = datosEntrenamienoRed(manoDerecha, manoIzquierda, pies)

% Funcion para crear los conjunmtos de datos para el entrenamiento de la
% red neuronal.

deDosEnDos = [manoDerecha(:,1:2:28) manoIzquierda(:,1:2:28) pies(:,1:2:28)];
deTresEnTres = [manoDerecha(:,1:3:27) manoIzquierda(:,1:3:27) pies(:,1:3:27)];
deCuatroEnCuatro = [manoDerecha(:,1:4:28) manoIzquierda(:,1:4:28) pies(:,1:4:28)];
deCincoEnCinco = [manoDerecha(:,1:5:25) manoIzquierda(:,1:5:25) pies(:,1:5:25)];
variosDatos = [manoDerecha(:,1:28) manoIzquierda(:,1:28) pies(:,1:28)];

datosEntrenamiento = [deDosEnDos deTresEnTres deCuatroEnCuatro deCincoEnCinco variosDatos];

end
```

B.7 Función para el entrenamiento de la red neuronal, clasificación de las tareas y obtención de la tasa de acierto y fallo a la hora de clasificar las tareas.

```
function [tasaAcierto,tasaFallo] = clasificadorDatos(datosEntrenamiento, datosClasificar, etiqueta)

% Entrenamiento de la red neuronal, clasificacion de los datos según la
% etiqueta que se le pasa por parámetro y calculo de la tasa de acierto y
% fallo de la clsificacion de las tareas en funcion de la etiqueta.

numAciertos = 0;
numFallos = 0;

% Definición de las capas de la red neuronal junto a las opciones de
% entrenamiento.

layers = [
    sequenceInputLayer(length(datosEntrenamiento)-1,"Name","sequence")
    convolution1dLayer(7,32,"Name","conv1d","Padding","same")
    reluLayer("Name","relu1")
    layerNormalizationLayer("Name","layernorm")
    maxPooling1dLayer(5,"Name","maxpool1d","Padding","same")
    convolution1dLayer(7,32,"Name","conv1d_1","Padding","same")
    reluLayer("Name","relu2")
    layerNormalizationLayer("Name","layernorm_2")
    maxPooling1dLayer(5,"Name","maxpool1d_2","Padding","same")
    convolution1dLayer(7,32,"Name","conv1d_2","Padding","same")
    reluLayer("Name","relu3")
    convolution1dLayer(7,32,"Name","conv1d_3","Padding","same")
    reluLayer("Name","relu4")
    convolution1dLayer(5,32,"Name","conv1d_4","Padding","same")
    reluLayer("Name","relu5")
    maxPooling1dLayer(7,"Name","maxpool1d_3","Padding","same")
    fullyConnectedLayer(4000,"Name","fc6","BiasLearnRateFactor",60)
    reluLayer("Name","relu6")
    dropoutLayer(0.7,"Name","drop6")
    fullyConnectedLayer(4000,"Name","fc7","BiasLearnRateFactor",65)
    reluLayer("Name","relu7")
    dropoutLayer(0.15,"Name","drop7")
    fullyConnectedLayer(3,"Name","fc8","BiasLearnRateFactor",65)
    softmaxLayer("Name","prob")
    classificationLayer("Name","output")];

options = trainingOptions('sgdm', ...
    'MiniBatchSize',30, ...
    'MaxEpochs',25, ...
    'InitialLearnRate',1e-4, ...
    'Shuffle','never', ...
    'Verbose',false, ...
    'Plots','none');
```

```
47
48  % Entrenamiento de la red neuronal
49
50  netTransfer = trainNetwork(datosEntrenamiento(1:end-1,:),categorical(datosEntrenamiento(end,:)),layers,options);
51
52  % Clasificación de los datos
53
54  class = classify(netTransfer, datosClasificar(1:end-1, 20:end));
55
56  % Calculo de la tasa de acierto y fallo
57
58  for j=1:length(class)
59      if(class(j) == categorical(etiqueta))
60          numAciertos = numAciertos + 1;
61      else
62          numFallos = numFallos + 1;
63      end
64  end
65
66  tasaAcierto = round((numAciertos/(numAciertos + numFallos)) * 100, 2);
67  tasaFallo = round((numFallos/(numAciertos + numFallos)) * 100, 2);
68
69  end
```

B.8 Código principal del fichero de PhysioNet.

```
1   clc; close all; clear
2
3   % Definimos las etiquetas de las diferentes tareas y una matriz que va a
4   % guardar los resultados medios de 20 ejecuciones
5
6   etiquetaPies = 0;
7   etiquetaManoIzquierda = 1;
8   etiquetaManoDerecha = 2;
9   tiradas = zeros(20, 8);
10  l = 1;
11
12
13  for k=1:20
14
15      f = 1;
16      iter = 1;
17      c = 1;
18      i = 1;
19      j = 1;
20
21      % Creamos una matriz que almacenará el porcentaje de acierto y fallo de
22      % las tres tareas y la ultima fila de esta matriz corresponde a la
23      % media de los acietos y fallos de cada tarea.
24
25      datosSujetos = zeros(13,7);
26
27      % Obtenemos 12 pacientes aleatorios y sin repetición de los 109 sujetos
28
29      sujetosCandidatos = zeros(1,12);
30
31      while i~=13
32          pacienteRandom = randi([1,109], 1,1);
33          if(~ismember(pacienteRandom, sujetosCandidatos))
34              sujetosCandidatos(:,iter) = pacienteRandom;
35              iter = iter + 1;
36              i = i +1;
37          end
38      end
39
```

```
    % Para los sujetos seleccionados aleatoriamente...

    for sujetos=1:length(sujetosCandidatos)

        % Obtenemos los datos correspondientes a la mano derecha, izquierda
        % y de ambos pies para todas las sesiones

        [mDS3,mIS3,mDS4,mIS4,pS5,pS6,mDS7,mIS7,mDS8,mIS8,pS9,pS10,mDS11,mIS11,mDS12,mIS12,pS13,pS14] = obtenerDatosSesiones(sujetosCandidatos(sujetos));

        % Agrupamos todos los datos de la mano derecha, mano izquierda y
        % de los pies

        pacienteManoDerecha = [mDS3 mDS4 mDS7 mDS8 mDS11 mDS12];
        pacienteManoIzquierda = [mIS3 mIS4 mIS7 mIS8 mIS11 mIS12];
        pacientePies = [pS5 pS6 pS9 pS10 pS13 pS14];

        % Componemos los datos de entrenamiento de la red neronal

        datosEntrenamiento = datosEntrenamienoRed(pacienteManoDerecha, pacienteManoIzquierda, pacientePies);

        % Clasificamos los datos en funcion de las tres tareas obteniendo
        % la tasa de acierto y fallo de cada una de ellas

        [tasaAcietoMD, tasaFalloMD] = clasificadorDatos(datosEntrenamiento, pacienteManoDerecha, etiquetaManoDerecha);
        [tasaAcietoMI, tasaFalloMI] = clasificadorDatos(datosEntrenamiento, pacienteManoIzquierda, etiquetaManoIzquierda);
        [tasaAcietoPies, tasaFalloPies] = clasificadorDatos(datosEntrenamiento, pacientePies, etiquetaPies);

        % Incorporamos los datos a la matriz comentada anteriormente

        datosSujetos(f,:) = [sujetos tasaAcietoMD tasaFalloMD tasaAcietoMI tasaFalloMI tasaAcietoPies tasaFalloPies];
        f = f + 1;
    end

    % A la ultima fila de la matriz anterior, añadimos el número total de
    % pacientes y la media de la tasa de acierto y fallo de las tareas.

    datosSujetos(end, :) = [max(datosSujetos(:,1)) round(mean(datosSujetos(1:end-1,2)), 2) round(mean(datosSujetos(1:end-1,3)), 2) round(mean(datosSujetos(1:end-1,4)), 2) round(mean(datosSujetos(1:end-1,5)), 2)

    tiradas(l,:) = [l datosSujetos(end,:)];
    l = l+1;

    % Guardamos los datos de las 20 ejecuciones en un fichero

    writetable(table(datosSujetos), 'Tiradas\Tiradas.xlsx', 'Sheet',k);
end
```

Universidad de Alcalá