UNIVERSIDAD MIGUEL HERNÁNDEZ DE ELCHE
ESCUELA POLITÉCNICA SUPERIOR DE ELCHE
GRADO EN INGENIERÍA ELECTRÓNICA Y
AUTOMÁTICA INDUSTRIAL

"PROCESAMIENTO DE SEÑALES EEG BASADO EN PYTHON"

TRABAJO DE FIN DE GRADO
Febrero - 2022

AUTOR: Fernando Gastón Vasconcellos Noailles
DIRECTOR/ES: José María Azorín Poveda
Eduardo Iáñez Martínez

Resumen

El presente trabajo se centra en el desarrollo de una arquitectura software en lenguaje Python capaz de preprocesar, extraer características y clasificar señales electro encefálicas (EEG) obtenidas a través de técnicas no invasivas.

El código de dicha arquitectura se ha concebido de forma dinámica y podrá adaptarse según las necesidades de una posterior utilización. Por tanto, el uso de esta herramienta facilita la visualización de la totalidad del proceso y la elección de los mejores métodos de clasificación independientemente de los datos de entrada. Este resultado se logra mediante el desarrollo generalista de la solución que incluye procesos agnósticos, construyendo código desacoplado de las definiciones de los datos tratados en este trabajo y aplicando técnicas ampliamente usadas en la industria del desarrollo de software.

A su vez, se implementan modelos de machine learning para la clasificación de los datos obtenidos. Junto con ellos, se desarrollan procesos que encuentran, de forma automática, los mejores parámetros de configuración de los modelos para cada grupo específico de datos a clasificar.

Abstract

This project focuses on the development of a software architecture in Python coding language able to pre-process, mine features and classify signals EEG acquired through non-invasive techniques.

The code of this software architecture has been conceived in a dynamic way to be adapted according to ulterior usage needs. As a consequence, when using this tool, a visualization of the whole process is provided as well as the best choice of the classification methods disregarding the input data. This result is achieved through a generalist development of the solution that includes agnostic procedures, creating loose coupling code from the definitions of the data processed in this project and applying techniques widespread used in the software development industry.

Likewise, machine learning models are used to classify the acquired data. At the same time, processes are developed to find out automatically the best configuration parameters for each specific classified group.

Agradecimientos

Primeramente, me gustaría agradecer a mis tutores José María Azorín Poveda y Eduardo Iáñez Martínez, por haberme permitido desarrollar el presente proyecto, así como por la atención y ayuda que me han prestado durante este periodo.

En segundo lugar, agradecer a mi entorno más cercano por la calidez, apoyo y motivación constante que he recibido por su parte durante todo este periodo de desarrollo del presente proyecto.

Debería, por supuesto, hacer especial mención a aquellos profesores y compañeros de carrera que, a lo largo de estos años, han fomentado el desarrollo de mi curiosidad, así como la formación que me han entregado y que finalmente me ha permitido crecer tanto profesional como personalmente.

Por último, agradezco profundamente mi familia y en especial a mis padres, quienes han sido mi principal fuente de motivación.

ÍNDICE GENERAL

Índice de figuras

Índice de tablas

Capítulo 1. Introducción

Gracias a la tecnología actual, el uso de reconocimiento de patrones ha hecho viable la optimización de la interacción entre las ondas cerebrales y las interfaces y su aplicación en ámbitos variados tales como la biomedicina y la industria.

En este ámbito, la evolución de la ingeniería, concretamente el desarrollo de la computación ha permitido un avance formidable en el campo de la IA y machine learning. La tecnología disponible hace posible la realización de innumerables tests reduciendo drásticamente el tiempo empleado y logrando resultados más pertinentes y exactos.

Este trabajo se propone desarrollar en lenguaje de programación Python una arquitectura de software que permita el procesamiento y clasificación de señales electroencefalográficas (EEG).
Los proyectos similares existentes han servido como punto de apoyo para el desarrollo de una herramienta que aporte un enfoque más pragmático, proporcionando al usuario una plataforma BMI (brain-machine interface o interfaces cerebro-computador) fácil de usar y adaptable a distintos usos.

En la construcción del sistema de procesamiento, se utilizaron los dos algoritmos más comunes: la transformada rápida de Fourier (STFT) y la transformada de wavelets (WT). Ha sido demostrado en trabajos de investigación anteriores, que la transformada rápida de Fourier para procesar señales EEG podría no ser muy eficaz en gran medida debido al carácter estático de las ventanas, lo que no permite la detección de variaciones de frecuencia en la señal.

1.1 Motivación

Este proyecto se encuadra dentro de los proyectos que se llevan a cabo en el Brain-Machine Interface Systems Lab de la UMH en el ámbito del desarrollo de interfaces hombre-máquina.

La intención del presente trabajo es generar un ejecutable que permita discernir la mejor combinación entre procesador de señales, características y clasificador.

Proveer al usuario que opera con sistemas BMI una herramienta de fácil acceso y utilización e integrable en otros sistemas constituye el desafío planteado al proyectar y realizar este trabajo.

A este respecto, uno de los mayores retos es el tratamiento de los datos primitivos, el limpiado de señales y la adecuación de los hiper parámetros de entrada de los modelos de clasificación para obtener el mayor rendimiento. Por ello, se aspira a que la herramienta sirva de enlace entre los fundamentos teóricos referentes a las BMI y la aplicación eficiente de metodologías computacionales.

Al mismo tiempo, se pretende también añadir librerías modernas que permitan una gestión de datos más eficiente, así como principios y reglas acreditados.

1.2 Objetivos

Como previamente se ha mencionado, el objetivo del presente trabajo es el de desarrollar una arquitectura que permita efectuar el procesamiento de señales EEG, extracción de características y clasificación de las mismas. Para ello, se enumeran las tareas que se pretenden realizar:

1. Proposición y construcción de una arquitectura desarrollada en Python incluyendo librerías de cálculo, gestión de datos y machine learning.

2. Optimización del código desarrollado para que pueda ser usado para distintos propósitos.

3. Implementación de algoritmos de procesamiento de señales, tales como la transformada rápida de Fourier o la transformada de wavelet.

4. Desarrollo de algoritmos de extracción de características, incluyendo la posibilidad de modificar los parámetros de entrada, junto con modelos de clasificación.

5. Comparación y optimización del rendimiento arrojado por los modelos de clasificación.

1.3 Estructura del capitulo

El presente trabajo se ha estructurado en las partes siguientes:

- Capítulo 1: contiene la introducción, la motivación, los objetivos y la estructura de los capítulos del proyecto.
- Capítulo 2: incluye la descripción del sistema nervioso y particularmente el funcionamiento neuronal basado en los potenciales de acción, la generación de ritmos cerebrales y su clasificación. Se presentan las herramientas de registro de la actividad cerebral que hacen posible la construcción de BMI (interfaces cerebro-computador o en inglés Brain-computer interface BCI o BMI Brain-Machine interface)tanto invasivas como no invasivas o mixtas.

  Se define el Machine learning (aprendizaje automático) y se describen los algoritmos de ML utilizados en nuestro trabajo. Se exponen brevemente las aplicaciones posibles de estas tecnologías. Por último, se presentan tres lenguajes de programación C,Matlab y Python y se argumenta la opción adoptada
- Capítulo 3: se detalla el desarrollo de la arquitectura de software que procese y clasifique las señales. Se detallan asimismo aspectos

relevantes del lenguaje empleado y las metodologías que garantizan su integración en otras soluciones.

- Capítulo 4: se detallan los tests realizados y los resultados obtenidos. Se establece la comparativa entre los datos resultantes del proceso.
- Capítulo 5: se exponen las conclusiones y líneas posibles de investigaciones futuras.

Capítulo 2. Estado del arte

Con el propósito de contextualizar y fundamentar nuestro trabajo y de hacer posible una comprensión pormenorizada de los procesos seguidos para lograr nuestros fines, revisaremos en este capítulo los fundamentos teóricos y los avances científicos y tecnológicos relativos al estudio del origen y procesamiento de las ondas cerebrales,los conceptos esenciales en relación con la encefalografía, las interfaces cerebro-máquina y el machine learning.

2.1 Comprender la estructura y el funcionamiento del sistema nervioso

Con el fin de comprender la generación de las ondas cerebrales describiremos brevemente los componentes y la organización del sistema nervioso.

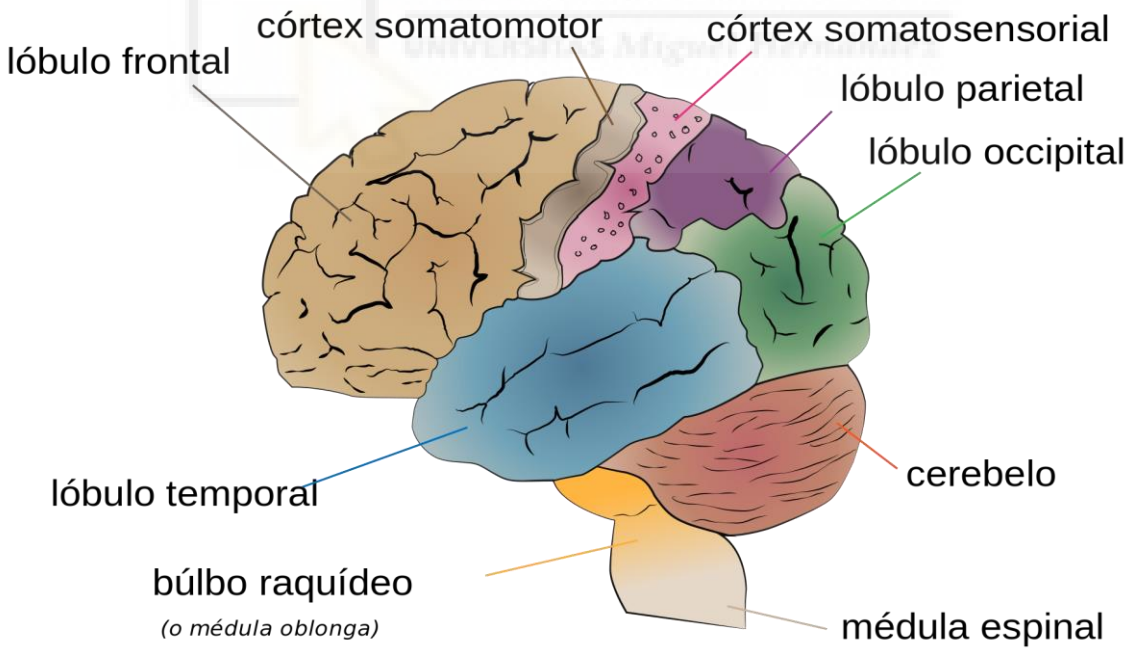

Figura 1: estructura del sistema nervioso

El sistema nervioso es una red que no tiene parangón en cuanto a la enorme complejidad de procesos y acciones de control que es capaz de realizar y que se encuadran en tres tipos de funciones básicas: la sensorial: recepción de

estímulos internos y externos, la integradora: análisis de la información y toma de decisiones y la motora: la generación de respuestas adecuadas.

Para su estudio, en el sistema nervioso se distinguen dos partes: el sistema central formado por el encéfalo y la médula y el sistema periférico que comprende los nervios craneales y espinales.

La médula espinal, que se sitúa dentro del conducto vertebral, regula el sistema motor, los reflejos para evitar el dolor, para sostener el cuerpo y los que rigen la circulación sanguínea, la digestión o el sistema urinario. En el encéfalo se encuentran: el cerebro, el cerebelo y el tallo cerebral.

La inervación sensitiva que hace posible el movimiento de los miembros se realiza a través de los nervios que conectan el encéfalo a la médula y que atraviesan el tallo cerebral. Además, en el tronco del encéfalo se encuentran los centros de control del aparato cardiovascular, del funcionamiento digestivo, de la respiración, del equilibrio, de los movimientos oculares y del equilibrio.

El cerebelo tiene por función la de coordinar temporalmente las tareas motoras.

El órgano de mayor volumen es el cerebro, constituido por los hemisferios cerebrales que se asocian funcionalmente con la mitad opuesta del cuerpo y que evidencian dos tipos de sustancias: una exterior, la gris y una interior, la blanca.

El córtex o corteza cerebral, capa externa de los hemisferios cerebrales, alberga la mayor parte de los centros de procesamiento del pensamiento.

En la figura 1 extraída de [53] se ilustra la estructura del sistema nervioso.

Cada porción del sistema nervioso cumple unas funciones específicas, pero es la corteza la que hace posible la utilización mental de toda la información guardada.

En la corteza se pueden distinguir cuatro áreas funcionales importantes, una en cada lóbulo:

- El lóbulo frontal contiene la corteza motora primaria que controla el movimiento.
- El lóbulo parietal es el lugar de terminación de vías que conducen modalidades del tacto, presión, dolor y temperatura.
- El lóbulo occipital contiene la corteza visual.
- El lóbulo temporal alberga la corteza auditiva.

2.1.1 Potencial de acción de las neuronas

La unidad básica estructural y funcional del sistema nervioso es la neurona. Las funciones de la neurona son recibir e integrar información que procede de los receptores sensitivos o de otras neuronas y transmitir información a otras neuronas u órganos. En general tienen tres partes: una dendrita que recibe las señales, el cuerpo de la célula que procesa la señal y un axón que emite una señal.

La conexión entre las neuronas hace posible la recepción y envío de señales a través de sinapsis axón-dendritas gracias a impulsos electroquímicos cambiantes en sus membranas. Estos cambios rápidos de potencial de las membranas se denominan potenciales de acción y se producen si la neurona es estimulada por encima de un determinado umbral. Cada potencial de acción comienza con un cambio desde un potencial negativo en reposo hacia uno positivo y termina con un potencial negativo volviendo al reposo. Durante el potencial de acción se produce una transferencia a través de la membrana de cargas positivas (iones Na+) al interior de la fibra y un regreso al exterior de estas cargas al final.

Las sucesivas fases del potencial de acción son las que siguen:

- Fase de reposo. El potencial de membrana es negativo antes del comienzo del potencial de acción. (-90 mV). Se dice que la membrana está polarizada.

  El interior de la célula contiene iones de Potasio (K+) y en su exterior aparece un exceso de iones de Sodio (Na +), de carga positiva.
- Fase de despolarización. La membrana se hace permeable permitiendo la entrada de un gran número de iones de sodio de carga positiva.La presencia de estos iones positivos puede resultar en un potencial cero o positivo.
- Fase de repolarización. Tras unas diezmilésimas de segundo, los canales de sodio se cierran y los canales de potasio se abren. La difusión hacia el exterior de los iones de potasio restablece el potencial de membrana en reposo negativo normal. Al final del potencial de acción, el retorno estado negativo produce el cierre de los canales de potasio hasta su estado original.

Esos cambios rápidos de potencial de las membranas se denominan potenciales de acción y se producen si la neurona es estimulada por encima de un determinado umbral.

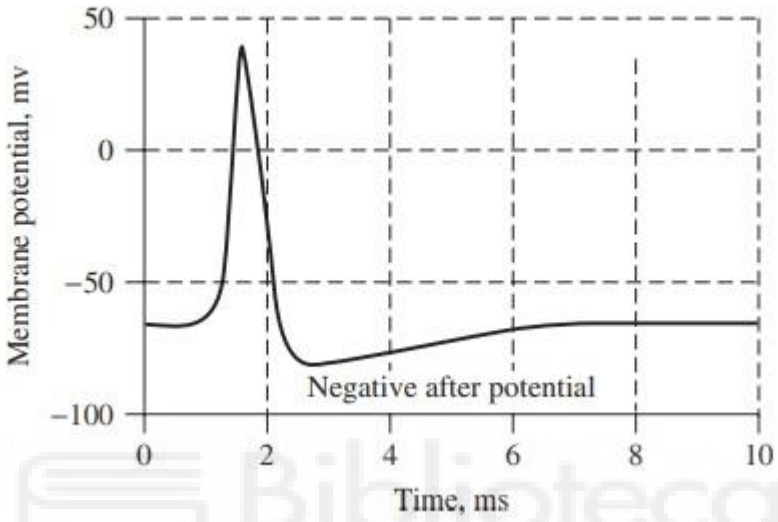

Figura 2: Ejemplo de potencial de acción neuronal

2.1.2 Ritmos cerebrales

El examen visual de las señales recogidas en un encefalograma permite diagnosticar distintas afecciones cerebrales. Se pueden conocer con exactitud los ritmos cerebrales que se manifiestan en las señales.

Las amplitudes y frecuencias de las mismas evidencian cambios de estado, como vigilia o sueño y permiten identificar la edad del paciente estudiado.

Hay cinco tipos de ondas cerebrales caracterizadas por diferentes rangos de frecuencias.

Las bandas de frecuencia que van de bajas frecuencias a altas se denominan alpha (α), theta (θ), beta (β), delta (δ) y gamma (γ).

Berger detectó las ondas alpha y beta en 1929, Jasper y Andrews en 1938 denominaron “gamma” las ondas con frecuencias superiores a 30 Hz.En 1936, Walter introdujo las ondas delta que presentan una frecuencia inferior a las

Alpha. También identificó las ondas theta como aquellas de entre 4 y 7.5 Hz. La noción de onda theta fue explicitada por Wolter y Dovey en 1944.
Si bien en el presente trabajo, analizamos ondas alfa y beta, es interesante caracterizar los cinco tipos de ondas existentes que se ilustran en la figura 3, extraída de [1].

- Las ondas α pueden ser localizadas en todas las regiones de los lóbulos posteriores del cerebro. Su frecuencia se sitúa en un rango de entre 8 y 13 Hz. Se piensa que las ondas alpha indican al mismo tiempo un estado de consciencia relajado, pero sin atención o concentración. Muchos sujetos producen ondas alpha cuando permanecen con los ojos cerrados lo que aparecería como un estado de espera o de percepción ocular rápida de las regiones visuales del cerebro. La producción de ondas alpha se detiene cuando el sujeto abre los ojos o experimenta ansiedad, concentración o atención.

  Se dice que Albert Einstein resolvía problemas matemáticos mientras permanecía en estado Alpha. Se desconoce por el momento el origen y la importancia fisiológica de las ondas alpha.

- Las ondas β con una frecuencia de entre 14 y 26 HZ y con una amplitud inferior a los 30 μV.Se registran en la región frontal y central y se relacionan con la actividad consciente del cerebro cuando el pensamiento y la atención están activos, concentrándose en el mundo exterior o resolviendo problemas concretos. Un nivel elevado de ondas β puede registrarse cuando un sujeto experimenta un estado de pánico. Un ritmo medio de ondas β se relaciona con el ritmo mu y se interrumpe con la actividad motora o la estimulación táctil.

- Las ondas θ tienen unas frecuencias entre 4 y 7.5 Hz.Se presume que se originan en el tálamo cerebral. Las ondas θ aparecen en el pasaje de la consciencia al sueño. Se las asocia con el acceso al material inconsciente, la inspiración creativa y la meditación profunda. Este tipo de ondas juegan un papel esencial en la infancia. Una gran cantidad de ondas θ en el adulto consciente son anormales y evidencian la presencia

de patologías. El cambio en el ritmo de las ondas θ son objeto de estudios sobre la maduración y las emociones.

- Las ondas δ son ondas de un rango de frecuencia entre 0.5 y 4 Hz y de amplitud entre 100 y 200 μV.Esta actividad ocurre en el sueño profundo, en la lactancia, anestesia, falta de oxígeno y en personas con enfermedades orgánicas serias del cerebro.

- Las ondas γ **son** ondas de un rango de frecuencia de entre 31 y 50 H y de entre 5 y 10 μV de amplitud. A pesar de las bajas amplitudes de estas ondas y su aparición que es poco frecuente, su detección, sobre todo en el área fronto central, puede ser útil para la confirmación de enfermedades cerebrales. Las ondas **γ** se relacionan también con la sincronización sensorial de los movimientos de los dedos índices derecho e izquierdo, de los movimientos de los dedos del pie derecho y de los movimientos de la lengua.

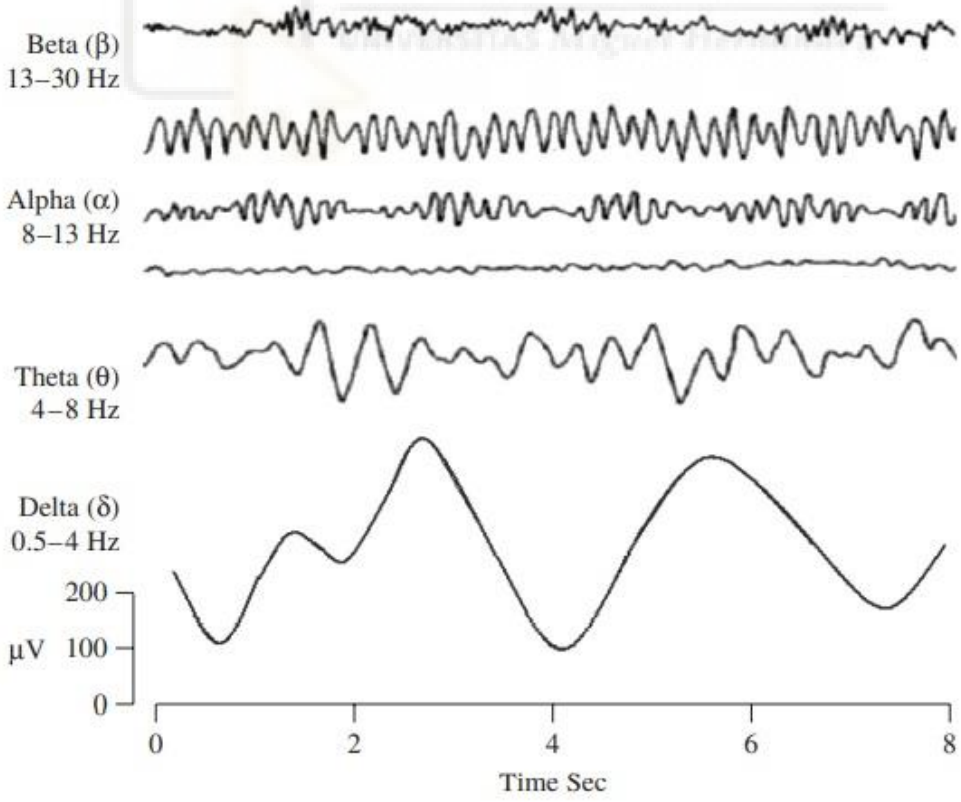

Figura 3: Los cuatro tipos principales de ritmos cerebrales

2.2 Herramientas para el registro de señales: el electroencefalograma

Los potenciales eléctricos recogidos se llaman ritmos cerebrales, y su registro completo se denomina electroencefalograma (EEG).

Las primeras actividades eléctricas neuronales se grabaron utilizando simples galvanómetros.

La invención por Hans Berger de la electroencefalografía (EEG) en 1929 despertó un gran interés por el estudio de la actividad cerebral. Sus descubrimientos fueron confirmados y desarrollados posteriormente por E.D.Adrian y B.H.Matthews en 1934.

Los sistemas más recientes de EEG estaban formados por electrodos, un conjunto de amplificadores bio-potenciales, filtros y agujas para materializar los registros.

Poco después de la llegada de estos sistemas, fue evidente que se debía crear un sistema computarizado que pudiera digitalizar y guardar las señales.

En la actualidad, estos sistemas hacen posible el muestreo, cuantificación y codificación de las señales, además de permitir su procesamiento.

La conversión de señales analógicas en señales digitales se realiza por medio de convertidores (ADC)

Los registros eléctricos realizados en el córtex cerebral o incluso en el cuero cabelludo evidencian la existencia de una actividad de intercomunicación neuronal constante en el encéfalo. El grado de excitación de las células neuronales determina la intensidad y los formatos de las variables en los distintos estados de salud o enfermedad, de vigilia o sueño, de consciencia o coma.

Es posible grabar las señales eléctricas del cerebro para detectar los potenciales de acción. El registro de la actividad de varias neuronas nos permite decodificar la información presente en esas células. Por ejemplo, en las áreas cerebrales relacionadas con el movimiento representa los movimientos que se desean hacer o transmiten información sensorial.

Una señal EEG es por lo tanto el registro de las corrientes sinápticas que fluyen en las dendritas cuando se activan las neuronas piramidales en el córtex cerebral.

Esta corriente genera un campo magnético que puede medirse por medio de un electromiograma (EMG) y un campo eléctrico secundario sobre el cuero cabelludo que se puede medir por medio de sistemas de EEG.

Como hemos mencionado más arriba, las corrientes cerebrales se originan por la extracción de los iones positivos de sodio, Na+, potasio, K+, calcio, CA++ y el ion negativo de cloro, a través de las membranas en la dirección determinada por su potencial.

La cabeza consta de capas superpuestas: el cuero cabelludo, el cráneo y el cerebro y otras capas intermedias. El cráneo amortigua las señales aproximadamente unas cien veces más que el tejido blando.

Una cuestión importante para nuestro trabajo, es la existencia de ruido que se genera o bien en el interior del cerebro (ruido interno) o sobre el cuero cabelludo (ruido de sistema o ruido externo).

Por lo tanto, solamente la presencia de un gran número de neuronas activas puede generar un potencial suficiente para permitir su registro por medio de electrodos ubicados sobre el cuero cabelludo.

Es de capital importancia el uso de la EEG para el diagnóstico de la epilepsia, las encefalopatías, tumores y ACV entre otras , y si bien su uso ha decrecido con la adopción de técnicas de imagen de alta resolución tales como la resonancia magnética o la tomografía computada, sigue siendo una técnica apreciada debido a su movilidad, sencillez y el hecho de que ofrece una resolución temporal en el rango del milisegundo.

De la EEG derivan otras técnicas como Potenciales evocados (EP)y el Potencial relacionado a eventos (ERPs).

La EEG y la MEG proporcionan una fuente inapreciable de información relacionada al estado funcional, fisiológico y patológico del cerebro.

De hecho, la electroencefalografía (EEG) y la magneto encefalografía (MEG) se han convertido en herramientas de diagnóstico cuantitativo sumamente poderosas para el análisis de patrones cerebrales y de señales, su procesamiento y mejora.

2.2.1 Interfaces cerebro-máquina

Una interfaz cerebro-máquina (BMI) es un sistema de comunicación en el cual un individuo envía mensajes u órdenes sin utilizar los trayectos cerebrales normales de los nervios y músculos periféricos. Es un método alternativo de acción sobre el mundo circundante gracias al cual el usuario interactúa directamente mediante su actividad cerebral.

Una BCI provee una conexión entre el cerebro y el mundo físico sin contacto físico alguno.[1]

Como cualquier sistema de comunicación o control una BCI presenta los siguientes componentes de manera esquemática ilustrados en la figura 4 extraída de [2].

- un material proporcionado por el usuario en su actividad electrofisiológica, es decir el EEG registrada en el cuero cabelludo o la superficie del cerebro o la actividad neuronal registrada en el interior del cerebro
- un conjunto de comandos para operar un dispositivo
- componentes que transforman el material de base (input) en comandos(output)
- un protocolo que determina el comienzo, el final y la gestión del tiempo de la

  operación.

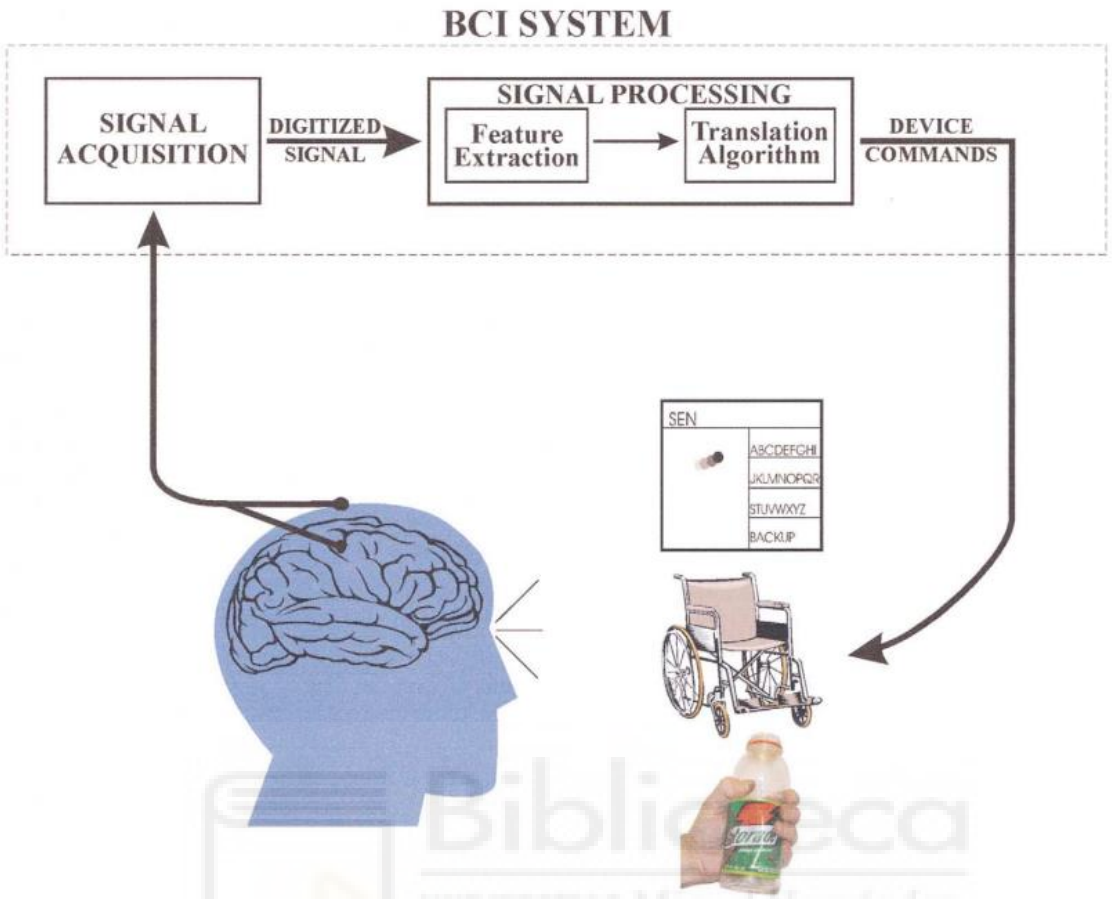

Figura 4: Diseño y operación básica de un sistema BCI

Este esquema puede ser transformado en un paradigma más ajustado al proyecto que nos ocupa. Propondremos una estructura de cuatro bloques:

1. Adquisición de las señales: a través de medios técnicos especializados se recopila la información cerebral analógica y se la convierte en una señal digital, se la amplifica y se aplican filtros para reducir el ruido y lograr una mejor captación.
2. Preprocesamiento de la señal: aplicación de filtros para mejorar la calidad de la señal con el objeto de extraer las características específicas.
3. Extracción de características: a la señal preprocesada se le aplican algoritmos que obtienen las especificaciones más representativas de la información neural que identifican un estado particular del cerebro.

4. Clasificación: el sistema predice el estado al cual pertenece una señal desconocida utilizando las características extraídas en 3.

2.2.1.2 Clasificación de las BMI

Las interfaces cerebro-máquina se pueden clasificar según el uso de metodologías de obtención de señales en invasivas, no invasivas o parcialmente invasivas.

- BMI no invasivas: son las BMI que utilizan electroencefalogramas (EEG) para controlar cursores u otros dispositivos como brazos robóticos.
  Este método se ha revelado eficaz para ayudar a pacientes que sufren parálisis a establecer una comunicación con el mundo exterior.
  Sin embargo, a pesar de poseer la ventaja de no exponer al paciente a los riesgos de una cirugía cerebral, las técnicas EEG proporcionan canales de comunicación de capacidad limitada debido a la existencia de ruidos internos y externos y la interferencia de otras señales o artefactos.
  A pesar de presentar estas desventajas, los métodos basados en el EEG pueden detectar la actividad cerebral y relacionarla con estímulos visuales, intenciones voluntarias o estados cognitivos.
  Se siguen explorando las posibilidades de las BMI basadas en EEG a fin de ofrecer soluciones prácticas para mejorar la vida de los pacientes tales como las BMI para deletrear o para restaurar movimientos a través del control de prótesis.
  Se trata de mejorar la resolución de las señales a través de artefactos como la electromiografía o la electrooculografía.
  Se han desarrollado sistemas basados en EEG que pueden detectar las relaciones entre la actividad cerebral y los estímulos visuales o se han utilizado potenciales visuales evocados (VEPs). Estos sistemas reposan sobre la habilidad de los pacientes al entrenamiento para operar una BMI.
  A fin de proporcionar un feedback a los pacientes en su entrenamiento se ha trabajado recientemente con sistemas de realidad virtual.
  Con el fin de obtener señales de mejor resolución y una disminución del ruido, se ha utilizado también la MEG, la magnetoencefalografía que analiza la actividad cerebral midiendo los campos magnéticos producidos

por las corrientes eléctricas en el cerebro y la fMRI que evalúa la actividad cerebral a través del flujo sanguíneo.

El objetivo esencial de estos enfoques es la detección y separación de señales de control dentro del material a analizar.

Una señal apropiada para nuestro análisis es aquélla:

➔ que puede ser caracterizada para un sujeto en particular

➔ que puede ser modulada para expresar una intención

➔ que puede ser detectada y rastreada sin errores.

- BMI invasivas: son las BMI basadas en registros obtenidos de conjuntos de células cerebrales individuales o de la actividad de múltiples neuronas a partir de la implantación directa de electrodos en el córtex cerebral, por lo general en el córtex frontal o parietal o en ambos.

  Estos enfoques surgen en los años 60 y 70 a partir de estudios pioneros llevados a cabo por Fetz y otros investigadores y posteriormente por Edward Smidt. Estas investigaciones se basan sobre todo en la experimentación animal.

  Si bien las señales obtenidas presentan una mayor calidad, el reto es el desarrollo de matrices de electrodos 3 D biocompatibles que produzcan mínimo daño en su implantación y una inflamación mínima posterior.
- BMI parcialmente invasivas:

  Con el objeto de mejorar la resolución de los potenciales cerebrales estudiados se ha utilizado la Electrocorticografía (ECoG) en la que la implantación de los electrodos requiere una cirugía de menor riesgo.

Las interfaces cerebro-máquina pueden clasificarse también en dos tipos:

- Interfaces espontáneas: se basan en el análisis de señales producidas durante tareas mentales realizadas por el sujeto. Son aquellas en las cuales se controla un dispositivo, un teclado, una silla de ruedas o un brazo robot.

- Interfaces evocadas: son las que hacen uso de un potencial evocado que refleja una respuesta automática del cerebro a un estímulo externo. Sin

embargo, la necesidad de estimulación externa restringe su uso a una serie limitada de tareas.

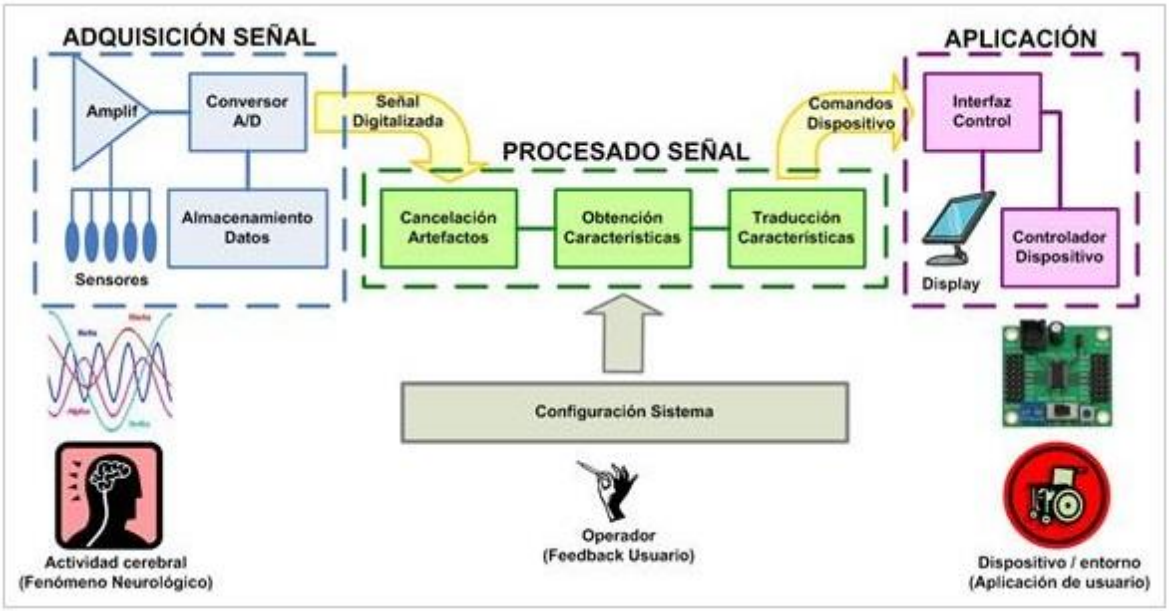

Figura 5: Modelo funcional de interfaz [54]

2.2.2 SISTEMA INTERNACIONAL 10-20 Y 10-10

Los sistemas de electrodos son métodos estándares adoptados por la Federación Internacional de Sociedades de electroencefalografía y neurología clínica para describir la localización de electrodos en el cuero cabelludo cuando se lleva a cabo un electroencefalograma.

El primer sistema adoptado fue el creado por Jasper en 1958 y se llamó 10-20 aludiendo a que las distancias reales entre electrodos adyacentes son o bien el 10% o bien el 20% de la distancia total del cráneo de la parte anterior a la posterior y de derecha a izquierda.

El lugar de emplazamiento de cada electrodo se etiqueta con una letra que identifica el lóbulo o el área cerebral que se está leyendo desde prefrontal (Fp), frontal (F), temporal (T), parietal (P), occipital (O), and central (C).

Del mismo modo, los electrodos con número par (2,4,6,8) son los situados en la parte derecha de la cabeza mientras que los impares (1,3,5,7) se colocan en la parte izquierda.

Las normas establecidas para el correcto emplazamiento del casco de electrodos son las siguientes:

1. La primera referencia son dos puntos en el cráneo: el nasión en la región frontal y el inión en la región posteroinferior.
2. En segundo lugar,se toman como referencia del mismo modo los dos puntos preauriculares (tragus)que se localizan delante de los pabellones auditivos.
3. Se mide la circunferencia del cráneo entre el nasión y el inión pasando por el vértex(línea media del cráneo).En el punto medio de esta trayectoria se sitúa el electrodo Cz.
4. Se mide la distancia entre los puntos preauriculares pasando por el vértex.En el punto medio de esta trayectoria se sitúa el electrodo Cz.

Este sistema resulta muy útil para uso clínico y para estudiar los potenciales relacionados a eventos (ERPs) en las BMIs.Sin embargo, los avances en el estudio de los potenciales evocados y espontáneos y la adopción de sistemas multi-canal hizo necesario el desarrollo de un nuevo sistema, el llamado sistema 10-10 que llevó el número de electrodos de 21 a 74 y que fue adoptado en 1985.

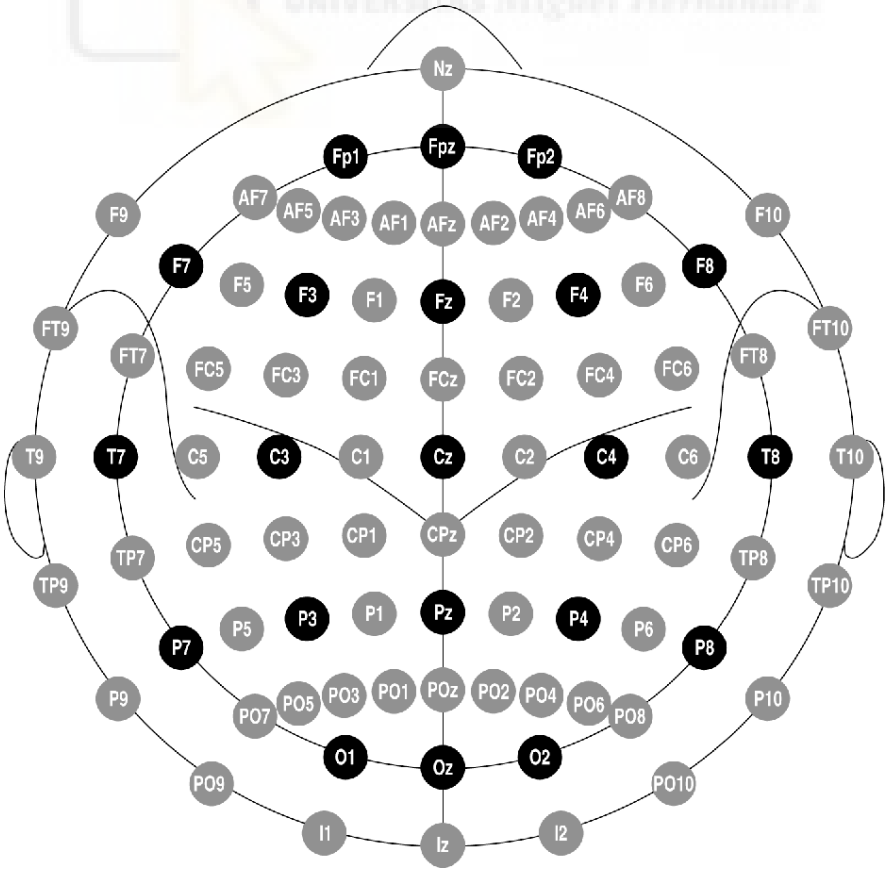

Figura 6: Sistema 10-20 [50]

Actualmente, las investigaciones utilizan muchos más canales que los 21 originales, pudiendo llegar a 64,128 y hasta a 256, para los cuales no existe de momento un estándar. Oostenveld [50] propone la extensión del sistema 10-10 para poder usar 345 electrodos en lo que ha dado en llamar el sistema 10-5 cuyo esquema se muestra en la figura 7.

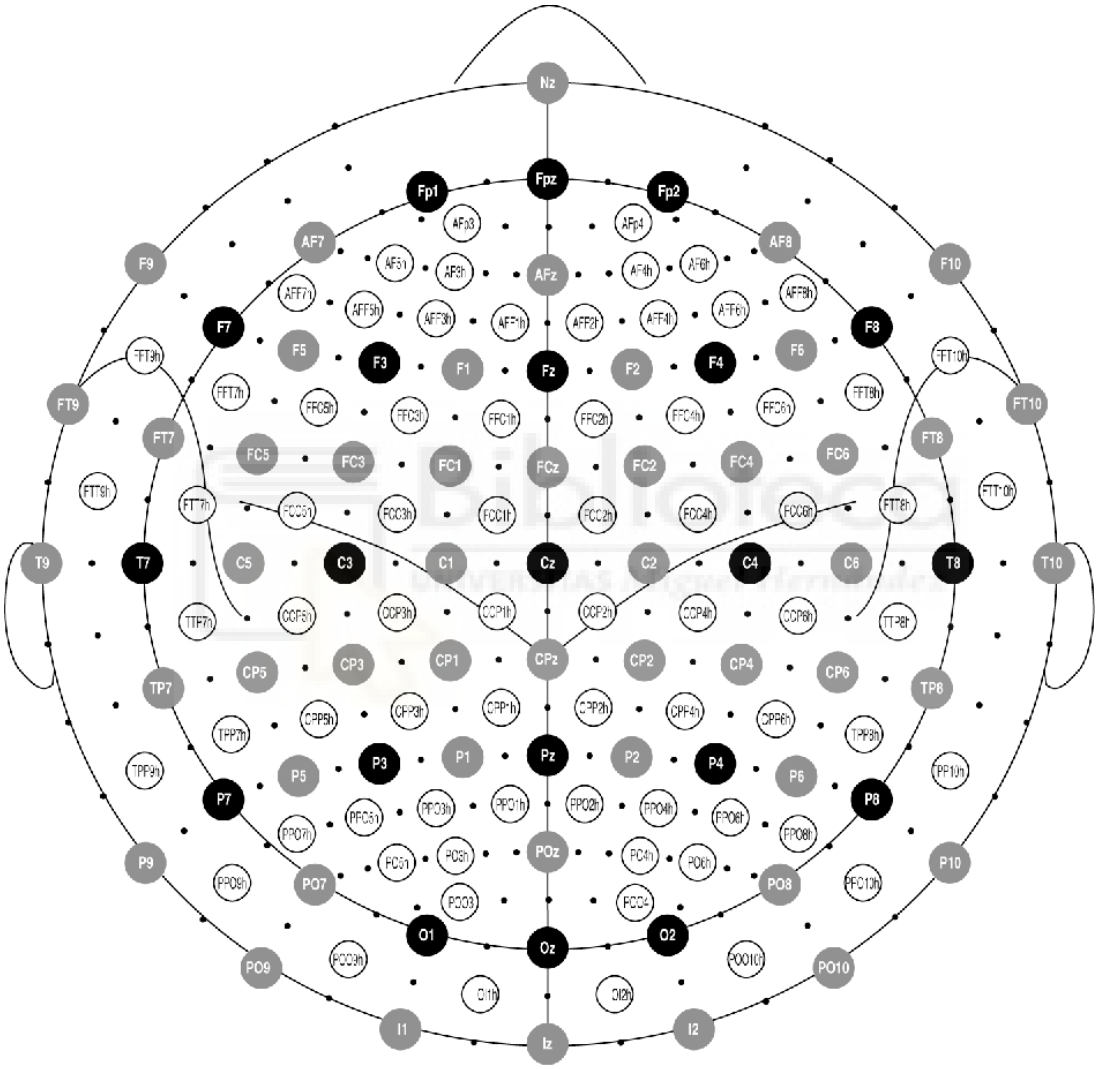

Figura 7: Sistema 10-5 [50]

2.3 Machine-learning

Nuestro trabajo se propone procesar y mejorar las señales cerebrales con el objeto de lograr mayor eficiencia en su utilización. Por lo tanto se encuadra en la disciplina de procesamiento de señales, rama de la ingeniería cuyo objeto es sintetizar, analizar y modificar señales.

Tras la extracción nuestra tarea consiste en lograr una optimización de las señales a través de la aplicación de filtros o a través de métodos de disección y supresión del ruido.

La sinergia existente entre los campos de procesamiento de señales y machine learning puede proporcionarnos un enfoque sumamente eficiente para resolver los problemas en relación con el tratamiento de las señales.

Por lo tanto, definiremos el concepto de machine learning.

ML “aprendizaje automático “es una ciencia que trata del desarrollo de algoritmos que aprenden de los datos.

Arthur Lee Samuel (1959) uno de los pioneros de la inteligencia artificial, ideó la expresión “machine learning” y la definió como el campo de estudio encuadrado en la inteligencia artificial que dota a las máquinas de la habilidad de aprender sin ser programadas explícitamente para ello.[18]

Es la construcción de sistemas capaces de identificar patrones en los datos y realizar predicciones.

Estos sistemas utilizan algoritmos que realizan las siguientes operaciones

1. aprenden de los datos a reconocer patrones
2. clasifican datos en categorías
3. predicen un resultado futuro

Los algoritmos del tipo machinen learning se pueden dividir en tres tipos:

- El ‘aprendizaje por refuerzo’ es la técnica mediante la cual una máquina aprende interactuando con el entorno realizando tareas de toma de decisiones.La máquina opera por medio de prueba y error hasta alcanzar la mejor manera de completar una tarea dada.Si el sistema es recompensado se entrena para aprender a modificar su conducta y llegar a una solución que no tiene una forma Impuesta definida.

- ‘Aprendizaje supervisado’, es un enfoque de aprendizaje predictivo cuyo objetivo es aprender de una serie de datos etiquetados.Estos datos proporcionan los ejemplos para entrenar a la máquina en la clasificación y predicción posterior.En el léxico de la máquina,”características” son los datos de entrada y “variables de respuestas” los datos de salida.En el caso del ‘aprendizaje no supervisado’, las máquinas buscan similitudes,patrones comunes.Los algoritmos agrupan datos que se parezcan.Se trata de un proceso de conocimiento por descubrimiento en el cual se separan los patrones del ruido, que carece de estructura.

El sistema de machine learning es muy eficiente si se lo compara con sistemas anteriores,ya que es capaz de adaptarse a los cambios que se producen en los datos con los que está aprendiendo.Por lo tanto, la máquina continúa optimizando su acción constantemente.
En el presente trabajo, se han utilizado los algoritmos siguientes:

2.3.1 KNN: K Nearest Neighbors.

Es uno de los más antiguos y más ampliamente utilizados para clasificación. Se considera un método muy poderoso de clasificación que permite la clasificación de una instancia desconocida usando un juego de entrenamiento de instancias clasificadas.
Es muy eficaz cuando hay muy poco o no hay conocimiento acerca de la distribución de los puntos de datos.
El valor k es el que especifica cuántos vecinos próximos se deben considerar para definir la clase de un punto de datos o una consulta.
KNN presenta las siguientes ventajas sobre otros algoritmos:

- tiempo de entrenamiento muy rápido
- fácil de aprender
- gran simplicidad algorítmica
- robusto frente a los puntos de data de entrenamiento que contienen ruido
- eficiente en amplias bases de datos de entrenamiento.

Sin embargo, en bases de datos demasiado amplias puede evidenciar limitaciones de memoria.

2.3.2 Support Vector Machines

Este algoritmo es un conjunto de métodos de aprendizaje supervisado que se aplican para resolver problemas de clasificación lineal y no lineal en los cuales construye un modelo que atribuye nuevos ejemplos a cada categoría.
Trabaja creando un mapa con los vectores de entrada dentro de un espacio de grandes dimensiones y construyendo el hiperplano de separación óptimo reduciendo el riesgo estructural.
Por lo tanto, un modelo SVM es una representación de ejemplos como puntos en el espacio, como si estuvieran separados por una separación nítida lo más ancha posible.A continuación, nuevos ejemplos se localizan en este mapa donde se predice su pertenencia a la brecha/hueco donde se los sitúa.
Este algoritmo presenta tres ventajas principales:

- Es eficiente en espacios de grandes dimensiones
- Cuenta con una memoria eficiente que es un subconjunto de puntos de entrenamiento en los vectores de soporte o en las funciones de decisión.
- Se le considera dotado de gran versatilidad ya que puede contener diferentes funciones kernel (lineal,polinomial,RBF)en calidad de funciones de decisión.

  Por contra,un aspecto que podría ser discutible es que este algoritmo hace clasificaciones binarias y soporta problemas que incluyan dos clases mientras que los problemas en la vida real comprenden más que dos clases.

2.3.3 Gradient Boosting Classifier

Es un conjunto de algoritmos de machine learning creado por Jerome Friedman, que incluye varios pequeños modelos que se combinan para formar un modelo mayor con un gran poder predictivo.
Es un algoritmo muy utilizado por su habilidad en la clasificación eficiente. El modelo que propone es el del árbol de decisiones. Gradient Boosting tiene tres componentes principales:

- Función pérdida: el papel de esta función es elaborar una evaluación del modelo cuando hace predicciones con los datos que se proporcionan.
- Aprendiz débil:es aquel que clasifica los datos pero lo hace tan mal,quizás como si operara adivinando al azar.En otras palabras, tienen una tasa de error muy elevada.El modelo típico es el de los árboles de decisión.
- Modelo aditivo: es un enfoque iterativo y secuencial que va agregando los árboles elaborados por el aprendiz débil uno a uno.Después de cada iteración deberíamos estar más cerca del modelo final.Dicho de otro modo,cada iteración reduce el valor de la función pérdida.

2.3.4 Naïve Bayes

El clasificador Naïve Bayes es una demostración clásica de la manera como las hipótesis generativas y las estimaciones de parámetros simplifican el proceso de aprendizaje. Se supone que las variables explicativas son independientes de los objetivos variables. Esta suposición reduce la complejidad y permite que el algoritmo pueda aplicarse en numerosas áreas. Su rendimiento depende de la calidad de la estimación de las distribuciones y de la selección de las variables informativas y explicativas. Sus ventajas son :

- Desviación estándar baja
- Necesidad de poco entrenamiento
- Poca complejidad del tiempo de predicción

2.3.5 Random forest

El clasificador Random forest se usa normalmente para resolver tareas de

clasificación múltiple en machine learning. Este algoritmo predice una clasificación correcta de resultados con un tiempo de entrenamiento mínimo si se le compara a otros clasificadores y puede generar varios árboles de decisión. El algoritmo está compuesto por una colección de árboles de decisión y cada árbol del conjunto está compuesto por una muestra de datos que se extraen de un esquema de entrenamiento. Un tercio de esta muestra se aparta para usarla como test de datos. Entonces se inyecta una instancia al azar lo que agrega más diversidad al conjunto de los datos y reduce la correlación entre los árboles de decisión.

Dependiendo del tipo de problema, la determinación de la predicción variará. Si se realiza un modelo de regresión, se obtiene un valor medio, si se trata de una tarea de clasificación, un voto de mayoría determinará el valor de la clase que se predice.

La muestra que se ha apartado anteriormente servirá como validación cruzada.

2.3.6 MLP Multilayer perceptron

Se trata de un algoritmo que se presenta con un modelo de una red neuronal artificial y que establece un mapa en el cual los conjuntos de datos de entrada corresponden a datos de salida apropiados.

Presenta múltiples capas y cada capa está totalmente conectada con la siguiente.

Los nodos de las capas son neuronas que poseen funciones de activación no lineales,excepto en el caso de los nodos de la capa interna.

Entre la capa de entrada y la capa de salida puede haber una o más capas no lineales ocultas.

2.3.7 Árbol de decisión

Es uno de los algoritmos de técnicas de aprendizaje supervisado más importantes. Se presenta en la forma de ramas, nodos y hojas y las decisiones se toman desde la raíz del árbol hasta la hoja.

En el extremo de cada rama, hay una hoja que presenta el resultado obtenido. Los nodos intermedios en el árbol contienen un test sobre un atributo en particular que distribuye puntos de datos en los diferentes sub-árboles..

El objetivo de la fase de aprendizaje es el de identificar grupos de ejemplos tan coherentes como sea posible mientras que en la fase de testeo el nuevo ejemplo se atribuye a la clase mayoritaria de la hoja basándose en una puntuación que corresponde a la proporción de ejemplos de entrenamiento en la hoja que pertenece a la misma clase.

Este algoritmo se usa frecuentemente porque presenta varias ventajas:

- flexibilidad y aplicabilidad a un amplio espectro de problemas
- El conjunto de reglas resultante es a la vez exclusivo y exhaustivo: una sola regla cubre cada instancia.

2.3.8 Procesos gaussianos

Se trata de un método genérico de aprendizaje supervisado que se utiliza para resolver problemas de regresión y de clasificación probabilística.

Presenta las siguientes ventajas:

- La predicción interpola las observaciones para los núcleos regulares.
- La predicción es probabilística (Gaussiana) de manera que hace posible el cálculo de intervalos empíricos de confianza para decidir partiendo de esta información el ajuste (en línea o adaptativo) de alguna predicción en alguna región de interés.
- Es versátil: permite la especificación de diferentes kernels comunes o personalizados.

Presentan asimismo desventajas:

- Utilizan toda la información de muestras o características para elaborar la predicción.
- No es eficaz en espacios de grandes dimensiones, sobre todo a partir de algunas docenas de prestaciones.

2.3.9 Red neuronal

Se trata de un algoritmo inspirado por la estructura de las redes neuronales del cerebro que consiste en un gran número de dispositivos/nodos/neuronas conectadas entre sí a través de enlaces. Cada neurona recibe una suma de pesos de los puntos de salida de las neuronas conectadas a sus enlaces de entrada.

Los pesos pueden representar valores a lo largo del proceso de aprendizaje hasta que se alcanzan los pesos óptimos y se conservan, en tanto que fortalezas de las interconexiones entre las neuronas.

Sus principales características son:

- habilidad para utilizar gran cantidad de información sensorial
- capacidad de procesamiento colectivo
- habilidad para aprender y adaptarse a los cambios y a la nueva información.

2.4 Aplicaciones

Tal vez las aplicaciones sean la parte más trascendente de nuestro proyecto ya que la creación de una arquitectura de software, apta al procesamiento de señales electroencefalográficas, tiene por objeto atender a un sector de nuestra población que evidencia necesidades especiales o dependencia.

Nuestra acción contribuye a la traducción de la actividad cerebral en acciones ya que permite la interacción entre los seres humanos y los dispositivos artificiales en el marco de la tecnología de las BCI-BMI.

De entre los múltiples trabajos que se realizan en este ámbito podríamos citar algunos de los que se llevan a cabo en el BMI Lab(Brain Machine Interface Systems Lab) de la Universidad Miguel Hernández de Elche:

- WALK Control de exoesqueletos para las extremidades inferiores por medio de interfaces cerebro computador para asistir a personas con discapacidades motoras. Se trata de crear estructuras robustas capaces de controlar los exoesqueletos para que puedan caminar y detenerse, girar a la derecha y a la izquierda, aumentar o disminuir la velocidad o

detenerse frente a obstáculos imprevistos.Se ha mejorado el rendimiento de las BMIs implementando estrategias de entrenamiento basadas en estimulación intracraneal y realidad virtual.

- REKINE:reconstruyendo trayectorias de marcha a partir de señales EEG. Se trata de interactuar con exoesqueletos robóticos por medio de BMIs basadas en señales electroencefalográficas.Puesto que la corteza cerebral aparece particularmente activa durante ciclos específicos del ciclo de la marcha, se trata de decodificar las señales que contienen información acerca del ángulo de las articulaciones .Se ha verificado que es posible establecer una relación entre los ángulos de los miembros inferiores y las señales usando modelos de regresión lineal. Por el momento, es necesario recopilar más señales para desarrollar, verificar y comparar los algoritmos.Las señales obtenidas y el algoritmo optimizado se integrarán en la base de datos de EUROBENCH para futuros desarrollos e investigaciones.

De forma más general, se podrían mencionar las aplicaciones siguientes:

- las interfaces oculares para interaccionar con dispositivos. En (Azorín et al.,2010) [55] se describe una interfaz ocular basada en electrooculografía que permite accionar un brazo robótico.
- prótesis y ortesis de exoesqueletos de miembros superiores o inferiores.
- utilización de señales electromiográficas (EMG) para detectar la intención del movimiento. La detección de las señales evidencia las estrategias del cerebro, la activación de los músculos, la fatiga. Las señales EEG son la información de entrada y se usan para accionar dispositivos robóticos, sistemas de rehabilitación o sillas de ruedas, por ejemplo.

2.5 Lenguajes de programación

La primera decisión con la que nos enfrentamos en el presente trabajo es la de elegir el lenguaje de programación más adecuado al desarrollo de la arquitectura

de software que procesa y clasifica las señales. Las tres opciones disponibles son el lenguaje de programación C, Matlab y Python.

En primer lugar, C se define como un lenguaje de programación compilado, de bajo nivel (lenguaje de programación de 3er nivel) cuyas principales ventajas son su rapidez y eficiencia sobre todo en sistemas integrados.
Sin embargo, presenta inconvenientes como la tendencia a la sobrecarga, la necesidad de establecer variables previamente a que las instrucciones se ejecuten y la obligación de compilar antes de ejecutar el código. Otros problemas se plantean cuando, si la indentación no es correcta o si se utilizan de forma errónea signos de puntuación o símbolos como llaves o paréntesis o si el formato numérico no es adecuado la compilación se detiene. Esta hipersensibilidad del lenguaje puede generar comandos o cálculos erróneos lo que ralentiza el proceso de programación.

Matlab (Matrix Laboratory) por contra, se creó como una plataforma de programación que era una matriz interactiva de cálculo y se convirtió en un lenguaje de programación de alto nivel o 4ª generación. Su ventaja más evidente si lo comparamos con el lenguaje C es que el código se interpreta en el momento de ejecución del programa, no se requiere compilación. Por un lado, si bien la velocidad de ejecución es menor con relación a C, permite la escritura dinámica y las sesiones interactivas donde el usuario escribe instrucciones que se ejecutan inmediatamente. Requiere menos tiempo para la codificación, para la depuración de los errores y para el desarrollo en general. Matlab es sumamente potente en cálculo y operaciones matriciales, es más rápido que Python y es particularmente eficaz en el procesado de imágenes con su Simulink Toolbox. Cuenta además con potentes librerías y su propio entorno de trabajo. Su gran inconveniente es que, si bien originariamente su creador Cleve Moler en 1960 lo distribuyó gratuitamente a las universidades, a partir de 1980 se convierte en un producto comercial cuyas licencias tienen un coste elevado.

Python es un lenguaje de programación de código abierto, de alto nivel y adecuado a un uso general desarrollado por Guido van Rossum en 1991. Se encuadra en el enfoque de programación orientada a objetos ya que apunta a

los datos más que a la lógica o las funciones. Es a simple vista muy similar a Matlab: se interpreta en el momento de la ejecución, permite sesiones interactivas, escritura dinámica y gestión automática de la memoria. Si bien es cierto que Matlab es más rápido en general Python presenta numerosas ventajas comparativas. Con Python es posible gestionar múltiples tareas simultáneamente, presenta una sintaxis sumamente intuitiva y clara y entornos de desarrollo integrado adaptables a las necesidades del usuario tales como Anaconda, Spyder y Júpiter que funcionan tan eficazmente como el entorno de desarrollo integrado que Matlab proporciona por defecto.De la misma forma que Matlab, Python proporciona un gran número de librerías entre las cuales cabe mencionar NumPy, SciPy, PyTorch, OpenCV Python, Keras, TensorFlow, Matplotlib, Theano, Requests, and NLTK. La gran ventaja es que el acceso a estos recursos es gratuito y que los usuarios pueden libremente diseñar sus propias herramientas (GUI toolkits). En lo que respecta a Machine-Learning (aprendizaje automático) y deep Learning (aprendizaje profundo) Python aventaja claramente a Matlab. Los marcos más usados de aprendizaje automático por ejemplo Scikit-learn están escritos en Python y puesto que el código es modificado y enriquecido permanentemente por la comunidad de usuarios la innovación es rápida y constante.

Las razones siguientes:

- Potentes herramientas de aprendizaje automático
- Sintaxis accesible y flexible para los usuarios (desarrolladores e investigadores)
- Carácter gratuito
- Innovación permanente

hacen que Python sea la mejor opción para nuestro proyecto.

Capítulo 3. Material y métodos

En el siguiente capítulo se describe el formato de los datos de entrada. Posterior y detalladamente, se describe el proceso de desarrollo de una arquitectura software para el procesamiento y clasificación de estas mismas señales. En el proceso de explicación del desarrollo del software, se detallarán también aspectos importantes sobre la elección del lenguaje de programación y las metodologías empleadas para asegurar una futura ampliación del software o su integración en otras soluciones.

3.1 Descripción del formato de datos EEG

Para la realización del trabajo de fin de grado se emplean un set determinado de archivos de datos reales, proporcionado por BCI Competitions, competición mundial que pretende proveer de datos neurocientíficos de alta calidad y de libre acceso a la comunidad científica.

En este estudio, se seleccionaron los datos de 3 sujetos (suponiendo un total de 12 archivos, 4 por usuario) que contienen las sesiones de registro de datos, siguiendo una estructura estandarizada. Se trata de sesiones de datos continuos multi-clase, lo que significa que en una misma sesión (en nuestro caso de 4 minutos) se efectúan múltiples tareas. A continuación, se detalla, como ejemplo, la estructura de los archivos con los datos del usuario 1, pudiendo no ser exactamente la misma cantidad de muestras para otros usuarios.
Por tanto, se trata de una matriz de datos que contiene 32 líneas (una por cada electrodo) y 122368 muestras, lo que da un total de casi 4 millones de puntos de datos. Un vector de 1 fila y 122368 muestras, conteniendo para cada muestra, un valor predeterminado que hace referencia a la tarea que se estaba realizando al momento de tomar la muestra.

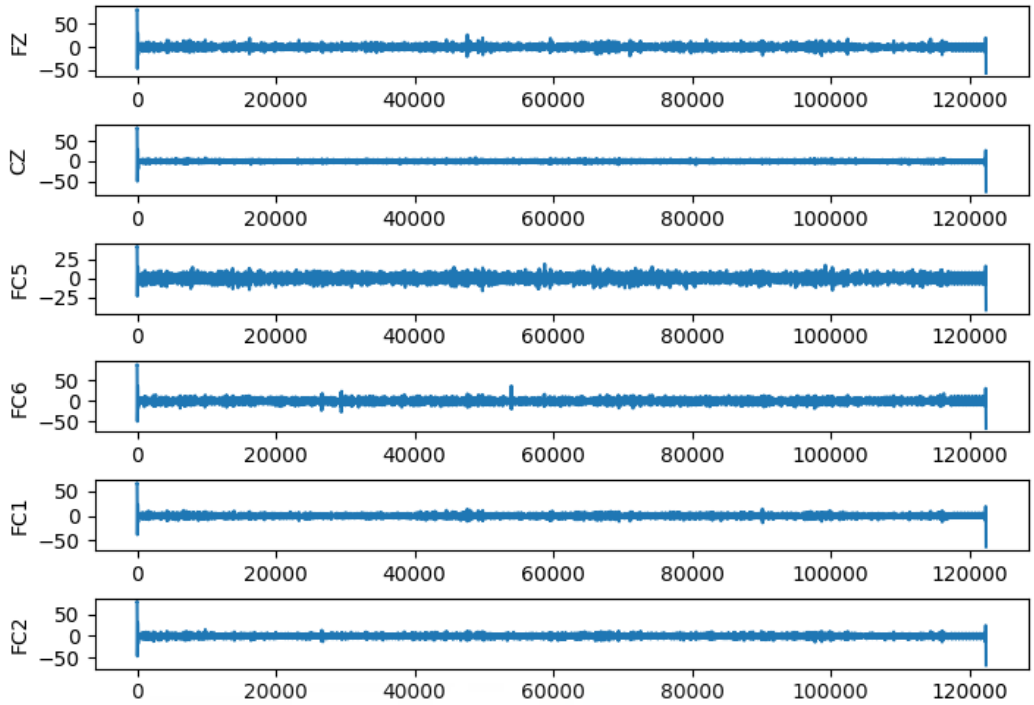

Figura 8: representación de las señales de los electrodos FZ, CZ, FC5, FC6, FC1, FC2

Por tanto, si por cada sujeto tenemos 4 archivos, esto se traduce en una matriz de 15 millones de datos y para cada una de los 15 millones de muestras, su correspondiente etiqueta. Los datos fueron digitalizados a razón de una muestra cada 1'95 milisegundos (512 Hz como frecuencia de muestreo) cuya banda de frecuencias es de 0-512 Hz.

A continuación, se detalla la estructura interna de las sesiones que ha sido dividida en diccionarios:

- conf: contiene la configuración y descripción de la estructura de los datos contenidos en el archivo. En este apartado del archivo podemos obtener el número de canales (electrodos), posición de los electrodos (ejes y/x), frecuencia de muestreo, descripción del set de data y códigos de configuración.

- data: matriz que contiene los datos obtenidos en cada punto del tiempo (122368 puntos de datos) de los 32 electrodos para cada momento del tiempo.

- task: vector que contiene la etiqueta de la tarea que se realizaba en cada punto del tiempo, por tanto, 122368 puntos cuyo valor es:
    - Imaginación de una palabra: 122
    - Imaginación de movimiento de mano derecha: 123
    - Imaginación de movimiento de mano izquierda: 127

- complete: booleano que expresa si la medición se completó.

- time: contiene dos marcas de tiempo, las cuales se corresponden al momento en el que comenzó la medición y el momento en que se detuvo la lectura de datos.

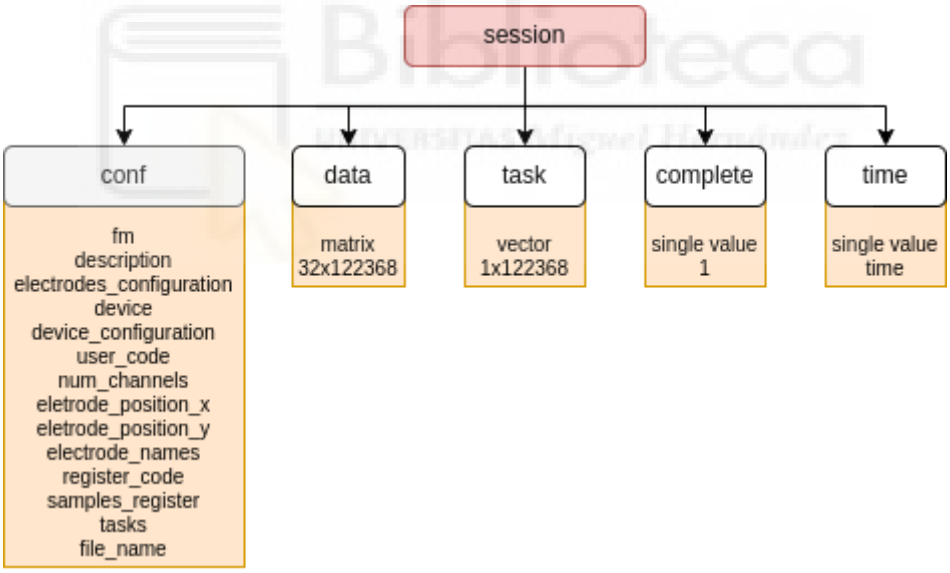

Figura 9: representación de la estructura interna de los archivos de BCI Competition III, dataset V

Puesto que los archivos se tratan en una arquitectura desarrollada con el lenguaje de programación Python y estos son de formato matlab (con extensión .mat), para poder trabajar con los datos que contienen, debemos adecuarlos a la forma en la que Python pueda tratarlos lo más eficientemente posible.

Para ello, se desarrolló una clase en Python que limpia y da formato a los datos contenidos en estos archivos, empleando dos librerías ampliamente usadas en el tratamiento y análisis de datos: Pandas y Numpy.

3.2 Arquitectura de software

Con el objeto de visualizar los objetivos a alcanzar, hemos construido un marco previo de interacción con el código desarrollado. Para ello se define una arquitectura a alto nivel con la intención de identificar los bloques lógicos de los que se debería componer el código que posteriormente se desarrolla para cubrir las necesidades que se establecieron.

3.2.1 Principios empleados

En el proceso de desarrollo del software se han aplicado distintas técnicas y metodologías, algunas de las cuales se exponen en manuales como *Código limpio: Manual de estilo para el desarrollo ágil de software*, escrito por Robert C. Martin, conocido como Uncle Bob. El autor trabaja en desarrollo y programación desde 1970 y es uno de los profesionales que lideró el Manifiesto Ágil (2001). Por esto y por la fuerte adopción de estas técnicas y metodologías es por lo que se decide desarrollar el código del presente trabajo tomando en cuenta esta corriente de código limpio. [56]

Un sistema informático nunca está totalmente finalizado, pues existe la necesidad constante de agregar funcionalidades y desarrollar actualizaciones. Con este pretexto, lo que se pretende es escribir un código que sea fácil de entender y por tanto, más fácil de mantener y actualizar.

Para lograr este fin, primeramente se aplicaron los principios SOLID en la mayoría de los casos, teniendo en cuenta las posibles limitaciones. Por lo tanto, si se hace un análisis del código, en ciertas partes podremos ver una fuerte influencia de estos principios y en otros, por cuestiones de eficiencia, no se aplicaron algunos de los principios que posteriormente se van a detallar.

En ingeniería de software, SOLID es el acrónimo de cinco principios cuyo objetivo es hacer los desarrollos fáciles de comprender, flexibles y mantenibles. Los principios por sus títulos en inglés son:

1. *Single responsibility principle*: Una clase debería tener una sola función que llevar a cabo.
2. *Open/Closed principle*: Una clase debería ser escalable sin modificar su comportamiento.
3. *Liskov substitution principle*: Una clase derivada de otra debería ser usada de la misma manera que la clase padre. La clase derivada hereda el comportamiento de la clase padre.
4. *Interface segregation principle*: Un cliente no debería estar expuesto a funcionalidades y métodos que no necesita.
5. *Dependency inversion principle*: Las entidades deben tener dependencias sobre abstracciones. En un árbol jerárquico de funciones, una función de nivel superior no debe depender de una función del nivel jerárquico inferior.

De esta forma, el proyecto se dividió en múltiples archivos de python, ilustrados en una estructura de árbol representado en la figura 10, cada uno de ellos contiene una lógica concreta para llevar a cabo un proceso. Dentro de cada uno de ellos, se observa una multiplicidad de funciones cada una de las cuales tiene un cometido concreto.

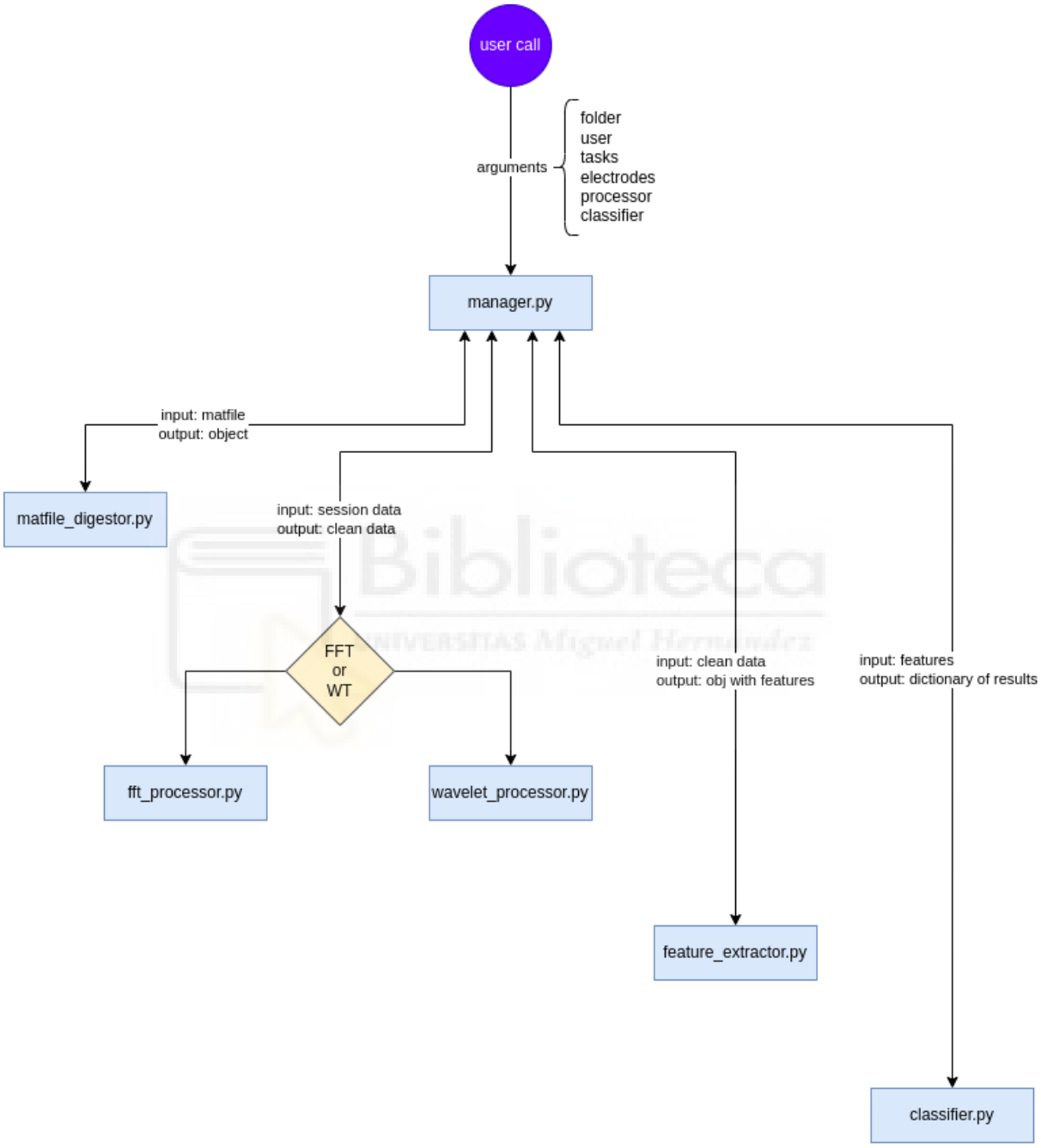

Figura 10: disección de la estructura de archivos

A continuación, se describen los archivos:

- manager.py

  Contiene, en código, el orden de ejecución a seguir. Dentro de este archivo, podremos encontrar la definición de los argumentos de entrada (tanto obligatorios como opcionales) los cuales definirán qué otras clases se llamarán durante el proceso para completar el proceso de BCI (procesamiento, extracción de características y clasificación de características)

- matfile_digestor.py

  Contiene la clase *MatFileDigestor* que traduce el formato del archivo introducido (archivo matlab con una estructura característica de diccionarios y listas anidadas) a un objeto que incluye distintos atributos en los cuales se dividen los datos extraídos del archivo.
  Por tanto, podremos acceder a las mismas etiquetas que previamente se describieron (conf, data, task) directamente seleccionando el atributo que tiene el mismo nombre dentro del objeto de esta clase.

- fft_processor.py

  Contiene la clase *FFTProcessor* la cual divide la señal, la procesa para extraer la banda de frecuencia deseada y calcula sobre cada una de las ventanas, la transformada rápida de Fourier.

- wavelet_processor.py

  Contiene la clase *WaveletProcessor* en la cual se efectúan las operaciones pertinentes para limpiar la señal de las frecuencias que están fuera del estudio a realizar y se construye un árbol de descomposiciones de wavelets que más adelante se explicará.

- feature_extractor.py

  Contiene la clase *FeatureExtractor* en el cual se encuentran los métodos de extracción de características sobre un vector de datos previamente limpiados y procesados. Entre ellos se encuentran características tanto

estadísticas como de resultados de cálculos específicos como la entropía, energía, porcentaje de energía por bandas, etc.

- classifier.py
  Contiene la clase Classifier que aúna todos los clasificadores que más adelante se presentarán, siendo posible seleccionar el clasificador con el que queremos trabajar. Esta clase también, para algunos de los clasificadores, provee al usuario la posibilidad de calcular un set de resultados de la clasificación con distintos parámetros de entrada que automáticamente se combinan entre sí para obtener la mejor configuración de ellos.

Como se comentó anteriormente, la atomización del código ha sido un elemento crucial en el desarrollo de la arquitectura.
Puesto que se trabaja con una ingente cantidad de datos, es esencial reducir el tiempo de ejecución del código para aumentar su eficiencia. Para ello, se añaden funciones de base de python y librerías con las que se pretende lograr este propósito.

3.2.2 Pruebas unitarias y tests funcionales

3.2.2.1 Pruebas unitarias

Las pruebas unitarias son a bajo nivel, por lo cual se prueba de forma individual las funciones y métodos (clases, componentes o módulos usados por nuestro software).

Al tratarse de pruebas específicas, generalmente se automatizan y, en algunos casos, se desarrollan sistemas completos en un servidor para llevarlas a cabo. A esto último se le conoce como integración continua.

En el presente trabajo el acercamiento es más simple, ejecutando estos tests en la misma máquina donde se desarrolla el software, utilizando métodos como *assert* de python.

El proceso de desarrollar los tests se hizo teniendo en cuenta las siguientes precisiones:

- Aislamos la funcionalidad hasta un punto en que no se puede atomizar más, y entonces escribir pruebas sobre ese código que es independiente del resto.
- Se debe verificar, también, que los nombres y tipos de los parámetros sean correctos, y así mismo el tipo y valor de lo que se devuelve como resultado.
- Dado que las pruebas unitarias no deben tener ningún tipo de dependencia, toda llamada/dependencia externa se imita con código auxiliar.

3.2.2.1 Pruebas funcionales

Las pruebas funcionales se centran en los requerimientos de una aplicación, verificando la salida de un proceso, tomando el código como un todo, sin centrar el estudio en lo que ocurre a bajo nivel mientras se lleva a cabo la ejecución. Para ello, se usaron datos que previamente se habían seleccionado para probar casos extremos y con ello mejorar el manejo de los errores por parte de las distintas funciones que componen el proceso completo.

3.2.3 Optimización del código

A continuación, se detallan algunas de las acciones que se tomaron para mejorar tanto la eficiencia del código/proceso como la legibilidad.

3.2.3.1 Métodos Base

En el caso de funciones de base de python, podemos destacar la inclusión de los generadores. Los generadores son funciones que crean iteradores en base al código que escribimos dentro de ellos.

```
def generate_electrodes_data(self: object, data: dict) ->
iter:
    for electrode in data.keys():
        if(electrode in self.electrodes):
            yield electrode, data[electrode]
```

A simple vista podría parecer una función normal con la particularidad que la palabra reservada para retornar la salida es *yield* y no *return* lo que devuelve un objeto iterador. Las ventajas que nos entrega este método son las siguientes:

- La función solo se ejecuta al momento de ser invocada.
- El control de la ejecución se le confiere a la función que invoca al generador.
- En nuestro caso, al tratarse de una función que devolveria una lista de tuplas si no fuera un generador, agregando la palabra reservada yield justo después del condicional del bucle, se devolverá un elemento de esa lista cada vez que se ejecute, otorgando control total y descendiendo la cantidad de memoria usada en el proceso, disminuyendo ampliamente el tiempo de ejecución total.

3.2.3.2 Anotaciones

Las anotaciones en la definición de funciones se introdujeron con el PEP 3107 (PEP: Python Enhancement Proposal) añadiendo la posibilidad de agregar anotaciones de metadatos.

```
def task_window_intervals(self: object, tasks: list, fm: int,
window_interval: int = 5, mode: str = None, overlapping: int =
None) -> dict:
     (…)
```

Estas anotaciones no mejoran directamente el código, puesto que por sí solas solo agregan un dato que nos ayuda a comprender, como desarrolladores, que tipo de dato espera la función. Con lo cual cualquier desarrollador externo, con

poco esfuerzo, puede cometer menos errores y hace más fácil implementar mejoras.

Además, python contiene el método __*annotations*__ con el cual podemos obtener un diccionario que describe los inputs y outputs de la función, dando de esta forma capacidad al desarrollador para comprender mejor el código y comportamiento de la función. Por otro lado, si incluimos librerías externas que hacen uso de estas anotaciones, podemos comprobar si la integridad y coherencia se cumplen.

3.2.3.3 Pandas

Pandas es un paquete de python creado para gestionar de forma veloz, potente y flexible grandes cantidades de datos estructurados. Además de esto, el uso de Pandas, concretamente los *DataFrame* (2D) y *Series* (1D), es increíblemente simple y eficiente.

Puesto que la cantidad de puntos de datos en este proyecto es de una gran magnitud (millones de puntos de datos a través de varios ficheros) y se descartó el uso de bases de datos para almacenarlos, se decidió formatear toda la información de las lecturas en estructuras de datos de Pandas.
Por ende, lo que se consiguió es una fuerte capacidad de filtrado y cálculo masivo, pudiendo operar con sets de millones de datos a la vez, siendo extremadamente sencillo, por ejemplo, calcular y almacenar características.
Podemos observar a lo largo del código producido, sobre todo la clasificación, constantes operaciones de unión y filtrado usando esta librería.

3.2.3.4 Numpy

Por supuesto, si incluimos Pandas como librería para gestionar grandes volúmenes de datos, Numpy es un módulo de python que está especializado en el cálculo matemático de características y formateo a bajo nivel de los tipos de datos.

En este punto, sumando estas librerías y métodos, se pudo descender el tiempo total de ejecución en un 98%, pasando de ejecutarse el código en

aproximadamente 2 horas a escasos 2-4 minutos para un set de datos de unos 15 millones de puntos. También se obtuvo una alta legibilidad en el proceso, permitiendo que otros desarrolladores escalen, mejoren y corrijan el código del proyecto.

3.3 Método propuesto

En el análisis de las señales de las lecturas efectuadas, se tomaron en cuenta dos métodos de procesamiento. En el presente trabajo se comenzó usando la transformada rápida de Fourier y calculando, sobre los datos procesados, la clasificación de los mismos con KNN (en inglés: K-Nearest-Neighbor).

Más adelante, cuando se logró reproducir y optimizar el código, inicialmente escrito en matlab, en python usando las dos herramientas previamente mencionadas, se decidió desarrollar un procesador de señales basado en la transformada de wavelet (WT) para descomponer y limpiar las señales originales y procesar los datos que más adelante alimentaria el extractor de características.

Una vez incluido como procesador alternativo la transformada de wavelet, se comparan los resultados obtenidos con ambos procesadores y se determina, de forma empírica, cuál de los dos es el más apropiado para el tipo de señales que tenemos que procesar.

A continuación, se exponen los detalles de lo previamente introducido.

3.3.1 Preprocesamiento

Los datos con los que se trabaja están basados en la configuración de electrodos 10-10, teniendo 32 electrodos disponibles. Para la elección de los electrodos, probamos con varias combinaciones de los mismos, teniendo en cuenta también, las regiones cerebrales en las que se producen las señales que distinguen unas tareas de otras.

Por tanto, de los 32 electrodos se eligieron, tras más de 40 test combinatorios, los siguientes: FZ, CZ, FC5, FC6, FC1, FC2, representados en el diagrama de electrodos de la figura 11.

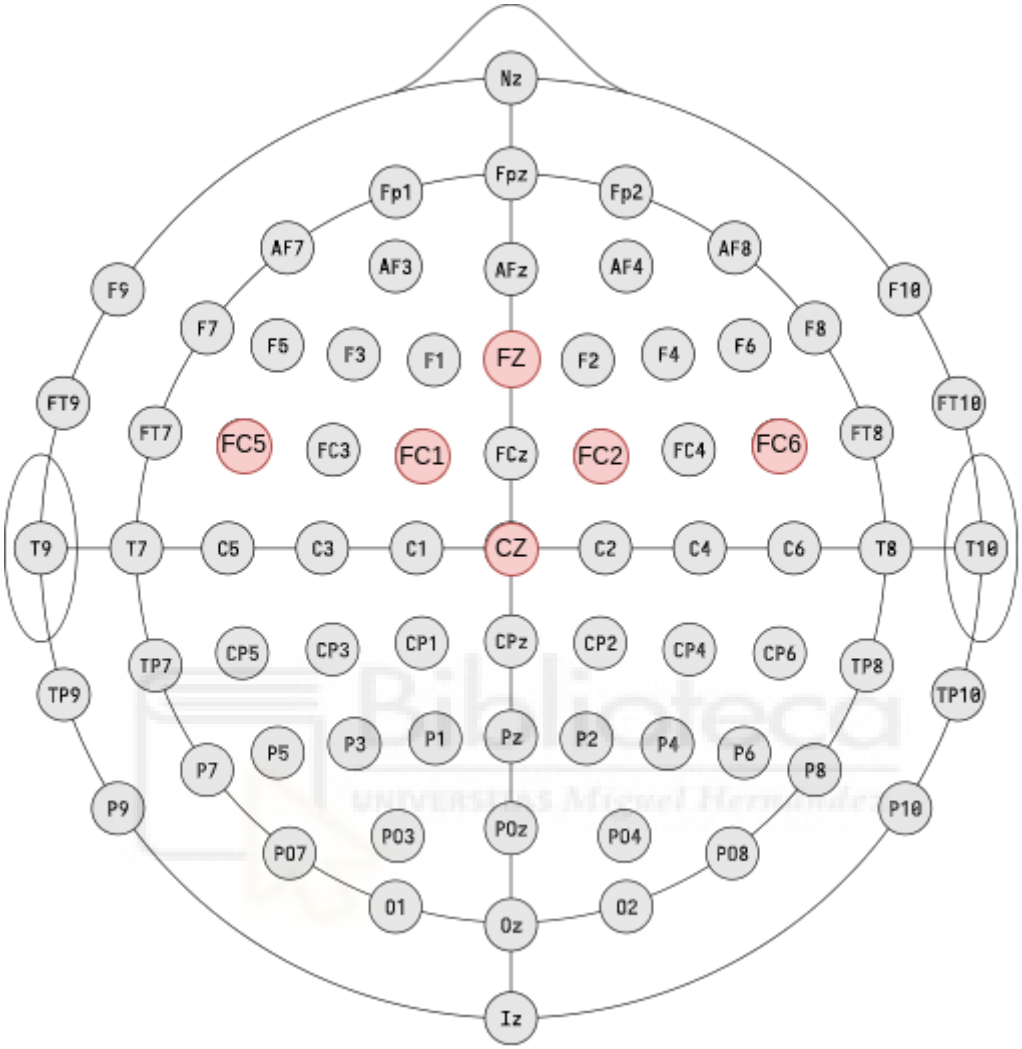

Figura 11: sistema 10-20 con los electrodos usados

Con respecto a la banda de frecuencias, en este caso las señales están comprendidas entre los 0 Hz y 512 Hz. Por lo tanto, hemos filtrado la señal de entrada para que tenga las frecuencias que se corresponden con las ondas α y β (de 8 Hz a 32 Hz).

Este filtrado de frecuencias se realiza con la transformada discreta de wavelet, la cual más adelante se explica, con la que descomponemos la señal de entrada en subdivisiones, cada cual con una banda de frecuencias. Una vez

descompuesta, se eliminan los términos que contienen las frecuencias que queremos eliminar y se reconstruye.

3.3.2 Procesamiento

A lo largo de la historia moderna se han desarrollado métodos y técnicas de procesamiento digital para la detección y evaluación de funciones en señales.

El objeto del análisis espectral es analizar en detalle los componentes armónicos de la señal en el dominio de la frecuencia y su comportamiento. Si queremos determinar el espectro de una señal simple, la transformada de Fourier resulta eficaz. Sin embargo, presenta ciertas limitaciones ya que no ofrece información en el tiempo, es decir, no especifica los momentos en que se producen ciertos eventos.

Con el objetivo de obtener las referencias temporales de los componentes de la señal, es imprescindible recurrir a otras transformadas que ofrezcan una representación tiempo/frecuencia dinámica. En este caso, la transformada Wavelet provee de información sobre el tiempo en el que se producen los eventos (como ruido y deltas) y hace posible la disección en funciones básicas que pueden ser llevadas al máximo detalle.

3.3.1.1 Transformada rápida de Fourier

La serie de Fourier es la representación de una función periódica como sumatorio infinito de funciones senoidales, cuya frecuencia es múltiplo de la frecuencia fundamental de la función.
De esta forma, multiplicando la señal que se quiere procesar por una serie de señales senoidales de distintas frecuencias podemos determinar qué frecuencias están presentes en la señal de estudio.
Considerando que la señal es discreta y la frecuencia es de magnitud finita, la transformada de Fourier se define matemáticamente:

$$F(t) = \int_{-\infty}^{\infty} y(t) \cdot e^{-j2\pi ft} dt$$

Siendo $y(t)$ la señal en el dominio del tiempo, $F(t)$ la señal en el dominio de la frecuencia, i la unidad imaginaria equivalente a $\sqrt{-1}$, y $e^{-j2\pi ft}$ la función exponencial compleja (siendo esta misma, una combinación de la función seno y coseno, ambas funciones presentando un desfase de $\frac{\pi}{2}$).

Por tanto, si efectuamos el producto escalar entre la señal a procesar y una señal senoidal de cierta frecuencia y en el resultado obtenemos una gran amplitud, esto quiere decir que hay solapamiento entre ambas y por lo tanto podemos asegurar que nuestra señal contiene la frecuencia de la señal con la que se efectuó el producto escalar.

La transformada de Fourier tiene una alta resolución en el dominio de la frecuencia y en contraposición una resolución muy pobre en el dominio del tiempo, lo que resulta en obtener información sobre las frecuencias presentes en la señal pero no cuando se producen en el tiempo.

Para corregir esta deficiencia, Denis Gabor adaptó la transformada de Fourier para poder analizar la señal en determinadas secciones de tiempo, haciendo uso de ventanas de tiempo. A esta adaptación se le conoce como transformada de Fourier por intervalos, con la cual se procesa la señal en el plano de tiempo y frecuencia.

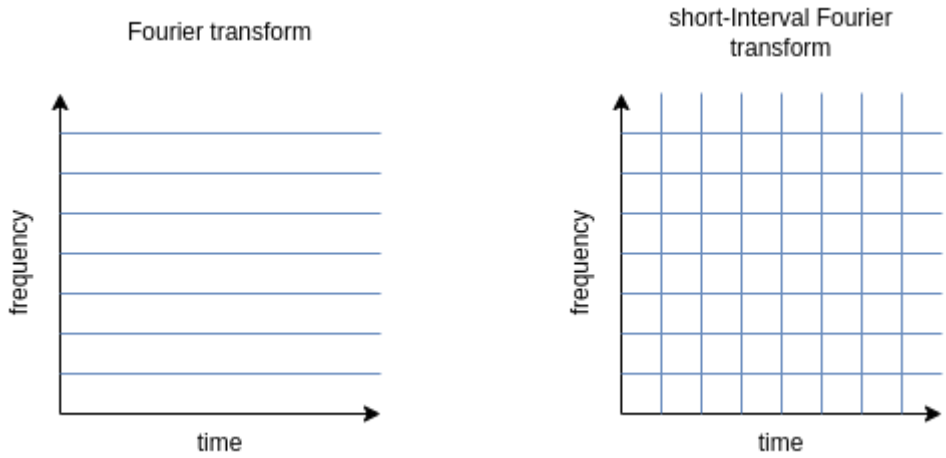

Figura 12: comparación de resolución entre la transformada de Fourier y la transformada de intervalos cortos de Fourier

Puesto que nuestro proyecto es de ámbito digital, se debe hacer uso de la transformación discreta de Fourier (DFT), que se describe matemáticamente como:

$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} kn} \quad k = 0, \ldots, N-1$$

Donde i es la unidad imaginaria y $e^{\frac{2\pi i}{N}}$ es la enésima raíz de la unidad. Puesto que el análisis de complejidad temporal (cuánto se estima que puede tardar un algoritmo en ejecutarse, en inglés denominado como *Big O notation*) para esta forma de calcular la transformada discreta de Fourier es de desarrollo exponencial $O(n^2)$.

La transformada rápida de Fourier (FFT) es un algoritmo que mejora la eficiencia de la DFT, pasando a describirse el rendimiento computacional como $O(n\ log\ n)$

3.3.1.2 Transformada Wavelet

El mayor problema de la transformada de Fourier, aun con su versión usando ventanas temporales, es que finalmente acabaremos encontrando una limitación teórica conocida como principio de incertidumbre. A medida que hacemos más pequeña la ventana temporal, tendremos más información sobre en qué momento del tiempo se encuentra una frecuencia específica, pero tendremos, como resultado, menos información sobre el valor de la frecuencia. Y por el contrario, si hacemos más grande la ventana, perderemos resolución a nivel temporal.

Un mejor acercamiento para analizar de forma dinámica el espectro de frecuencias es haciendo uso de la transformada de wavelets (WT), la cual tiene una alta resolución en ambos dominios de frecuencia y tiempo.

Esto se consigue cambiando dinámicamente la longitud/escala de las ventanas con las cuales se procesa la señal.

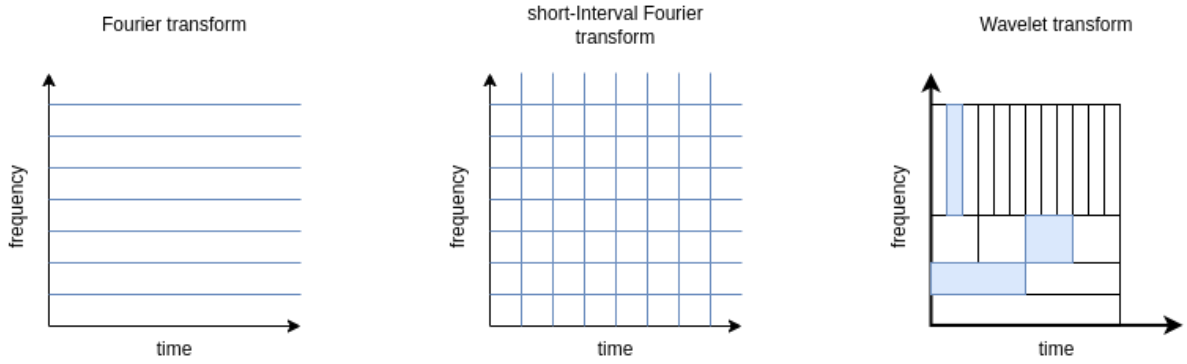

Figura 13: representación de la resolución dinámica obtenida con la transformada de wavelet en contraposición con las obtenidas con las transformadas de Fourier

Por lo tanto, la transformada de wavelet consigue:

- Para valores pequeños de frecuencia una alta resolución en el dominio de la frecuencia y baja resolución en el dominio del tiempo
- Para valores grandes de frecuencia una baja resolución en el dominio de la frecuencia y alta resolución en el dominio del tiempo

Con lo cual, en las escalas en las que son interesantes las características dependientes del tiempo tiene una alta resolución en el dominio del tiempo y en las escalas en las que son interesantes las características dependientes de la frecuencia tiene una alta resolución en el dominio de la frecuencia.

La transformada de Wavelets usa unas funciones que se denominan ondículas, cada una de las que usa con una escala distinta.

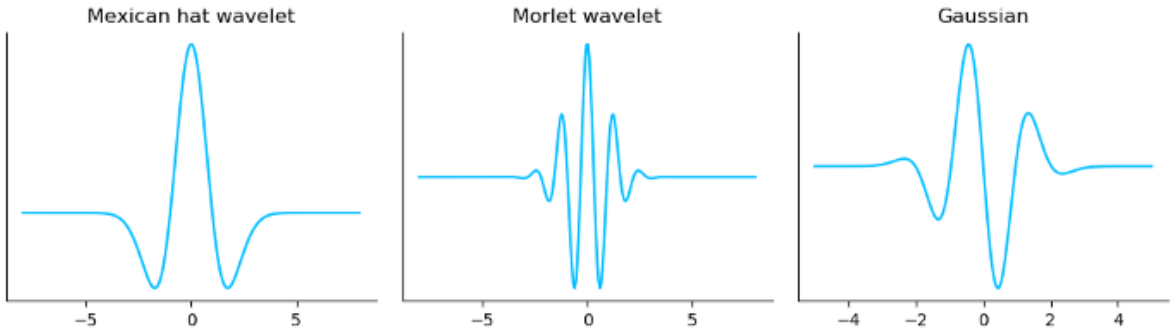

Figura 14: Ondículas extraída de [57]

Como la ondícula está localizada en un instante de tiempo, podemos multiplicarla por la señal en distintos momentos del tiempo, de esta forma, comenzamos al inicio de la señal y desplazamos la ondícula a lo largo del tiempo total de la señal. A este proceso se le denomina convolución.

Como en el caso de la transformada de Fourier, tenemos transformada continua y discreta de wavelets. En el presente trabajo se exploran ambas para distintos cometidos. Para la transformada continua de wavelet (CWT), la ecuación matemática:

$$CWT(b,a) = \frac{1}{\sqrt{|a|}} \int_{-\infty}^{\infty} s(t)\psi^* \left(\frac{1}{a}(t-b)\right) dt$$

Siendo $\psi_a^b(t)$ la wavelet madre:

$$\psi_a^b(t) = \frac{1}{\sqrt{a}} \psi(\frac{t-b}{a})$$

La transformada discreta de wavelet cubre las redundancias de la transformada continua de wavelet, descomponiendo la señal de entrada en un conjunto de paquetes ortogonales, estando los factores a y b limitados a valores discretos. Estos valores se denominan aproximación y detalle, siendo la aproximación el coeficiente resultante de aplicar un filtro de paso bajo y el detalle el coeficiente resultante de aplicar un filtro de paso alto.

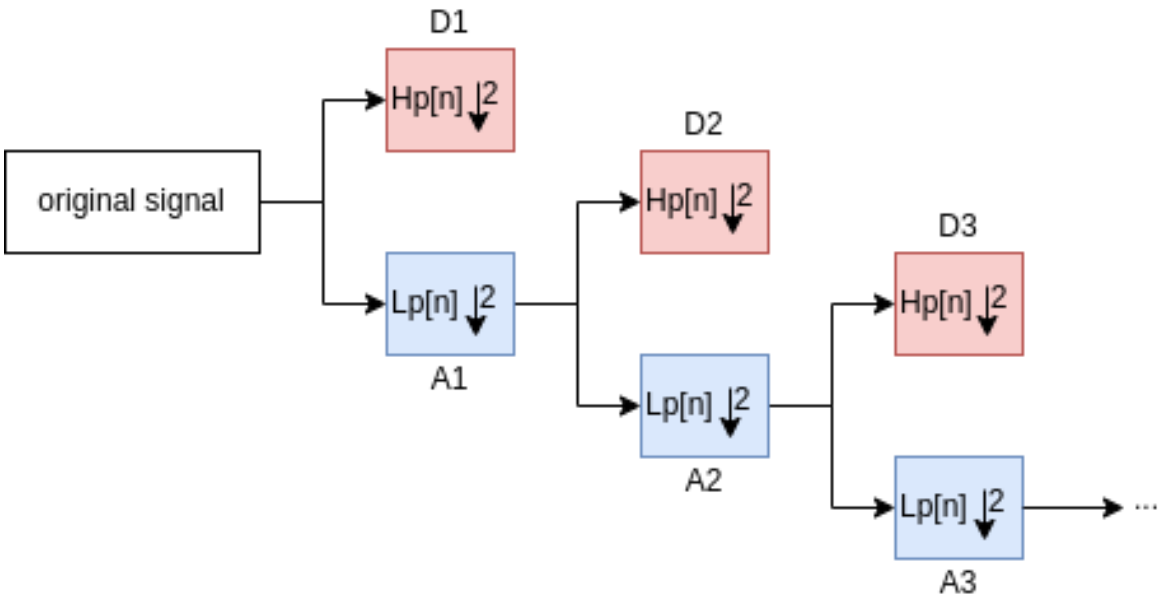

Figura 15: representación de las descomposiciones efectuadas usando la transformada de wavelet

3.3.1.2 Limpiado de frecuencias

A partir de las técnicas anteriormente descritas para el procesamiento de señales, se pasa a describir brevemente el uso de las mismas para el limpiado de frecuencias de las señales a clasificar. En este sentido, puesto que las frecuencias en las que se produce la actividad cerebral al momento de efectuar las tareas 122, 123 y 127 están contenidas principalmente en la banda de frecuencias de 8 a 32 Hz, debemos limpiar la señal de ruido blanco y otras frecuencias cerebrales no relevantes. Se pueden aplicar, para ello, las transformadas previamente descritas a nivel programático. En el caso de la transformada de wavelet, primero se filtra en múltiples iteraciones la señal, tanto a altas frecuencias como a bajas frecuencias, dando como resultado un vector de coeficientes. De estos, eliminamos las bandas de frecuencias no relevantes y se reconstruye la señal con los coeficientes restantes, resultando en una señal limpia. El resultado se puede observar en la figura 16

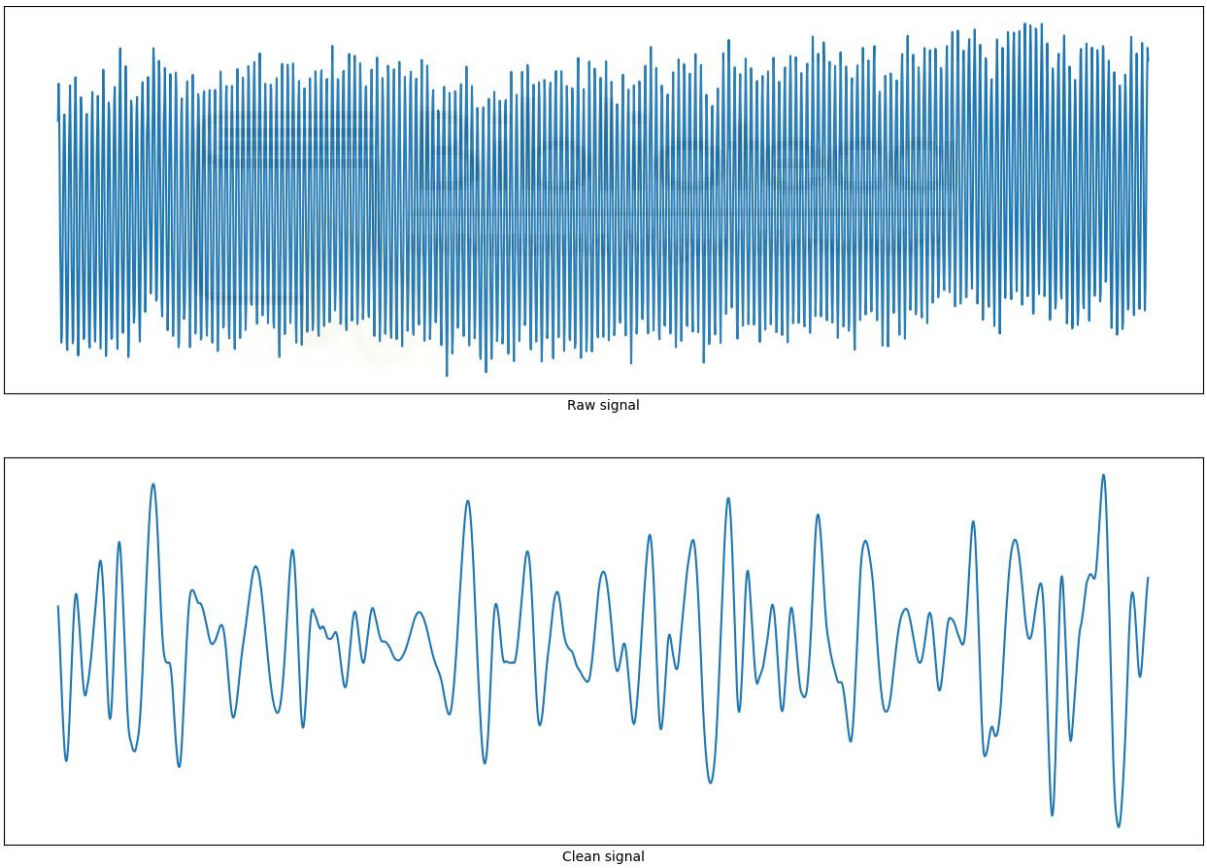

Figura 16: arriba, fragmento de la señal original del electrodo CZ; abajo, fragmento de la señal limpia del electrodo CZ

3.3.3 Extracción de características

3.3.3.1 Wavelet Packet Decomposition

La descomposición en paquetes de Wavelet puede ser vista como una generalización de la transformada discreta de wavelet, la cual nos provee una alta resolución multi-dominio (frecuencial y temporal) para el análisis de señales no estacionarias. Esta forma de transformada genera una descomposición completa de la señal de entrada, generando para cada nivel, un fragmento que representa una de las mitades de la descomposición (cada nivel de cálculo divide la señal entre 2, lo que genera dos paquetes de datos), cuantas veces como niveles de descomposición se requieran.

Este proceso construye un árbol compuesto por paquetes de señales que se encuentran en una subbanda determinada. Los paquetes son llamados detalle y aproximación. Cada pareja de paquetes descompuestos de la señal de entrada contiene un paquete con las altas frecuencias (aproximación) y otro con las bajas frecuencias (detalle).

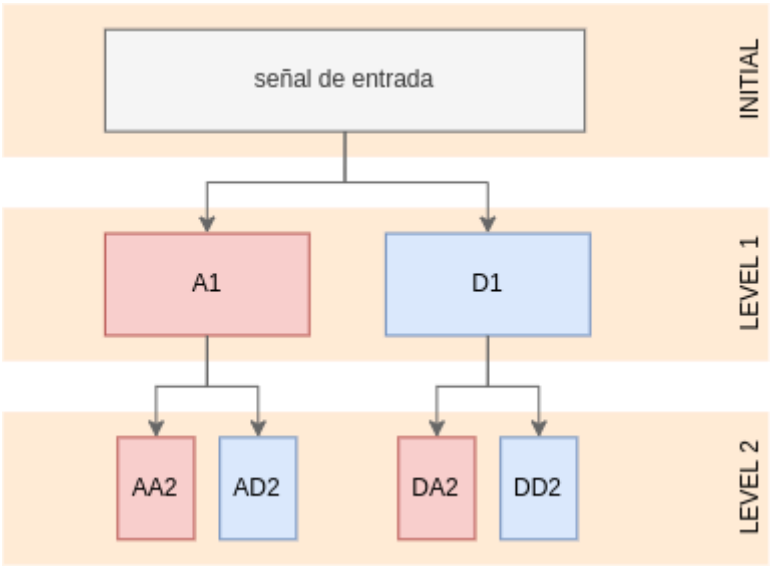

Figura 17: representación de la descomposición de una señal de entrada usando wavelet packet decomposition

Por lo tanto, se efectúa un filtrado de frecuencias altas y bajas a lo largo de todo el árbol. Un paquete de WPD está representado por la función:

$$\psi_{j,k}^{i}(t) = 2^{\frac{-j}{2}}\psi^{i}(2^{j} \cdot t - k)$$

Siendo i el parámetro de modulación, j el parámetro de dilatación y k el parámetro de traslación. A su vez, $i = 1,2,\ldots,j^{n}$ siendo n el nivel de descomposición del árbol de WPD.

La descomposición, a nivel matemático, se obtiene de la recursión de relaciones de las siguientes ecuaciones:

$$\psi^{2i} = \frac{1}{\sqrt{2}}\sum_{-\infty}^{\infty} h(k)\psi^{i}(\frac{t}{2} - k)$$

$$\psi^{2i+1} = \frac{1}{\sqrt{2}}\sum_{-\infty}^{\infty} g(k)\psi^{i}(\frac{t}{2} - k)$$

Siendo ψ^{i} la wavelet madre, $h(k)$ y $g(k)$ filtros (opuestos) cuadráticos asociados a la escala de la wavelet madre.

Sobre estos paquetes de datos se calculan, más adelante, las características que luego usaremos para clasificar las señales. Previo a la extracción de características, se forman las ventanas temporales sobre las cuales se aplicará el método de WPD.

Para ello, a nivel programático, se dividió el vector de tareas (el cual contiene para cada momento del tiempo, la tarea que se estaba realizando) en sendos vectores que contenían el inicio y final temporal de cada uno de esos bloques de tareas y se almacenaron los datos en un diccionario, cuya clave era cada una de las tareas.

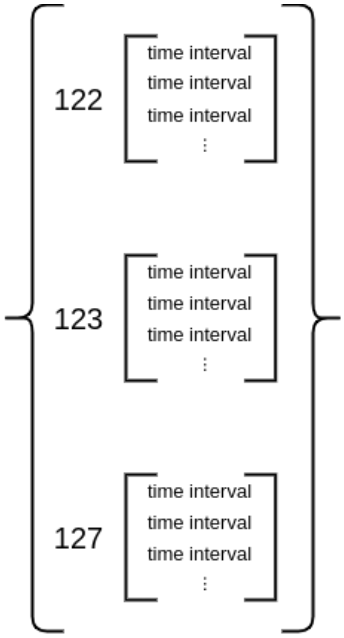

Figura 18: representación de la salida del diccionario de tareas

Una vez divididos los momentos temporales, se procede a eliminar de cada grupo el primer segundo, para evitar solapamientos o artefactos que pueden descender la ratio de acierto de los modelos de clasificación.

A continuación, se fragmentan los vectores temporales en ventas de 5 segundos con 2 segundos de solapamiento. Se ha observado durante el desarrollo del presente trabajo que el solapamiento en las ventanas mejora considerablemente la ratio de acierto en la clasificación.

Por lo tanto, teniendo en cuenta que en cada archivo tenemos aproximadamente 4 minutos de sesión, esto quiere decir que tendremos alrededor de 48 ventanas A su vez, si multiplicamos por 4 archivos el total de ventanas son 192.

Sobre estas ventanas, aplicamos a cada una de ellas el método de WPD, descomponiendo las ventanas en un árbol de 6 niveles. De cada nivel, extraemos la aproximación y detalle que se encuentran en los extremos opuestos (Fig. 19).

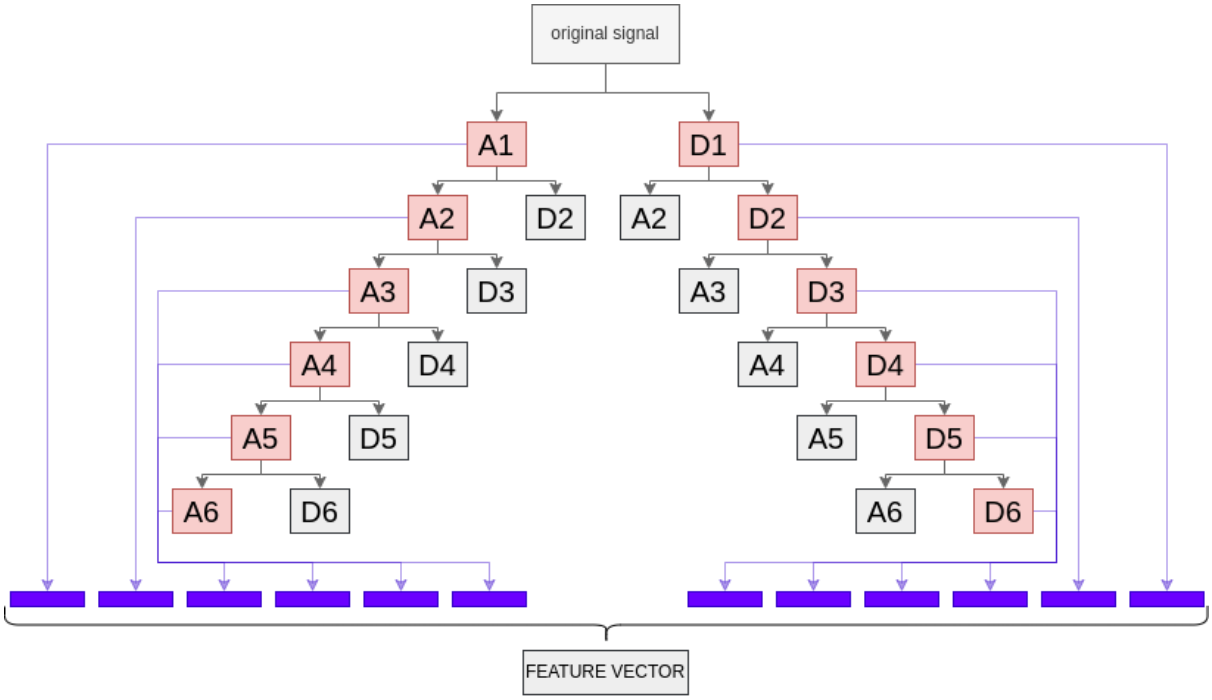

Figura 19: representación de la extracción de características que se efectúa usando wavelet packet decomposition

Sobre cada uno de ellos se calculan las siguientes características:

- MAV (Mean Absolute Values): Media absoluta de los coeficientes de cada subbanda.
- AVP (Average Power): Promedio de la potencia de los coeficientes de cada subbanda.
- SD (Standard deviation): Desviación estándar de los valores de los coeficientes de cada subbanda.
- SKEW (Skewness): Asimetría de los coeficientes.
- KURT (Kurtosis): Medida del apuntalamiento de la distribución.
- ZC (Zero Crossing): Número de cortes con el eje cero en Y
- MC (Mean Crossing): Media del número de cortes con el eje Y, y = mean(y)
- Entropy: entropía de la señal.
- n5: Quinto percentil
- n25: Percentil 25
- n75: Percentil 75
- n95: Percentil 95

- Median: Mediana de los coeficientes.
- Mean: Media de los coeficientes.
- std: Desviación estándar ignorando los coeficientes nulos.
- var: Varianza
- rms: Media aritmética a lo largo de un eje, ignorando coeficientes nulos.
- EnergySubBand: Lista de energías por subbanda.
- PercentageSubBand: Porcentaje de representación de las energías individuales sobre el total de energías.
- Energytot: Energía total de la ventana.

En el apartado de resultados veremos que algunas de las características tuvieron un desempeño exiguo, por lo que en la mayoría de los casos habría que descartarlos. Se deja constancia en el trabajo para dar cuenta de la variedad de características y para proporcionar un entorno más completo de clasificación, así como de la extensión y minuciosidad del proceso de testeo

3.3.4 Clasificación de características

En el capítulo 2 del presente trabajo se nombraron y explicaron de forma resumida los algoritmos de clasificación usados en la clasificación de características. A continuación, se detalla brevemente el proceso de clasificación con dichos algoritmos, centrando la exposición en el uso de librerías de libre uso.

3.3.4.1 Antecedentes

Al inicio del desarrollo se contaba con un código construido en matlab, el cual incorporaba el procesamiento y extracción de características con FFT y la clasificación de estas últimas con el algoritmo KNN.
Cuando se comenzó a desarrollar en Python se buscaron alternativas a la solución originalmente aportada, de tal manera que pudiéramos usar una herramienta ampliamente testeada y usada que nos permitiera clasificar los datos con simples llamadas a métodos. La solución que se propone es la implementación de la librería scikit-learn, la cual incorpora no solo el clasificador

KNN, sino también todos aquellos descritos en el capítulo 2 (salvo la red neuronal convolucional). A su vez, esta nos permite agregar muchos más clasificadores que los inicialmente planteados porque la implementación de los modelos se efectúa de forma sencilla, centrando nuestros esfuerzos en el formateo de los datos a introducir y el porcentaje de datos destinados a entrenamiento/testing.

3.3.4.2 Clasificadores en Sklearn

De esta librería se han usado todos los clasificadores que previamente se han descrito y cuya lista se detalla a continuación:

- KNN
- SVM
- Gradient Boosting Classifier
- Naive bayes
- Random forest
- MLP
- Decision Tree
- Procesos Gaussianos

Se ha escogido esta librería porque ofrece la posibilidad de realizar el testing de distintos parámetros para el mismo set de data y el mismo clasificador de forma automática. En esta línea, se han usado métodos tales como GridSearchCV, lo que nos permite generar un subproceso en el que se prueban todas las combinaciones posibles sobre los parámetros de entrada para encontrar el marco con mayor porcentaje de acierto.
Además, se han usado otros métodos que nos ayudan a ser más rápidos tanto en desarrollo como en ejecución. Algunas de ellas son:

- train_test_split: Método que nos permite dividir el set completo de datos en dos grupos, uno para entrenamiento y otro para testeo. Nos permite introducir múltiples parámetros de entrada de los cuales hemos usado:

    - Tamaño de test: porcentaje de los datos que queremos destinar a test y por extensión, los que queremos usar para entrenamiento.
    - Aleatoriedad y elección de estado de aleatoriedad.
- metrics: Método para calcular el porcentaje de acierto sobre los sets de datos que se devuelven de los clasificadores.
- cross_val_score: Evalúa la puntuación por validación cruzada.

Del mismo modo, los test de clasificación se han conducido de múltiples formas. Dependiendo del algoritmo de clasificación, en algunos casos, la diferencia de puntuación era desdeñable, por tanto se optó por la clasificación multi-clase para descender el consumo computacional que supone generar conjuntos separados de datos para múltiples características. Por tanto, los metodos de validacion usados han sido:

- Holdout method (Fig. 20)
- K-Fold Cross Validation (Fig. 21)

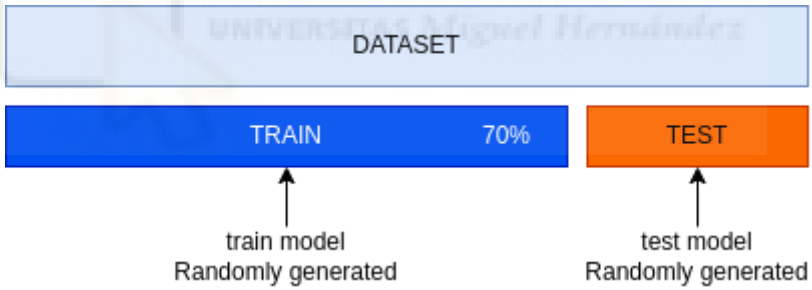

Figura 20: representación del método Holdout

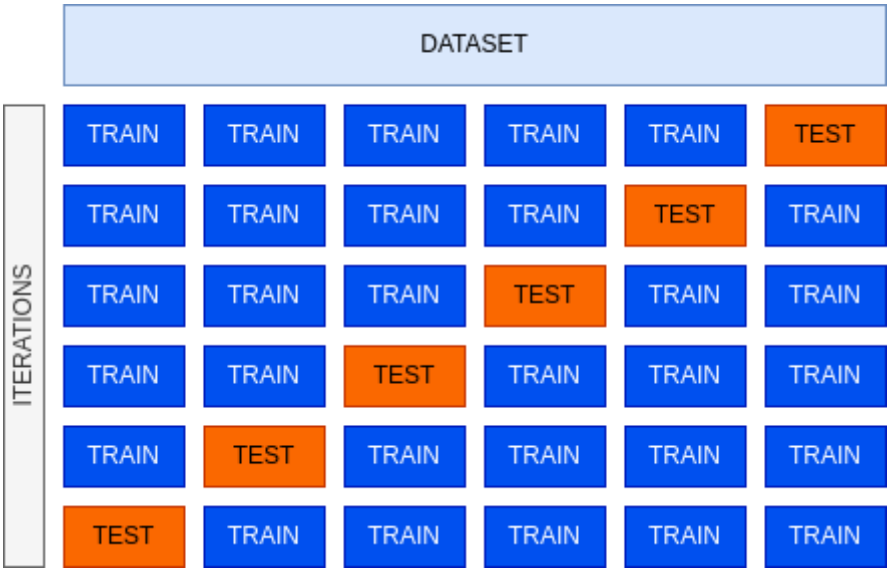

Figura 21: representación del método de validación cruzada K-Fold

A su vez, como se ha comentado anteriormente, se realizaron los tests con dos clases (separando del conjunto de datos una de las clases y clasificando las dos restantes) y con múltiples clases (teniendo en un mismo conjunto de datos las tres etiquetas 122, 123 y 127).

Inicialmente el conjunto de características era demasiado grande para gestionar el cálculo de forma eficiente, por ello en el desarrollo de los flujos de trabajo para clasificar los datos, se implementó un proceso de agregación de características. Reduciendo de esta forma el conjunto de datos, no solo aumentamos la eficiencia, sino que en la mayoría de casos aumentamos el puntaje de las clasificaciones. Esta reducción se llevó a cabo calculando medias sobre los datos obtenidos de los distintos electrodos, como se muestra en la figura 22.

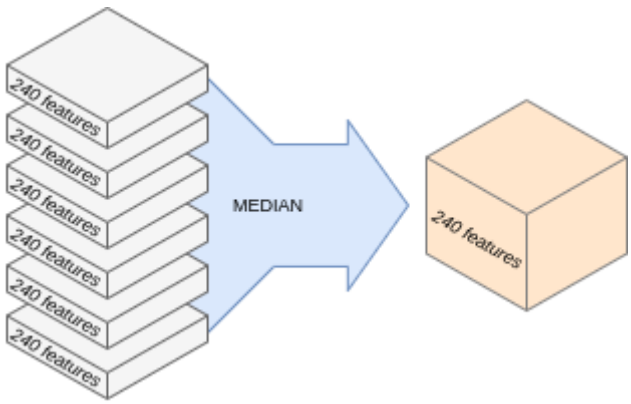

Figura 22: representación de la reducción del vector de características

Por tanto, pasamos de tener 196 ventanas x 8 electrodos x 240 características a finalmente obtener una matriz de 196 ventanas x 240 características.

Capítulo 4. Resultados

En el siguiente capítulo se muestran y detallan los tests realizados durante el desarrollo del presente trabajo, así como los resultados obtenidos. Todos los tests se han realizado de forma offline para cada uno de los sujetos y se han comparado los resultados globalmente.

Por consiguiente, se comentan, primero, los resultados obtenidos procesando las señales con FFT, obteniendo el espectro de frecuencias de cada ventana y clasificando los datos obtenidos con KNN variando la K. Seguidamente, se expondrán los resultados obtenidos de distintas configuraciones, tomando como base de procesamiento la transformada wavelet (WT), como extractor de características el algoritmo wavelet packet decomposition (WPD).
Por último, se mostrará una comparativa de los resultados obtenidos.

4.1 FFT + KNN

Para esta primera configuración vamos a procesar la señal con la transformada rápida de Fourier, generando ventanas de señales de 128 muestras. Extraemos características de las ventanas calculando el espectro en frecuencia entre 8 Hz y 32 Hz. Clasificaremos las mismas con el algoritmo KNN variando el valor de K (número de vecinos). Se evaluará la clasificación diferenciando entre 2 tareas. Los resultados obtenidos se representan en la tabla 1.

Tabla 1. Comparación de resultados entre las características MAV, AVP, SD, SKEW cuando se efectúan clasificaciones por pares de tareas usando knn.

Task		122-123		122-127		123-127	
user	K	score	std	score	std	score	std
1	1	58.33	3.57	58.02	06.03	70.01	7.72
	3	61.16	3.86	62.03	6.62	73.7	8.37
	5	62.07	3.14	62.8	7.37	76.39	8.97
	15	65.38	2.48	66.31	9.11	76.41	9.69
2	1	50.5	2.2	62.1	4.71	63.46	03.07
	3	50.41	4.73	63.09	6.53	66.58	2.32
	5	50.37	5.92	66.31	04.08	67.04	2.86
	15	50.23	6.99	68.14	3.56	71.19	3.71
3	1	52.5	1.73	58.31	1.82	65.58	01.09
	3	52.41	1.48	61.97	2.3	70.54	0.7
	5	52.5	2.79	63.39	1.88	71.32	1.54
	15	53.37	1.76	66.1	3.51	73.36	2.58

4.2 Transformada wavelet + KNN

Para este, como para las siguientes pruebas, los primeros resultados que se muestran son individuales para cada una de las características y posteriormente se compara el rendimiento de todas ellas. En este caso, usamos el modelo de clasificación de vecinos cercanos, aplicando como algoritmo “ball tree” y realizando una validación cruzada de 4 iteraciones.

Tabla 2. Comparación de resultados entre las características MAV, AVP, SD, SKEW cuando se efectúan clasificaciones por pares de tareas usando knn.

Característica		MAV						AVP						SD						SKEW					
Sujeto		S1		S2		S3		S1		S2		S3		S1		S2		S3		S1		S2		S3	
tareas	K	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std
122-123	1	58,62	9,06	41,74	2,92	51,56	5,13	59,37	5,29	47,99	9,56	46,88	4,4	57,42	7,56	44,64	6,68	44,6	3,39	53,45	6,34	43,3	3,86	39,06	3,07
	3	64,91	4,9	39,96	5,28	53,12	9,68	56,9	3,42	48,44	7,7	47,82	13,59	60,65	2,58	50	2,53	54,69	7,63	58,23	5,9	55,36	7,82	42,19	4,23
	5	61,03	5,72	43,75	10,13	51,56	7,07	57,59	8,56	46,43	3,71	43,13	5,16	63,29	8,04	50	6,19	57,81	7,67	56,62	2,05	53,57	3,65	37,5	4,39
	15	72,1	6,31	41,07	7,91	50	5,53	64,43	9,26	42,86	2,32	49,24	3,55	63,92	12,11	45,09	6,39	51,56	6,86	44,02	5,42	48,21	1,79	36,88	6,87
122-127	1	67,48	4,78	64,06	2,95	59,9	9,16	70,93	8,91	72,3	2,61	65,62	8,58	69,88	7,05	75	6,17	59,99	2,27	43,31	5,41	46,88	4,55	50,8	5,78
	3	67,82	8,86	73,44	2,24	65,62	6,84	72,65	9,51	72,25	5,07	63,21	11,73	69,33	3,65	75	7,15	60,94	9,48	46,45	7,24	48,44	7,46	53,84	2,57
	5	75,77	9,19	72,3	2,61	63,16	8,7	73,71	5,76	69,27	7	64,63	6,75	71,49	4,64	71,88	6,52	64,06	6,86	43,42	9,55	54,69	4,41	44,7	8,8
	15	74,04	8,84	73,86	3,67	67,19	7,11	75,82	6,26	70,79	2,4	62,5	5,18	71,57	8,57	73,86	2,42	68,75	6,91	41,26	6,5	51,56	7,49	50	7,5
123-127	1	86,99	4,25	66,67	6,84	73,44	3,49	79,8	5,16	60,25	7,28	59,38	8,12	86,59	3,22	64,34	8,07	71,88	3,41	50	9,84	47,94	7,38	46,88	2,21
	3	91,29	7,09	70,83	9,16	78,12	5,58	82,58	5,38	67,08	4,11	65,62	6	91	4,63	67,12	4,51	71,88	6	54,33	6,88	44,1	11,31	43,75	1,35
	5	89,9	5,01	75	6,42	78,12	4,06	86,99	6,9	73,95	4,61	68,75	4,06	89,9	2,92	78,08	7,18	71,88	7,65	57,11	4,9	45,49	4,54	48,44	3,83
	15	91,41	6,24	79,17	6,87	76,56	3,83	85,48	7,82	77,78	8,78	70,31	3,49	86,87	7	81,94	8,77	76,56	5,12	50	0,66	50,35	8,44	45,31	3,49

Tabla 3. Comparación de resultados entre las características KURT, ZC, MC, entropy cuando se efectúan clasificaciones por pares de tareas usando knn.

Característica		KURT						ZC						MC						entropy					
Sujeto		S1		S2		S3		S1		S2		S3		S1		S2		S3		S1		S2		S3	
tareas	K	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std
122-123	1	59,37	6,44	62,5	7,95	53,74	9,14	67,76	5,46	54,91	9,5	53,88	3,74	66,09	9	51,79	2,96	51,56	4,33	49,14	5,95	42,86	3,09	50	1,69
	3	60,07	4,78	60,71	10,07	56,25	7	74,48	6,6	46,88	4,44	53,84	2,29	63,79	4,39	53,12	3,36	60,94	4,96	47,41	6,11	42,86	3,09	50	0,66
	5	59,31	2,28	46,88	7,13	59,99	2,27	73,82	5,37	51,79	3,85	50,76	7,2	73,19	5,93	60,04	5,28	57,81	6,91	47,41	6,11	42,86	3,09	50	0,66
	15	60,34	5,49	57,14	6,96	57,81	5,02	76,32	3,6	53,79	9,17	57,81	1,92	72,15	3,82	50,45	7,92	56,25	10,48	50,81	3,72	42,86	3,09	50	0,66
122-127	1	50,76	5,01	50	6,45	46,16	4,84	58,72	1,29	59,38	7,62	59,99	9,31	65,55	11,18	53,12	1,72	61,51	11,2	47,77	4,08	43,75	4,45	50	0,66
	3	53,79	7,77	53,12	6,57	53,88	3,89	64,58	5,82	62,5	5,63	63,07	7,97	61,19	8,43	53,12	4,46	59,99	5,56	47,77	3,26	39,06	4,08	50	0,66
	5	48,17	6,47	54,69	5,56	52,37	8,22	70,93	9,54	57,81	6,3	72,3	6,09	62,43	8,53	58,52	5,36	60,04	9,01	46,26	4,09	36,93	6,13	50	0,66
	15	41,8	7,81	54,69	5,47	47,73	12,41	66,67	4,83	62,5	5,04	70,83	5,34	66,84	3,58	60,94	7,56	67,76	5,32	42,75	2,07	37,5	4,25	50	0,66
123-127	1	55,05	1,21	42,71	4,63	54,69	5,12	79,17	7,29	71,25	5,38	70,31	9,21	75,65	5,7	64,38	6,84	71,88	6,63	50,88	4,82	52,78	2,41	50	0
	3	65,77	7	52,95	7,53	45,31	5,85	80,56	5,34	67,71	2,74	73,44	4,62	82,84	6,65	59,72	5,35	71,88	8,66	52,94	4,22	52,78	2,41	50	0
	5	61,11	6,37	55,56	5,34	45,31	6,44	82,84	6,75	65,24	2,96	78,12	9,11	82,84	6,75	65,28	3,48	71,88	10,68	55,72	4,82	55,56	2,63	50	0
	15	63,89	6,07	51,39	4,54	40,62	1,35	82,76	8,05	63,89	4,89	75	8,08	82,76	10,2	68,06	2,67	68,75	6,77	55,72	4,82	55,56	2,63	50	0

Tabla 4. Comparación de resultados entre las características n5, n25, n75, n95 cuando se efectúan clasificaciones por pares de tareas usando knn.

Característica		n5						n25						n75						n95					
Sujeto		S1		S2		S3		S1		S2		S3		S1		S2		S3		S1		S2		S3	
tareas	K	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std
122-123	1	49,08	7,81	53,57	6,86	40,06	4,25	54,14	10,34	38,39	5,11	44,6	5,88	61,09	10,36	56,47	4,48	47,68	4,47	59,25	5,92	46,65	6,9	51,56	9,37
	3	56,9	11,82	60,04	8,25	44,6	4,6	65,02	9,68	42,86	3,24	46,88	4,49	64,54	10,65	57,14	1,74	52,32	7,91	62,07	9,1	44,64	7,18	53,84	4,84
	5	56,73	7,89	55,36	3,65	43,75	10,31	64,43	9,83	45,09	5,7	41,57	5,74	59,96	6,09	55,36	9,62	53,12	6,7	62,76	5,38	44,64	7,82	44,6	3,88
	15	51,72	13,35	56,47	4,34	45,31	4,02	75,43	17,79	48,66	7,21	43,09	10,18	67,2	3,68	51,79	9,56	56,25	5,12	56,51	13,38	51,79	11,92	47,77	7,89
122-127	1	67,82	7,59	66,24	9,78	52,23	6,7	69,88	15,21	67,71	4,51	56,25	8,25	59,87	7,63	65,62	2,17	58,48	2,55	66,09	8,81	65,62	5,84	58,48	7,99
	3	69,54	9,11	72,35	3,85	56,25	11,32	71,49	7,04	70,31	1,8	58,48	8,87	66,79	8,82	72,3	7,8	62,5	2,59	63,76	5,79	71,88	0,94	58,43	5,1
	5	68,15	9,31	73,44	4,35	55,35	6,39	71,6	12,19	65,62	0,45	61,55	6,75	67,82	7,15	66,15	6,64	61,55	2,36	69,54	5,15	73,44	8,49	63,07	4,45
	15	71,26	10,33	72,3	4,63	71,88	7,81	69,88	8,81	67,71	3,64	64,06	3,1	70,93	5,92	70,74	3,61	63,12	4,77	69,75	8,63	68,75	6,28	67,66	5,71
123-127	1	83,96	8,01	69,1	2,86	59,38	6,39	84,31	9,03	66,15	5,26	65,62	4,06	91,29	3,88	65,97	7,85	62,5	4,06	90,03	6,72	64,41	9,11	67,19	9,21
	3	84,22	8,18	75,34	1,99	65,62	4,06	90,03	8,91	68,47	2,67	68,75	4,06	92,8	4,09	75,34	4,67	71,88	4,94	88,51	9,15	79,47	3,56	71,88	8,38
	5	87,12	10,31	75	4	67,19	2,59	91,41	10,73	73,61	5,9	70,31	2,59	92,8	3	73,99	3,29	70,31	5,63	88,56	5,1	79,47	6,21	70,31	6,44
	15	81,44	11,07	76,39	7,41	71,88	4,69	89,9	8,07	75	5,66	71,88	3,41	91,42	2,86	77,78	3,21	68,75	4,06	91,42	4	74,02	10,68	75	4,69

Tabla 5. Comparación de resultados entre las características median, mean, std, var cuando se efectúan clasificaciones por pares de tareas usando knn.

Característica	median	mean	std	var

Sujeto		S1		S2		S3		S1		S2		S3		S1		S2		S3		S1		S2		S3	
tareas	K	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std
122-123	1	54,09	2,55	50,22	7,94	43,75	2,66	58,34	9,29	50,22	6,79	53,12	1,72	57,42	7,56	44,64	6,68	44,6	3,39	56,73	2,33	46,88	4,53	47,73	5,76
	3	54,2	5,2	48,21	7,22	46,16	7,5	60,34	3,43	45,31	7,28	50	4,43	60,65	2,58	50	2,53	54,69	7,63	55,92	1,49	45,09	2,78	49,24	2,31
	5	52,59	3,04	50	6,39	46,21	8,61	58,62	1,92	50	8,6	51,56	7,85	63,29	8,04	50	6,19	57,81	7,67	59,03	6,58	48,21	5,28	46,88	3,81
	15	53,28	3,28	50	7,73	48,44	3,99	51,67	7,81	48,21	4,78	45,31	8,96	63,92	12,11	45,09	6,39	51,56	6,86	61,68	11,53	41,07	4,79	53,12	5,24
122-127	1	47,85	4,13	52,32	5,22	49,2	7,92	55,26	3,75	53,84	7,2	51,56	3,99	69,88	7,05	75	6,17	59,99	2,27	64,38	5,75	67,71	4,51	66,15	3,38
	3	46,59	7,3	41,52	9,26	55,4	2,33	58,72	8,77	48,44	2,96	52,27	3,88	69,33	3,65	75	7,15	60,94	9,48	69,54	8,78	69,27	5,85	64,68	6,24
	5	48,38	7,25	46,07	7,75	50,76	7,84	49,65	6,86	53,12	4,55	53,88	3,89	71,49	4,64	71,88	6,52	64,06	6,86	71,26	6,62	73,86	4,6	66,15	6,89
	15	41,92	8,26	51,56	8,37	58,48	5,57	54,92	3,9	50	9,64	57,01	6,76	71,57	8,57	73,86	2,42	68,75	6,91	74,43	7,65	73,44	3,19	61,55	5,7
123-127	1	58,58	6,47	47,22	4,9	50	9,63	54,92	5,68	48,44	5,7	45,31	10,68	86,59	3,22	64,34	8,07	71,88	3,41	82,7	4,95	67,04	6,94	68,75	4,94
	3	53,1	11,61	52,6	8,36	50	4,42	56,73	7,4	43,06	3,36	46,88	4,69	91	4,63	67,12	4,51	71,88	6	82,58	6,04	71,21	6,75	70,31	2,59
	5	63,64	14,57	60,07	7,02	45,31	3,83	51,31	7,42	45,31	8,87	43,75	11,48	89,9	2,92	78,08	7,18	71,88	7,65	85,61	8,28	73,95	7,84	73,44	2,59
	15	62,66	8,96	53,42	7,14	53,12	4,94	52,02	4,84	43,75	7,65	45,31	2,59	86,87	7	81,94	8,77	76,56	5,12	89,9	8,47	76,73	8,25	70,31	2,59

Tabla 6. Comparación de resultados entre las características rms, EnergySubBand, PercentageSubBand, EnergyTot cuando se efectúan clasificaciones por pares de tareas usando knn.

Característica		rms						EnergySubBand						PercentageSubBand						EnergyTot					
Sujeto		S1		S2		S3		S1		S2		S3		S1		S2		S3		S1		S2		S3	
tareas	K	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std
122-123	1	58,62	9,06	41,74	2,92	51,56	5,13	60,34	3,3	40,18	9,31	43,09	3,91	56,62	5,61	38,39	4,07	44,7	9,96	60,59	8,2	51,34	8,69	48,44	6,07
	3	64,91	4,9	39,96	5,28	53,12	9,68	62,07	7,25	47,1	11,1	46,88	6,02	63,29	9,05	41,74	5,14	44,65	10,24	64,03	8,48	53,35	8,33	51,56	6,34
	5	61,03	5,72	43,75	10,13	51,56	7,07	64,37	7,62	46,43	5,51	51,56	7,59	55,01	6,69	38,84	9,29	51,56	7,23	68,35	7,71	46,65	5,4	53,88	3,74
	15	72,1	6,31	41,07	7,91	50	5,53	68,92	11,7	46,43	9,51	57,81	10,2	63,79	9,03	55,13	5,34	49,15	6,82	64,43	8,52	55,36	6,8	50	2,99
122-127	1	67,48	4,78	64,06	2,95	59,9	9,16	69,78	9,64	73,82	5,96	71,88	8,55	58,2	5,32	50,8	8,14	58,48	1,32	74,26	10,21	60,94	5,36	53,79	5,18
	3	67,82	8,86	73,44	2,24	65,62	6,84	72,65	6,39	70,79	7,78	66,19	9,64	61,23	7,47	61,6	6,33	61,51	5,73	78,76	9,42	68,75	4,18	54,69	3,07
	5	75,77	9,19	72,3	2,61	63,16	8,7	74,43	8,7	70,74	4,67	67,66	5,99	53,74	7,93	60,04	4,76	59,99	5,9	77,29	9,06	67,71	3,64	56,96	6,11
	15	74,04	8,84	73,86	3,67	67,19	7,11	72,99	7,99	76,56	3,22	75	7,32	59,4	6,63	55,45	9,27	60,94	1,89	71,93	15,31	75,38	2,62	64,63	3,62
123-127	1	86,99	4,25	66,67	6,84	73,44	3,49	84,22	6,54	66,67	6,31	68,75	4,94	65,69	6,36	58,93	3,34	53,12	7,16	77,12	9,57	63,68	4,28	62,5	6
	3	91,29	7,09	70,83	9,16	78,12	5,58	85,61	6,97	76,39	8,33	68,75	7,12	69,7	6,08	58,9	6,28	56,25	11,13	82,83	9,44	71,21	5,51	59,38	5,18
	5	89,9	5,01	75	6,42	78,12	4,06	84,09	6,78	76,73	5,63	71,88	6,25	65,4	11,23	62,33	4,01	57,81	9,21	84,09	6,78	69,86	9,82	68,75	4,69
	15	91,41	6,24	79,17	6,87	76,56	3,83	89,77	5,13	80,56	8,7	78,12	4,06	71,09	2,9	64,58	3,03	60,94	12,2	86,99	6,9	80,56	6,73	70,31	4,62

Sobre los resultados obtenidos, podemos observar que las características que mayor rendimiento obtienen son EnergyTot, ZC y AVP, estando sus valores en torno al 75% con relativas bajas desviaciones estándar. Si comparamos estos resultados con los obtenidos por la FFT junto con KNN, la mejoría de rendimiento está en torno al 18%.

4.3 Transformada wavelet + Support Vector Machines

En este caso, exploramos los valores obtenidos de la clasificación de los coeficientes con Support Vector Machines para cada uno de los sujetos. Se clasifica usando *GridSearchCV*. En cuanto a los parámetros de entrada, introducimos un vector que contiene valores de C (parámetro de regularización, valores entre 0.001 y 1000), kernel (en este caso probamos todos los disponibles, los cuales son *rbf, linear, poly, sigmoid*) y gamma (coeficientes para los kernel *rbf*, *poly* y *sigmoid*)

Tabla 7. Comparación de resultados entre características cuando se efectúan clasificaciones en parejas de tareas para el clasificador máquinas de vector soporte.

Tareas	122 - 123						122 - 127						123 - 127					
Sujeto	S1		S2		S3		S1		S2		S3		S1		S2		S3	
Característica	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std
MAV	69,44	7,86	57,32	11,14	48,72	7,25	65,2	5,7	64,1	7,25	58,97	9,59	95,24	3,37	81,43	3,09	84,62	6,28
AVP	61,11	7,86	60,1	2,5	53,85	6,28	57,51	3,15	51,28	3,63	66,67	7,25	100	0	74,29	7,1	76,92	6,28
SD	66,67	13,61	57,32	5,61	46,15	10,88	54,76	7,43	56,41	7,25	74,36	7,25	92,86	5,83	79,05	5,87	66,67	3,63
SKEW	41,67	6,8	51,26	5,33	43,59	19,19	57,88	14,2	53,85	6,28	53,85	10,88	61,9	3,37	41,75	4,54	48,72	9,59
KURT	41,67	13,61	54,55	9,36	53,85	0	44,69	10,82	46,15	12,56	53,85	12,56	59,52	6,73	67,62	7,94	48,72	13,07
ZC	72,22	3,93	57,32	5,61	51,28	9,59	57,69	13,69	64,1	3,63	46,15	6,28	90,48	3,37	77,14	12,12	79,49	7,25
MC	63,89	14,16	60,1	2,5	43,59	3,63	50,37	10,83	66,67	9,59	41,03	3,63	88,1	6,73	65,4	10,33	74,36	3,63
entropy	55,56	3,93	60,1	2,5	56,41	3,63	60,07	2,07	64,1	3,63	56,41	3,63	47,62	3,37	58,1	5,99	56,41	3,63
n5	52,78	3,93	62,88	3,44	41,03	3,63	57,51	3,15	64,1	9,59	58,97	3,63	95,24	3,37	79,05	0,67	84,62	0
n25	66,67	6,8	65,4	8,4	43,59	3,63	62,64	10,19	64,1	9,59	51,28	19,19	88,1	3,37	76,83	2,47	84,62	6,28
n75	72,22	10,39	60,35	9,39	53,85	10,88	50,18	8,14	56,41	9,59	56,41	3,63	95,24	3,37	74,44	8,78	82,05	3,63
n95	80,56	10,39	54,29	3,41	64,1	18,13	62,09	13,24	64,1	7,25	66,67	3,63	90,48	3,37	72,06	5,9	87,18	3,63
median	47,22	3,93	54,55	15,05	46,15	12,56	55,31	10,82	58,97	9,59	46,15	10,88	71,43	5,83	46,51	2,92	48,72	9,59
mean	38,89	14,16	42,68	5,61	33,33	9,59	45,24	11,58	61,54	6,28	46,15	6,28	54,76	6,73	62,7	4,05	46,15	10,88
std	66,67	13,61	57,32	5,61	46,15	10,88	54,76	7,43	56,41	7,25	74,36	7,25	92,86	5,83	79,05	5,87	66,67	3,63
var	61,11	7,86	54,04	6,07	56,41	3,63	57,51	3,15	56,41	3,63	69,23	10,88	100	0	74,29	7,1	76,92	6,28
rms	69,44	7,86	57,32	11,14	48,72	7,25	65,2	5,7	64,1	7,25	58,97	9,59	95,24	3,37	81,43	3,09	84,62	6,28
EnergySubBand	72,22	10,39	57,07	1,79	56,41	3,63	50,18	8,14	61,54	16,62	71,79	7,25	100	0	74,44	8,78	76,92	6,28
PercentageSubBand	47,22	15,71	54,55	9,36	48,72	13,07	52,38	4,6	61,54	12,56	43,59	13,07	71,43	5,83	57,94	15,83	56,41	9,59
Energytot	69,44	10,39	60,1	2,5	56,41	3,63	52,38	10	64,1	3,63	48,72	3,63	88,1	6,73	76,83	6,33	74,36	7,25

Tabla 8. Comparación de resultados entre características cuando se efectúan clasificaciones multi-etiqueta para el clasificador máquinas de vector soporte.

Feature	Best params	S1		S2		S3		time consumed (ms)
		Score	std	Score	std	Score	std	
ZC	C=10, gamma='auto'	72,12	9,31	57,04	3,78	55,56	7,77	0,23
MC	C=100, gamma=0.1, kernel='sigmoid'	71,32	5,91	51,11	6,37	53,33	8,64	0,24
n25	C=1, gamma='auto', kernel='linear'	68,33	4,07	57,78	6,02	64,44	7,63	0,2
n95	C=1000, gamma=0.1	66,14	9,3	58,52	2,77	65,19	10,63	0,24
MAV	C=100, gamma='auto', kernel='linear'	66,11	6,62	59,26	5,24	64,44	6,87	0,19
rms	C=100, gamma='auto', kernel='linear'	66,11	6,62	59,26	5,24	64,44	6,87	0,2
AVP	C=100, gamma=1, kernel='poly'	65,42	5,12	58,52	2,77	58,52	6,37	0,55
n75	C=1, gamma='auto', kernel='poly'	65,42	3,9	57,04	5,54	58,52	2,77	0,2
SD	C=1000, gamma='auto', kernel='linear'	64,74	6,25	59,26	4,06	58,52	6,37	0,22
std	C=1000, gamma='auto', kernel='linear'	64,74	6,25	59,26	4,06	58,52	6,37	0,22
var	C=100, gamma=1, kernel='poly'	64,71	4,97	59,26	2,34	60	7,91	0,41
EnergySubBand	C=0.1, gamma=1, kernel='poly'	64,68	8,71	59,26	4,06	58,52	4,91	0,21
n5	C=0.01, gamma=1, kernel='poly'	63,12	9,33	60	4,32	59,26	9,07	0,19
EnergyTot	C=1000, gamma='auto', kernel='poly'	59,52	4,32	58,52	3,63	51,85	8,45	0,18
PercentageSubBand	C=0.1, gamma=1, kernel='poly'	51,51	6,44	47,41	4,91	46,67	7,63	0,21
KURT	C=10, gamma=0.02	44,84	4,13	41,48	1,48	42,96	5,02	0,26
median	C=10, gamma=0.1, kernel='sigmoid'	42,62	4,71	42,22	1,81	40	16,13	0,25
SKEW	C=10, gamma=1, kernel='sigmoid'	41,22	6,14	41,48	1,48	42,22	8,64	0,25
mean	C=1000, gamma=1, kernel='poly'	40,53	8,73	41,48	1,48	37,78	9,19	0,32
Entropy	C=1000, gamma=0.1, kernel='sigmoid'	38,23	3,65	41,48	1,48	34,81	1,81	0,26

4.3 Transformada wavelet + Gradient Boosting

En este caso, exploramos los valores obtenidos de la clasificación realizada con el modelo Gradient Boosting para cada uno de los sujetos. Se clasifica usando *GridSearchCV.*

En cuanto a los parámetros de entrada, introducimos un vector que contiene múltiples vectores de posibles valores decimales para los campos de ratio de aprendizaje, submuestra, número de estimadores y profundidad máxima.

Tabla 9. Comparación de resultados entre características cuando se efectúan clasificaciones en parejas de tareas para el clasificador Gradient Boosting.

Tareas	122 - 123						122 - 127						123 - 127					
Sujeto	S1		S2		S3		S1		S2		S3		S1		S2		S3	
Característica	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std
MAV	77,8	10,4	46,154	6,281	46,154	6,281	59,7	15	56,41	3,626	56,41	3,626	92,9	5,8	76,923	6,281	76,923	6,281
AVP	61,1	14,2	51,282	9,594	51,282	9,594	54,8	7,4	74,359	9,594	74,359	9,594	90,5	3,4	69,231	10,879	69,231	10,879
SD	66,7	6,8	56,41	9,594	56,41	9,594	59,9	4,4	64,103	9,594	64,103	9,594	92,9	5,8	66,667	3,626	66,667	3,626
SKEW	38,9	14,2	51,282	9,594	51,282	9,594	60,1	2,1	66,667	9,594	66,667	9,594	54,8	3,4	43,59	9,594	43,59	9,594
KURT	66,7	6,8	53,846	6,281	53,846	6,281	52,4	4,6	43,59	15,806	43,59	15,806	64,3	0	41,026	3,626	41,026	3,626
ZC	77,8	3,9	43,59	9,594	43,59	9,594	60,1	6,6	56,41	7,252	56,41	7,252	85,7	5,8	69,231	10,879	69,231	10,879
MC	61,1	10,4	38,462	16,617	38,462	16,617	50	9,4	61,538	10,879	61,538	10,879	83,3	3,4	69,231	10,879	69,231	10,879
entropy	52,8	3,9	51,282	9,594	51,282	9,594	60,1	2,1	51,282	9,594	51,282	9,594	45,2	6,7	56,41	3,626	56,41	3,626
n5	72,2	10,4	38,462	6,281	38,462	6,281	64,8	4,7	71,795	9,594	71,795	9,594	97,6	3,4	69,231	12,561	69,231	12,561
n25	69,4	3,9	43,59	13,074	43,59	13,074	62,5	12,6	56,41	15,806	56,41	15,806	88,1	3,4	79,487	7,252	79,487	7,252
n75	72,2	10,4	53,846	16,617	53,846	16,617	60,1	12,7	48,718	9,594	48,718	9,594	97,6	3,4	74,359	9,594	74,359	9,594
n95	75	6,8	38,462	0	38,462	0	62,1	11,7	64,103	13,074	64,103	13,074	95,2	3,4	76,923	6,281	76,923	6,281
median	58,3	13,6	43,59	19,188	43,59	19,188	60,1	6,6	56,41	3,626	56,41	3,626	52,4	8,9	41,026	3,626	41,026	3,626
mean	36,1	10,4	48,718	20,19	48,718	20,19	54,9	1,6	41,026	9,594	41,026	9,594	76,2	3,4	43,59	14,505	43,59	14,505
std	66,7	6,8	56,41	9,594	56,41	9,594	59,9	4,4	64,103	9,594	64,103	9,594	90,5	3,4	69,231	0	69,231	0
var	50	0	43,59	9,594	43,59	9,594	45,1	1,6	69,231	6,281	69,231	6,281	90,5	3,4	69,231	10,879	69,231	10,879
rms	75	6,8	38,462	6,281	38,462	6,281	59,7	15	56,41	3,626	56,41	3,626	92,9	5,8	74,359	9,594	74,359	9,594
EnergySubBand	58,3	11,8	43,59	7,252	43,59	7,252	52,2	8,5	74,359	9,594	74,359	9,594	90,5	3,4	69,231	10,879	69,231	10,879
PercentageSubBand	41,7	11,8	43,59	7,252	43,59	7,252	47,6	16,1	28,205	3,626	28,205	3,626	64,3	5,8	61,538	10,879	61,538	10,879
Energytot	72,2	17,1	56,41	9,594	56,41	9,594	47,4	1,8	64,103	3,626	64,103	3,626	90,5	3,4	61,538	6,281	61,538	6,281

Tabla 10. Comparación de resultados entre características cuando se efectúan clasificaciones multi-etiqueta para el clasificador Gradient Boosting.

Feature	Best params	S1		S2		S3		time consumed (ms)
		Score	std	Score	std	Score	std	
MAV	learning_rate=0.6, max_depth=4, min_samples_split=5, n_estimators=600	72,9	6,3	48,3	3,2	46,4	10,53	1,06
AVP	learning_rate=0.6, min_samples_split=5, n_estimators=2000	68,6	7,5	36,3	5	31,05	0,74	3,43
SD	min_samples_split=4, n_estimators=50	74,7	10,5	44,9	5,5	41,23	6,2	0,6
SKEW	max_depth=4, min_samples_split=5, n_estimators=1150	29,6	12	25,7	7,9	34,47	6,46	3,16
KURT	learning_rate=0.5, min_samples_split=4, n_estimators=50	35,8	8,5	32,9	7	39,65	2,16	0,64
ZC	learning_rate=1.0, max_depth=4, min_samples_split=5, n_estimators=50	59,1	9,2	29,4	5,3	50,09	9,32	0,7
MC	learning_rate=0.6, max_depth=4, min_samples_split=4, n_estimators=50	59,5	4	41,4	4,4	37,9	8,72	1,1
entropy	learning_rate=0.9, min_samples_split=5, n_estimators=50	18	5,2	31	3,6	27,63	2,84	4,61
n5	max_depth=4, min_samples_split=4	55,8	15	38,1	7,1	34,47	6,46	1,47
n25	learning_rate=0.2, min_samples_split=5, n_estimators=50	56,7	12,2	43	5,4	41,4	0,99	1
n75	learning_rate=0.5, max_depth=4, min_samples_split=5, n_estimators=1700	74,3	15,6	32,7	4,6	34,56	9,18	6,67
n95	learning_rate=0.8, max_depth=4, min_samples_split=5, n_estimators=1700	61,8	5,4	48,3	11	48,33	11	2,12
median	learning_rate=1.0, max_depth=4, min_samples_split=4, n_estimators=1150	36,5	13	36,3	15,7	34,47	2,18	1,51
mean	learning_rate=0.5, max_depth=4, min_samples_split=5, n_estimators=1150	25,5	8	34,6	9,4	34,56	3,23	1,68
std	learning_rate=0.9, max_depth=4, min_samples_split=5, n_estimators=1700	71,6	9,2	48,4	6	41,23	6,2	2,14
var	learning_rate=0.5, max_depth=4, min_samples_split=5, n_estimators=1150	67,7	4	41,4	4,4	32,81	6,75	1,66
rms	learning_rate=1.0, min_samples_split=4, n_estimators=50	70,6	9,8	48,3	3,2	48,16	11,78	0,46
EnergySubBand	learning_rate=0.5, max_depth=4, min_samples_split=5, n_estimators=1150	63,6	9,7	41,5	5,1	30,97	3,57	1,79
PercentageSubBand	learning_rate=0.4, max_depth=4, min_samples_split=5, n_estimators=1150	46,8	13,7	32,5	9,8	25,79	3,67	2,17
Energytot	min_samples_split=5, n_estimators=50	60,6	10,7	53,5	6,9	39,65	2,16	0,37

En este caso, al momento de efectuar los test de clasificación, se observó una gran diferencia del número de estimadores, lo que es el número total de árboles secuenciales, lo cual nos indica que para aquellas características que tienen un valor mayor a 100 sería recomendable modificar el resto de los parámetros hasta encontrar una combinación que haga más robusto el modelo para ese grupo de datos a clasificar.

4.4 Transformada wavelet + Naive Bayes GaussianNB

En este caso, exploramos los valores obtenidos de la clasificación realizada con el modelo Gradient Boosting para cada uno de los sujetos. Para este clasificador no se efectúan cálculos de combinación entre parámetros puesto que la diferencia de rendimiento observada ha sido prácticamente nula.

Tabla 11. Comparación de resultados entre clasificadores cuando se efectúan clasificaciones multi etiqueta.

Tareas	122 - 123						122 - 127						123 - 127					
Sujeto	S1		S2		S3		S1		S2		S3		S1		S2		S3	
Característica	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std
MAV	75	11,79	49,24	19,49	41,03	9,59	72,16	13,44	56,41	13,07	46,15	6,28	88,1	3,37	71,75	12,06	74,36	3,63
AVP	66,67	0	51,77	18,97	53,85	6,28	61,9	23,02	56,41	7,25	51,28	3,63	80,95	12,14	79,05	11,68	82,05	3,63
SD	66,67	0	57,32	5,61	46,15	6,28	64,65	18,53	48,72	13,07	48,72	9,59	85,71	5,83	72,06	11,7	79,49	7,25
SKEW	41,67	11,79	63,13	6,79	56,41	13,07	47,44	1,81	58,97	7,25	46,15	6,28	50	5,83	60,32	12,5	33,33	9,59
KURT	50	11,79	62,88	10,22	48,72	13,07	32,23	8,14	61,54	0	41,03	7,25	59,52	3,37	58,25	4,54	41,03	7,25
ZC	77,78	10,39	45,96	6,07	58,97	3,63	67,4	15,96	66,67	3,63	61,54	6,28	85,71	5,83	74,44	8,78	76,92	10,88
MC	80,56	10,39	43,18	8,73	58,97	3,63	60,26	17,3	64,1	9,59	53,85	0	83,33	8,91	67,62	7,94	74,36	3,63
entropy	47,22	7,86	51,77	13,21	53,85	0	45,05	1,55	56,41	7,25	53,85	0	54,76	3,37	48,89	1,57	41,03	3,63
n5	66,67	18	48,99	10,36	46,15	10,88	54,58	13,47	53,85	6,28	53,85	10,88	90,48	3,37	71,9	6,42	74,36	3,63
n25	72,22	10,39	48,74	5,33	38,46	10,88	52,56	6,54	43,59	9,59	51,28	18,13	90,48	3,37	67,14	12,62	74,36	3,63
n75	80,56	10,39	40,4	10	43,59	9,59	42,67	7,98	56,41	7,25	53,85	6,28	90,48	8,91	69,52	7,41	69,23	6,28
n95	69,44	15,71	43,18	12,99	41,03	9,59	57,33	13,49	58,97	9,59	48,72	19,19	90,48	3,37	74,29	7,1	79,49	7,25
median	69,44	15,71	51,52	7,13	43,59	9,59	57,33	4,92	41,03	3,63	51,28	13,07	73,81	13,47	58,25	9,41	56,41	19,19
mean	50	6,8	54,55	6,43	51,28	3,63	49,82	8,14	66,67	7,25	41,03	3,63	66,67	8,91	51,59	19,47	43,59	13,07
std	66,67	0	57,32	5,61	46,15	6,28	64,65	18,53	48,72	13,07	48,72	9,59	85,71	5,83	72,06	11,7	79,49	7,25
var	66,67	0	51,77	18,97	53,85	12,56	64,47	19,58	56,41	7,25	51,28	3,63	83,33	8,91	79,05	11,68	82,05	3,63
rms	75	11,79	49,24	19,49	41,03	9,59	72,16	13,44	56,41	13,07	46,15	6,28	88,1	3,37	71,75	12,06	74,36	3,63
EnergySubBand	66,67	0	51,77	18,97	51,28	9,59	61,9	23,02	56,41	7,25	43,59	9,59	85,71	5,83	76,67	9,06	82,05	3,63
PercentageSubBand	63,89	3,93	37,63	12,44	48,72	7,25	39,56	12,43	51,28	7,25	41,03	3,63	57,14	11,66	51,27	4,37	61,54	16,62
Energytot	63,89	10,39	65,66	1,43	53,85	0	62,09	13,24	58,97	3,63	41,03	7,25	88,1	3,37	78,89	10,33	76,92	0

Tabla 12. Comparación de resultados entre clasificadores cuando se efectúan clasificaciones multi etiqueta.

Feature	S1		S2		S3	
	Score	std	Score	std	Score	std
MAV	72,9	6,3	48,3	3,2	46,4	10,53
AVP	68,6	7,5	36,3	5	31,05	0,74
SD	74,7	10,5	44,9	5,5	41,23	6,2
SKEW	29,6	12	25,7	7,9	34,47	6,46
KURT	35,8	8,5	32,9	7	39,65	2,16
ZC	59,1	9,2	29,4	5,3	50,09	9,32
MC	59,5	4	41,4	4,4	37,9	8,72
entropy	18	5,2	31	3,6	27,63	2,84
n5	55,8	15	38,1	7,1	34,47	6,46
n25	56,7	12,2	43	5,4	41,4	0,99
n75	74,3	15,6	32,7	4,6	34,56	9,18
n95	61,8	5,4	48,3	11	48,33	11
median	36,5	13	36,3	15,7	34,47	2,18
mean	25,5	8	34,6	9,4	34,56	3,23
std	71,6	9,2	48,4	6	41,23	6,2
var	67,7	4	41,4	4,4	32,81	6,75
rms	70,6	9,8	48,3	3,2	48,16	11,78
EnergySubBand	63,6	9,7	41,5	5,1	30,97	3,57
PercentageSubBand	46,8	13,7	32,5	9,8	25,79	3,67
Energytot	60,6	10,7	53,5	6,9	39,65	2,16

4.5 Transformada wavelet + Random Forest

En este caso, exploramos los valores obtenidos de la clasificación realizada con el modelo Random Forest para cada uno de los sujetos. Para este clasificador, como para el caso anterior, no se efectúan cálculos de combinación entre parámetros puesto que la diferencia de rendimiento observada ha sido prácticamente nula.

Tabla 13. Comparación de resultados entre clasificadores cuando se efectúan clasificaciones multi etiqueta.

Tareas	122 - 123						122 - 127						123 - 127					
Sujeto	S1		S2		S3		S1		S2		S3		S1		S2		S3	
Característica	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std
MAV	78,07	9,25	53,42	1,12	42,89	12,13	79,74	6,89	51,75	1,24	36,14	3,47	81,4	4,54	48,25	1,24	41,05	15,92
AVP	71,23	8,35	43,16	3,1	37,72	9,69	76,49	12,29	41,32	3,38	34,39	5,85	74,65	6,84	41,4	0,99	35,96	10,15
SD	79,65	8,18	46,58	1,12	32,72	4,59	76,23	6,38	48,33	3,19	32,81	2,93	77,89	8,69	48,25	1,24	41,14	11,73
SKEW	30,35	7,74	34,21	11,16	30,96	7,05	32,02	8,02	39,39	11,85	34,3	7,87	30,26	10,44	39,39	11,85	29,39	5,3
KURT	37,28	2,06	42,98	5,41	32,81	2,93	35,53	6,73	41,32	3,38	39,82	7,27	37,19	5,69	39,56	3,85	34,56	3,23
ZC	62,54	12,38	34,74	16,58	41,32	3,38	64,3	11,17	34,65	12,73	48,42	7,34	62,63	8,83	36,32	11,63	51,75	4,47
MC	62,54	9,31	41,4	12,93	44,82	6,45	64,3	7,62	36,32	7,84	46,58	4,44	67,63	15,86	36,32	11,63	48,33	11
entropy	30,53	8,2	30,96	3,57	29,39	5,3	34,04	9,96	30,96	3,57	29,39	3,1	30,53	8,2	30,96	3,57	29,39	5,3
n5	67,98	8,02	50,09	13,38	34,3	9,94	69,65	7,74	50,09	13,38	34,56	3,23	67,98	9,88	48,42	14,2	46,49	11,03
n25	79,82	14,02	46,49	3,28	41,4	0,99	74,74	13,96	51,58	5,95	37,98	3,01	73,16	14,13	44,65	7,63	43,16	3,1
n75	79,65	12,26	48,16	5,29	41,58	11,87	76,32	8,37	48,42	10,46	41,49	11,68	76,32	12,39	44,91	9,27	41,49	11,68
n95	74,74	7,84	51,84	5,29	50	2,15	71,23	4,44	53,51	3,28	53,42	4,44	76,4	8,24	48,33	6,86	53,33	5,18
median	45,79	4,23	34,56	9,18	32,72	4,59	42,37	6,13	36,14	11,08	31,05	4,36	37,46	9,31	37,98	6,78	36,23	0,87
mean	32,37	7,18	37,98	10,95	32,98	11,22	28,95	5,58	48,42	5,95	33,07	12,78	32,37	9,21	38,07	7,14	29,47	7,04
std	72,81	8,73	48,33	6,86	35,96	10,15	79,56	7,4	46,58	4,44	42,81	12,35	77,98	10,23	51,84	8,06	39,47	7,75
var	69,47	8,2	46,67	5,18	34,3	7,87	69,56	6,78	39,65	2,16	34,3	7,87	71,23	4,44	43,16	3,1	37,72	9,69
rms	79,74	6,89	48,33	3,19	37,81	8,21	77,98	6,16	48,33	6,86	44,65	13	78,07	9,25	50	2,15	42,98	5,41
EnergySubBand	72,98	9,19	43,07	1,36	34,39	5,85	72,98	9,19	43,16	3,1	34,3	7,87	72,98	9,19	41,32	3,38	34,39	3,97
PercentageSubBand	49,39	14,11	18,86	4,34	27,54	4,64	47,81	15,81	25,7	10,82	24,12	6,48	49,39	9,95	17,11	5,98	25,96	4,73
Energytot	64,21	8,19	53,51	6,91	39,65	2,16	64,21	8,19	53,51	6,91	39,65	2,16	64,21	8,19	53,51	6,91	39,65	2,16

Tabla 14. Comparación de resultados entre clasificadores cuando se efectúan clasificaciones multi etiqueta.

Feature	S1		S2		S3	
	Score	std	Score	std	Score	std
MAV	67,98	8,02	9,52	6,73	34,13	26,39
AVP	72,89	10,22	9,52	13,47	39,68	12,5
SD	69,47	8,2	15,08	11,72	24,6	12,94
SKEW	40,61	3,36	25,4	8,09	61,11	23,52
KURT	30,53	4,15	34,92	5,94	44,44	9,78
ZC	57,81	9,69	54,76	3,37	54,76	23,57
MC	57,54	8,87	39,68	12,5	40,48	8,91
entropy	32,37	7,18	24,6	5,61	34,13	12,35
n5	68,07	11,41	35,71	10,1	25,4	18,37
n25	69,39	7,54	35,71	10,1	39,68	12,5
n75	69,65	6,57	41,27	21,41	34,92	5,94
n95	73,07	8,14	30,95	14,68	40,48	8,91
median	40,7	4,2	30,16	2,24	30,95	22,08
mean	37,37	3,35	35,71	15,43	30,95	14,68
std	74,56	4,13	14,29	20,2	30,16	11,88
var	61,05	4,35	15,08	11,72	34,92	5,94
rms	67,98	8,02	14,29	11,66	34,13	26,39
EnergySubBand	76,14	9,82	15,08	11,72	39,68	12,5
PercentageSubBand	57,72	9,03	30,16	2,24	35,71	15,43
Energytot	60,79	9,71	4,76	6,73	30,95	14,68

Los aspectos a remarcar sobre este test de clasificación con Random Forest son la clara diferencia de rendimiento entre el sujeto 1 con los sujetos 2 y 3, habiendo en ocasiones diferencias de hasta 30 puntos. Esto podría ser porque los datos de los sujetos 2 y 3 están mucho menos diferenciados entre ellos y por tanto, al ser un modelo que se realimenta de iteraciones anteriores, podría estar arrastrando resultados erróneos a lo largo del cálculo de clasificación.

4.6 Transformada wavelet + MLP

En este caso, exploramos los valores obtenidos de la clasificación realizada con el clasificador perceptrón multicapa para cada uno de los sujetos. En este caso se usó la función para calcular la mejor combinación de parámetros, pero lamentablemente aumentaba el tiempo de cómputo exponencialmente, por tanto, al descubrir que el rendimiento obtenido no era mayor que cuando se ejecutaba

el modelo con los parámetros por defecto, se eligió seguir adelante con esta última opción.

Tabla 15. Comparación de resultados entre clasificadores cuando se efectúan clasificaciones multi etiqueta.

Tareas	122 - 123						122 - 127						123 - 127					
Sujeto	S1		S2		S3		S1		S2		S3		S1		S2		S3	
Característica	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std
MAV	72,22	10,39	43,59	7,25	46,15	6,28	62,64	5	48,72	3,63	61,54	10,88	95,24	3,37	82,05	9,59	82,05	9,59
AVP	55,56	10,39	48,72	3,63	58,97	7,25	55,13	4,8	66,67	15,81	64,1	28,32	83,33	23,57	64,1	9,59	64,1	3,63
SD	61,11	7,86	48,72	3,63	46,15	10,88	57,69	8,31	43,59	7,25	58,97	7,25	90,48	8,91	69,23	6,28	69,23	6,28
SKEW	38,89	17,12	53,85	18,84	38,46	6,28	50,37	14	51,28	9,59	46,15	6,28	64,29	0	51,28	3,63	48,72	9,59
KURT	50	11,79	56,41	3,63	53,85	12,56	47,25	11,73	64,1	13,07	51,28	7,25	61,9	8,91	48,72	13,07	51,28	7,25
ZC	72,22	7,86	53,85	6,28	61,54	6,28	65,02	3,18	56,41	3,63	51,28	15,81	80,95	8,91	84,62	6,28	79,49	7,25
MC	69,44	10,39	53,85	16,62	51,28	9,59	63	17,08	35,9	13,07	41,03	3,63	80,95	8,91	82,05	7,25	79,49	3,63
entropy	55,56	3,93	56,41	3,63	58,97	7,25	57,51	3,15	51,28	9,59	56,41	3,63	54,76	3,37	56,41	3,63	56,41	3,63
n5	47,22	7,86	41,03	9,59	41,03	3,63	52,38	4,6	61,54	6,28	64,1	3,63	100	0	79,49	7,25	76,92	6,28
n25	80,56	17,12	33,33	3,63	41,03	3,63	52,93	17,49	48,72	13,07	58,97	22,06	88,1	3,37	87,18	3,63	84,62	6,28
n75	77,78	3,93	43,59	7,25	38,46	0	52,38	4,6	61,54	6,28	53,85	0	88,1	6,73	74,36	7,25	71,79	9,59
n95	66,67	6,8	35,9	15,81	46,15	12,56	65,38	11,32	58,97	3,63	56,41	3,63	95,24	3,37	82,05	3,63	84,62	0
median	47,22	3,93	43,59	14,5	41,03	9,59	59,89	4,42	66,67	14,5	53,85	22,65	57,14	11,66	51,28	9,59	58,97	3,63
mean	61,11	10,39	30,77	6,28	43,59	7,25	52,2	15,19	58,97	15,81	58,97	9,59	40,48	13,47	51,28	14,5	46,15	6,28
std	61,11	3,93	51,28	3,63	53,85	0	57,69	5,44	61,54	6,28	56,41	9,59	95,24	3,37	69,23	6,28	69,23	6,28
var	55,56	10,39	56,41	3,63	46,15	16,62	52,38	10	48,72	13,07	53,85	10,88	100	0	64,1	9,59	61,54	6,28
rms	72,22	3,93	46,15	6,28	48,72	3,63	57,69	5,44	61,54	6,28	56,41	3,63	80,95	22,08	87,18	3,63	84,62	6,28
EnergySubBand	69,44	10,39	58,97	7,25	51,28	9,59	54,95	6,47	51,28	15,81	66,67	15,81	97,62	3,37	66,67	7,25	66,67	3,63
PercentageSubBand	63,89	3,93	56,41	3,63	53,85	0	37,36	5	51,28	7,25	43,59	7,25	73,81	17,82	53,85	10,88	53,85	6,28
Energytot	61,11	10,39	56,41	3,63	56,41	3,63	60,07	2,07	56,41	3,63	53,85	6,28	78,57	20,2	56,41	3,63	56,41	3,63

Tabla 16. Comparación de resultados entre clasificadores cuando se efectúan clasificaciones multi etiqueta.

Feature	S1		S2		S3	
	Score	std	Score	std	Score	std
MAV	72,9	1,86	58,77	10,6	46,67	5,18
AVP	68,6	14,28	48,42	7,34	20,7	4,33
SD	74,7	5,87	51,93	9,47	25,79	7,1
SKEW	29,6	9,95	36,23	8,64	34,56	3,23
KURT	35,8	9,08	42,98	8,13	27,72	10,18
ZC	59,1	2,09	39,74	9,22	37,89	8,72
MC	59,5	10,33	32,63	8,29	46,58	1,12
entropy	18	2,05	39,56	3,85	34,47	2,18
n5	55,8	5,77	55,18	2,15	53,33	5,18
n25	56,7	6,78	41,23	7,55	46,58	1,12
n75	74,3	5,87	46,58	1,12	32,89	7,04
n95	61,8	8,14	53,6	7,44	39,65	6,45
median	36,5	5,28	41,49	5,07	36,32	4,96
mean	25,5	1,49	39,65	6,45	20,79	11,5
std	71,6	16,04	46,67	5,18	25,88	4,34
var	67,7	9,03	51,84	5,29	25,88	0,62
rms	70,6	1,86	53,6	7,44	37,98	6,78
EnergySubBand	63,6	6,42	51,84	8,06	32,54	8,81
PercentageSubBand	46,8	10,21	32,72	12,99	39,56	5,77
Energytot	60,6	2,05	44,91	3,47	37,98	3,01

Observamos que, como en el caso del clasificador Random Forest, los resultados de los sujetos 2 y 3 son relativamente más bajos que los obtenidos para el sujeto 1. También debemos destacar que el tiempo invertido en la clasificación aumentó considerablemente con respecto al resto de clasificadores, con iteraciones con tiempos hasta 10 veces superiores.

4.7 Transformada wavelet + Decision tree

En este caso, exploramos los valores obtenidos de la clasificación al usar el modelo de árbol de decisión para cada uno de los sujetos. Al ser un modelo desarrollado con relativa sencillez, no se hace necesaria la evaluación de múltiples parámetros de entrada. Los tests efectuados se realizan aplicando una validación cruzada 4-k.

Tabla 17. Comparación de resultados entre clasificadores cuando se efectúan clasificaciones multi etiqueta.

Tareas	122 - 123						122 - 127						123 - 127					
Sujeto	S1		S2		S3		S1		S2		S3		S1		S2		S3	
Característica	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std
MAV	80,56	7,86	45,96	9,12	35,9	7,25	56,78	20,7	66,67	9,59	69,23	6,28	92,86	5,83	79,05	5,87	79,49	7,25
AVP	66,67	13,61	62,88	3,44	46,15	6,28	59,71	10,4	71,79	13,07	76,92	10,88	90,48	3,37	72,06	0,9	76,92	10,88
SD	75	11,79	68,43	4,8	48,72	7,25	54,58	11,91	69,23	12,56	71,79	9,59	90,48	3,37	71,75	12,06	69,23	6,28
SKEW	38,89	14,16	54,55	6,43	43,59	9,59	57,51	3,15	56,41	9,59	64,1	7,25	52,38	3,37	41,75	9,41	46,15	12,56
KURT	58,33	0	51,52	7,13	53,85	6,28	52,2	10,61	53,85	12,56	48,72	7,25	57,14	10,1	55,71	4,21	38,46	0
ZC	72,22	3,93	57,32	5,61	51,28	3,63	57,51	15,7	61,54	16,62	58,97	3,63	85,71	5,83	55,87	6,1	69,23	6,28
MC	69,44	7,86	49,24	16,95	53,85	0	52,56	6,54	66,67	3,63	43,59	3,63	88,1	3,37	65,08	5,94	74,36	3,63
entropy	52,78	3,93	60,1	2,5	51,28	9,59	60,07	2,07	64,1	3,63	51,28	9,59	42,86	5,83	58,1	5,99	56,41	3,63
n5	63,89	10,39	60,1	2,5	51,28	7,25	49,82	8,14	66,67	3,63	66,67	13,07	97,62	3,37	81,27	6,97	69,23	10,88
n25	72,22	10,39	65,91	5,9	53,85	6,28	59,89	9,92	61,54	6,28	35,9	15,81	88,1	3,37	72,06	5,9	82,05	7,25
n75	77,78	14,16	54,55	9,36	53,85	16,62	64,65	13,62	64,1	3,63	53,85	22,65	95,24	3,37	74,44	8,78	74,36	7,25
n95	77,78	7,86	51,52	7,13	58,97	18,13	60,07	6,61	74,36	15,81	61,54	12,56	95,24	3,37	76,83	6,33	79,49	3,63
median	55,56	14,16	59,6	10	56,41	3,63	67,77	12,05	48,72	3,63	53,85	0	52,38	6,73	48,73	4,37	53,85	6,28
mean	27,78	3,93	51,52	7,13	43,59	15,81	65,02	3,18	58,97	3,63	43,59	7,25	57,14	11,66	51,59	13,23	48,72	7,25
std	75	11,79	68,43	4,8	48,72	7,25	54,58	11,91	69,23	12,56	71,79	9,59	90,48	3,37	71,75	12,06	69,23	6,28
var	69,44	14,16	51,77	13,21	43,59	3,63	59,71	10,4	71,79	13,07	76,92	10,88	90,48	3,37	76,67	3,75	76,92	10,88
rms	80,56	7,86	45,96	9,12	35,9	7,25	56,78	20,7	66,67	9,59	69,23	6,28	92,86	5,83	79,05	5,87	79,49	7,25
EnergySubBand	66,67	13,61	62,88	3,44	43,59	9,59	59,71	10,4	71,79	13,07	76,92	10,88	90,48	3,37	76,51	7,18	74,36	9,59
PercentageSubBand	58,33	6,8	26,01	8,07	35,9	7,25	39,93	6,61	48,72	13,07	38,46	10,88	71,43	5,83	46,35	7,64	69,23	12,56
Energytot	72,22	7,86	51,77	9,06	48,72	7,25	62,45	12,63	74,36	7,25	74,36	9,59	90,48	3,37	69,68	3,89	71,79	3,63

Tabla 18. Comparación de resultados entre clasificadores cuando se efectúan clasificaciones multi etiqueta.

Feature	S1		S2		S3	
	Score	std	Score	std	Score	std
MAV	71,23	8,35	50,18	7,51	43,07	4,51
AVP	69,39	4,85	46,93	15,51	41,32	6,95
SD	62,46	11,42	50,26	10,79	36,23	8,64
SKEW	38,86	5,54	39,56	5,77	36,05	7,65
KURT	45,88	7,77	32,81	2,93	34,56	3,23
ZC	54,39	6,2	29,39	5,3	51,84	5,29
MC	64,39	8,21	32,89	10,25	32,72	4,59
entropy	29,04	9,46	36,23	0,87	27,63	2,84
n5	67,89	5,78	45	9,57	46,49	3,28
n25	69,74	14,45	50	2,15	41,49	5,07
n75	71,23	16,42	47,02	19,22	37,98	3,01
n95	69,47	4,15	41,67	13,43	42,98	5,41
median	42,46	3,47	36,23	4,38	32,81	2,93
mean	32,19	2,09	43,16	3,1	27,63	2,84
std	62,46	11,42	50,26	10,79	36,23	8,64
var	74,56	4,13	46,93	15,51	41,32	6,95
rms	71,23	8,35	50,18	7,51	43,07	4,51
EnergySubBand	67,72	5,2	46,93	15,51	36,05	6,33
PercentageSubBand	47,89	18,24	34,47	6,46	31,32	13,22
Energytot	60,96	2,98	58,68	3,38	37,89	1,49

Como ya hemos observado en otros modelos, la clasificación de los datos por parejas de tareas arroja unos resultados relativamente buenos, puesto que las características se pueden diferenciar con mas facilidad. En el caso de la clasificación multi etiqueta observamos una fuerte caída del rendimiento, haciéndose más notable para el sujeto 3.

4.8 Transformada wavelet + Gaussian process

En este caso, exploramos los valores obtenidos de la clasificación al usar el modelo de procesos gausianos para cada uno de los sujetos. Se usa RBF (por su nombre en inglés radial-basis function) como kernel, con la escala de longitud en su valor por defecto. Los tests efectuados se realizan aplicando una validación cruzada 4-k.

Tabla 19. Comparación de resultados entre clasificadores cuando se efectúan clasificaciones multi etiqueta.

Tareas	122 - 123						122 - 127						123 - 127					
Sujeto	S1		S2		S3		S1		S2		S3		S1		S2		S3	
Característica	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std	score	std
MAV	77,78	10,39	54,29	3,41	41,03	3,63	65,38	11,32	51,28	14,5	58,97	13,07	95,24	6,73	81,43	3,09	82,05	9,59
AVP	58,33	11,79	65,91	5,9	51,28	3,63	60,07	2,07	56,41	7,25	46,15	10,88	66,67	23,57	81,43	3,09	58,97	3,63
SD	63,89	3,93	48,74	11	53,85	0	54,95	6,47	56,41	3,63	48,72	20,19	95,24	3,37	81,43	3,09	56,41	3,63
SKEW	38,89	7,86	57,07	1,79	41,03	7,25	45,05	1,55	66,67	7,25	48,72	22,06	52,38	6,73	34,76	4,71	43,59	9,59
KURT	50	20,41	62,88	3,44	46,15	6,28	47,62	10	61,54	12,56	53,85	16,62	59,52	6,73	48,73	4,37	48,72	13,07
ZC	72,22	7,86	57,32	5,61	53,85	6,28	67,58	12,78	61,54	6,28	53,85	22,65	80,95	12,14	79,37	8,98	79,49	7,25
MC	47,22	3,93	42,93	13,72	53,85	6,28	60,44	17,67	69,23	0	46,15	12,56	85,71	5,83	67,62	5,39	76,92	6,28
entropy	55,56	3,93	60,1	2,5	58,97	7,25	60,07	2,07	64,1	3,63	58,97	7,25	52,38	3,37	53,65	7,64	56,41	3,63
n5	50	6,8	60,1	2,5	66,67	9,59	52,56	12,69	64,1	9,59	64,1	3,63	97,62	3,37	81,43	3,09	79,49	7,25
n25	80,56	7,86	57,32	5,61	41,03	9,59	62,64	5	53,85	10,88	51,28	9,59	92,86	0	78,89	6,22	87,18	3,63
n75	75	11,79	68,94	13,68	43,59	9,59	54,95	1,55	66,67	9,59	48,72	3,63	92,86	0	81,27	6,97	74,36	9,59
n95	77,78	10,39	57,58	12,69	38,46	6,28	47,62	10	64,1	7,25	61,54	6,28	95,24	3,37	76,83	2,47	87,18	3,63
median	50	6,8	51,52	7,13	38,46	10,88	57,69	5,44	58,97	7,25	51,28	3,63	61,9	6,73	53,33	7,5	61,54	0
mean	61,11	3,93	48,74	5,33	33,33	9,59	55,13	4,8	61,54	0	41,03	15,81	42,86	5,83	46,51	8,75	38,46	10,88
std	63,89	3,93	48,74	11	53,85	0	54,95	6,47	56,41	3,63	48,72	20,19	95,24	3,37	81,43	3,09	56,41	3,63
var	58,33	11,79	65,91	5,9	53,85	0	57,51	3,15	58,97	3,63	46,15	10,88	100	0	81,43	3,09	58,97	3,63
rms	77,78	10,39	54,29	3,41	41,03	3,63	65,38	11,32	51,28	14,5	58,97	13,07	95,24	6,73	81,43	3,09	82,05	9,59
EnergySubBand	66,67	13,61	63,13	6,79	51,28	3,63	54,95	6,47	58,97	3,63	46,15	10,88	100	0	79,05	0,67	58,97	3,63
PercentageSubBand	63,89	3,93	54,55	9,36	53,85	0	45,05	12,66	56,41	3,63	35,9	13,07	73,81	8,91	53,33	7,5	58,97	3,63
Energytot	47,22	10,39	60,1	2,5	51,28	9,59	60,07	2,07	64,1	3,63	58,97	3,63	69,05	16,84	74,44	3,02	56,41	3,63

Tabla 20. Comparación de resultados entre clasificadores cuando se efectúan clasificaciones multi etiqueta.

Feature	S1		S2		S3	
	Score	std	Score	std	Score	std
MAV	74,56	0,62	58,68	3,38	39,65	2,16
AVP	64,39	4,17	50,18	7,51	25,79	3,67
SD	65,96	7,02	53,6	6,08	29,21	4,09
SKEW	44,04	4,3	37,98	3,01	36,23	4,38
KURT	42,37	2,05	39,65	2,16	38,16	9,55
ZC	62,72	6,13	34,56	3,23	43,07	4,51
MC	57,37	10,79	36,32	4,96	43,25	10,39
entropy	42,37	2,05	37,89	1,49	36,23	0,87
n5	71,4	9,06	56,93	1,36	46,58	1,12
n25	67,81	6,14	51,75	4,47	41,32	6,95
n75	66,23	8,08	58,77	6,2	37,89	4,55
n95	73,07	8,14	58,77	6,2	37,98	6,78
median	42,37	2,05	36,23	0,87	29,39	5,3
mean	37,28	2,06	36,32	4,96	22,28	12,05
std	65,96	7,02	53,6	6,08	29,21	4,09
var	64,47	6,73	50,18	7,51	25,79	3,67
rms	74,56	0,62	58,68	3,38	39,65	2,16
EnergySubBand	66,14	8,32	50,18	7,51	25,79	3,67
PercentageSubBand	44,04	1,36	37,72	9,69	29,56	10,3
Energytot	66,14	1,61	53,68	9,91	41,4	0,99

Analizando los datos obtenidos, volvemos a observar una fuerte bajada de rendimiento en la clasificación multi etiqueta debido a la poca diferenciación de los valores internos de las características. Un aspecto para remarcar es el tiempo invertido para entrenar y clasificar los datos, siendo unas 32000 veces más lento que otros modelos, como SVM (support vector machines), lo que no es nada desdeñable.

4.9 Discusión

En el presente trabajo se planteó el objetivo de proporcionar una arquitectura en lenguaje Python capaz de procesar y clasificar datos obtenidos de distintos sujetos cuando realizan idénticas tareas por medio de técnicas no invasivas.

Tras la obtención de los resultados, como punto de partida, al comparar los resultados de las clasificaciones para los 3 sujetos descubrimos la diferencia de puntuación existente entre el sujeto 1 y el resto de los sujetos. Esto es debido, principalmente, a que los datos de las tareas del sujeto 1 muestran una alta disimilitud entre ellos, obteniendo de esta forma un mayor rendimiento y mejores clasificaciones.[49] Podemos observar esta diferencia de puntuación en los resultados obtenidos en la BCI competition III Data set V en la que en casi la totalidad de los participantes obtuvieron peores resultados con los sujetos 2 y 3 en comparación con el sujeto 1, pudiendo observar diferencias de puntuación de entorno al 30% entre el sujeto 1 y 3.[50]

Por tanto, se toman los resultados del sujeto 1 para comparar el rendimiento tanto de los modelos de clasificación como de la elección de los grupos de electrodos puesto que son los resultados con más consistencia en todo el proceso de testing y cálculo de puntuaciones. Con el anterior pretexto, podemos determinar que la solución obtenida en base a los tests realizados son:

- Sobre la elección de electrodos, se repite el mismo procedimiento en múltiples ocasiones para distintos grupos de electrodos. Para este análisis usamos como procesador la transformada de wavelet y como clasificador KNN, de esta forma, la comparación es coherente. Por lo cual, podemos concluir que la mejor combinación es la de los electrodos FZ, CZ, FC5, FC6, FC1, FC2.

- En cuanto al procesamiento de señales y extracción de características, se compara el rendimiento de la transformada rápida de Fourier y la transformada de wavelet junto con la WPD (Wavelet Packet Decomposition). Esta última nos arroja un resultado ligeramente superior al obtenido con la transformada rápida de Fourier. Se establecen las mismas condiciones para ambos, siendo el clasificador a usar KNN y comparando se evidencia, que, en el mejor de los casos, se produce un incremento de, aproximadamente, un 21%.
  Sin embargo, si observamos los resultados de todas las características, en 13 de las 19 consideradas si bien no llegan al 21%, muestran una mejora de al menos 8,6%.

En lo que respecta la clasificación de las señales para su posterior utilización, se compararon los resultados de siete clasificadores en lo que concierne la puntuación, la desviación y el tiempo. Los resultados se dividen en dos grupos: resultados multi etiqueta y resultados obtenidos clasificando por pares. A su vez, para el cálculo de estos resultados, se han tomado los resultados de las 5 características con las mejores puntuaciones y calculado la media de estas, resultando en las tablas __ y __.

Tabla 21. Comparación de resultados entre clasificadores cuando se efectúan clasificaciones multi etiqueta.

Clasificador usuario	SVM	Gradient Boosting	Naive Bayes Gaussian	Random Forest	MLP	Decision tree	Gaussian process	KNN
S1	68,71	72,79	69,65	73,2	72,79	71,56	72,19	64,81
S2	59,41	50,74	47,81	40,42	54,52	51,71	58,36	49,3
S3	63,64	46,68	44,65	46,93	46,15	45,25	43,04	42,97
media	**63,69**	54,68	52,02	50,24	55,82	54,14	55,33	50,86

Tabla 22. Comparación de resultados entre clasificadores cuando se efectúan clasificaciones multi etiqueta.

Clasificador usuario	SVM	Gradient Boosting	Naive Bayes Gaussian	Random Forest	MLP	Decision tree	Gaussian process	KNN
S1	73,32	74,47	75,16	78,04	72,13	74,5	75,1	64,81
S2	66,97	61,5	61,74	50,77	54,32	69,27	67,76	49,3
S3	66,09	61,5	60,15	45,39	60,93	62,72	59,69	42,97
media	**68,65**	65,29	65,04	55	61,62	68,49	66,92	50,86

Tabla 23. Comparación de tiempos de ejecución de los clasificadores empleados.

Clasificador	Media tiempo consumido (ms)
SVM	0,22
Gradient Boosting	4,6
Naive Bayes Gaussian	0,18
Random Forest	7,27
MLP	0,7
Decision tree	0,12
Gaussian process	71,57
KNN	0,33

En base a los resultados obtenidos, podemos concluir, tras haber comparado los datos calculados tanto en clasificaciones multi etiqueta como las efectuadas por pares de tareas, el clasificador que nos arroja la mayor puntuación es máquinas de vector soporte, con una puntuación media de 63,69% para la clasificación multi etiqueta y 68,65% para la clasificación por pares. Además, observamos que al efectuar los cálculos de clasificación por pares, el rendimiento aumenta una media de 14,24%, siendo notablemente superior la mejoría en casos como el clasificador de árbol de decisión, para el cual el aumento de rendimiento fue de un 26,50%.

En cuanto a las características, se observa que, para cada clasificador, la combinación de ellos que mayor rendimiento nos arroja es cambiante, lo que podría deberse a la forma que tiene cada uno de los clasificadores de predecir el tipo de tarea al que pertenecen los datos que se están clasificando. Para la mejora del rendimiento se ha usado en la mayoría de los tests la función *GridSearchCV,* la cual nos proporciona la posibilidad de evaluar para un mismo set de datos, que combinación de valores en los parámetros de entrada del modelo de clasificación provoca que el mismo efectúe mejores clasificaciones y por tanto, ajustando el modelo a los datos que se quiere clasificar.

En este sentido, encontramos una limitación debido a que el hardware que se ha usado para el cálculo de clasificaciones no está completamente optimizado para ello. Por ello, en algunos de los tests realizados, en los que se buscaba la mejor configuración de un modelo para un determinado cuadro de datos, hemos observado que el tiempo de cómputo podía incluso llegar a las 5 horas. Si bien, una vez entrenado el modelo podemos almacenarlo en memoria y hacer uso de este en un futuro, sería deseable utilizar plataformas de hardware especializadas en llevar a cabo estos procesos de cómputo pesado como las ofrecidas por amazon web services.

Capítulo 5. Conclusión

En el presente trabajo se ha desarrollado una arquitectura software completamente en Python y para el cual se han usado distintas librerías usadas ampliamente en Machine Learning. En este sentido, se ha hecho posible la conversión de un código ya existente en Matlab, con el cual se procesaban y extraían características por medio de la transformada rápida de Fourier y finalmente se clasificaban con el algoritmo de vecinos cercanos.

Tras la obtención de los resultados y discusión de los mismos podemos concluir:

- Al desarrollar la solución en Python, incluso habiendo construido un algoritmo que limpia y extrae de un archivo de matlab los datos que están estructurados para la BCI Competition, dotamos a cualquier estudiante o investigador, con conocimientos en este lenguaje, de un código que se puede usar de forma libre, escalar y ampliar con nuevos procesadores, extractores de características o clasificadores.

- Por otro lado, comparando los resultados obtenidos en este proyecto, concluimos que la mejor configuración, de las propuestas, es el uso de la transformada de wavelet tanto para el procesamiento de la señal como para la extracción de características (implementando el algoritmo de wavelet packet decomposition) y máquinas de vector soporte como clasificador.

- Por último, pese a que los resultados en algunas ocasiones son considerablemente mejores que los que se obtuvieron previamente a este trabajo, queda todavía un amplio margen de mejora, sobre todo en la configuración de los parámetros de entrada de los modelos. Aun así, no es desdeñable la mejora de rendimiento en la clasificación de las señales EEG, arrojando valores que superan las mejores puntuaciones previamente obtenidas hasta en un, aproximadamente, 30%.

5.1 Trabajos futuros

Durante el trabajo se han detectado puntos que se podrían mejorar/ampliar en futuros trabajos. Se indagó brevemente en la inclusión de una red neuronal convolucional capaz de clasificar las señales a través de los diagramas obtenidos de los cálculos de la transformada continua de wavelet. Finalmente, no se tuvo en cuenta por que se carecía de una amplia variedad de imágenes para el correcto entrenamiento de la red, arrojando resultados muy bajos.

Por otro lado, las pruebas realizadas para evaluar el rendimiento de la arquitectura desarrollada se han efectuado en un entorno offline. En este sentido, sería interesante plantear un flujo de procesos que pudieran ejecutarse de forma online.

Bibliografía

[1] Sanei, S., & Chambers, J. A. (2007). EEG signal processing. Centre of Digital Signal Processing Cardiff University, UK John Wiley & Sons.

[2] Wolpaw JR,Birbaumer,N, McFarland DJ.,Pfurtscheller;G.,Vaughan,TM. (2002) Brain–computer interfaces for communication and control,Ireland, Elsevier,*Clinical Neurophysiology 113 (2002) 767–791*

[3]Cichocki,A.& S. Sanei,S. (2007) EEG/MEG Signal Processing,UK,Hindawi Publishing Corporation Computational Intelligence and Neuroscience Volume 2007, Article ID 97026, 2 pages 1&2,doi:10.1155/2007/97026

[4] Guyton A.C. y Hall J.E. (2016)Tratado de fisiología médica ,España,Elsevier.

[5]Crossman,A.R y Neary,D.(2007) Neuroanatomía,España,Elsevier.

[6] Guger,G., Schlögl,A., Neuper,C, Walterspacher,D., Strein,T.,& Pfurtscheller,G.(2001) Rapid Prototyping of an EEG-Based Brain–Computer Interface (BCI)IEEE Transactions on Neural Systems and Rehabilitations engineering, Vol. 9, NO. 1, March 2001

[7]Géron,A.(2017)Hands-on Machine learning with Scikit-Learn and TensorFlow, USA,O'Reilly.

[8]Madi,S & Baba-Ali,R.,Comparison of classification techniques for wall following robot navigation and improvements to the KNN ,Algiers,algorithm,DOI: 10.5121/csit.2019.91806.

[9]Talla,S. , Venigalla,P., Shaik,A., Vuyyuru,M.(2019) Multiclass Classification Using Random Forest Classifier, India,*International Journal of Scientific Research in Computer Science*, *Engineering and Information Technology,*ISSN :

2456-3307 UGC Journal No : 64718,Volume 5, Issue 2, March-April -2019 | http://ijsrcseit.com,DOI : https://doi.org/10.32628/CSEIT183821

[10] Xu,B., Fu,Y., Miao,L,Wang,Z,Li,H.(2011),Classification of fNIRS Data Using Wavelets and Support Vector Machine during Speed and Force Imagination ,*Proceedings of the 2011 IEEE International Conference on Robotics Biometrics,*December 2011 Phuket, Thailand.

[11] Wang,D. , Miao,D, Xie,C (2011)Best basis-based wavelet packet entropy feature extraction and hierarchical EEG classification for epileptic detection,China,Elsevier, *Expert Systems with Applications,*Volume 38, Issue 11, October 2011,pp 14314-14320.https://doi.org/10.1016/j.eswa.2011.05.096

[12] Ceres,R., Mañanas,M.A.,Azorín ,J.M. (2011) Interfaces y sistemas en rehabilitación y compensación funcional para la autonomía personal y para la terapia clínica,in *Revista Iberoamericana de Automática e Informática Industrial,*ISSN:1697-7912.Vol 8 Núm 2,Abril 2011,pp 5 a 15,DOI:10.4995/RIAI.2011.02.03

[13] Dornhege,G., Millán,J.dR.,Hinterberger,T., McFarland,D.J. & Müller,K.R. (2007) Toward brain-computer Interfacing, Cambridge, Massachusetts,The MIT Press.

[14]Lebedev,M.A. & Nicolelis,M.A.L (2006) ,Brain–machine interfaces: past, present and future,*TRENDS in Neurosciences Vol.29 No.9* ,USA, Elsevier Ltd. All rights reserved. doi:10.1016/j.tins.2006.07.004.

[15] Liu,N.H.,Chiang ,CY,Chu,H.C. (2013) Recognizing the Degree of Human Attention Using EEG Signals from Mobile Sensors,ISSN 1424-8220 www.mdpi.com/journal/sensors,licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution license.

[16] Smith,E.J. Introduction to EEG An Introduction to EEG (ebme.co.uk) (consultado el 30/1/2022)

[17] 10-20 system (EEG) 10–20 system (EEG) - Wikipedia (consultado el 28/1/2022)

[18] Machine learning,¿qué es y cómo funciona? Te contamos qué es el 'machine learning' y cómo funciona (bbva.com) (consultado el 30/1/2022)

[19] Electroencephalogram (EEG),The Gale Encyclopedia of Science,Elsevier. Electroencephalography | Encyclopedia.com (consultado el 30/1/2022)

[20] Fu,Y, Xu,B, Lili Pei,L. , Li,H (2010)The Architecture and Key Technologies of A BMI Based Telerobot System over Internet,China,2010 3rd International Congress on Image and Signal Processing (CISP2010).

[21] Mathuranathan,(2020) Introduction to Signal Processing for Machine Learning
Introduction to Signal Processing for Machine Learning - GaussianWaves (consultado el 5/2/2022)

[22] Random forest ,IBM cloud learn hub What is Random Forest? | IBM (consultado el 5/2/2022)

[23] Procesos gaussianos,Unipython Procesos Gaussianos - ▷ Cursos de Programación de 0 a Experto © Garantizados (unipython.com) (Consultado el 5/2/2022)

[24] Kvartalnyi,N Gradient boosting classifier https://medium.com/geekculture/gradient-boosting-classifier-f7a6834979d8 (consultado el 9/2/2022)

[25] Taspinar,A.A guide for using the Wavelet Transform in Machine Learning https://ataspinar.com/2018/12/21/a-guide-for-using-the-wavelet-transform-in-machine-learning/ (consultado el 8/2/2022)

[26] Wang,D., Miao,D.,Xie,C. (2011)Best basis-based wavelet packet entropy feature extraction and hierarchical EEG classification for epileptic detection,China, *Expert Systems with Applications,Volume 38, Issue 11, October 2011, Pages 14314-14320*
Elsevier .doi:10.1016/j.eswa.2011.05.096.

[27] Al-Qazzaz,N.K., Ali,S. H. B. , Ahmad,S.A.,Islam,M.S. & Escudero,J.(2015) Selection of Mother Wavelet Functions for Multi-Channel EEG Signal Analysis during a Working Memory Task,*Sensors 2015, 15, 29015-29035; doi:10.3390/s151129015.*

[28] Xuesheng Peng,X., Chen,R. ,Yu,K ,Ye,F & Xue,W.(2020) An Improved Weighted K-Nearest Neighbor Algorithm for Indoor Localization,China,*Electronics 2020, 9, 2117; doi:10.3390/electronics9122117,http://www.mdpi.com/journal/electronics*

[29] Alazrai,R.,Momani,M.,Khudair,H.A, Daoud,M.I. (2017)EEG-based tonic cold pain recognition system using wavelet transform,Amman,Jordan,Neural Comput & Applic (2019) 31:3187–3200,DOI 10.1007/s00521-017-3263-6

[30] Henríquez Muñoz, C.N.(2014) Estudio de técnicas de análisis y clasificación de señales EEG en el contexto de sistemas BCI (Brain Computer Interface),UAM. Departamento de Ingeniería Informática,http://hdl.handle.net/10486/660477.

[31] Puri,D. , Ingle,R., Kachare,P. , Patil,M. & Awale,R.(2017)Wavelet Packet Sub-band Based Classification of Alcoholic and Controlled State EEG Signals ,Mumbai,India,© 2017- Atlantis Press . (http://creativecommons.org/licens)es/by-nc/4)

[32] Du,E., Wang,W, Zhang,C & Ren,J.(2014)Feature Extraction of Frequency Bands Power Based on Wavelet Packet Decomposition,China,Atlantis Press.

[33] Subasi,A. (2006)EEG signal classification using wavelet feature extraction and a mixture of expert model ,Turkía,Elsevier,*Expert Systems with Applications* 32 (2007) 1084–1093,doi:10.1016/j.eswa.2006.02.005

[34]Geng,X., Li,D., Chen,H., Yu,P., Yan,H., Yue,M.(2021)An improved feature extraction algorithms of EEG signals based on motor imagery brain-computer interface,China,Elsevier BV on behalf of Faculty of Engineering, Alexandria University,*Alexandria Engineering Journal*.https://doi.org/10.1016/j.aej.2021.10.034,http://creativecommons.org/licenses/by-nc-nd/4.0/.

[35] Zhang,X., Tanaka,T., Jin,J. (2005)Application of wavelet transform in pulsed ultrasonic modulation voltammetry,Japon,Elsevier,*Ultrasonics Sonochemistry* 13 (2006) 133–138,doi:10.1016/j.ultsonch.2005.01.005,http://www.elsevier.com/locate/ultsonch

[36] Mamun,Md,Al-Kadi,M. , Marufuzzaman ,M.(2013)Effectiveness of Wavelet Denoising on Electroencephalogram Signals ,Malaysia,*Journal of Applied Research and Technology,*pp 150-160.

[37] Albaqami ,H.,Hassan,G.M. , Subasi,A. , Datta,A. .(2021)Automatic detection of abnormal EEG signals using wavelet feature extraction and gradient boosting decision tree,Elsevier,https://www.sciencedirect.com/journal/biomedical-signal-processing-and-control.https://doi.org/10.1016/j.bspc.2021.102957

[38] Alhassana,S., AlDammas,M.A , Soudania,A. (2019)Energy-efficient Sensor-based EEG Features' Extraction for Epilepsy Detection,Saudi Arabia,Elsevier,*Procedia Computer Science 160* (2019) 273–28,http://creativecommons.org/licenses/by-nc-nd/4.0/

[39] Kang.Y, Liu,H. , Aziz.M.M.A. , Kassim,K.A.(2018)A wavelet transform method for studying the energy distribution characteristics of microseismicities associated rock failure,China,Elsevier,*Journal of Traffic and Transportation Engineering,Volume 6, Issue 6, December 2019, pp 631-646,https://doi.org/10.1016/j.jtte.2018.03.007*
http://creativecommons.org/licenses/by-nc-nd/4.0/

[40] Ting,W, Guo-zheng,Y , Bang-hua,Y, Hong,S .(2007)EEG feature extraction based on wavelet packet decomposition for brain computer interface,China,Elsevier,*Measurement 41* (2008) 618–625doi:10.1016/j.measurement.2007.07.007

[41] sklearn.model_selection sklearn.model_selection.GridSearchCV — scikit-learn 1.0.2 documentation (consultado el 8/2/2022)

[42]Hong,S.(2021)Drowsiness Detection Based on Intelligent Systems with Nonlinear Features for Optimal Placement of Encephalogram Electrodes on the Cerebral Area,China.
(a) The 10-10 electrode placement system (Bold: 10-20 system). (b) The... | Download Scientific Diagram (researchgate.net) DOI:10.3390/s21041255

[43] Cortés,J.A., Medina,F.A,Chaves,J.A.(2007) Del análisis de Fourier a las wavelets análisis de Fourier,(Colombia) *Scientia et Technica Año XIII, No 34,* Mayo de 2007. Universidad Tecnológica de Pereira. ISSN 0122-1701

[44] Advancing technology for humanity IEEE Xplore (consultado el 30/1/2022)

[45] Too,J., Abdullah,A.R ,Saad,N.M.(2019) Classification of Hand Movements based on Discrete Wavelet Transform and Enhanced Feature Extraction,Malaysia,(IJACSA) *International Journal of Advanced Computer Science and Applications*, Vol. 10, No. 6, 2019

[46] Elsevier Elsevier | Una empresa de análisis de la información | Empowering Knowledge (consultado el 30/1/2022)

[47] Machine learnia https://www.youtube.com/channel/UCmpptkXu8iIFe6kfDK5o7VQ (consultado el 12/2/2022)

[48] Tung,T.H. ,Wang,S.H ,Huang,C.C.,Su,T.Y. & Lo,C.M. (2020) Use of Discrete Wavelet Transform to Assess Impedance Fluctuations Obtained from Cellular Micromotion,Taiwan,*Sensors 2020,* 20, 3250; doi:10.3390/s20113250,http://www.mdpi.com/journal/sensors

[49] Fangohr H. (2004) A Comparison of C, MATLAB, and Python as Teaching Languages in Engineering. In: Bubak M., van Albada G.D., Sloot P.M.A., Dongarra J. (eds) Computational Science - ICCS 2004. ICCS 2004. Lecture Notes in Computer Science, vol 3039. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-25944-2_157

[50] Oostenveld,R., Praamstra,P.(2001) The five percent electrode system for high-resolution EEG and ERP measurements,(Ireland)Elsevier,*Clinical Neurophysiology 112 (2001) pp 713-719*

[51] Galán, F.(2005) “BCI Competition III . Data Set V : Algorithm Description.”

[52] BCI Competition III, Final results https://www.bbci.de/competition/iii/results/index.html (consultado el 21/2/2022)

[53] http://training.seer.cancer.gov/module_anatomy/unit5_3_nerve_org1_cns.html Brain | SEER Training (cancer.gov) (consultado el 21/2/2022)

[54] Interfaz cerebro-computadora - Wikipedia, la enciclopedia libre (consultado el 21/2/2022

[55] Azorin, J.& Iáñez,E. & Fernandez,E. & Sabater-Navarro, J. (2010). Interacción ocular con Robots: Una ayuda para discapacitados. Dyna. 85. 768-776. 10.6036/3738.

[56] Martin, R. (2008). Clean Code: A Handbook of Agile Software Craftsmanship.

[57] Multiple Time Series Classification by Using Continuous Wavelet Transformation https://towardsdatascience.com/multiple-time-series-classification-by-using-continuous-wavelet-transformation-d29df97c0442 (consultado el 24/2/2022)