Universidad de Valladolid

Escuela Técnica Superior de Ingenieros de Telecomunicación

Grado en Ingeniería de Tecnologías de Telecomunicación

Predicción de edad a partir de datos de ECG

Autor:
Alberto Martínez Fraile

Tutores:
Rodrigo de Luis García
Rafael Navarro González

Departamento de Teoría de la Señal y Comunicaciones e Ingeniería Telemática

Curso: 2023-2024

TÍTULO:	Predicción de edad a partir de datos de ECG
AUTOR:	Alberto Martínez Fraile
TUTORES:	Rodrigo de Luis García
	Rafael Navarro González
DEPARTAMENTO:	Teoría de la Señal y Comunicaciones e Ingeniería Telemática

TRIBUNAL DE EVALUACIÓN

PRESIDENTE:	Pablo Casaseca de la Higuera
SECRETARIO:	Federico Simmross Wattenberg
VOCAL:	Rodrigo de Luis García
SUPLENTE 1:	Santiago Aja Fernández
SUPLENTE 2:	Miguel Ángel Martínez Fernández

Agradecimientos

No puedo evitar empezar esta sección agradeciendo en primer lugar a mi madre, la persona que me ha apoyado a lo largo toda mi etapa educativa y me ha motivado día tras día a seguir adelante, a pesar de las dificultades que puedan surgir. También quiero agradecer a mi hermana por el apoyo que, en diversas ocasiones, me brindó.

Mi agradecimiento también va para mis tutores del proyecto, Rodrigo y Rafael, quienes me han ayudado a realizar este trabajo. A Rodri, por haber creado los códigos responsables del procesamiento y volcado de los datos en el servidor, además de evaluar los resultados obtenidos y detectar varios fallos de los que no me habría percatado. Y a Rafa, por proporcionarme los recursos necesarios para adentrarme en el desarrollo del aprendizaje automático y por brindarme su apoyo durante la realización del proyecto.

No quiero olvidar agradecer a los miembros del Laboratorio de Procesado de Imágenes (LPI) de la Universidad de Valladolid, quienes me ayudaron con el trabajo en el servidor del grupo. También a los administradores del LPI, en especial a Federico, quien me facilitó el acceso remoto al servidor.

Finalmente, quiero agradecer a mis amigos, quienes no solo me han apoyado durante el tiempo en el que realicé este trabajo, si no que han sido una gran ayuda para superar de forma amena toda la etapa universitaria.

Resumen

El electrocardiograma (ECG) es una herramienta esencial para el diagnóstico de enfermedades cardiovasculares. Es una opción recurrida por los profesionales sanitarios ya que es barata, rápida y eficaz. El aprendizaje automático es capaz de identificar patrones y características que no son detectadas en el análisis convencional.

Por ello, en este Trabajo Fin de Grado se plantea la aplicación del aprendizaje automático, una tecnología que ha experimentado avances significativos en los últimos años. En este proyecto ha sido usado el aprendizaje automático para estimar la edad cardiaca del paciente a través del electrocardiograma que le ha sido realizado. Para lograr esta tarea se alimenta un modelo de aprendizaje automático con datos etiquetados provenientes de electrocardiogramas, a partir de los cuales es capaz de detectar patrones que asocia con la edad cardiaca. Conocer la edad estimada por el programa es útil para detectar la presencia de patologías cardiacas, ya que si la edad estimada es mucho mayor a la real se puede determinar que la persona presenta algún problema cardiaco y debe ser derivado a la consulta de un especialista para una evaluación en detalle.

La discrepancia entre la edad estimada y la edad cronológica del paciente podría indicar la presencia de patologías cardiacas, lo que justifica su uso como una herramienta complementaria para identificar pacientes que podrían beneficiarse de una evaluación clínica especializada.

Es también posible aplicar el aprendizaje automático para detectar patologías concretas, aunque este no es el objetivo del proyecto.

Para resolver esta tarea se han empleado dos redes de aprendizaje profundo con diferentes arquitecturas: una red convolucional 2D que emplea las imágenes de los espectrogramas resultantes de las señales extraídas de los ECG como entradas y otra red convolucional 1D que emplea las señales crudas, las procesa y las pasa al modelo para estimar la edad cardiaca. El resultado de ambas redes es que la edad cronológica de un paciente se relaciona con la edad cardiaca predicha, por lo que ambos modelos sí son capaces de estimar la conocida como edad cardiaca. Al comparar ambas redes se concluye que el empleo de espectrogramas mejora la capacidad del modelo para aprender a detectar la información que contienen las señales.

Investigaciones previas han resuelto este mismo problema. No obstante, las redes diseñadas en esos estudios ofrecían peores resultados cuando se entrenaban con la base de datos empleada en este mismo proyecto.

Palabras Clave

Edad Cardíaca, Aprendizaje Automático, CNN, Electrocardiograma, Estimación de edad.

Abstract

The electrocardiogram (ECG) is an essential tool for diagnosing cardiovascular diseases. It is a preferred option for healthcare professionals because it is inexpensive, fast, and effective. Machine learning can identify patterns and features that are not detected through conventional analysis.

For this reason, this project proposes the application of machine learning, a technology that has undergone significant advancements in recent years. In this project, machine learning has been used to estimate the cardiac age of a patient through their electrocardiogram. To accomplish this task, a machine learning model is fed with labelled data from electrocardiograms, enabling it to detect patterns associated with cardiac age. Knowing the estimated age provided by the program is useful for detecting the presence of cardiac pathologies, as an estimated age significantly higher than the actual age can indicate that the person has a cardiac problem and should be referred to a specialist for a detailed evaluation.

The discrepancy between the estimated age and the patient's chronological age could indicate the presence of cardiac pathologies, which justifies its use as a complementary tool to identify patients who might benefit from specialized clinical evaluation.

It is also possible to apply machine learning to detect specific pathologies, although this is not the objective of this project.

To address this task, two deep learning networks with different architectures were employed: a 2D convolutional network that uses spectrogram images derived from the signals extracted from ECGs as inputs, and a 1D convolutional network that processes raw signals and inputs them into the model to estimate cardiac age. The results from both networks indicate that the chronological age of a patient correlates with the predicted cardiac age, demonstrating that both models can estimate what is known as cardiac age. Comparing the two networks reveals that the use of spectrograms improves the model's ability to learn to detect the information contained in the signals.

Previous research has addressed this same problem. However, the networks designed in those studies produced worse results when trained on the database used in this project.

Keywords

Cardiac Age, Machine Learning, CNN, Electrocardiogram, Age estimation.

Índice

Capítulo 1. Introducción

1.1 Contexto

El electrocardiograma o ECG es una herramienta esencial en la evaluación de la salud cardiaca de una persona, ya que permite evaluar el estado del sistema cardiovascular del organismo. El ECG es usado para detectar posibles problemas en el sistema cardiaco o para estudiar la respuesta del cuerpo humano ante diferentes estímulos físicos. También es empleada con el fin de estudiar la evolución de un problema conocido. El electrocardiograma es una herramienta versátil, barata y poco invasiva altamente usada en todo el mundo. Sin embargo esta herramienta contiene gran información que es difícil de percibir por los procedimientos comunes, por lo que la aplicación del aprendizaje automático puede extraer información novedosa de gran utilidad.

El envejecimiento es un proceso natural e inevitable que tiene un impacto directo en la salud de las personas. A medida que envejecemos, el cuerpo sufre cambios que aumentan la vulnerabilidad a una amplia variedad de enfermedades, como la hipertensión o la insuficiencia cardíaca, y a afecciones en el sistema inmunológico, el cual se debilita con el tiempo. Estos cambios afectan directamente en el estilo de vida, ya que pueden traer consigo limitaciones en la movilidad o cambios en los sentidos. Es importante tener una medida de cuantificación del envejecimiento, porque esto significaría cuantificar el riesgo de contraer una variedad significativa de enfermedades y poder tratar de evitar en la medida de lo posible cualquier efecto adverso consecuente.

En la sociedad está comúnmente aceptado emplear la edad cronológica como un indicador de salud, cuanto mayor es esta edad, peor debería ser nuestra salud. Sin embargo, es sabido que no siempre es así, nuestra salud depende en mayor medida de factores externos a la edad, como son nuestros hábitos alimenticios, comportamiento social o calidad del sueño. Es por esto por lo que se buscó un indicativo más certero para evaluar la salud, y se propuso la edad biológica [1]. La edad biológica es un concepto que evalúa el estado del organismo, de manera que tiene en cuenta cómo funciona cada sistema en el interior del cuerpo humano. Se puede estimar evaluando numerosos biomarcadores, como la longitud de los telómeros, la capacidad pulmonar o el estrés cardiovascular. Un concepto relacionado es la denominada edad cardíaca, esta evalúa el estado y funcionamiento del sistema cardiovascular, por lo que influye directamente en la edad biológica. Una posibilidad barata, rápida y económica de estudiar la edad cardiaca es a partir del ECG.

En los últimos años han surgido pocos estudios que tratan de estimar la edad cardíaca por medio de diferentes métodos como son el estudio de la estructura del corazón mediante ecocardiografía o, más recientemente, modelos predictivos basados en inteligencia artificial y aprendizaje profundo que estiman la edad cardiaca a partir de las señales de ECG de los pacientes. Este último método es rápido y preciso, ya que únicamente requiere de la acción humana para capturar las muestras de ECG.

En este trabajo se ha realizado el diseño, planificación y desarrollo de un método predictivo basado en aprendizaje automático, que emplee datos derivados de los ECG para estimar la edad del paciente asociado. Ingeniosamente se propone emplear como datos de entrada al modelo

una representación alternativa de la señal de los ECG a través de espectrogramas. Esta modificación se propone en base a que los espectrogramas permiten representar las señales de los ECG en el dominio frecuencial, facilitando la extracción de características reconocibles en este dominio. Además, al convertir las señales a imágenes es posible emplear redes neuronales efectivas en el análisis de imágenes.

1.2 Objetivos

El objetivo del presente trabajo es estudiar la viabilidad de la predicción de edad a partir de datos de ECG usando una representación alternativa de la señal del ECG a través de espectrogramas.

Además, se utilizarán los resultados obtenidos a partir de los modelos de aprendizaje automático para relacionar el rendimiento de este sistema con el obtenido mediante modelos que emplean representaciones directas de las señales como datos de entrada. El objetivo de este paso es probar la utilidad de los espectrogramas en este tipo de tareas. También se interpretarán los resultados obtenidos para estimar la importancia relativa de los diferentes canales de un ECG para la predicción de edad.

1.3 Estructura del documento

El presente documento se organiza en los seis capítulos listados en el índice. A continuación, se detalla el contenido de cada capítulo:

1. El primer capítulo corresponde a la introducción, en esta se presentan al lector las ideas básicas que conforman el trasfondo del trabajo. La introducción se divide en tres subpartes: el contexto, donde se expone de forma breve el marco contextual que abarca el proyecto y la motivación que llevaron al autor a realizar este proyecto, los objetivos propuestos para la finalización de este trabajo fin de grado y la estructura del presente documento.
2. En el capítulo segundo se expone el estado del arte de los distintos temas principales que componen el trabajo. Este capítulo inicia con un punto en el que se estudia la literatura actual sobre la aplicación del aprendizaje automático en la medicina. En este punto se abordan conceptos clave y una breve historia de la inteligencia artificial, seguidos de una explicación del aprendizaje automático y los distintos tipos que lo componen. En este mismo punto se detallan las bases de las redes neuronales, se explica el concepto de hiperparámetros y se muestran los ejemplos más típicos. El punto finaliza con una clase sobre como la inteligencia artificial está transformando la industria médica.

   A este punto le sigue un apartado en el que se explican los fundamentos teóricos del electrocardiograma, así como su interpretación médica. Tras este se explicará más detalladamente el concepto de edad biológica y se expondrá su importancia para la evaluación de la salud. Finalizando el capítulo se centrara especial atención en el concepto de edad cardíaca y se analizarán estudios previos de la estimación de edad a partir de datos de ECG.

3. El tercer capítulo concentra la metodología seguida para la elaboración del trabajo fin de grado. Para empezar el capítulo se describen los datos empleados, su fuente, sus características y se realiza un estudio población sobre ellos. Después se explica la decisión tomada a la hora de dividir los datos. El siguiente apartado de este capítulo resume los pasos seguidos en el proceso y a continuación los pasos empleados en el preprocesamiento de los datos antes de pasarlos al modelo. Una vez descrito todo lo relacionado con los datos, se explica el diseño escogido para el modelo de aprendizaje automático y las técnicas implementadas sobre este para mejorar los resultados. El capítulo termina con un compendio de las herramientas empleadas.
4. En el cuarto capítulo se evalúan los resultados obtenidos y se explican las métricas usadas para ello.
5. El quinto capítulo se reflexiona sobre los resultados, exponiendo sus implicaciones y comparándolos con los resultados de estudios similares.
6. El capítulo sexto reúne las conclusiones sacadas tras la evaluación de los resultados. También en este capítulo se enumeran diferentes aplicaciones clínicas posibles y las líneas de investigación propuestas por el autor.

Al sexto capítulo le sigue una sección de referencias en la que se recogen todas las referencias empleadas durante la realización de este documento.

Finaliza el documento el apartado de anexo, en el que se comparte el repositorio online que contiene el código y se explican las partes más interesantes de este.

Capítulo 2. Estado del Arte

2.1Aprendizaje Automático En Medicina

2.1.1 Concepto y Breve Historia de la Inteligencia Artificial

En el presente punto se introducirá el concepto del aprendizaje automático relacionándolo con la inteligencia artificial y se explicará su relevancia y uso en medicina. Se empezará desde el amplio concepto de la inteligencia artificial indagando en la historia de esta, tras esto se introducirá el aprendizaje automático y se expondrá la base de varios conceptos relacionados. Para terminar con el aprendizaje automático se explicará cada categoría que incluye y los algoritmos relacionados a cada una de ellas. El punto terminará con una introducción al aprendizaje automático aplicado en medicina.

La inteligencia artificial es la tecnología que simula la inteligencia del ser humano en un ordenador, es un término general, que engloba varias tecnologías distintas. Si nos remitimos a la historia, la primera inteligencia artificial creada fue el programa *Logic Theorist*, que era capaz de solucionar problemas matemáticos [2]. En la Conferencia de Dartmouth del verano de 1956 se acuñó por primera vez el término de Inteligencia Artificial gracias al trabajo de John McCarthy. Esta conferencia duró 2 meses y tuvo 47 asistentes, entre los organizadores también estaba Claude Shannon. En la conferencia se sentaron las bases de la inteligencia artificial y se estableció un grupo de investigadores prestigiosos que trabajaran en desarrollar las primeras inteligencias artificiales basadas en neuronas, imitando el comportamiento del cerebro humano [3].

2.1.2 Aprendizaje Automático

El aprendizaje automático, conocido en inglés como *machine learning*, es una rama de la inteligencia artificial que se basa en el desarrollo de algoritmos capaces de aprender e identificar patrones a partir de volúmenes de datos. Estos algoritmos se adaptan a distintas tareas sin la necesidad de haber sido programados explícitamente para la tarea en cuestión.

El aprendizaje automático se divide en tres categorías distintas: aprendizaje supervisado, aprendizaje no supervisado y aprendizaje reforzado.

El aprendizaje supervisado se basa en entrenar el modelo con datos etiquetados, es decir, la salida que debería causar un dato al pasar por el modelo también se pasa al modelo para que pueda tomarla de referencia en su entrenamiento. El aprendizaje supervisado es útil para clasificar entidades a partir de una foto, entre otras tareas. En esta categoría se pueden encontrar diferentes algoritmos con diferentes funciones y objetivos. Por ejemplo, es en el aprendizaje supervisado donde más se emplean modelos de redes neuronales, basados en capas de neuronas conectadas entre sí como el de la figura 1. Debido a la importancia de las redes neuronales se dedicará el punto 2.1.3 a explicarlas.

Las redes neuronales son ampliamente usadas por su potencia, ya que cuantas más capas tenga el modelo más parámetros tendrá para aprender y más útil será en tareas tan complejas como identificar patrones en imágenes. En el caso de que el algoritmo se entrene mediante convoluciones entre los datos de entrada y filtros se conoce como red neuronal convolucional. Si el modelo cuenta con muchas capas ocultas, o intermedias, entra dentro de la definición de *deep learning*.

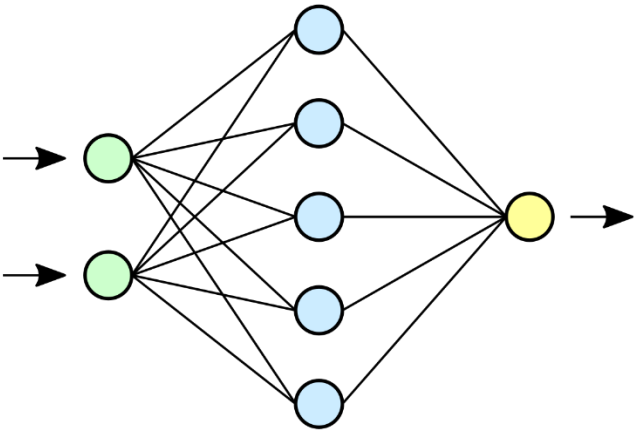

Figura 1. Red neuronal simple [4].

Otro algoritmo ampliamente usado en tareas de aprendizaje automático supervisado es el árbol de decisión, el cual emplea estructuras en forma de árbol para tomar decisiones a partir de los datos. El último algoritmo interesante de aprendizaje supervisado es la máquina de soporte vectorial, esta se usa para clasificar datos en dos clases mediante una frontera de decisión. Para realizar esta tarea se mapean los datos en tres dimensiones y se calcula el hiperplano que mejor separe estos datos [5]. En la figura 2 se muestra un ejemplo.

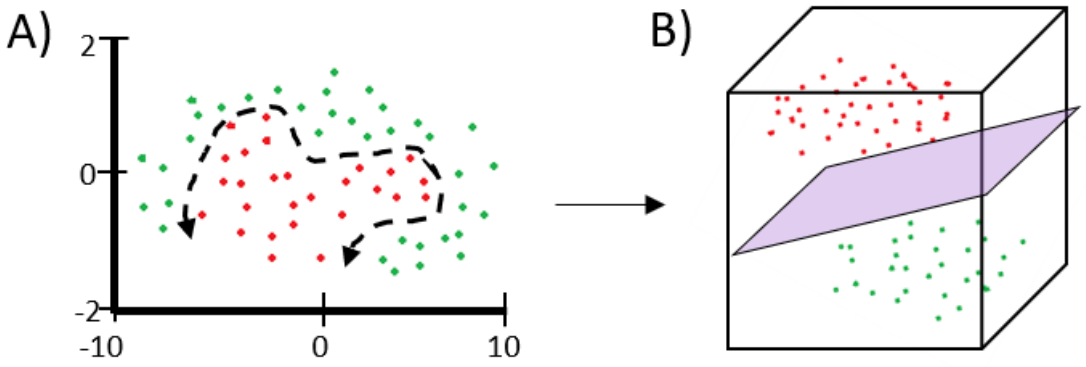

Figura 2. Ejemplo de máquina de soporte vectorial [6].

El aprendizaje no supervisado se refiere a aquellos modelos a los que se pasan los datos sin ninguna etiqueta y es el propio modelo el que busca patrones sin ninguna instrucción previa. Algunos ejemplos de uso son: separación de objetos por colores, ordenamiento de temperaturas para posteriormente clasificarlas por estaciones, étc [7]. Algoritmos de aprendizaje no supervisado muy utilizados incluyen los algoritmos de agrupación, como el algoritmo k-medias usado en análisis de datos para segmentar esos datos en k grupos o los autocodificadores, que son algoritmos basados en redes neuronales que aprenden a codificar y decodificar las muestras

para reducir su tamaño asumiendo mínimas pérdidas de codificación. Estos algoritmos son muy útiles para simplificar el estudio de grandes conjuntos de datos.

La tercera categoría de aprendizaje automático es el aprendizaje reforzado. En esta categoría los modelos actúan sin instrucción, pero recibiendo después de realizar la acción una puntuación indicando como de bien han realizado su tarea, así el modelo se ajusta dependiendo de la puntuación recibida. Es útil en robótica o Internet de las Cosas [8]. En cuanto a sus aplicaciones en la medicina, estos modelos podrían ser usados para diseñar tratamientos personalizados adaptados a cada paciente según la respuesta que den, por ejemplo, podrían ser usados para estimar la cantidad de medicamento necesaria en un paciente donde la puntuación que recibe el modelo se basa en cómo le sienta la medicina a la persona.

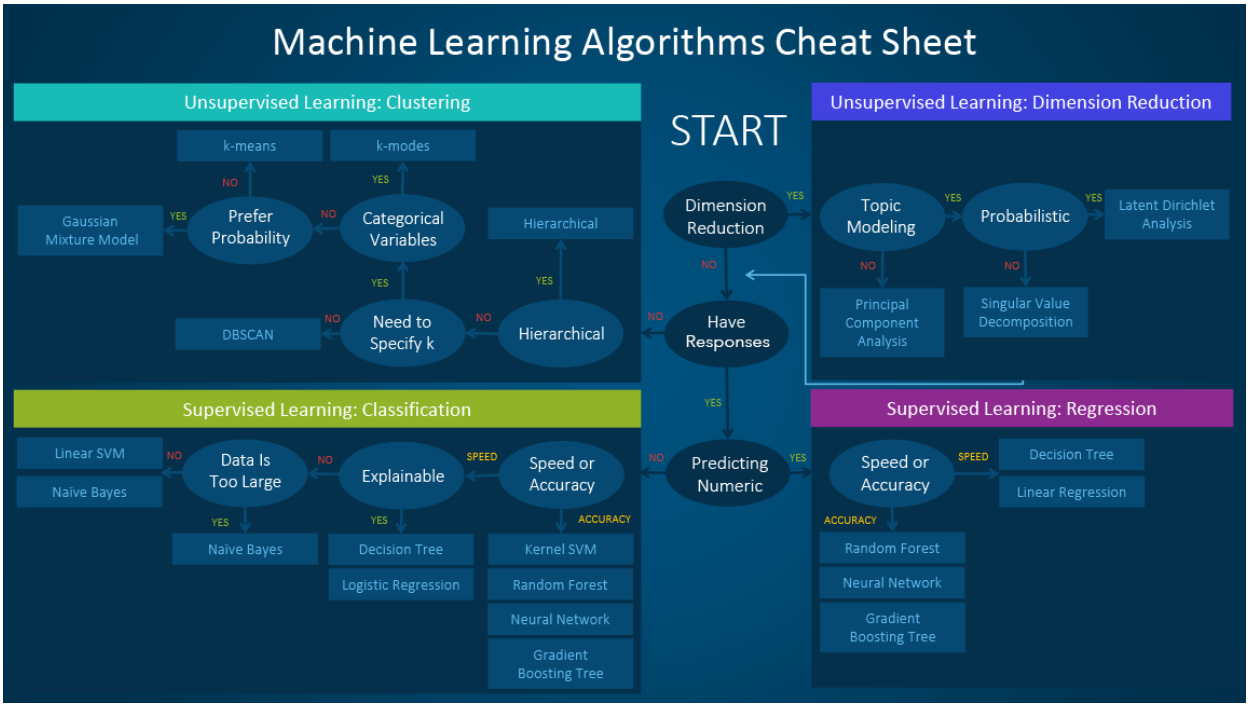

Figura 3. Hoja esquemática sobre los algoritmos de ML [9].

La forma de diseñar un algoritmo de aprendizaje automático no está estandarizada, aunque usualmente sigue un flujo de trabajo lineal.

Inicialmente se identifica el objetivo del algoritmo, detectar objetos, analizar texto, agrupar datos, étc. Una vez se ha identificado el objetivo se tienen que obtener los datos que se usarán en el entrenamiento del modelo. Este paso es muy importante ya que sin datos un modelo es un simple código inservible, además los datos deben ser fiables y, en el caso de ser etiquetados, deben de llevar la etiqueta correcta por lo que deben de haber sido corregidos.

Tras conseguir los datos a emplear el siguiente paso es decidir qué tipo de algoritmo debemos crear, ya que hay numerosos tipos adecuados para cada tipo de problema como podemos ver en la figura 3. Los datos en crudo pueden no estar en el formato óptimo, usualmente las redes neuronales que se emplean para la detección de objetos en imágenes requieren de tensores en vez de imágenes, por lo que se han de convertir las imágenes a tensores antes de pasarlas al modelo.

Una vez se ha decidido que arquitectura de modelo usar y se han obtenido los datos en el formato adecuado, se empieza a programar el modelo. Para esto se cuenta con dos opciones, diseñar un modelo capa a capa o importar un modelo ya creado. Los modelos ya creados tienen la ventaja de que ya han sido estudiados cuidadosamente y no contienen errores, también podemos importar los pesos resultantes de entrenar estos modelos con imágenes estándar en la industria, lo cual se conoce como *transfer learning*.

El siguiente paso es entrenar el modelo, aunque se haya decidido importar los pesos es necesario entrenar al algoritmo con los nuevos datos para que se adapte a estos. Tras el entrenamiento es importante asegurarse de que el modelo que se haya creado sea preciso, para ello se evalúa la salida del modelo con imágenes que hayan sido usadas en la etapa de entrenamiento y se debe estudiar la precisión que resulta. En este paso se deben insertar evaluaciones estadísticas para asegurarse de que el rendimiento del modelo no solo es bueno en los datos de entrenamiento, sino que también se generaliza bien a datos no vistos. Esto implica calcular las métricas de rendimiento adecuadas según la naturaleza del problema y del rendimiento de nuestro modelo.

Dependiendo de las conclusiones a las que se lleguen se decide implementar el modelo creado al problema real con los nuevos pesos generados, o se debe volver atrás, para mejorar y optimizar el modelo y volver a entrenarlo.

2.1.3 Conceptos sobre Redes Neuronales

La neurona es el elemento más básico de una red neuronal, esta es una recreación artificial simple de las neuronas del cerebro humano. La neurona humana, como la de la figura 4, tiene un comportamiento sencillo, esta recibe señales eléctricas a través de las dendritas, las cuales se juntan en el soma y el resultado viaja por el axón hasta el final, donde están conectadas a otras neuronas.

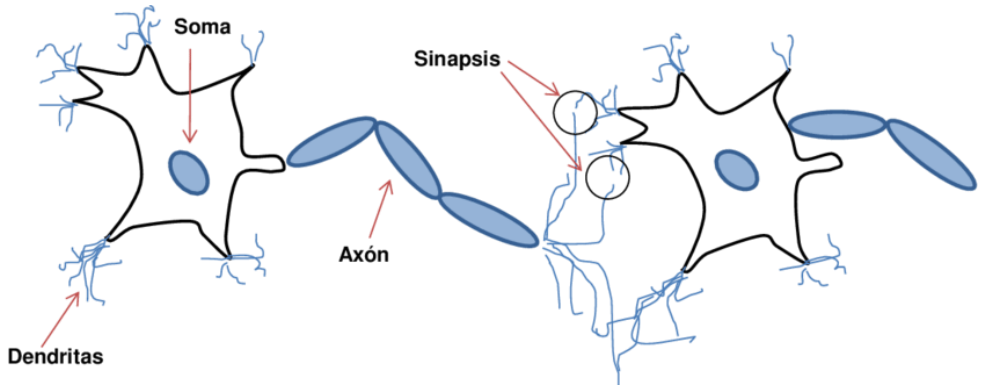

Figura 4. Estructura simple de una neurona [10].

El comportamiento de una neurona humana es el que trata de replicar una neurona artificial como se muestra en la figura 5. La neurona toma los pesos (*w*) que le pasa cada entrada y lo multiplica por los valores de esas entradas, entonces computa la suma entre estos y unos sesgos y pasa la salida a la siguiente conexión. Es posible que en una neurona se aplique una función de activación, las cuales son funciones no lineales que alteran la salida, pero no cambian su comportamiento con el entrenamiento del modelo. Una de las funciones de activación más usadas en tareas de regresión es la sigmoide empleada en la figura 5, la cual normaliza la salida

entre 0 y 1, siendo útil en clasificación binaria o regresión ya que permite normalizar el rango de salida al que se desee. Otras funciones de activación conocidas son la ReLU (*Rectified Linear Unit*), que pone a 0 las salidas negativas y la *Softmax*, la cual cuenta con una salida vectorial que puede ser usada cuando realizamos una clasificación con múltiples clases para dar una probabilidad a cada una de las clases.

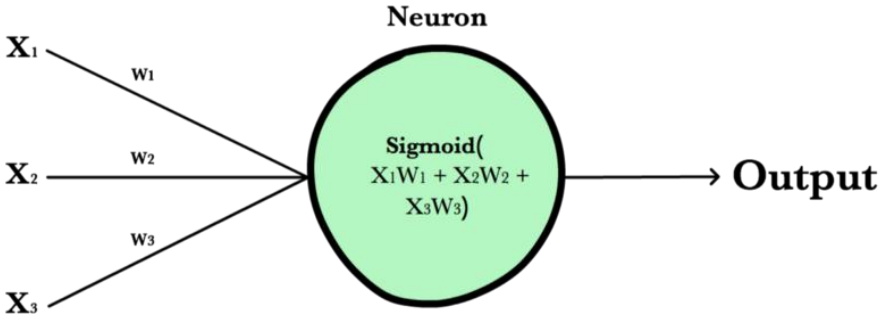

Figura 5. Neurona artificial simple [11].

Las capas con las que cuenta una red neuronal se clasifican en capas de entrada, capas ocultas y capas de salida. Las capas de entrada son aquellas a las que se pasan inicialmente los datos originales, se encargan de transmitir estos datos a las siguientes capas. Las capas ocultas son las capas intermedias y son las encargadas de aprender a interpretar los datos del modelo, por ejemplo, pueden realizar convoluciones entre imágenes de entrada y filtros para detectar la presencia de patrones en una imagen. Las capas ocultas tienen pesos que se modifican para aprender a partir de los datos. La última capa es la capa de salida, esta se encarga de dar el formato adecuado a la salida del modelo, por lo que usualmente contienen funciones de activación acordes a la naturaleza del problema.

Entre las capas más comunes de redes neuronales se encuentran las capas convolucionales, en las cuales se aplican filtros a los datos para detectar patrones. Las capas convolucionales usualmente cuentan con una operación de *pooling*, el cual reduce las dimensiones de los datos eliminando información redundante, lo cual agiliza el proceso y reduce la carga computacional. Otro tipo común de capas son las capas densas, en esta cada neurona de la capa está conectada a todas las neuronas de la capa anterior. Estas capas son útiles para ajustar la dimensión de la salida del modelo, ya que transforman las activaciones de la última capa oculta a un formato interpretable, adecuado para las tareas específicas de clasificación o regresión.

El entrenamiento de una red neuronal consiste en ajustar los pesos y los sesgos de cada neurona en función del error calculado. Este proceso se divide en dos etapas: *forward pass* y *back propagation*. El proceso por el cual los datos pasan de una capa a la siguiente se llama *forward pass*, o paso hacia delante. Durante este proceso se calcula la salida de cada neurona dentro de cada capa y se transmite esta información a la siguiente neurona. El proceso contrario se conoce mediante *back propagation*, durante este se calcula el gradiente de la pérdida en cada neurona y se actualizan los pesos de acorde a este gradiente con el objetivo de minimizar el error. En cada época se pasa al modelo todo el conjunto de datos de entrenamiento dividido en lotes, cuyo tamaño depende de la cantidad de datos disponibles. Los parámetros de las neuronas se actualizan después de cada lote.

2.1.4 Hiperparámetros de los modelos

Al diseñar e implementar modelos de aprendizaje automático podemos encontrarnos con ciertos problemas comunes cuya solución es compartida para diferentes modelos y aplicaciones. Por ejemplo, puede suceder que al entrenar un modelo este aprenda bien los datos de entrenamiento, pero no sea capaz de generalizar para datos no vistos, lo cual se conoce como *overfitting* o sobreajuste, es común si el modelo que empleamos es demasiado complejo para los datos que debe analizar. Por el contrario, puede ocurrir que el modelo no pueda aprender con los datos de entrenamiento disponibles, esto puede pasar si no contamos con un número elevado de datos o estos están mal etiquetados. Este problema se llama *underfitting* o subajuste.

En las siguientes gráficas se puede ver la pérdida de entrenamiento (azul) y validación (naranja) según se iban completando las épocas. En la primera imagen vemos un comportamiento óptimo, la pérdida de validación acompaña a la de entrenamiento. En la segunda imagen, sin embargo, la pérdida de validación deja de descender en épocas tempranas debido al sobreajuste.

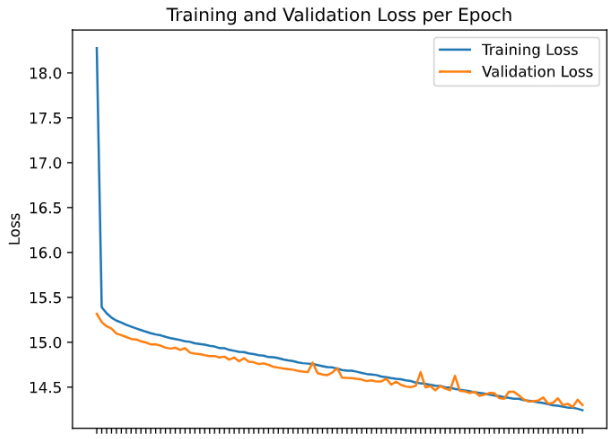

Figura 6. Gráfica del error en caso óptimo.

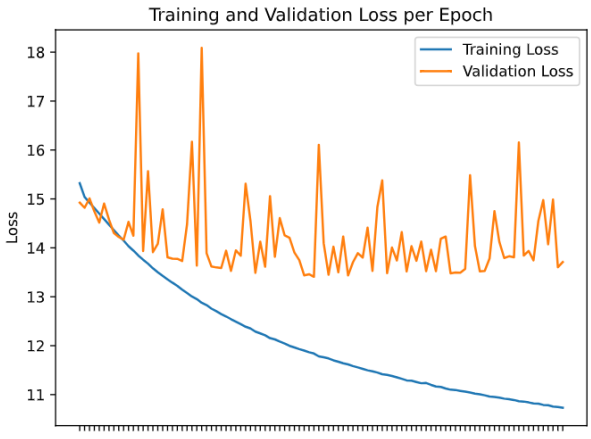

Figura 7. Gráfica del error en caso de sobreajuste.

Además de recopilar los datos adecuados, la mayor dificultad de diseñar e integrar un algoritmo de aprendizaje automático, como una red neuronal, es elegir adecuadamente todos los parámetros que se emplean en el diseño del algoritmo, estos se conocen como hiperparámetros. Este proceso se conoce como ajuste de hiperparámetros o *hyperparameter tunning.* En el caso de que lo hagamos a partir de modelos de *transfer learning* se le conoce como *fine tunning*.

Ciertos hiperparámetros son compartidos en la mayoría de los algoritmos de aprendizaje automático, otros en unos pocos tipos y otros son específicos de ciertos tipos. A continuación se describirán los seis hiperparámetros más comunes entre los algoritmos de aprendizaje automático.

El primer hiperparámetro compartido por la mayoría de los algoritmos es la tasa de aprendizaje, esta se refiere a la tasa de cambio de los pesos en cada iteración del entrenamiento, si la tasa de aprendizaje es alta, se llegará más rápidamente a una pérdida baja, sin embargo, no será capaz de llegar a la menor tasa posible, ya que evitará esta tasa. Si se escoge una tasa menor se llegará a una pérdida mínima, pero el modelo tardará más en converger. Además, se pueden emplear técnicas para que la tasa de aprendizaje varíe según se entrene el modelo, para empezar con una tasa de aprendizaje alta y disminuirla según se llega a un error mínimo.

El segundo hiperparámetro compartido por la mayoría de los algoritmos es el número de épocas, este se define como la cantidad de veces que el modelo se entrena con todos los datos, es decir, en cada época el modelo se entrena con todos los datos y esto sucede de nuevo por cada época escogida. Este es decisivo, ya que con un valor pequeño de este parámetro el modelo no aprenderá lo suficientemente bien, pero con un valor más alto el modelo sobre aprenderá los datos de entrenamiento y se dará *overfitting*.

El tamaño del lote es el tercero de los parámetros compartidos por los algoritmos de aprendizaje automático. En cada época se pasan todos los datos divididos en lotes cuyo tamaño se escoge antes del entrenamiento. Cada vez que se pasa un lote de datos al modelo se denomina iteración, en cada iteración se actualizan los pesos. De esta manera la actualización de los pesos se realiza múltiples veces en una misma época, en vez de una única vez.

El cuarto parámetro empleados en la mayoría de los algoritmos de aprendizaje automático es la función de pérdida o función de error. Esta se define como la función que sigue el modelo para evaluar el error entre las predicciones y la salida esperada y se emplea para modificar los pesos del algoritmo. Dependiendo de la función de error empleada el modelo tendrá más o menos en cuenta ciertos rangos de errores, por ejemplo, una función MSE cuantifica en mayor medida los errores grandes que una función de Huber, haciendo que los datos que disten de la norma del conjunto completo de datos y, por lo tanto, causen un mayor error en la predicción por ser más diferentes del resto, influyan más o menos en la actualización de los pesos. Esta explicación es más fácil de entender si atendemos a la figura 8, dónde se representan la mayoría de las funciones de pérdida comunes.

Analizando la representación gráfica de cada función de pérdida se puede observar que para errores pequeños la función de Huber toma un comportamiento cuadrático, similar al comportamiento de la función MSE, pero si el error es mayor su comportamiento es lineal, influyendo en menor medida errores más grandes. La función MAE tiene un comportamiento estrictamente lineal para todos los posibles errores.

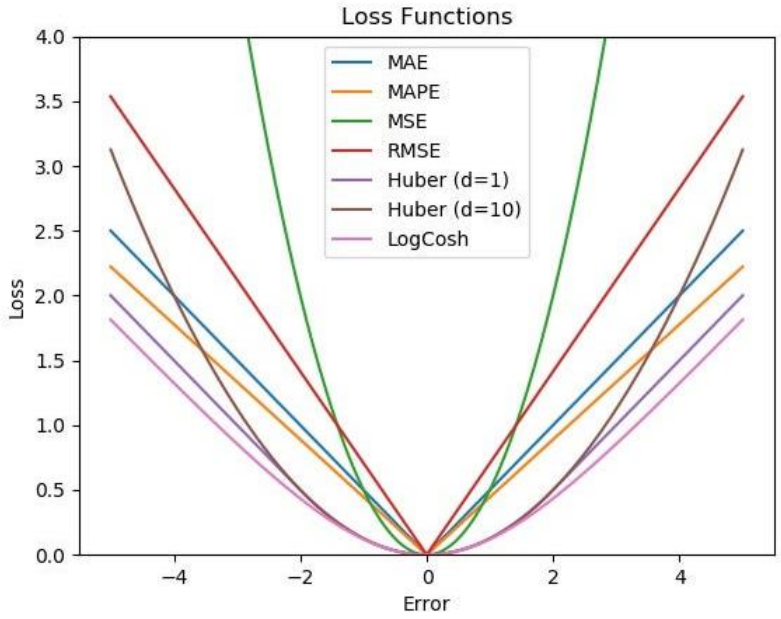

Figura 8. Funciones de pérdida más comunes [12].

El quinto parámetro configurable en la mayoría de los algoritmos de aprendizaje automático es el algoritmo de optimización u optimizador. Este se encarga de ajustar los pesos en cada iteración durante el entrenamiento con el fin de llegar al mínimo de la función de pérdida elegida. Esto lo logra basándose en el error de la iteración actual y empleando la tasa de aprendizaje para decidir la magnitud del valor que añade a los pesos. Hay varios tipos de optimizadores según el algoritmo diseñado, por ejemplo, para algoritmos sencillos de regresión lineal o redes neuronales simples se puede emplear el algoritmo SGD o *Stochastic Gradient Descent*. Este algoritmo divide los lotes de datos en lotes más pequeños y evalúa el gradiente de la función de pérdida para cada sublote, entonces modifica los pesos en función de la dirección negativa al gradiente, ya que el gradiente indica la dirección de mayor incremento. Es un algoritmo simple y rápido. Al dividir aleatoriamente los datos en lotes más pequeños añade un ruido al proceso, ya que se asegura de que los datos que puedan haber sido divididos en lotes siguiendo una relación entre sí queden aleatoriamente divididos. El ruido añadido hace que la pérdida no se quede en un posible mínimo local. Aunque pueda parecer perjudicial que una vez encuentre un mínimo el propio algoritmo salga de este, es positivo ya que su objetivo es llegar al mínimo global.

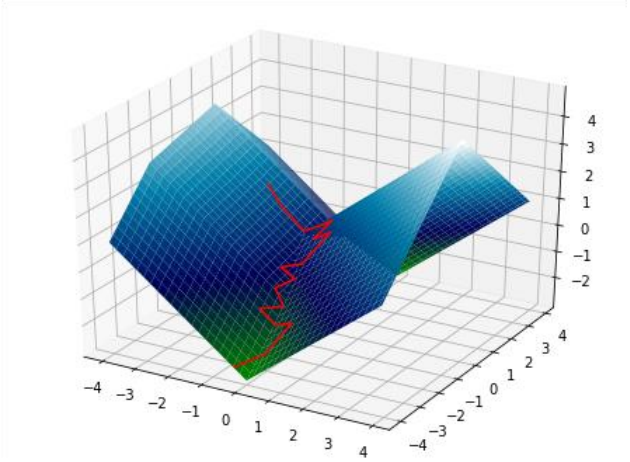

Figura 9. Algoritmo SGD en un punto de silla lineal [13].

Otro algoritmo de optimización muy utilizado, sobre todo en las redes neuronales, es el optimizador Adam. Este algoritmo fue introducido en el artículo de conferencia *Adam: A Method for Stochastic Optimization*, por Diederik P. Kingma, cofundador de la empresa OpenAI, y por Jimmy Lei Ba [14]. Adam, que viene de *Adaptive Moment Estimation*, es un algoritmo más complejo que el SGD y es altamente usado en modelos avanzados con muchos parámetros. Es más preciso que el SGD porque no solo calcula el gradiente de la iteración actual, sino que también tiene en cuenta el gradiente de la iteración anterior en términos conocidos como momentos.

El algoritmo Adam es un algoritmo preciso porque cada parámetro se ajusta individualmente basándose en sus propios gradientes, rápidamente reemplazó a los algoritmos utilizados en su momento ya que superaba a todos en precisión y era computacionalmente eficaz para las bases de datos más probadas [14].

En 2017 surgió un documento que mejoraba directamente el optimizador Adam, creando el nuevo AdamW [15]. Adam añade directamente un factor de decaimiento a los momentos, influyendo directamente en la actualización de los parámetros, lo cual AdamW trata de mejorar.

AdamW funciona exactamente igual que Adam, pero tras calcular la actualización de los pesos elimina los factores de decaimiento que el algoritmo Adam añadía artificialmente.

Cada algoritmo de aprendizaje automático tiene también algoritmos propios que pueden ser configurados, como el *dropout*, el cual permite apagar aleatoriamente el aprendizaje de neuronas en redes neuronales o el número de árboles en su respectivo algoritmo.

El último hiperparámetro común configurable por el programador es la arquitectura del modelo. Dependiendo de la naturaleza del problema y de la cantidad de datos disponibles es más útil emplear una arquitectura u otra. Un modelo con un mayor número de capas profundas puede aprender patrones más complejos que un modelo con un número reducido de capas, sin embargo, será propenso al sobreajuste si se entrena con una cantidad limitada de datos. Lo mismo sucede con el número de neuronas en cada capa. La arquitectura de un modelo también está definida por el tipo de capas que la conforman, como pueden ser las capas densas explicadas anteriormente, o las funciones de activación que las acompañan.

2.1.5 IA en la Medicina

El funcionamiento del aprendizaje automático no es ampliamente conocido en la sociedad, incluso entre profesionales sanitarios. Esto sucede porque los modelos se comportan como cajas negras, dificultando entender el porqué de los resultados. Sin embargo, su interpretabilidad es crucial, el modelo debería ser capaz de indicar al profesional que aspectos han influenciado más en su respuesta. Para lograr explicar la salida del modelo contamos con varias técnicas, siendo la más utilizada los mapas de saliencia. Estos mapas muestran sobre los datos los aspectos que más han influenciado en la decisión del modelo generando una imagen de los datos y resaltando las áreas de mayor influencia a modo de mapa de calor. En la figura 10 se puede ver un ejemplo.

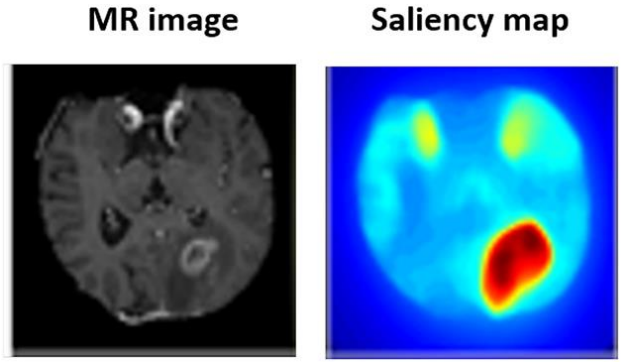

Figura 10. Mapa de saliencia de una imagen por resonancia magnética. Imagen editada, original en [16].

La Inteligencia Artificial puede ser aplicada en todas las etapas del flujo de trabajo médico, ya que se puede emplear en el diagnóstico, en el tratamiento, en la producción de fármacos y otros medicamentos y en la administración, gestión y educación en la materia. En este trabajo se centran los esfuerzos en la aplicación de la IA en la etapa del diagnóstico.

Actualmente existen varios proyectos alrededor del mundo que emplean soluciones basadas en aprendizaje automático a nivel profesional. El super ordenador Watson de IBM sustenta

aplicaciones de aprendizaje automático, en hospitales de todo el mundo se usan sus capacidades en aplicaciones de oncología. Este super ordenador da indicaciones a los oncólogos basándose en la salud en tiempo real del paciente. Según el estudio [17] en un 72.8% de los casos analizados, Watson daba instrucciones adecuadas, motivo por el cual su uso es positivo para los hospitales aunque no cuenta con la precisión suficiente para reemplazar a los oncólogos.

Streams, la aplicación desarrollada por DeepMind en colaboración con Google, permite a los doctores y enfermeros recibir alertas automáticas de cuando los pacientes estén en riesgo de tener problemas de infección de riñón, analiza los resultados de todas las pruebas que el paciente reciba y manda una alerta directa al móvil de los clínicos, evitando que el tiempo que puede pasar desde que el paciente pide consulta hasta que un profesional analiza los resultados. Esta aplicación ha sido usada en el Servicio Nacional de Salud del Reino Unido, aunque también ha sido motivo de controversia en cuanto a la protección de los datos de los pacientes [18]. Este caso resalta la complejidad de la problemática de la protección de datos y la privacidad, ya que cumplir con la normativa vigente puede ralentizar el desarrollo y la implementación de la tecnología. No obstante, el cumplimiento de la regulación es esencial para garantizar la seguridad de las personas, por lo que el aprendizaje automático debe mejorar en cuanto a precisión y eficiencia a la par que adherirse a los estándares que fijan los organismos reguladores.

Otras aplicaciones del aprendizaje automático en ámbitos profesionales incluyen algoritmos de detección de enfermedades coronarias, osteoporosis o la esteatosis hepática [19] y es de esperar que con el paso de los años y el avance de la investigación se desarrollen aplicaciones aún más precisas que cumplan con la normativa correspondiente.

Respecto a algoritmos que se empleen para predecir la edad cardiaca a partir de electrocardiogramas se pueden encontrar muchas arquitecturas distintas con distintos resultados. Entre los algoritmos más utilizados en los últimos años se encuentran las redes neuronales, estas se pueden entrenar a partir de los datos de las señales de los electrocardiogramas directamente, o a partir de la imagen de las señales creadas mediante el espectrograma. En la actualidad, sin embargo, la mayoría de estos algoritmos se emplean para clasificar los ECG en función de la enfermedad de la persona a la que se la haya tomado la prueba, sin embargo, es posible emplear estas arquitecturas para la tarea de estimación de edad cardiaca.

En el caso de usar las señales directamente, podemos encontrar arquitecturas de redes neuronales convolucionales 1D, las cuales obtienen muy buenos resultados [20, 21, 22]. Estas redes son ideales para detectar patrones a lo largo de la señal, al igual que puntos de interés como máximos o ceros. Por esto son muy útiles para capturar dependencias temporales, además de ser más simples que otras redes que empleen datos de formato más avanzado.

También podemos encontrar arquitecturas de redes neuronales convolucionales, las cuales son más complejas y computacionalmente complejas, pero aprovechan otras características y son capaces de detectar regiones locales en una imagen que puedan ser de utilidad para el modelo. Es cierto que en cuanto a los espectrogramas ofrecen peores resultados, sin embargo, estas redes se usan en otros ámbitos de la medicina en los que no se pueden obtener señales o directamente ofrece mayor precisión el uso de imágenes.

El avance de la tecnología hace que se puedan realizar electrocardiogramas desde dispositivos portátiles como relojes inteligentes, por lo que si somos capaces de insertar estas arquitecturas dentro de los dispositivos se lograrían hazañas tales como que una persona pudiera conocer si padece de ciertas enfermedades cardiacas en cualquier lugar y momento, o del estado general de su sistema cardiaco según la edad vascular. Sin embargo, para implementar estos modelos a nivel hospitalario se deben seguir las normas correspondientes y debes ser aprobados por los organismos reguladores, lo cual es un proceso largo y estricto en el que es tan importante la eficacia del modelo como garantizar la seguridad de los datos y la privacidad de las personas.

Los algoritmos de aprendizaje automático en medicina deben ser extremadamente precisos debido a su impacto directo en la salud, lo que limita su adopción profesional a unos pocos proyectos que cumplen con los altos estándares requeridos. Incluso estos algoritmos necesitan supervisión experta para garantizar que sus resultados sean óptimos y seguros para los pacientes.

2.2 Electrocardiograma

El electrocardiograma, también conocido como ECG, es una prueba médica en la que se registra la actividad eléctrica que se genera en el corazón y pasa a través de este, mediante una serie de sensores llamados electrodos que se adhieren a ciertos puntos del cuerpo humano, es una prueba rápida, indolora y económica [23]. Esta prueba permite detectar posibles problemas cardíacos, como arritmias o signos de infarto, que se manifiestan como alteraciones en el trazado del ECG.

Los electrocardiogramas no solo se emplean en ambientes médicos, gracias a su facilidad y disponibilidad hay numerosos equipos en el mercado que permiten a cualquier persona realizarse un ECG en casa con un nivel alto de fiabilidad. Además, con el paso de los años y el avance de la tecnología surgen aún más equipos que permiten la realización de un ECG incluso a pie de calle, como son los dispositivos Holter, los cuales registran el electrocardiograma del paciente durante 24 horas.

Las máquinas que registran electrocardiogramas a nivel clínico se llaman electrocardiógrafos, estas actúan de unidad central con los electrodos y cuentan con componentes electrónicos de muy alta precisión como amplificadores de bajo ruido y filtros precisos. El funcionamiento de estas máquinas es sencillo, las células del cuerpo humano contienen concentraciones de iones dentro y fuera de las membranas celulares, lo cual crea una diferencia de potencial. Cuando estas células se polarizan y despolarizan esta diferencia varía. Los electrodos captan este incremento de la diferencia de potencial de los músculos del corazón y lo registran en forma de señal eléctrica. Finalmente, estas señales son tratadas en la unidad central de procesamiento, la cual se encarga de filtrar, amplificar y digitalizar las señales [24].

Según la conexión que se registre entre los diferentes electrodos se habla de una u otra derivación, habiendo un total de 12 derivaciones posibles empleando 10 electrodos. En ambientes de investigación se ha llegado a estudiar la realización de 120 derivaciones, sin embargo esto no es habitual ya que llevarlo a un ambiente clínico sería extremadamente complicado [25]. En la figura 11 se puede observar con un círculo la colocación de cada uno de los 10 sensores en el cuerpo humano y con letras se muestra la derivación calculada. Aunque hay varias formas de colocar los electrodos esta es la más común.

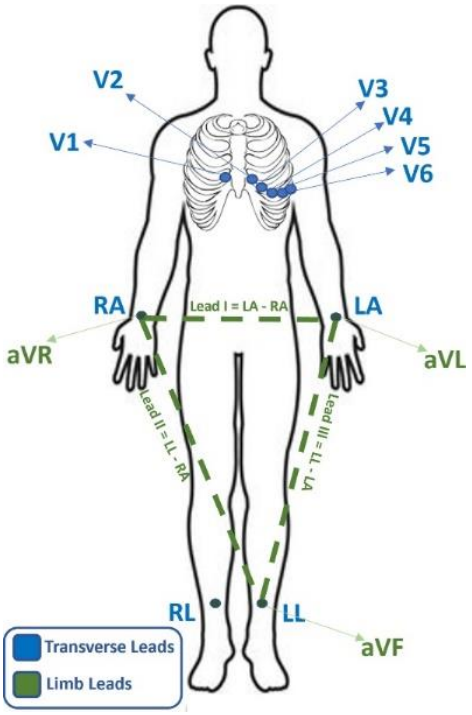

Figura 11. Esquema de posicionamiento de electrodos [26]

En la imagen podemos observar 12 derivaciones, DI, DII, DIII, aVR, aVL, aVF, V1, V2, V3, V4, V5, V6. Las derivaciones de las extremidades DI, DII y DIII fueron descritas por Einthoven en 1901, mientras que aVR, aVL y aVF se obtienen de un cálculo matemático entre los voltajes de los puntos medidos, de ahí que su nomenclatura empiece por “a”, que viene de derivaciones aumentadas.

En hospitales es común la realización del electrocardiograma de 12 derivaciones, ya que ofrece datos más precisos que difícilmente se podrían obtener con otras derivaciones. En dispositivos portátiles es habitual que los ECG se obtengan de una única derivación y sin la necesidad de adherir electrodos al cuerpo, sin embargo, estos ofrecen únicamente la información suficiente como para detectar ciertos problemas cardiacos, limitando su uso a un mero seguimiento diario por parte de la persona con la posibilidad de detectar ligeras inconsistencias como las arritmias [27].

Las señales eléctricas registradas por los electrocardiógrafos tienen una forma común y fácilmente reconocible, en el caso de que la forma de la señal registrada diste de esta se puede detectar que la persona puede tener un problema en su sistema cardiaco. En la señal se identifican varios puntos e intervalos de interés. Entre los puntos más destacados se encuentra la onda P, que refleja la despolarización auricular, el complejo de ondas QRS, formado por los picos que representan la despolarización de los ventrículos, y la onda T, asociada con la repolarización ventricular. En algunas ocasiones también puede observarse la onda U, cuyo motivo de aparición no está claro entre los expertos. Entre los intervalos relevantes para el diagnóstico de enfermedades se encuentran el intervalo PR, el intervalo RR, o el segmento ST, los cuales son claves para evaluar el rendimiento del sistema cardiaco. Todos estos tienen un rango de tiempos estandarizado en el que se puede declarar al paciente como sano. Todo esto, junto más información, se puede ver en la figura 12.

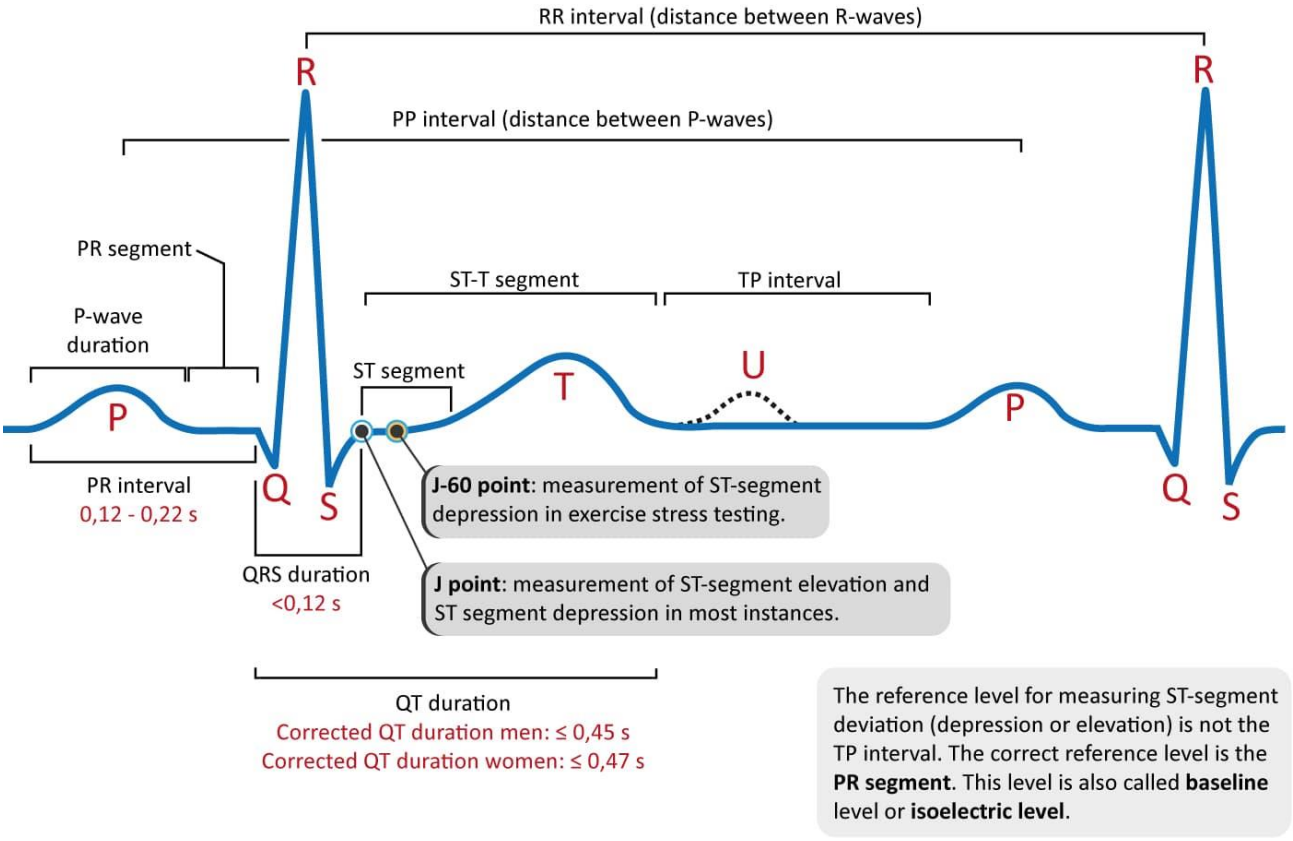

Figura 12. Forma de onda típica y puntos importantes [28]

Para entender cómo surge cada onda de la señal hay que conocer cómo funciona el ciclo cardiaco, es decir, los movimientos que realiza el corazón en cada latido. Este se divide en sístole, o contracción, y diástole, o expansión. El ciclo empieza cuando se genera un impulso eléctrico en el nodo sinoatrial (ver figura 13), este se propaga por las paredes de las aurículas y genera su contracción antes de pasar a los ventrículos. La despolarización causada en las aurículas es la que genera la onda P. De las paredes de las aurículas la señal pasa al nodo atrio ventricular, donde se mantiene hasta que los ventrículos se llenan de sangre. El complejo QRS se genera por la despolarización de los ventrículos, una vez que el impulso eléctrico recorre sus paredes pasa al tabique central del corazón llamado haz de His y de este hacia cada lado del corazón por las ramas derecha e izquierda. Finalmente, la señal eléctrica sube por las fibras de Purkinje repolarizando los ventrículos, lo que causa la onda T. [29]. El origen de la onda U no es completamente conocido, pero se teoriza que ocurre por la repolarización tardía de las fibras de Purkinje [30]. Por otro lado, la amplitud de las ondas depende en mayor medida de la cantidad de masa muscular del órgano y de la orientación de este.

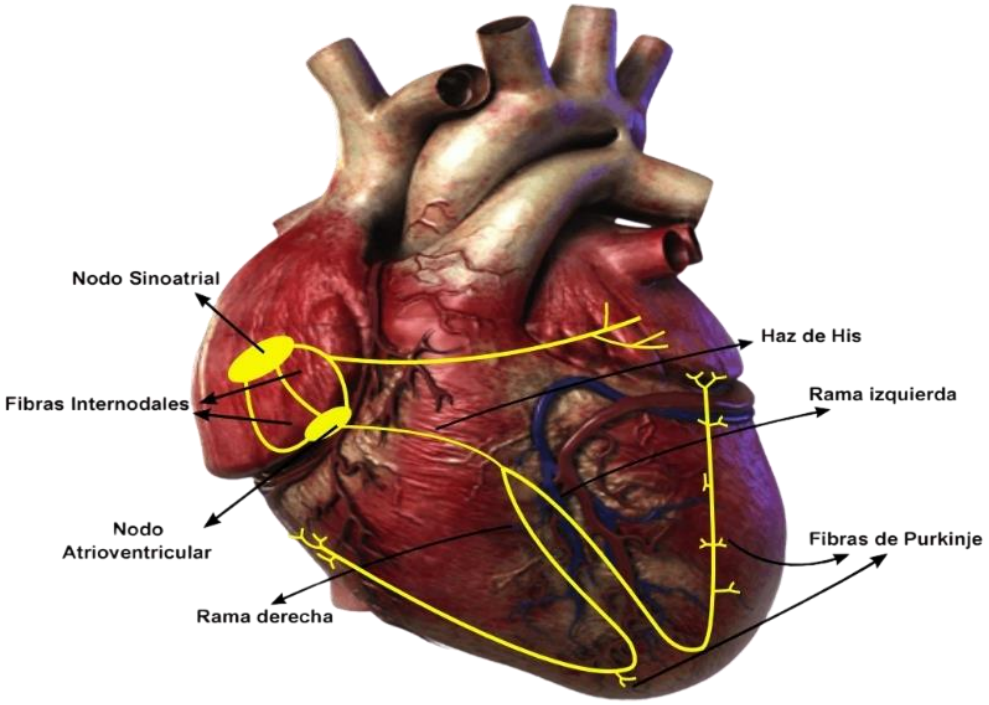

Figura 13. Partes del corazón. Imagen editada, original en [31]

En casos de personas con problemas cardiacos pueden aparecer otras ondas como la onda G o la onda Delta [32].

El ECG es una herramienta fundamental para la detección de irregularidades en el funcionamiento del sistema cardiaco de cualquier persona. El electrocardiograma se ha vuelto fundamental en hospitales de todo el mundo por su accesibilidad, bajo coste y mínima invasión, se emplea para diagnosticar ritmos cardiacos anormales y para prevenir afecciones tan importantes como los infartos.

2.3 Edades Biológicas

La edad cronológica es el tiempo que ha transcurrido desde que una persona nació, este término es universalmente conocido y ha sido usado para evaluar distintos aspectos de la vida de una persona, incluyendo la salud. En general cuanto mayor es la edad cronológica de una persona, peor es su salud. Esto es parcialmente correcto ya que el paso del tiempo contribuye a la aparición de ciertas enfermedades, como la hipertensión, el alzhéimer y el párkinson. Sin embargo, la salud y la edad cronológica no se correlacionan al completo, la salud de una persona depende también de otros factores como sus hábitos alimenticios, su estilo de vida y su entorno; Hábitos como el tabaquismo, el consumo alto de alcohol y el sedentarismo son causas comunes de los problemas de salud.

Los profesionales sanitarios que estudian el envejecimiento, conocidos como gerontólogos, comenzaron a usar el término de edad biológica a finales del siglo XX cuando estudiaron cómo diferentes factores biológicos afectan a nuestra salud. La edad biológica se refiere al estudio del nivel de salud de una persona, medida en años, empleando diferentes biomarcadores (por ejemplo, la estructura de proteínas que tienen función en diferentes procesos a nivel celular o la

composición de nuestra sangre). La edad biológica es una medida útil para conocer el estado de salud de una persona y es mucho más precisa que la edad cronológica [1].

En el 2013 Steve Horvath, profesor de genética humana y de bioestadística en la UCLA, publicó el primer reloj epigenético, una prueba capaz de estimar la edad biológica de diferentes partes del cuerpo humano [33]. Sin embargo, el primer reloj epigenético se realizaba a partir únicamente de muestras de la persona, en los últimos años se han incluido variables ambientales como el tabaquismo en la ecuación, lo que ha permitido tener una estimación más precisa de la edad biológica de una persona [34].

El cuerpo humano cuenta con once sistemas de órganos diferentes, por lo que hablar de una edad biológica en general puede ser impreciso, que una persona tenga una edad biológica favorable en conjunto no quiere decir que cada uno de sus órganos o sistemas esté en condiciones favorables. Por este motivo puede ser conveniente hablar de la edad de cada sistema. En el artículo *Heterogeneous aging across multiple organ systems and prediction of chronic disease and mortality* [35] concluyen que la edad de un sistema afecta al desarrollo del resto de sistemas.

La edad biológica se puede calcular a través de diferentes enfoques. Por ejemplo, se puede estudiar a partir de la longitud de los telomeros, que es la parte en el extremo de los cromosomas que se acorta con la edad celular, por lo que puede ser un indicativo de envejecimiento celular [36]. Otro aspecto estudiado para calcular la edad biológica es la edad epigenética, basada en un modelo matemático capaz de predecir la edad midiendo el nivel de metilación del ADN [37]. También se ha estudiado la edad cerebral, la cual se calcula mediante imágenes cerebrales en las que se exploran cambios en la estructura cerebral causados por la edad o derivados de otros problemas [38].

2.4 Edad Cardiaca

La edad cardiaca es la edad biológica del sistema cardiovascular, esto es el corazón y los conductos que trasportan la sangre alrededor del cuerpo humano.

Estimar la edad cardiaca puede parecer sencillo, basta una simple búsqueda en Internet para encontrar calculadoras de edad cardiaca, sin embargo y como es de esperar, estas calculadoras dan un valor poco fiable ya que no tienen en cuenta aspectos tan importantes como los genes o el estilo de vida. Por ello es importante emplear métodos que estimen la edad cardiaca de la persona empleando resultados de pruebas que se les haya tomado personalmente, como es el caso del electrocardiograma.

Aunque no está clara si la definición de edad cardiaca abarca en su conjunto a la edad predicha mediante datos extraídos de los electrocardiogramas [39], ya que no es un tema que se haya discutido profesionalmente. En varios artículos y en el presente proyecto se emplearán ambas terminologías indistintamente.

Además del electrocardiograma, se emplean otras pruebas como la ecocardiografía, prueba en la que se emplean ultrasonidos para extraer fotografías de cada parte del corazón. También se utiliza la resonancia magnética cardíaca, prueba mediante la cual se extrae información sobre el tejido cardiaco. Además de técnicas de imagen se emplean otro tipo de pruebas como la

medición de los niveles de péptidos natriuréticos, empleados para medir el estrés en el corazón y relacionarlo con la aparición de ataques cardíacos [40].

En los últimos años han surgido numerosos artículos exponiendo distintas formas de calcular la edad cardiaca a partir del ECG, por ejemplo, en [41] realizan un modelo de regresión lineal a partir de ciertos parámetros del electrocardiograma. Además, estos estudios dan cuenta de que en personas con posibles problemas cardiacos su modelo predecía una edad aún más distante de la edad cronológica. En [42] emplean un modelo estadístico bayesiano tanto a personas comunes como a atletas y concluyen que los atletas profesionales tienen una edad predicha mayor a su edad real, indicando que someter al cuerpo a esfuerzos físicos enormes es perjudicial para la salud. Más recientemente se han implementados modelos de aprendizaje automático para estimar la edad cardiaca, los cuales son métodos más rápidos y precisos que los tradicionales.

2.4.1 Estimación de edad a partir del ECG

En este punto veremos los resultados obtenidos por diferentes investigadores en la tarea de estimar la edad cardiaca a partir de datos de electrocardiogramas.

Desde finales del siglo XX han aparecido numerosos artículos diseñando modelos para estimar la aparición de enfermedades cardiacas a partir de datos de electrocardiograma. Inicialmente se exploraron técnicas como la regresión logística a partir de parámetros extraídos de los ECG o el uso de árboles de decisión. En el artículo *Can functional cardiac age be predicted from the ECG in a normal healthy population?* [41] los autores obtienen diferentes parámetros del electrocardiograma, como la longitud del intervalo RR y del segmento ST y los relacionan con otros parámetros clínicos como el índice de masa corporal para hacer una estimación de la edad de una persona sana. En *Predicting "heart age" using electrocardiography* [42] los autores desarrollan un modelo estadístico bayesiano y lo aplican en sujetos sanos de 20 años. Los autores de este último *paper* aplican su modelo a atletas y descubren que los atletas avanzados tienen una edad predicha mayor a su edad real, lo cual indica que el deporte en exceso causó en estos atletas la aparición de condiciones cardíacas desfavorables para la salud.

Con el pasar de los años los investigadores abandonaron las técnicas tradicionales de estimación y centraron sus esfuerzos en aplicar modelos de aprendizaje automático. Según han pasado los años, los avances en esta tarea han ido acompañando a los avances en el aprendizaje automático, ya que se han ido creando modelos y técnicas más precisas y óptimas. En el 2009 se publicó un artículo bajo el nombre *Added Value of a Resting ECG Neural Network That Predicts Cardiovascular Mortality*, en el que se exploraba el uso de redes neuronales artificiales para la predicción de enfermedades cardiovasculares [43].

Son varios los artículos que hacen uso de redes convolucionales 1D para tratar de estimar la edad a partir de las señales de los ECG. Estas redes son simples y no requieren una carga computacional elevada. En *Age and Sex Estimation Using Artificial Intelligence From Standard 12-Lead ECGs* [44] Attia, et al. estudiaron implementar una red convolucional para estimar la edad y el género de la persona. Lograron un MAE de 6.9 años y un R^2 igual a 0.7 para un conjunto de datos de entrenamiento con edades comprendidas alrededor de los 58.6 años con una desviación estándar de 16.2 años. La red del artículo anterior ha sido usada por varios investigadores para sus propios proyectos, en *The 12-lead electrocardiogram as a biomarker of*

biological age [45] emplean esta misma red con una base de datos diferente obteniendo un MAE de 7.4 años, la base de datos tiene una distribución de edades similar a la usada para entrenar el modelo en el artículo original.
El artículo *Deep learning for ECG analysis: Benchmarks and insights from PTB-XL* [46] expone como las redes neuronales convolucionales son las elegidas por otros autores para la misma tarea. En esta publicación emplean los electrocardiogramas de 12 derivaciones, 10 segundos de duración y una frecuencia de muestreo de 500 Hz de la base de datos pública PTB-XL para lograr un MAE de 6.86 años en gente sana y 7.38 años en gente enferma, mostrando como estos modelos pueden ser usados para detectar problemas en el sistema cardiaco. En *Deep learning-derived cardiovascular age shares a genetic basis with other cardiac phenotypes* [21] los autores logran un menor MAE mediante su propia red convolucional 1D aplicada sobre una base de datos perteneciente al *UK Biobank*, sin embargo, el coeficiente de Pearson es de tan solo 0.53 lo cual no es favorable.

Mientras que las redes convolucionales 1D usadas en los anteriores *papers* provienen de diseños propios, existen numerosas redes ya estudiadas y diseñadas que pueden ser usadas para esta tarea. El tipo de red más usado para la estimación de edad a partir de los datos crudos de los electrocardiogramas es la red residual, ResNet. En *Automatic diagnosis of the 12-lead ECG using a deep neural network* [47] Lima, et al. crearon una red ResNet y la entrenaron con datos de la base de datos CODE, su objetivo era detectar enfermedad cardiovasculares. Un año más tarde los mismos autores publicaron el artículo *Deep neural network-estimated electrocardiographic age as a mortality predictor* [48], en el que mostraban como modificaron la ResNet para la tarea de predicción de edad, obteniendo un MAE de 8.38 años y un coeficiente de Pearson de 0.84 en la misma base de datos.

En 2024 todavía varios autores han seguido usando redes residuales para estimar la edad electrocardiográfica. En *Artificial intelligence estimated electrocardiographic age as a recurrence predictor after atrial fibrillation catheter ablation* [49] emplearon la red descrita por Lima y su equipo y la validaron con la base de datos del *UK Biobank* obteniendo un MAE de 8.79 años y un coeficiente de Pearson igual a 0.6. Antes de pasar los datos al modelo los preprocesaron, de manera que cumplían con las características de los datos que empleó Lima para entrenar su modelo original. Este resultado demuestra que un modelo es robusto y generalizable siempre que se realice el preprocesamiento adecuado, aunque varíe la base de datos empleada.

Las redes mencionadas hasta ahora son redes genéricas, cuyo propósito inicial no era el análisis de las señales extraídas de electrocardiogramas, sin embargo, existen también redes creadas específicamente para esta tarea. En *A Deep-Learning Algorithm (ECG12Net) for Detecting Hypokalemia and Hyperkalemia by Electrocardiography: Algorithm Development* [50] se describe una red especialmente creada para este propósito, la ECG12Net. En *Electrocardiogram-based heart age estimation by a deep learning model provides more information on the incidence of cardiovascular disorders* [51] los autores logran una precisión con un MAE de 6.9 años respecto a la edad cronológica de los pacientes aplicando la red ECG12Net a su propia base de datos. Recientemente se ha publicado el estudio *Enhancing ECG-based heart age: impact of acquisition parameters and generalization strategies for varying signal morphologies and corruptions* [52], en el que comparan la red ECG12Net con la red de Attia y la red de Lima aplicándolas sobre las bases de datos PTB-XL y CODE. Finalmente obtienen un resultado más preciso con la red de Attia aplicada sobre la PTB-XL, logrando un MAE de 7.8 años. Aún mejor es el resultado alcanzado por investigadores surcoreanos y publicado en *Artificial intelligence-*

estimated biological heart age using a 12-lead electrocardiogram predicts mortality and cardiovascular outcomes [53]. En este proyecto los investigadores logran un MAE de 6 años sobre una base de datos privada.

Cabe resaltar las conclusiones a las que se llega en el estudio *Can Artificial Intelligence Identify Physiologically "Old" Hearts* [54], en el que estudian la relación entre la disminución de una proteína llamada CD34+ y la edad cardiaca. En este estudio demuestran que la implementación de redes profundas para estimar la edad cardiaca a partir de datos de electrocardiogramas ofrece resultados similares que realizar pruebas invasivas, tardías y costosas en las que se extrae ADN de células cardiacas del paciente y se estudia el tamaño de los telómeros y la cantidad de CD34+ en sus células. Esta conclusión pone de manifiesto la eficacia e importancia del uso de redes neuronales, ya que son más rápidas y no requieren de intervenciones al paciente. La idea es clara, la inclusión de inteligencia artificial en procedimientos médicos sería un avance extraordinario que liberaría a los profesionales sanitarios de horas de trabajo y haría más accesible la medicina.

Capítulo 3. Metodología

3.1Descripción de los datos

3.1.1 Limitaciones en la obtención de datos médicos

Para entrenar un modelo de *deep learning* hacen falta cantidades enormes de datos, esto es miles de datos de entrada. En el caso de modelos supervisados todos estos datos deben estar correctamente etiquetados, además en modelos entrenados con pruebas médicas se requiere que sean los profesionales sanitarios los que obtengan y etiqueten los datos, debido a la fragilidad de los mismos. Por todo ello es imposible para un estudiante lograr crear una base de datos grande correctamente etiquetada, lo cual lleva a buscar bases públicas disponibles en la Internet.

En el caso de señales de ECG correctamente etiquetadas con la edad del paciente es posible obtener dos grandes bases de datos en la Internet, la CODE-15% y la SaMi-Trop.

La CODE-15% es una base de datos pública que contiene el 15% de las señales ECG de una base de datos privada llamada CODE [55]. La CODE-15% cuenta con señales de los 12 canales del electrocardiograma, obtenidas a partir de 345779 pruebas a 233770 pacientes. Esta base de datos fue creada por la Red de Telesalud de Minas Gerais, TNMG por sus siglas en inglés, entre los años 2010 y 2016 en Brasil. El TNMG es un sistema público de telesalud del estado de Minas Gerais [56]. Esta base de datos fue usada por primera vez en el *paper*, *Automatic Diagnosis of the 12-Lead ECG Using a Deep Neural Network* [47], publicado en 2020 por Antônio H Ribeiro, entre otros. Esta base de datos es ideal para resolver el problema principal del proyecto, ya que cuenta con muchas señales etiquetadas con varios campos, como la edad de la persona, su género, su estado de salud y en el caso de que el paciente falleciera, el tiempo que vivió desde que se le realizó la prueba.

La base de datos de SaMi-Trop, que viene de *Sao Paulo-Minas Gerais Tropical Medicine Research Center*, es una base de datos privada. financiada por el *National Institute of Health* de los Estados Unidos de América. Un conjunto de datos obtenidos de la SaMi-Trop puede encontrarse en línea en [57]. Esta base de datos incluye señales de ECG obtenidas de pacientes de la zona de Minas Gerais, en Brasil, junto con anotaciones como la edad de la persona, su género y en el caso de que el paciente falleciera, el tiempo que vivió desde que se le realizó la prueba. La mayor diferencia con la base CODE-15% es que los pacientes de los que se obtuvieron las pruebas padecen de miocardiopatía debida a la enfermedad crónica de Chagas, una enfermedad parasitaria causada por la chinche de chagas que hace que el corazón aumente de tamaño. Entrenar una red a partir de datos de personas enfermas hace que la red aprenda bien a sobre estos datos, pero luego generalice mal sobre datos de personas enfermas.

3.1.2 Características de la CODE-15%

La base de datos CODE-15% cuenta con señales de los 12 canales del electrocardiograma, obtenidas a partir de 345779 pruebas a 233770 pacientes. Esta base de datos cuenta con los

exámenes realizados al 15% de los pacientes de los registrados en la base de datos CODE, la cual es una base de datos de acceso restringido a la cual no se permite acceder públicamente [58].

El repositorio en el que se aloja la base de datos contiene el fichero “exams.csv”, este fichero contiene valores separados por coma con la información ordenada en las siguientes columnas:

- "exam_id": este es el identificador asignado a cada examen.
- "age": esta es la edad del paciente a la hora de realizarla el examen, la edad mínima de los pacientes registrados en esta base de datos es de 17 años, y la edad mínima es de 81 años.
- "is_male": esta es la variable que define el género del paciente, es “1” si el paciente es hombre.
- "nn_predicted_age": esta es la edad predicha por su modelo, descrito en [48].
- "1dAVb": esta variable indica si la persona tiene un bloqueo atrio ventricular de primer grado.
- "RBBB": esta variable indica si la persona tiene un bloque de rama derecha.
- "LBBB": esta variable indica si la persona tiene un bloqueo de rama izquierda.
- "SB": esta variable indica si la persona sufre de bradicardia sinusal.
- "AF": esta variable indica si la persona sufre de fibrilación auricular.
- "ST": esta variable indica si la persona sufre de taquicardia sinusal.
- "patient_id": este es el identificador asignado a cada paciente.
- "normal_ecg": esta variable indica si el paciente tiene un ECG considerado normal.
- "death": esta variable indica si el paciente falleció un tiempo después a la realización del examen.
- "timey": esta variable indica el tiempo que pasó desde que se realizó la prueba al paciente hasta que este falleció, o en su defecto, hasta que se dejó de seguir su caso.
- "trace_file": esta variable identifica en que fichero hdf5 de la base de datos están almacenados los ficheros correspondientes al paciente.

El comienzo del fichero en formato csv puede verse en la figura 14. Este fichero, al igual que el resto de la base de datos, es de acceso público en [56].

```
exam_id,age,is_male,nn_predicted_age,1dAVb,RBBB,LBBB,SB,ST,AF,patient_id,death,timey,normal_ecg,trace_file
1169160,38,True,40.160484,False,False,False,False,False,False,523632,False,2.098628,True,exams_part13.hdf5
2873686,73,True,67.05944,False,False,False,False,False,False,1724173,False,6.657529,False,exams_part13.hdf5
168405,67,True,79.62174,False,False,False,False,False,True,51421,False,4.282188,False,exams_part13.hdf5
271011,41,True,69.75026,False,False,False,False,False,False,1737282,False,4.038353,True,exams_part13.hdf5
384368,73,True,78.87346,False,False,False,False,False,False,331652,False,3.786298,False,exams_part13.hdf5
2950575,61,True,70.905174,False,False,False,False,False,False,17423,,,False,exams_part13.hdf5
1467619,33,False,48.628563,False,False,False,False,False,False,1351337,False,1.498629,False,exams_part13.hdf5
1537328,24,True,29.179337,False,False,False,False,False,False,1519774,False,1.367122,True,exams_part13.hdf5
981735,82,True,82.22461,False,False,False,False,False,False,1192754,False,2.468491,False,exams_part13.hdf5
```

Figura 14. Comienzo del fichero "exams.csv”

Además de este fichero, en la base de datos se encuentra un conjunto de 18 ficheros llamados “exams_part{i}.hdf5” con extensión hdf5, un estándar de almacenamiento ampliamente usado en la investigación por su facilidad para manejar grandes cantidades de datos. Estos ficheros hdf5 contienen dos conjuntos de datos:

- “exam_id”: contiene el id del identificador asignado a la prueba, es el mismo que en el fichero csv.

- “tracings”: una matriz con dimensión (N, 4096, 12) cuyas filas contienen la señal correspondiente a cada canal de un ECG, DI, DII, DIII, AVR, AVL, AVF, V1, V2, V3, V4, V5, V6.

Para facilitar el tratamiento de los datos, ambos conjuntos están ordenados en el mismo patrón.

Las señales que contiene la base de datos están muestreadas a 400 hercios y tienen una duración de 10 segundos.

3.1.3 Estudio poblacional de la CODE-15%

Para entrenar una red neuronal es importante que los datos de entrada sean homogéneos, es decir, todos los datos en la base de datos deben formar una distribución unitaria. En el caso que atañe a este trabajo fin de grado esto se traduce en que el número de casos presentes en la base de datos debe ser el mismo para cada grupo de edades.

Para determinar la distribución de edades dentro de la CODE-15% se ha realizado un programa que grafica la distribución de edades a partir del fichero csv mencionado en el punto anterior. La distribución se muestra en la figura 15.

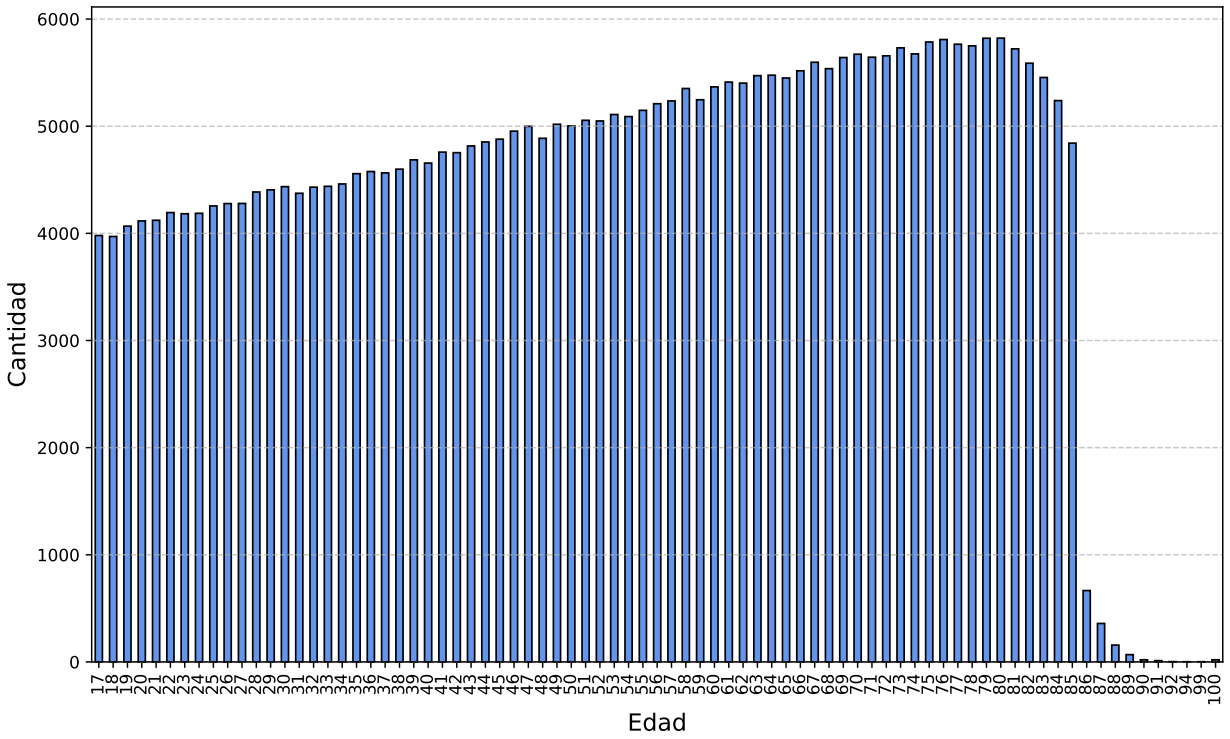

Figura 15. Histograma de las edades de los pacientes en la base de datos CODE-15%

Como se aprecia en la imagen esta distribución no es unitaria, si no que se encuentran más pacientes de años avanzados que pacientes jóvenes, de hecho, la edad media de esta base de datos es de 53.24 años y la mediana de edad es de 54 años. Es importante recordar que en CODE-15% se encuentran los electrocardiogramas realizados tanto a pacientes sanos como a pacientes enfermos, por lo que antes de emplear estos datos para entrenar una red neuronal debemos filtrar únicamente los pacientes etiquetados como sanos, además de los ECG que han

sido marcados como normales por los médicos. De esta forma la distribución resultante es la que se puede ver en la figura 16.

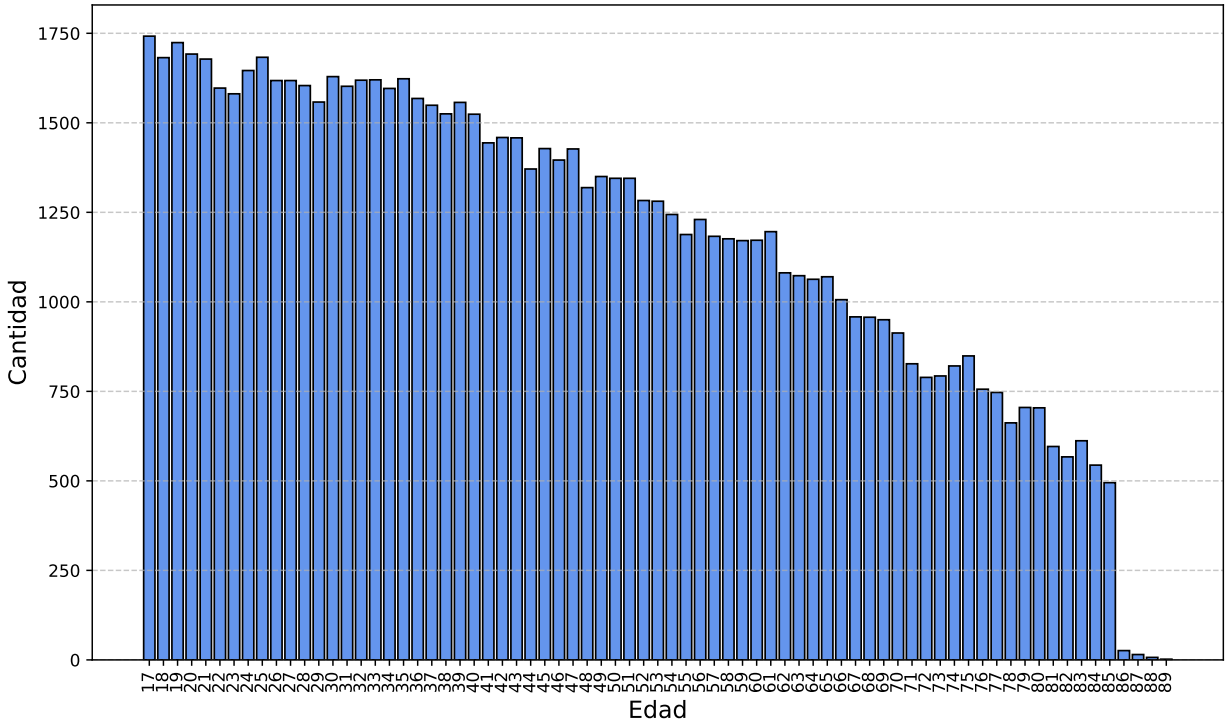

Figura 16. Histograma de las edades de los pacientes sanos en la base de datos CODE-15%

Como se aprecia en la figura, después de filtrar los pacientes sanos la base tiene una mayor cantidad de pacientes jóvenes que de pacientes ancianos, lo cual es de esperar ya que como se comentó en puntos anteriores el avance de la edad cronológica suele traer consigo enfermedades y otros problemas de la salud. Esto puede causar un problema en el entrenamiento de la red ya que puede suceder que la red aprenda demasiado bien a estimar la edad de las personas jóvenes, pero mal en el caso de personas mayores. Otro problema que se puede apreciar estudiando la gráfica es el rango de valores de esta, mientras que los pacientes sanos empleados para entrenar llegan únicamente a los 89 años de edad, en la base de datos CODE-15% hay pacientes con 100 años, por lo que si limitamos la salida de nuestro modelo a los 89 años estamos creando un error mínimo añadido para personas mayores a esta edad, por ejemplo, estamos añadiendo un error de 11 años a la predicción de edad en personas de 100 años.

Otras características de la población de la base de datos toman importancia cuando nos proponemos estudiar cómo afecta la disfunción del sistema cardíaco a cada clase de persona. Es de interés estudiar cómo afectan los problemas cardíacos según el género de la persona, sus hábitos o problemas conocidos en su sistema.

De los 345779 electrocardiogramas registrados en la base de datos, 206576 de ellos pertenecen a mujeres, lo que representa un 59.74% de la población. Además, un 1.65% de la población de la base de datos tiene un bloqueo atrio ventricular de primer grado, un 2.8% tiene un bloque de rama derecha, un 1.74% padece un bloqueo de rama izquierda, un 1.62% sufre de bradicardia sinusal, un 2.19% tiene taquicardia sinusal y un 2.03% sufre de fibrilación auricular. En total un 10.92% de los pacientes sufren de algún problema cardíaco recogido en la base de datos

correspondiente a 37775 pacientes, frente al 89.08% de los pacientes sin estas patologías registradas.

Además, en la base de datos se recogen el número de pacientes que fallecieron en el tiempo de seguimiento posterior a la prueba, como se mencionó anteriormente. Así podemos observar que 8341 pacientes fallecieron en un tiempo posterior, lo que corresponde al 2.41%. Porcentaje aún más alto si únicamente lo ponemos en la perspectiva de los pacientes enfermos, el cual sería del 22.08%.

3.2 División de los datos

A la hora de diseñar una red neuronal no solo es importante conseguir una base de datos fiable y correctamente etiquetada, sino que hay que optimizar correctamente la división de los datos entre los conjuntos de entrenamiento, validación y prueba.

El conjunto de entrenamiento contiene los datos que serán usados para entrenar al modelo. El modelo deberá ser capaz de obtener características de los datos de entrenamiento a partir de instrucciones que se pasan junto a los datos, por ejemplo, en el caso de estimación de la edad a partir de espectrogramas el modelo debe ser capaz de identificar patrones en los espectrogramas relacionados con la edad y comparar estos con la etiqueta de edad pasada junto a las imágenes. Este conjunto debe ser el más grande de todos, ya que cuántos más datos tenga disponible el modelo para aprender, mejor generalizará a todos ellos, evitando memorizar unos pocos. En este conjunto deben residir las imágenes cuya información deba aprender a predecir el modelo, por ejemplo, en el caso de predecir la edad de un paciente, es importante que en este conjunto solo haya casos de pacientes sanos, para evitar que el modelo aprenda información de características no relevante para el problema a resolver.

El conjunto de validación se emplea para evaluar el modelo con imágenes que no ha visto, es decir, que no ha aprendido. De esta forma se obtiene un error válido extrapolable a datos nuevos, como pueden ser los obtenidos tras la aplicación del modelo. El error de entrenamiento, por otra parte, no es un indicador fiel, ya que de poco sirve saber que el modelo aprende a predecir sobre los datos que le han sido pasados ya etiquetados. Al igual que en el conjunto de entrenamiento, en este conjunto deben residir las imágenes cuya información deba aprender a predecir el modelo, por ejemplo, en el caso de predecir la edad de un paciente, es importante que en este conjunto solo haya casos de pacientes sanos, para evitar que el modelo aprenda en función de un error causado por características no relevantes para el problema a resolver, por ejemplo, no interesa que el modelo aprenda a estimar correctamente la edad de personas enfermas ya que el objetivo final es que estas personas vean a través de una predicción elevada, que tienen problemas cardiacos.

Como se explicó en el punto 2.1.4, el entrenamiento de la red se divide en varias épocas en las que se van modificando los parámetros de las neuronas para lograr la mayor precisión posible, es decir, para minimizar la función de pérdida. En cada época se pasa al modelo tanto el conjunto de entrenamiento como el conjunto de validación. En un primer momento el modelo se entrena con el conjunto de entrenamiento al completo, aunque dividido en lotes según el tamaño de lote escogido, tras entrenarse el modelo se evalúa con el conjunto de validación, el cual contiene datos que no han sido entregados todavía al modelo. En función del error generado al evaluar el conjunto de validación se modificarán de una forma u otra los parámetros de la red.

Adicionalmente, es posible modificar hiperparámetros del código en función del error en el conjunto de validación, por ejemplo, es útil modificar la tasa de aprendizaje del modelo para no saltar el mínimo de la función de pérdida según se acercan los resultados.

Por último, está el conjunto de prueba. Este conjunto no se pasa en ningún momento del entrenamiento al modelo, pues se reserva para validar el modelo una vez ya haya terminado de converger. Debe ser, por lo tanto, un conjunto de datos pequeño, igual o menor al conjunto de validación.

La tarea por resolver en cuanto a la división de los datos es elegir el porcentaje de la base de datos que ocupa cada conjunto. Es común elegir un 70% de la base de datos para el conjunto de entrenamiento y un 15% del conjunto de los datos para los conjuntos de validación y prueba.

Además, es muy importante asegurarse de que los datos se distribuyen aleatoriamente entre los diferentes conjuntos para evitar sesgos innecesarios, como que la mayoría de los datos que contienen una información útil para el modelo sean divididos entre los conjuntos de validación y prueba, pero no se encuentren en el conjunto de entrenamiento.

En el caso de pruebas médicas, en el que cada prueba realizada a cada paciente se divida en varias instancias, como puede ser una señal ECG dividida entre los 12 canales que contiene, es importante asegurarse de que las pruebas realizadas a cada paciente no se encuentren en varios conjuntos a la vez, como puede ser que unos canales de un paciente se encuentren en el conjunto de entrenamiento y otros del mismo paciente se encuentren en el conjunto de validación. Si esto pasara estaríamos validando el modelo con datos de los mismos pacientes que se emplean para entrenarlo, por lo que estaríamos obteniendo un error para imágenes con características ya aprendidas por el modelo resultando en un error de validación menor al real.

3.3 Esquema general del procesado

En este apartado se describirá, sin entrar en detalle, el flujo del proceso completo del proyecto, desde la carga y preprocesado de los datos hasta la obtención del resultado final.

Primeramente, se han descargado los datos en formato hdf5 de la página oficial que aloja la base de datos CODE-15% [55]. Tras ello se han preprocesado y se han guardado en el servidor de trabajo siguiendo los pasos que se explican en el punto 3.4. Seguidamente se ha realizado una copia de estos datos en formato .mat y se han trasformado a su representación frecuencial en forma de espectrograma para cada derivación de los electros, como se describe también en el punto 3.4.

Una vez los datos fueron cargados en el servidor, el programa creado carga y preprocesa los datos de cada derivación por separado, aplicando el logaritmo, convirtiendo la imagen a un tensor y normalizando este según la media y la desviación típica. Este paso suaviza variaciones en los espectrogramas, se asegura de que los datos estén en un formato adecuado para el modelo y acelera la convergencia de este. Tras esto el modelo carga estos datos en dos conjuntos, el conjunto de entrenamiento que contiene aproximadamente un 88% de los datos y el conjunto de validación con un 12% de los datos.

A continuación, los datos son pasados por el modelo destinado a cada derivación y son divididos en lotes de tamaño escogido 32. En cada época se realizan dos etapas, primeramente se entrena el modelo con los lotes y después se valida el resultado de ese entrenamiento. Según se va completando cada época, se muestra por pantalla el error resultante y se guarda, con el

objetivo de que en la época en la que el error llegue al mínimo absoluto se guarden los pesos. Estos pesos permiten volver a obtener el error mínimo al pasar de nuevo las imágenes de validación por el modelo.

Cuando se completa la última época se devuelve al modelo el estado de la época en la que obtuvo menor error. Posteriormente se evalúan distintos parámetros estadísticos, como el MAE o el coeficiente de Pearson, para poder evaluar el resultado. Tras la generación de las métricas se generan las imágenes y el fichero con formato csv correspondiente.

Finalmente, se ha empleado un modelo sencillo de regresión lineal en MATLAB. Los resultados obtenidos en cada derivación fueron pasados a este modelo, el cual los juntó y obtuvo un resultado final.

3.4 Preprocesamiento de los datos

En el presente proyecto se han empleado datos obtenidos a partir de las señales ECG de los 12 canales. Estos datos han sido preprocesados en dos formatos distintos.

En la base de datos CODE-15% los datos se encuentran distribuidos en formato hdf5. Primero se han obtenido las señales guardadas en dicho formato y se han preprocesado en MATLAB.

El código empleado para esta tarea realiza los siguientes pasos:

1) Lee los datos del fichero exams.csv y carga sus datos.
2) Lee los datos de los archivos hdf5 y carga tanto las señales como el ID de los exámenes correspondientes.
3) Dentro de cada adquisición recorta la señal a 7 segundos y 2800 muestras y realiza un *detrend*, mediante el cual se nivela la señal sobre un cierto valor medio.

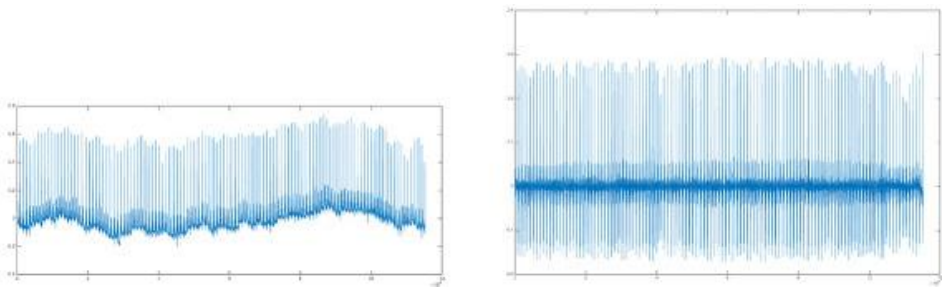

Figura 17. Ejemplo de señal ECG antes y después de aplicar "detrend" [59].

4) Filtra las señales según la información obtenida del fichero csv, de manera que únicamente carga las señales correspondientes a pacientes sanos, útiles para el entrenamiento.
5) Guarda estas señales preprocesadas en matrices con forma 12x2800, correspondientes a 2800 valores de los 12 canales de un ECG, en formato mat en un servidor para poder ser pasadas al modelo.

En las siguientes figuras se puede ver el resultado final de este proceso.

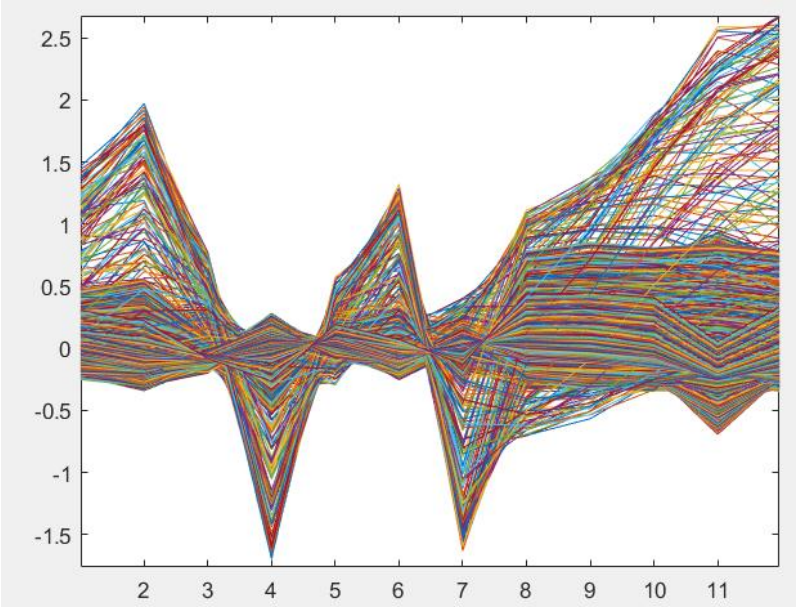

Figura 18. Gráfico de la matriz de los 12 canales de un ECG. Representación temporal completa.

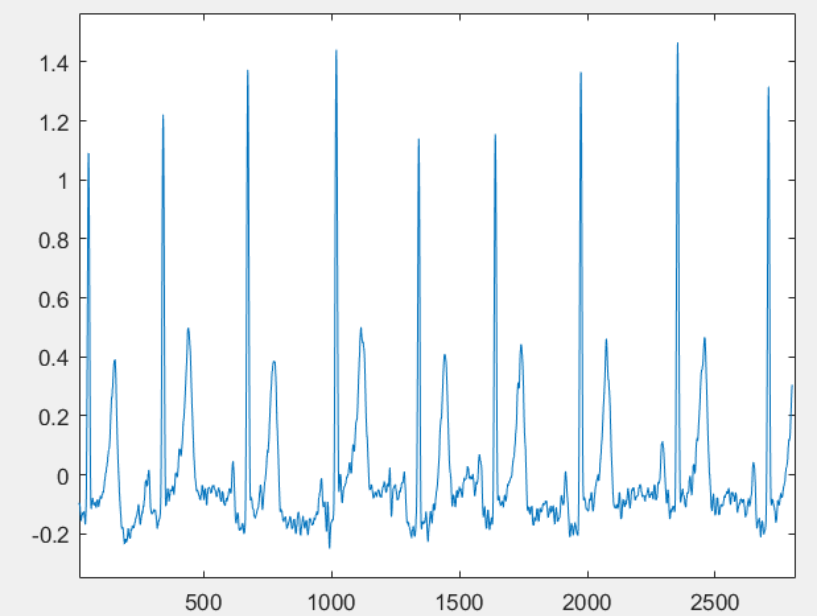

Figura 19. Gráfico del canal 1 de la matriz mostrada en la figura 19.

En este proyecto se han creado modelos que emplean estas señales directamente para entrenar el algoritmo. Pero también se ha explorado la vía de obtener los espectrogramas de estos datos y estudiar cómo afecta este nuevo formato a la precisión del modelo.

El espectrograma es una herramienta fundamental en el análisis de señales, ya que permite ver una gráfica de la frecuencia de la señal a lo largo del tiempo, además la intensidad del color de la gráfica define la amplitud. El espectrograma se obtiene dividiendo la señal en ventanas y haciendo la Trasformada de Fourier de cada ventana, esto se realiza con la Transformada de Fourier de Corto Tiempo (STFT).

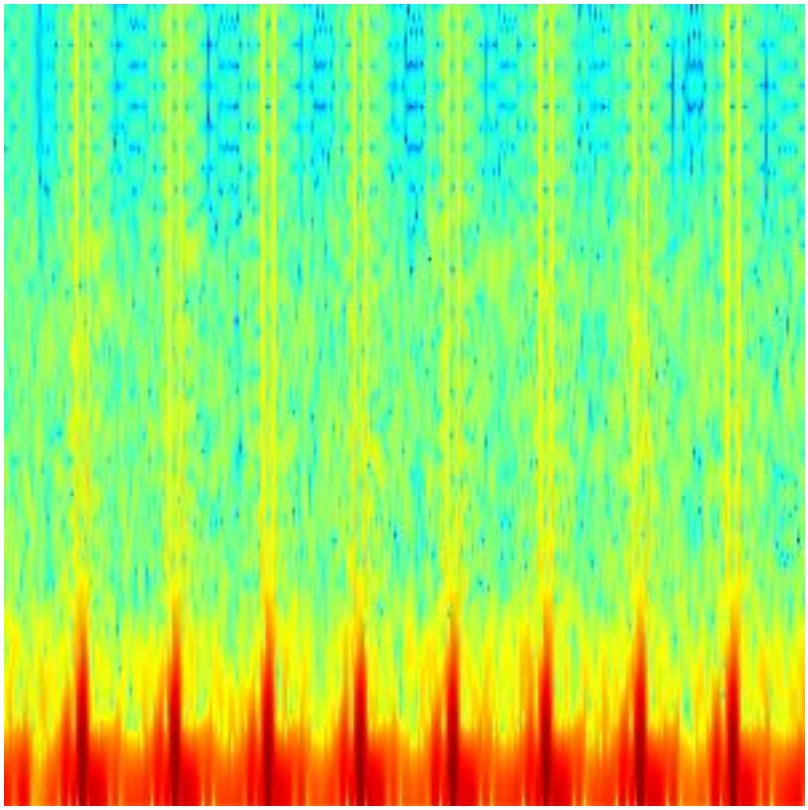

Figura 20. Espectrograma coloreado obtenido de una señal ECG

El uso del espectrograma como herramienta fundamental en el análisis de señales abarca numerosas industrias, como el procesamiento de audio, el análisis de señales de radiofrecuencia, la sismología y la acústica. En el presente trabajo se estudia el caso de uso de los espectrogramas para la mejora de los resultados de redes neuronales. Para ello se ha fijado un tamaño común de las señales de ECG recortándolas para y se ha obtenido el espectrograma en escala de grises de la señal resultante. Tanto el tamaño como el mapa de colores del espectrograma se han escogido así por lograr una mayor eficiencia computacional. Un ejemplo del espectrograma obtenido a partir de una señal de ECG se puede ver en la figura 20.

Para convertir las señales de formato mat en espectrogramas se ha realizado un código en MATLAB que realiza los siguientes pasos:

1) Lee los datos de los ficheros de formato mat que contienen las señales ya preprocesadas.
2) Carga cada canal individualmente para obtener el espectrograma de un único canal.
3) Obtiene el espectrograma empleando una ventana de Hamming de 38 muestras que suaviza las señales con un solapamiento de 26 muestras entre ventanas consecutivas, 447 puntos en la FFT y una frecuencia de muestreo de 400 hercios, ya que esta es la frecuencia de muestreo de los datos en la base de datos CODE-15%.
4) Escala la imagen a un tamaño de 224x224 píxeles y normaliza sus valores entre 0 y 65535, que corresponde a una profundidad de 16 bits en escala de grises.
5) Guarda las imágenes generadas en formato tiff en el servidor. Las imágenes se guardan en diferentes carpetas dependiendo de la derivación empleada.

3.5 Diseño de los modelos escogidos

Durante el desarrollo del proyecto se han probado varios modelos de redes neuronales profundas. El conjunto de redes empleadas abarca tanto redes convolucionales 2D, las cuales emplean imágenes como entrada, como redes convolucionales 1D, las cuales emplean señales unidimensionales como datos de entrada.

Entre las redes 2D probadas se encuentran las redes CNN sencillas, las redes de tipo ResNet18, DenseNet121, DenseNet161, VGG, y un transformer visual.

Estas redes se escogieron por su complejidad, ya que son redes de gran tamaño con millones de parámetros a entrenar capaces de identificar y aprender patrones en imágenes. Entre las redes 2D probadas la red que mejores resultados resultó fue la DenseNet121, seguida de la DenseNet161 y la ResNet18. Mientras que la red ResNet18 es una red residual, las redes DenseNet121 y DenseNet161 son redes compuestas por bloques densos, los cuales se explicará más adelante en este mismo punto. La principal diferente entre ambas redes DenseNet es su tamaño, ya que la DenseNet121 consta de menos parámetros entrenables.

ImageNet es un proyecto que aloja una gran base de datos empleada para comparar el rendimiento de distintos modelos de redes neuronales en una misma tarea, la clasificación de imágenes [60]. Las más de 14 millones de imágenes etiquetadas de *ImageNet* se clasifican en 22000 clases, la tarea a resolver por las redes es clasificar correctamente estas imágenes.

Los resultados de las pruebas son compartidos públicamente y recogidos en páginas como *Papers with Code* [61]. Según la tabla de resultados alojada en esta página, la red DenseNet121 queda en el puesto 448, mientras que la red DenseNet161 queda en el puesto 369 [62]. Esto, sin embargo, no significa que la red DenseNet161 sea superior a la DenseNet121, ya que otros problemas pueden arrojar resultados distintos. Tras probar ambas redes en la tarea de asociar la edad de un paciente con el espectrograma obtenido a partir de su ECG, la red DenseNet121 obtuvo mejores resultados.

Las redes DenseNet fueron introducidas en el artículo *Densely Connected Convolutional Networks*, publicado por primera vez en 2016 [63]. Estas redes funcionan mediante un diseño de conexiones densas entre las capas gracias a las cuales cada capa toma como entrada no solo la salida de la capa inmediatamente anterior, sino también la salida de capas previas, reutilizando información adquirida por capas previas que de otra forma se perdería. Las redes DenseNet están formadas por varios bloques densos que están separados por capas de transición. Dentro de cada bloque denso, cada capa se conecta a todas las anteriores. Estas conexiones hacen que cada capa no tenga que aprender de cero patrones reconocidos en capas anteriores, requiriendo menos parámetros entrenables que redes como las residuales. En la figura 21 se puede apreciar la estructura de un bloque residual, en ella es fácil ver como cada capa se conecta con las anteriores. En la figura 22 se aprecia la tabla que recoge las capas que tienen las redes densas predefinidas en PyTorch, en esta se puede ver que la DenseNet121 es la de menor tamaño, aunque ha sido la que mejores resultados ha arrojado en la tarea que abarca el presente trabajo.

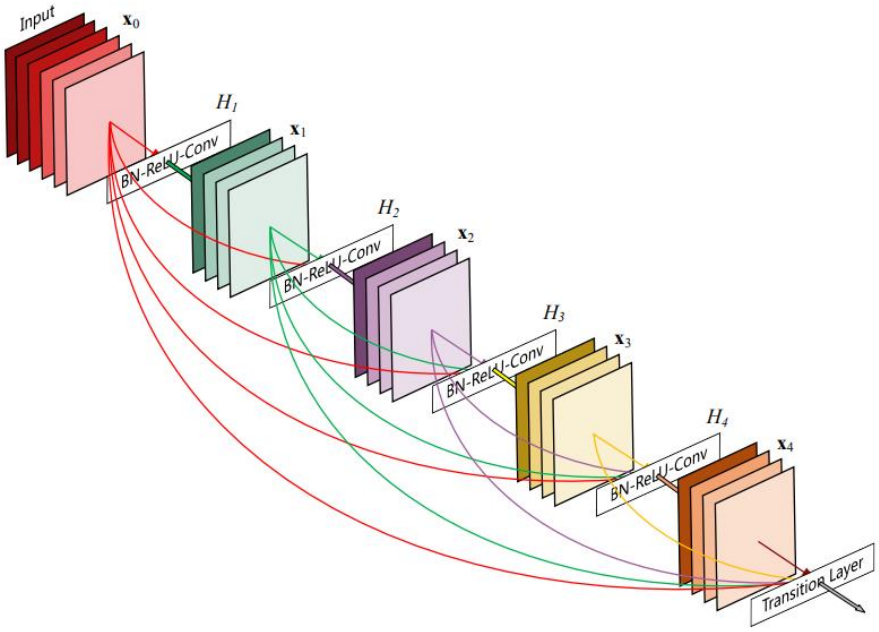

Figura 21. Esquema bloque denso [63].

Layers	Output Size	DenseNet-121	DenseNet-169	DenseNet-201	DenseNet-264
Convolution	112 × 112	7 × 7 conv, stride 2			
Pooling	56 × 56	3 × 3 max pool, stride 2			
Dense Block (1)	56 × 56	[1 × 1 conv; 3 × 3 conv] × 6	[1 × 1 conv; 3 × 3 conv] × 6	[1 × 1 conv; 3 × 3 conv] × 6	[1 × 1 conv; 3 × 3 conv] × 6
Transition Layer (1)	56 × 56	1 × 1 conv			
	28 × 28	2 × 2 average pool, stride 2			
Dense Block (2)	28 × 28	[1 × 1 conv; 3 × 3 conv] × 12	[1 × 1 conv; 3 × 3 conv] × 12	[1 × 1 conv; 3 × 3 conv] × 12	[1 × 1 conv; 3 × 3 conv] × 12
Transition Layer (2)	28 × 28	1 × 1 conv			
	14 × 14	2 × 2 average pool, stride 2			
Dense Block (3)	14 × 14	[1 × 1 conv; 3 × 3 conv] × 24	[1 × 1 conv; 3 × 3 conv] × 32	[1 × 1 conv; 3 × 3 conv] × 48	[1 × 1 conv; 3 × 3 conv] × 64
Transition Layer (3)	14 × 14	1 × 1 conv			
	7 × 7	2 × 2 average pool, stride 2			
Dense Block (4)	7 × 7	[1 × 1 conv; 3 × 3 conv] × 16	[1 × 1 conv; 3 × 3 conv] × 32	[1 × 1 conv; 3 × 3 conv] × 32	[1 × 1 conv; 3 × 3 conv] × 48
Classification Layer	1 × 1	7 × 7 global average pool			
		1000D fully-connected, softmax			

Figura 22. Tabla de las capas predeterminadas de las redes de tipo DenseNet integradas en PyTorch [64].

Las redes mencionadas son usadas para clasificar imágenes, como las que se emplean en el proyecto *ImageNet*, es por eso que la última capa de las redes densas mostradas en la figura 23 tiene 1000 salidas. Sin embargo, la tarea de asociar una edad con una imagen no es una tarea de clasificación, si no de regresión. Es decir, si quiero emplear el código público de estas redes para mi tarea debo primero modificarlas de manera que la última capa no tenga una salida por cada clase de imágenes, sino una única salida. Para ello he importado la red DenseNet121 y modificado su última capa para que dé un único valor de salida.

Para lograr esto se ha creado un módulo de tres capas lineales, encargadas de disminuir el tamaño de los datos de los 1024 datos que entran al módulo hasta 1 único valor de salida. Antes de cada capa lineal he configurado una regularización *Dropout*, la cual apaga aleatoriamente

neuronas en el entrenamiento para que no sobre aprendan. Esto favorece que las neuronas generalicen y no se sobreentrenen con el conjunto de entrenamiento. Tras cada capa lineal he implementado una capa de normalización por lotes y una capa de activación ReLU que hace 0 las salidas de cada capa si estas toman valores negativos. Finalmente, la salida de la última capa lineal la conecto a una función de activación sigmoide que la normaliza entre 0 y 1. Posteriormente, en el entrenamiento y en la validación tomo el valor de salida que arroja cada dato de entrada y lo normalizo entre el rango de edades que quiero contemplar. Es importante estudiar el rango de edades que queremos que el modelo sea capaz de predecir, ya que mediante diferentes experimentos hemos visto que un rango mayor arroja errores ligeramente inferiores a los resultantes por un rango menor.

Tras entrenar los modelos con derivaciones individuales se exploraron diferentes formas de obtener un único resultado final. Se consideró realizar la media de las 12 predicciones, la mediana y entrenar un modelo sencillo de *machine learning*. Se probaron las tres opciones y se obtuvo un mejor resultado con un modelo simple de regresión lineal, por lo que esta es la opción escogida para este proyecto.

Se han contemplado además varias arquitecturas de redes 1D diseñadas con el fin de competir en precisión con la red 2D entrenada con espectrogramas. Estas arquitecturas han sido las de redes convolucionales sencillas 1D, las de redes RCNN que mezclan capas residuales y capas convolucionales, las de redes con memoria a largo y corto plazo o LSTM por sus siglas en inglés y las de redes residuales de tipo ResNet. Finalmente se ha obtenido un mejor resultado empleando redes ResNet, por lo que estas han sido las elegidas para enfrentar a la red 2D, la DenseNet121.

En el caso de la DenseNet se trabajó sobre una red importada, la cual fue editada para la tarea de regresión. En el caso de la ResNet 1D se ha creado la red capa por capa, lo cual da más libertad al autor para aplicar las técnicas que asume necesarias para la mejora del programa. La red diseñada consta de 14 capas significativas, entre las que se encuentran 2 capas convolucionales iniciales, 6 bloques residuales y 3 capas finales. Los bloques residuales tratan de mitigar el desvanecimiento del gradiente mediante normalización en lotes y *dropout*. Esta red escala el dato de entrada hasta 1024 canales, empleando cada canal para aprender patrones durante el entrenamiento. La salida final es de un único canal que muestra la edad predicha por el modelo.

3.5.1 Técnicas Avanzadas para la Mejora del Modelo

Además de escoger la arquitectura de modelo más adecuada para el problema, es de interés optimizar los parámetros configurables en el código. Por ello y tras varias iteraciones con distintas configuraciones de la red 2D, se decidió emplear una tasa de aprendizaje inicial de 0.0002, un tamaño de lote de 32 y un número de épocas lo suficientemente alto para permitir al programa acercarse al mínimo de la función de pérdida, en este caso 25 épocas. Para el caso de la red 1D la tasa de aprendizaje inicial fue de 0.00001, un tamaño de lote de 32 y 125 épocas, ya que su convergencia fue más lenta.

Los datos se pueden modificar y adaptar antes de pasarlos al modelo, es recomendable ya que mejora la salida. En este caso las imágenes de los espectrogramas fueron convertidas a tensores y normalizadas. Esto es indispensable ya que los modelos realmente aceptan tensores como entrada para realizar operaciones convolucionales entre los datos y los filtros. Las señales crudas de entrada al modelo 1D sufrieron un preprocesamiento similar ya que fueron normalizadas y convertidas a tensores. Sin embargo, el empleo de señales 1D permitía aplicar

técnicas para minimizar el ruido, por lo que previamente estas señales se suavizaron con un filtro de media móvil y se filtraron mediante un filtro paso bajo de Butterworth de orden 3.

Respecto a la configuración de hiperparámetros escogida en ambos modelos para optimizar y mejorar el modelo se experimentó con varias opciones hasta dar con la más adecuada. La función de pérdida escogida es la función de Huber, caracterizada por funcionar como MSE para errores pequeños, aprovechando la suavidad de su curva, y como MAE para errores grandes, evitando que errores grandes afecten en mayor medida al cálculo de la pérdida.

El optimizador seleccionado en los dos programas ha sido el AdamW, explicado en la sección 2.1.4. Este ofrece mejores resultados que los optimizadores más comunes como el optimizador Adam, usado muy a menudo para problemas del estilo.

También se implementó un programador dinámico de la tasa de aprendizaje, mediante el cual el valor de la tasa de aprendizaje cambia en cada época permitiendo acercarse más al mínimo de la función de pérdida. Para el modelo 2D se seleccionó el programador *ReduceLROnPlateau* por reducir la tasa de aprendizaje cuando el modelo lleva varias épocas sin mejorar, basándose en el error del conjunto de validación. En el caso del modelo 1D ofreció mejores resultados el programador *CosineAnnealingWarmRestarts*, el cual funciona bajando la tasa de aprendizaje de manera cosenoidal y volviendo a subirla cuando la pérdida no mejore, evitando estancarse en mínimos locales.

Además, se decidió usar un objeto que escala los gradientes de los pesos automáticamente para que las operaciones que requieres de mayor precisión empleen 32 bits, mientras que las que requieren de menos precisión usen 16 bits. Este concepto se conoce como *Mixed Precision Training* y es útil para agilizar el entrenamiento y reducir la memoria necesaria para la tarea.

Finalmente, utilizo un limitador en los gradientes de cambio de los pesos. Mediante este limito los gradientes cuya norma sea mayor a 1 para evitar la explosión de los gradientes, evento que ocurre cuando los gradientes de cambio son demasiado grandes, dificultando la estabilidad del proceso de entrenamiento.

En ambos programas se incluyó un fragmento de código que evalúa el uso de cada tarjeta gráfica disponible en la máquina en la que se ejecuta y selecciona que se emplee la de menor uso, evitando interferencias con otros programas en funcionamiento.

3.5.2 Herramientas empleadas

El código empleado inicialmente para el procesamiento de los datos fue programado en MATLAB, un lenguaje de programación especialmente diseñado para el ámbito matemático. Esto permitió generar un código sin demasiada dificultad que realizara la tarea de filtrar, recortar, preprocesar y convertir a espectrogramas las señales compartidas en la base de datos CODE15.

Programar en MATLAB es una tarea intuitiva gracias a herramientas procedentes de la *Signal Processing Toolbox* y de la *MATLAB Image Processing Toolbox*. La primera contiene las instrucciones y herramientas necesarias para procesar y generar las señales y convertirlas en espectrogramas [65]. Por su parte, la *MATLAB Image Processing Toolbox* contiene las instrucciones que facilitan la generación y guardado de las imágenes [66].

El código relacionado con el modelo neuronal fue escrito enteramente en Python, pues este lenguaje de programación contiene muchas librerías ya creadas que facilitan enormemente el trabajo relacionado con redes neuronales. Se han empleado las siguientes librerías para esta tarea:

- *os*: facilita el trabajo con elementos del sistema operativo, por ejemplo, permite leer el directorio de la base de datos y gestionar los ficheros de salida.
- *subprocess*: permite ejecutar comandos del sistema y capturar su salida. Ha sido empleado para evaluar el uso a tiempo real de las diferentes tarjetas gráficas del servidor para así elegir trabajar con la tarjeta de menor uso.
- *time*: librería que incluye comandos para trabajar con el tiempo. Ha sido usada para monitorear el tiempo de entrenamiento del código.
- *re*: ha sido usada para filtrar ficheros según la cadena de texto de su nombre. Por ejemplo, este módulo ha facilitado trabajar con una selección de directorios dentro de la base de datos, filtrados según su nombre.
- *numpy*: esta librería es esencial en una gran parte de los trabajos que se realizan con Python, proporciona innumerables funciones relacionadas con la matemática y es ideal para realizar diferentes operaciones matemáticas. Ha sido empleada para calcular la media y la desviación típica de los datos de entrada para luego normalizarlos, también para transformarlos en arrays de números y tensores.
- *pandas*: otra de las librerías indispensables de Python. Es ideal para trabajar con grandes volúmenes de datos ya que permite crear y gestionar grandes conjuntos de datos.
- *seaborn*: librería muy útil para crear gráficos de distintos tipos, como los gráficos calculados de predicciones vs. targets.

Figura 23. Gráfico resultante de calcular las predicciones del modelo tras entrenarle con los espectrogramas derivados del segundo canal de ECG.

- *matplotlib.pyplot*: librería base de Python para visualizar gráficos y diagramas. Ha sido empleada junto a seaborn para la creación de los gráficos, permitiendo poner el título y los ejes a los gráficos y guardárlos.
- *PIL*: módulo que permite cargar imágenes. Ha sido empleada para cargar las imágenes desde la base de datos a el código, para luego procesarlas como tensores mediante otras librerías.
- *tqdm*: esta librería permite agregar barras de progreso al código. Esto facilita en entendimiento del flujo del programa y ha sido empleado para monitorear el entrenamiento y la validación según sucedían.
- *scipy*: librería que contiene módulos para realizar análisis matemático sobre los datos. Usada para calcular directamente el coeficiente de correlación de Pearson y verificar resultados.
- *scikit-learn*: librería comúnmente empleada para tareas de aprendizaje automático. En este caso ha sido usada para calcular diferentes métricas de evaluación, como el MAE o el MSE. Se eligió este módulo por su simplicidad.
- *PyTorch*: esta es la librería principal del proyecto, pues permite trabajar sobre modelos de redes neuronales profundas ya creados y diseñados [67]. También incluye las diferentes funciones de activación, algoritmos de optimización, herramientas de gestión de los datos y trabajo en la GPU.
- *torchvision*: módulo dentro de PyTorch que contiene herramientas que facilitan la transformación y normalización de los datos y los modelos de redes neuronales.
- *torch_lr_finder*: módulo dentro de PyTorch que permite encontrar la tasa de aprendizaje óptima para el inicio del entrenamiento del programa.

El código final, así como parte de las versiones anteriores, fueron ejecutados en remoto mediante el protocolo SSH en un servidor. El servidor remoto utilizado en este proyecto cuenta con el sistema operativo Debian, un sistema ampliamente usado por su disponibilidad y facilidad de manejo basado en el kernel de Linux. Esta versión en particular está diseñada para arquitecturas de 64 bits. El servidor está equipado con cuatro potentes tarjetas gráficas NVIDIA RTX A5000, cada una con una capacidad de 24564 MB de memoria disponible, creando un entorno de trabajo ideal para proyectos tan intensos como el llevado a cabo.

Los costosos recursos computacionales han sido cedidos muy amablemente por el Laboratorio de Procesado de Imagen de la Universidad de Valladolid [68], un grupo de trabajo comprometido con el avance en la investigación científica y tecnológica. Este apoyo ha sido fundamental para el desarrollo del proyecto, ya que si no el tiempo de desarrollo de los diferentes programas hubiera sido enorme, ya que el programa tarda en ejecutarse en el servidor 3 horas por cada derivación lo cual se hubiera podido extender a un día de no contar con estos potentes recursos.

Capítulo 4. Resultados

4.1 Métricas empleadas

En este apartado se explicarán las distintas métricas empleadas para la evaluación del rendimiento del modelo.

Durante el entrenamiento y validación del modelo los pesos se han calculado para minimizar la función de pérdida, el error resultante se ha ido registrando a lo largo de las épocas. Este error depende de la función de pérdida escogida, en el caso de este proyecto se ha escogido la función de Huber. Sin embargo, al no ser tan conocida y fácil de interpretar como los siguientes parámetros estadísticos, la pérdida de Huber no ha sido empleada para validar los resultados finales.

La métrica más comúnmente empleada para estudiar el error en estos sistemas es el error absoluto medio, MAE por sus siglas en inglés. El MAE sigue la siguiente fórmula:

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Dónde y_i es la predicción y $\hat{y}_i$ es el valor verdadero. En la fórmula se puede ver que el MAE es una métrica que cuantifica la magnitud media de los errores, de manera que todos los errores computan igual al valor final del MAE. La unidad de medida del MAE coincide con la unidad de medida de las predicciones, en el caso del presente proyecto el MAE se mide en años.

La segunda métrica que se ha empleado para validar los resultados finales del modelo es el error cuadrático medio, o MSE por sus siglas en inglés. El MSE sigue la siguiente fórmula:

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Dónde y_i es la predicción y $\hat{y}_i$ es el valor verdadero. Al analizar la fórmula es fácil entender que al contrario de lo que pasa en el MAE, en el MSE cada error computa distinto, de manera que errores más grandes influenciarán en mayor medida al valor total del MSE. Emplear el MSE en vez del MAE puede ser útil para que el modelo penalice más a los errores más grandes, de manera que los casos que disten de la media de la totalidad de los datos son más influyentes en el valor del error.

En la siguiente imagen se puede apreciar la ponderación del error según los parámetros explicados. Se puede apreciar que la función de Huber es parecida a la función del MAE, pero no es lineal si no que está suavizada. También es fácil entender como el MSE pondera en mayor medida los errores mayores a 1 comparándolo con el MAE y la función de Huber.

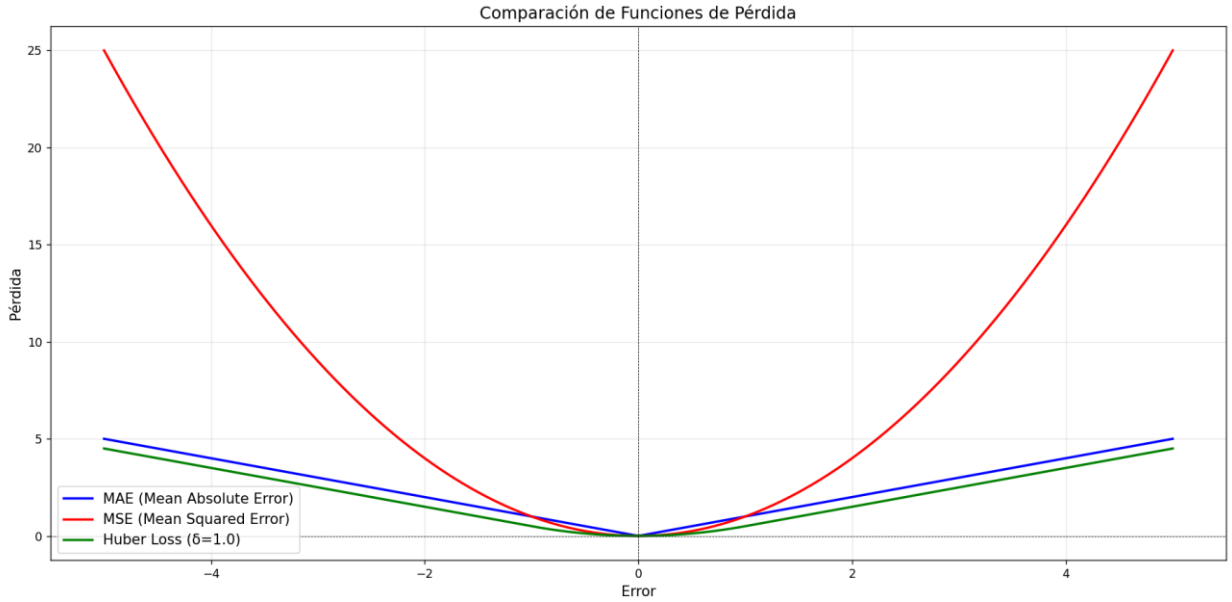

Figura 214. Gráfica comparativa de las diferentes métricas empleadas.

Por último, se han empleados dos parámetros adicionales, el coeficiente de correlación de Pearson r y el coeficiente de determinación R^2.

El coeficiente de correlación de Pearson mide la relación lineal entre dos variables, como son en este caso las edades predichas y las edades reales. Es decir, el coeficiente de relación de Pearson indica que tan bien se correlacionan las variables, cuantifica si los datos predichos siguen el patrón de los datos reales, aunque no indica directamente si el modelo tiene un error bajo. El coeficiente de Pearson está acotado entre -1 y 1, indicando un valor de -1 una relación lineal negativa perfecta, el valor 0 una nula relación y el valor 1 una relación lineal positiva perfecta. La fórmula del coeficiente de correlación es la siguiente:

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

Donde x_i e y_i son las variables de estudio, la edad predicha y la edad real en este caso, y sus valores con una barra encima son los valores medios. Además, n es el número total de las observaciones.

El coeficiente de determinación R^2 determina que proporción de las diferencias en las edades reales se debe a la precisión del modelo, siendo el restante de la proporción resultados del azar. Se relaciona con el coeficiente de correlación de Pearson como el cuadrado de este, de manera que, si existe una relación lineal perfecta, $r^2 = R^2 = 1$. El valor del coeficiente de determinación está acotado entre 0 y 1 por ser una proporción, indicando un valor de 1 que la variable dependiente, edad predicha, es completamente explicable por la variable independiente, edad real.

La fórmula del coeficiente de determinación es la siguiente:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Donde y_i es la variable independiente, edad real, $\hat{y}_i$ es la variable dependiente, edad predicha y $\bar{y}$ es la media de los valores observados.

4.2 Evaluación de los resultados

Un electrocardiograma recoge una señal por cada derivación estudiada y cada señal puede contener información distinta de la del resto relevante para el problema. Es por esto que generar espectrogramas que respondan a las señales de todas las derivaciones a la vez para luego pasarlas al modelo de aprendizaje automático carece de sentido. En este proyecto se ha usado un programa por cada derivación, para así obtener una predicción de edad basada en cada canal con el fin de estudiar qué canales influencian más a la predicción de edad. Además, este enfoque ofrece una gran ventaja, al obtener 12 resultados de predicción de edad es posible crear un modelo que encuentre una combinación óptima de estos resultados y logre predecir la edad de la persona con un mayor acierto. Esta hipótesis queda demostrada al realizar el trabajo.

Por lo presentado anteriormente se expondrán en este apartado los resultados obtenidos a partir de cada derivación y al finalizar se mostrará el resultado conjunto.

En la siguiente tabla se recogen los resultados obtenidos en el conjunto de validación para cada una de las 12 derivaciones con el modelo de red 2D DenseNet121:

Derivación	MAE (años)	MSE (años^2)	*r*	R^2
DI	10.1376	171.4817	0.7143	0.5053
DII	9.3376	146.0705	0.7607	0.5786
DIII	10.3036	176.1342	0.7027	0.4919
AVR	9.7581	164.6060	0.7339	0.5252
AVL	10.2550	177.8272	0.7060	0.4870
AVF	9.6250	157.1594	0.7401	0.5466
V1	9.9685	167.4794	0.7203	0.5169
V2	10.4156	184.3416	0.6913	0.4682
V3	9.8964	166.8300	0.7246	0.5188
V4	9.3909	150.6480	0.7539	0.5654
V5	9.5432	154.3367	0.7453	0.5548
V6	9.7572	164.2422	0.7295	0.5262

Para interpretar mejor esta información se ha realizado un gráfico para cada columna de la tabla anterior.

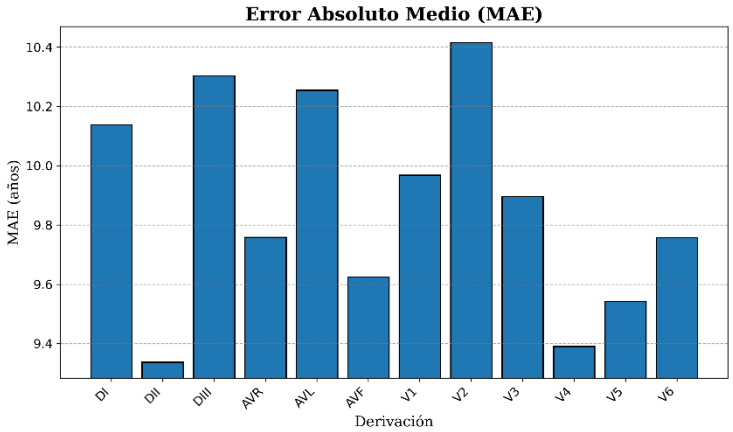

Figura 25. Gráfica del MAE en cada derivación para el caso 2D.

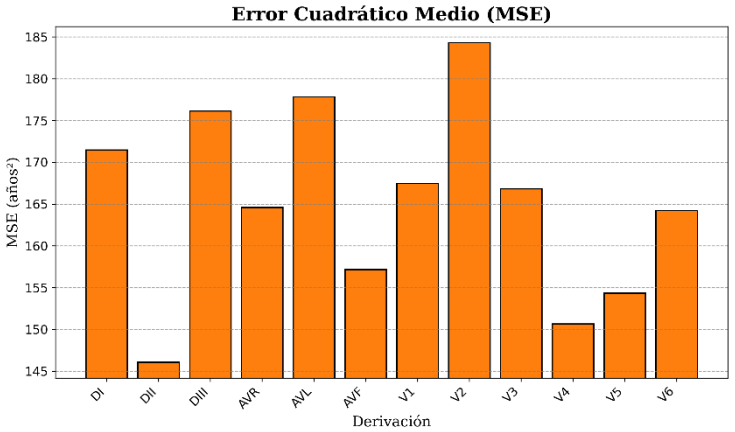

Figura 26. Gráfica del MSE en cada derivación para el caso 2D.

Observando las gráficas anteriores es inmediato ver que las derivaciones que mejor MAE y MSE obtuvieron fueron la DII, la V4 y la V5, ya que las columnas que representan estas derivaciones tienen una altura menor. Indicando este hecho que estas derivaciones muestran información asociada a la edad del paciente en el espectrograma calculado.

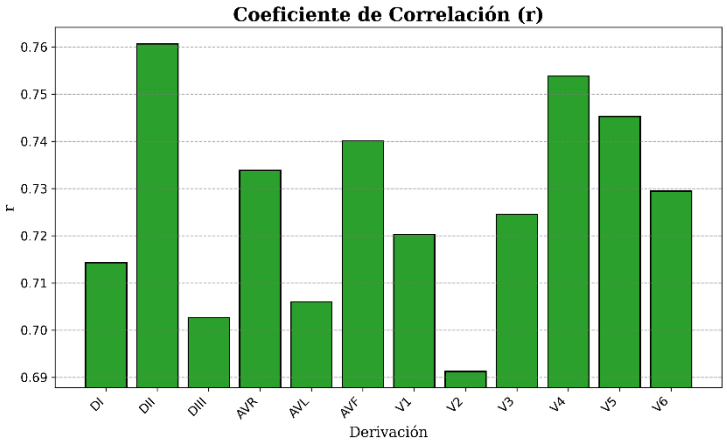

Figura 22. Gráfica de r en cada derivación para el caso 2D.

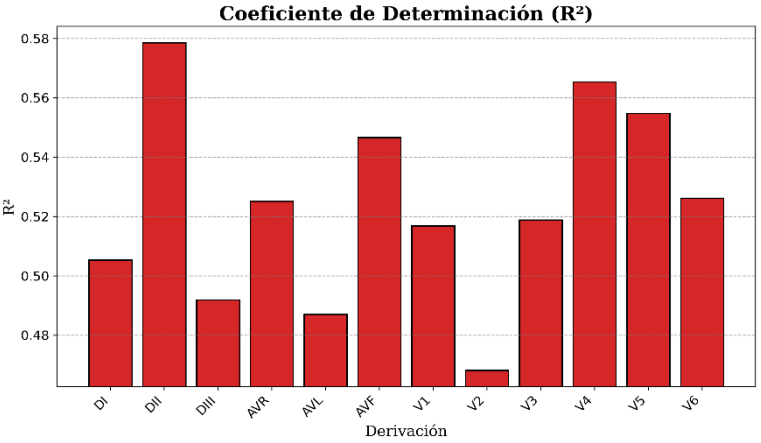

Figura 28. Gráfica de R^2 en cada derivación para el caso 2D.

En las gráficas anteriores podemos evaluar lo intuido anteriormente, las derivaciones que mejor expresan la información relacionada con la edad con la DII, V4 y V5, por ser las que mayor coeficiente de correlación de Pearson y coeficiente de determinación presentan.

Es normal que haya tanta variación entre el MAE obtenido con cada derivación, ya que el posicionamiento de los electrodos determina en gran medida la información que tendrá la señal eléctrica captada y el posible ruido asociado, como se vio anteriormente en la figura 11. La derivación DII es la que más información ha ofrecido al modelo, esto puede ser porque DII se mide entre la pierna izquierda y el brazo derecho, captando una señal que pasa por todo el corazón. V4 y V5 también resultan en un MAE pequeño porque reflejan la actividad eléctrica en el ventrículo izquierdo, el cual se altera fácilmente a medida que se envejece como resultado de la hipertrofia ventricular. Por otra parte, V2 ha sido la que peor resultado ha generado, esto no es del todo lógico ya que el posicionamiento de los electrodos causa que la señal debiera tener tanta información como V1 y V3. Se puede teorizar que este comportamiento es causado por el

ruido, ya que la derivación V2 es muy sensible al ruido. DIII y aVL también han ofrecido pésimos resultados, aplicando la misma teoría que con DII se puede pensar que como DIII se obtiene entre el brazo y la pierna izquierdos la señal eléctrica no captura patrones tan importantes como el resto de las derivaciones. Como aVL se relaciona matemáticamente con DIII es lógico pensar que esta va a contener información de un nivel de interés similar al de DIII. Por último, aVR, aVF, V1, V3 y V6 contienen información útil para el modelo debido a la colocación de los electrodos.

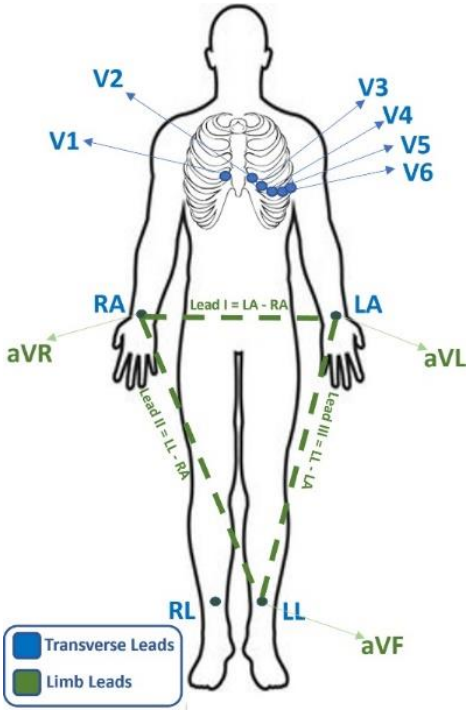

Figura 11. Esquema de posicionamiento de electrodos [26]

Finalmente, los resultados obtenidos para cada derivación fueron empleados en un modelo simple de regresión lineal, como se mencionó en el capítulo 3. Este modelo obtuvo una predicción con un MAE igual a 7.97 años, mejorando enormemente la predicción obtenida con derivaciones individuales.

Uno de los objetivos del trabajo es estudiar la eficiencia de emplear espectrogramas en vez de señales crudas en la tarea de estimación de edad mediante inteligencia artificial. Debido a esto también se han obtenido resultados con la red 1D ResNet, la cual ha sido diseñada para aceptar señales en vez de imágenes.

Los resultados obtenidos con esta red han sido recogidos en la siguiente tabla, similar a la mostrada anteriormente para el caso de la red 2D.

Derivación	MAE (años)	MSE (años^2)	*r*	R^2
DI	10.37	180.48	0.69	0.47
DII	9.87	165.58	0.72	0.51
DIII	10.37	182.78	0.69	0.46
AVR	10.07	171.00	0.71	0.50
AVL	10.44	183.90	0.69	0.46
AVF	10.06	173.88	0.71	0.49
V1	10.48	187.66	0.69	0.45
V2	10.50	187.46	0.68	0.45
V3	10.17	174.80	0.70	0.49
V4	9.75	162.37	0.73	0.52
V5	9.75	163.23	0.73	0.52
V6	10.06	174.83	0.71	0.49

Para interpretar mejor esta información se ha realizado un gráfico para cada columna de la tabla anterior.

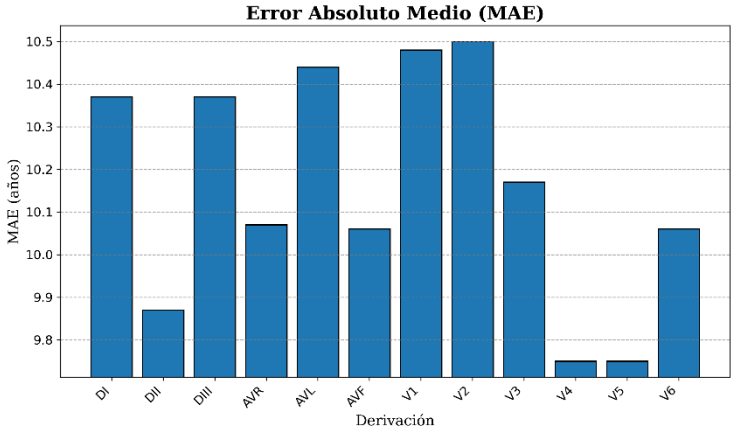

Figura 23. Gráfica del MAE en cada derivación para el caso 1D.

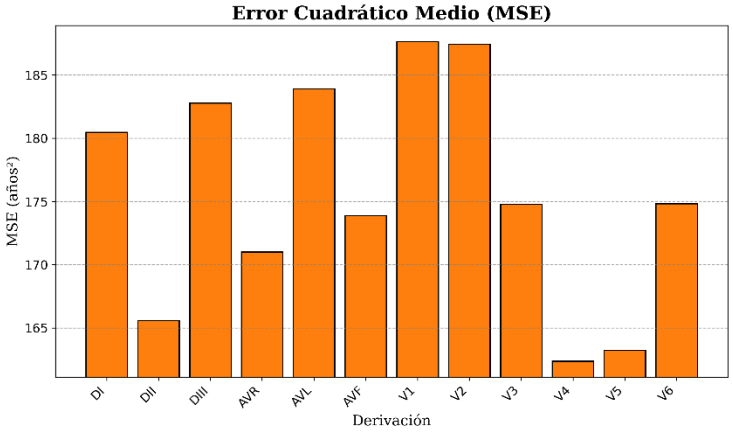

Figura 30. Gráfica del MSE en cada derivación para el caso 1D.

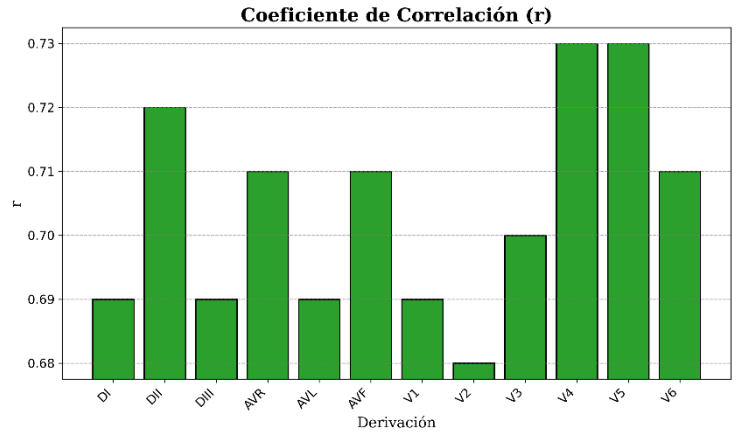

Figura 31. Gráfica de r en cada derivación para el caso 1D.

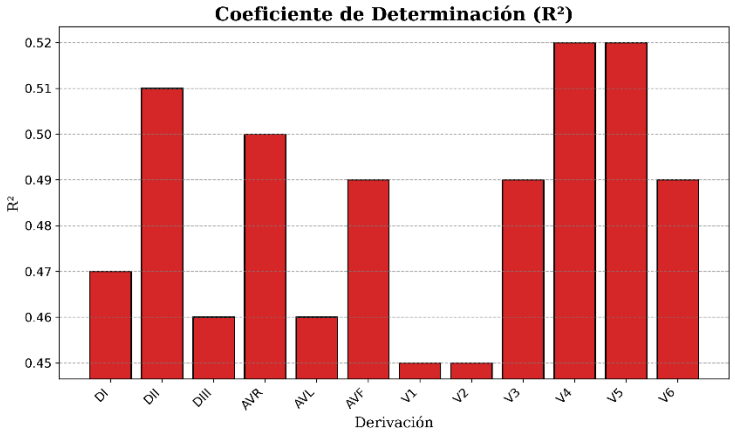

Figura 32. Gráfica de R^2 en cada derivación para el caso 1D.

Observando las gráficas anteriores se obtienen las mismas conclusiones que las obtenidas para el caso 2D: las derivaciones que más información relacionada con la edad ofrecen al modelo son la DII, V4 y V5, aunque esta vez la derivación DII ha resultado inferior a las otras dos.

Estos resultados no hacen más que reafirmar lo que se teorizó anteriormente, las derivaciones DII, V4 y V5 son más propensas a cambiar según el estado del sistema cardiovascular de la persona se deteriora.

Comparando los resultados del caso 1D con los del caso 2D resulta que la red 2D ha sido más precisa en su estimación para todas las derivaciones, lo cual resalta la utilidad de los espectrogramas y permite concluir que el empleo de espectrogramas en la evaluación de señales de electrocardiografía ofrece mejores resultados que el empleo de las señales directas para el problema de la estimación de edad.

Capítulo 5. Discusión

5.1 Implicaciones

Los resultados obtenidos durante la realización de este proyecto aportan un avance al campo del aprendizaje automático a partir de señales. Queda demostrado que la trasformación de las señales en espectrogramas puede mejorar notablemente las predicciones de los modelos, ya que de esta forma se pueden aplicar redes complejas que aprovechen las características espaciales de la imagen, como puede ser la relación entre las diferentes frecuencias y el tiempo de una señal. Además, según los resultados obtenidos podemos concluir que mediante la representación frecuencial de la señal en forma de espectrograma la red puede captar patrones frecuenciales que difícilmente captaría un modelo 1D aplicado sobre señales crudas.

En el presente proyecto también se ha estudiado la información que muestra cada una de las derivaciones de un electrocardiograma con el objetivo de enfrentar las distintas derivaciones entre sí y analizar cuáles de estas contienen más información relativa al envejecimiento del sistema cardiaco. Como se vio en el capítulo anterior las derivaciones que más información guardan al respecto son la DII, la V4 y la V5, lo cual sucede en los dos modelos estudiados.

La aceptación de las aplicaciones de aprendizaje automático en la industria médica implicaría un gran impacto positivo, ya que agilizaría los procesos permitiendo a los especialistas atender a más pacientes y reducir las largas listas de espera.

El modelo 2D diseñado ha demostrado aprovechar eficientemente las fortalezas de las redes densas aplicadas a la representación frecuencial de las señales.

5.2 Comparación con estudios similares

Si se comparan los resultados mostrados en el capítulo 4 con los resultados que han obtenido diferentes investigadores para el mismo problema, recogidos en el capítulo 2, se llega a una conclusión positiva: el modelo descrito en este documento obtiene mejores resultados que aquellos obtenidos por el resto de los investigadores que hayan entrenado sus modelos con la misma base de datos.

Antes de hacer una comparación entre los resultados obtenidos en este trabajo y los obtenidos en estudios similares es importante recalcar la importancia de las características de las bases de datos y su influencia en los resultados finales. Las bases de datos que emplean los distintos estudios son diferentes entre sí, cada una cuenta con diferentes características como la distribución de los datos, el rango de edades que comprenden, la frecuencia de muestreo de las señales y la longitud de las mismas. Es por esto por lo que, aunque se obtenga un MAE menor en un estudio que en otro, esto no significa que el primer estudio sea mejor que el segundo, ya que debido a las características de su base de datos puede ser más fácil obtener resultados más precisos. Por ejemplo, si una base de datos tiene un rango de edades comprendidas entre los 30 y los 60 años, mientras que otra base de datos tiene un rango de edades comprendidas entre los 17 y los 88 años, fácilmente el MAE obtenido al aplicar un modelo sobre la primera base de datos será menor que el obtenido al aplicar el mismo MAE sobre la segunda base de datos.

Como se describió en el capítulo 3, la base de datos empleada en este proyecto, la CODE-15%, es ruidosa, tiene un rango de edades entre los 17 y los 100 años y proviene de extraer un 15% de los datos de la base CODE.

El *paper* más comúnmente referido al tratar la cuestión principal de este proyecto es *Deep neural network-estimated electrocardiographic age as a mortality predictor* [48], realizado por Lima, et al. En este artículo los autores diseñan y entrenan una red de tipo residual y la aplican sobre las matrices de las señales. La base de datos que emplean para ello es la CODE y obtienen un MAE de 8.38 años y un coeficiente de Pearson de 0.84. Comparando el resultado final obtenido en el proyecto actual de 7.97 años para una base de datos de un tamaño igual al 15% de la base de datos empleada por Lima está claro que es una mejora notable, no solo se ha obtenido un menor MAE, si no que se ha hecho empleando una cantidad reducida de datos.

En *Age and Sex Estimation Using Artificial Intelligence From Standard 12-Lead ECGs* [44], Attia, et al. obtuvieron un MAE menor igual a 6.9 años, sin embargo sería injusto comparar estos resultados a los obtenidos en este trabajo debido a que Attia hace uso de una base de datos cuyo rango de edades es notablemente menor. Lo mismo sucede si comparamos los resultados actuales al MAE de 7.4 años obtenido en *The 12-lead electrocardiogram as a biomarker of biological age* [45].

El resto de los trabajos mencionados en el punto 2.4.1 emplean una base de datos diferente a la CODE, por lo que no es justo comparar los resultados aquí obtenidos con los suyos. Si quisiéramos comparar la eficacia del modelo aquí descrito a la del modelo creado por varios investigadores deberíamos aplicar el modelo diseñado a sus bases de datos.

El proyecto presentado en este documento es innovador, ya que no existen investigaciones públicas en la que se intente estimar la edad de una persona a través de la conversión de las señales del electrocardiograma a su espectrograma. Este enfoque ha resultado mejorar las predicciones a partir de una base de datos más limitada.

Capítulo 6. Conclusión y líneas futuras

6.1 Resumen de los resultados

Los espectrogramas han resultado ser una herramienta exitosa en el área del aprendizaje automático, ya que los resultados obtenidos empleando los espectrogramas como datos de entrada a redes convolucionales 2D han superado a los resultados que se comparten en los estudios en los que tratan de predecir la edad de una persona a través de la señal cruda de su electrocardiograma. Además estos resultados han superado los obtenidos mediante la red 1D también diseñada en este proyecto, la cual emplea las señales como datos de entrada.

El apartado 4.1.3 del escrito *Estimating age and gender from electrocardiogram signals: A comprehensive review of the past decade* [69] menciona que diferentes estudios han concluido que usar las 12 derivaciones de un ECG empeora la estimación, ya que algunas de las derivaciones son calculadas matemáticamente a partir de las otras por lo que ofrecen información redundante. Este artículo menciona que usando únicamente 4 derivaciones se obtienen mejores resultados que con un conjunto más elevado. En el presente trabajo podemos añadir sobre esto que ciertas derivaciones de los electrocardiogramas no solo contienen información redundante, si no que la información que contienen relacionada con la edad del paciente es más difícil de detectar por los modelos de aprendizaje automático. Esto se traduce en que una posible mejora de los modelos empleados para el problema tratado en este proyecto es emplear un número limitado de canales, en vez de los 12 disponibles.

Según los resultados de ambos modelos, las derivaciones con más información útil para el modelo de predicción de edad a partir de datos de espectrogramas son la DII, la V4 y la V5, puesto que individualmente han resultado en un menor MAE.

Quedan cumplidos los objetivos del trabajo, estudiar la viabilidad de la predicción de edad a partir de datos de ECG usando una representación alternativa de la señal del ECG a través de espectrogramas y estimar la importancia relativa de los diferentes canales de un ECG para la predicción de edad.

6.2 Aplicaciones clínicas

La aplicación directa de estas herramientas en la medicina es extremadamente útil en la identificación temprana de problemas cardiacos, además de permitir a cualquier persona llevar un control de su estado de salud si se somete a la realización periódica de un electrocardiograma sin tener que consultar con un especialista. Los ECG son una herramienta recurrente aun cuando no se investigan problemas en el sistema cardiaco, por lo que los centros hospitalarios suelen contar con bases de datos de electrocardiogramas ya realizados a sus pacientes. Aprovechar estas pruebas ya realizadas y almacenadas en las bases de datos puede suponer la detección de problemas sin que el paciente tenga si quiera que pasar por el centro.

Los modelos diseñados en este proyecto pueden ser aplicado directamente en la industria médica y su elección depende de la potencia disponible en el centro. Utilizar un modelo ya entrenado requiere de una potencia computacional baja, por lo que supone un coste pequeño.

Sin embargo, antes de ser aplicado debe ser probado con datos obtenidos en los hospitales de interés para adaptar la red al nuevo formato de los datos y calibrar el modelo si fuera necesario.

Aplicar esta herramienta no solo ayudaría a identificar prematuramente problemas en el sistema cardiovascular, sino que además el emplear tecnología de vanguardia y mostrar el estado de salud real de la población española podría concienciar a la sociedad sobre los diferentes hábitos que pueden ser perjudiciales para la salud.

6.3 Líneas de investigación futuras

Con base en los resultados del modelo se proponen varias líneas de investigación futuras que continuarían el trabajo ya realizado en el proyecto:

- Mejora del modelo actual: aunque el modelo ofrece buenos resultados está lejos de ser perfecto, varios cambios podrían hacer que los resultados mejorasen. Aunque se haya dedicado una gran cantidad de tiempo y esfuerzo a la optimización del modelo aún puede mejorarse añadiendo y disminuyendo el número de capas escogido y realizando un ajuste de los hiperparámetros todavía más extenso empleando herramientas de búsqueda automática de hiperparámetros.

  Una parte importante del modelo es el preprocesamiento de los datos, se propone buscar la mejora de la etapa de preprocesado con el objetivo de que los nuevos datos faciliten el aprendizaje de la red.

- Reemplazo del modelo: como se menciona en el punto 3.5, diseño de los modelos escogidos, se ha tratado de buscar el mejor modelo para aplicar a la base de datos seleccionada. De esta forma se han explorado diferentes arquitecturas variadas que trabajan con distintos mecanismos para lograr resolver el problema. Por ello se propone emplear redes vanguardistas diseñadas recientemente que puedan beneficiarse de los datos para obtener resultados más correctos.
- Ampliación de la base de datos: como ya se ha explicado, la base de datos escogida es la CODE-15%, la cual cuenta con varias etiquetas útiles para el proyecto. Se propone la búsqueda y empleo de diferentes bases de datos que permitan generalizar sobre un rango de datos más extenso.
- Calibración de la red: la red creada ha sido entrenada con miles de datos por lo que debería ser capaz de generalizar lo aprendido sobre datos de fuera de la base de datos, sin embargo, esto es un escenario ideal ya que cada base de datos cuenta con sus propias características. Se propone pues estudiar la aplicación directa del modelo mediante la validación de la red con datos reales cedidos por centros hospitalarios cercanos.
- Estudio de la edad biológica: en el capítulo 2 se describe el concepto de edad biológica y se estipula que la edad cardiaca es solo un subconjunto de la edad biológica. Se propone integral el modelo actual a un sistema completo que mediante diferentes técnicas evalúe el estado de salud general de un paciente, integrando modelos que predigan la edad de cada sistema.

Referencias

[1] Li, Z., Li, X., Xu, Z., Wang, L., Yang, L., & Ren, W. (2023). Progress in biological age research. Frontiers in Public Health, 11, 1074274. https://doi.org/10.3389/fpubh.2023.1074274

[2] «Logic Theorist», Wikipedia. 4 de agosto de 2024. Accedido: 18 de agosto de 2024. Disponible en línea en: https://en.wikipedia.org/w/index.php?title=Logic_Theorist&oldid=1238596146

[3] «Dartmouth workshop», Wikipedia. 16 de agosto de 2024. Accedido: 18 de agosto de 2024. Disponible en línea en: https://en.wikipedia.org/w/index.php?title=Dartmouth_workshop&oldid=1240707438

[4] Mysid. (2006). A simplified view of an artificial neural network [Imagen]. Disponible en línea en: https://commons.wikimedia.org/wiki/File:Neural_network.svg#/media/File:Neural_network.svg

[5] Chollet, F. (2021). Deep learning with Python (2nd ed.). Manning Publications.

[6] Brianne. (s.f.). Support Vector Machine (SVM) Model. Discovery in the Post-Genomic Age. Accedido: 20 de agosto de 2024. Disponible en línea en: https://www.raybiotech.com/learning-center/support-vector-machine-model/

[7] Google Cloud. (s.f.). ¿Qué es el aprendizaje no supervisado? Accedido: 18 de agosto de 2024. Disponible en línea en: https://cloud.google.com/discover/what-is-unsupervised-learning

[8] Gayhardt, L. (s.f.). Machine Learning Algorithm Cheat Sheet. Azure Machine Learning. Accedido: 18 de agosto de 2024. Disponible en línea en: https://learn.microsoft.com/en-us/azure/machine-learning/algorithm-cheat-sheet?view=azureml-api-1

[9] SAS Data Science Blog. (s.f.). Which machine learning algorithm should I use? Accedido: 26 de agosto de 2024. Disponible en línea en: https://blogs.sas.com/content/subconsciousmusings/2020/12/09/machine-learning-algorithm-use/

[10] ResearchGate. (s.f.). Figura III.3. Estructura de una Neurona Biológica [Figura]. Accedido: 18 de agosto de 2024. Disponible en línea en: https://www.researchgate.net/figure/Figura-III3-Estructura-de-una-Neurona-Biologica_fig2_315762548

[11] ML Glossary Documentation. (s.f.). Concepts. Accedido: 18 de agosto de 2024. Disponible en línea en: https://ml-cheatsheet.readthedocs.io/en/latest/nn_concepts.html

[12] Hirekerur, R. (s.f.). A Comprehensive Guide To Loss Functions — Part 1 : Regression. Analytics Vidhya. Accedido: 28 de agosto de 2024. Disponible en línea en: https://medium.com/analytics-vidhya/a-comprehensive-guide-to-loss-functions-part-1-regression-ff8b847675d6

[13] Li-Bland, D. (s.f.). David Li-Bland's Blog - Saddle Points and Stochastic Gradient Descent. Accedido: 29 de agosto de 2024. Disponible en línea en: https://davidlibland.github.io/posts/2018-11-10-saddle-points-and-sdg.html

[14] Kingma, D. P., & Ba, J. (2017). Adam: A Method for Stochastic Optimization. arXiv. Disponible en línea en: http://arxiv.org/abs/1412.6980

[15] Loshchilov, I., & Hutter, F. (2019). Decoupled Weight Decay Regularization. arXiv. Disponible en línea en: http://arxiv.org/abs/1711.05101

[16] Ben-Ahmed, O., Fernandez-Maloigne, C., Julian, A., & Paccalin, M. (2019). Visual saliency for medical imaging and computer-aided diagnosis. En Neurological Disorders and Imaging Physics, Volume 3: Application to autism spectrum disorders and Alzheimer's. IOP Publishing. Disponible en línea en: https://doi.org/10.1088/978-0-7503-1793-1ch9

[17] Zou, F., Tang, Y., Liu, C., Ma, J., & Hu, C. (2020). Concordance Study Between IBM Watson for Oncology and Real Clinical Practice for Cervical Cancer Patients in China: A Retrospective Analysis. Frontiers in Genetics, 11, 200. Disponible en línea en: https://doi.org/10.3389/fgene.2020.00200

[18] Crouch, H. (s.f.). Google DeepMind's Streams technology branded "phenomenal". Digital Health. Accedido: 26 de agosto de 2024. Disponible en línea en: https://www.digitalhealth.net/2017/12/google-deepmind-streams-royal-free/

[19] Nanox AI. (s.f.). Nanox AI's solutions help identify early signs of chronic disease within CT scans. Accedido: 26 de agosto de 2024. Disponible en línea en: https://www.nanox.vision/ai#solutions

[20] Adib, A., Zhu, W.-P., & Ahmad, M. O. (2022). Adult and Non-Adult Classification Using ECG. En 2022 IEEE 7th Forum on Research and Technologies for Society and Industry Innovation (RTSI) (pp. 174-179). Disponible en línea en: https://doi.org/10.1109/RTSI55261.2022.9905194

[21] Libiseller-Egger, J., et al. (2022). Deep learning-derived cardiovascular age shares a genetic basis with other cardiac phenotypes. Scientific Reports, 12(1), 22625. Disponible en línea en: https://doi.org/10.1038/s41598-022-27254-z

[22] Toya, T., et al. (2021). Vascular Aging Detected by Peripheral Endothelial Dysfunction Is Associated With ECG-Derived Physiological Aging. Journal of the American Heart Association, 10(3), e018656. Disponible en línea en: https://doi.org/10.1161/JAHA.120.018656

[23] NHS.uk. (s.f.). Electrocardiogram (ECG). Accedido: 12 de agosto de 2024. Disponible en línea: https://www.nhs.uk/conditions/electrocardiogram/

[24] Kesto, N. M. (2013). Electrocardiography Circuit Design. Semantic Scholar. Accedido: 12 de agosto de 2024. Disponible en línea en: https://www.semanticscholar.org/paper/Electrocardiography-Circuit-Design-Kesto/d4134baa3ea1f19eb21e9dc9b9dc33ad0ba89a34

[25] Puurtinen, M., Viik, J., Takano, N., Malmivuo, J., & Hyttinen, J. (2009). Estimating the measuring sensitivity of unipolar and bipolar ECG with lead field method and FDM models. Computer Methods and Programs in Biomedicine, 94, 161-167. Disponible en línea en: https://doi.org/10.1016/j.cmpb.2008.12.005

[26] Ansari, M. Y., Qaraqe, M., Charafeddine, F., Serpedin, E., Righetti, R., & Qaraqe, K. (2023). Estimating age and gender from electrocardiogram signals: A comprehensive review of the past decade. Artificial Intelligence in Medicine, 146, 102690. Disponible en línea en: https://doi.org/10.1016/j.artmed.2023.102690

[27] Atrial Fibrillation Institute. (s.f.). Atrial Fibrillation: A Guide to Wearable ECG Smart Watches. Accedido: 13 de agosto de 2024. Disponible en línea en: https://afibinstitute.com.au/atrial-fibrillation-a-guide-to-wearable-ecg-smart-watches/

[28] Cardiovascular Education. (s.f.). Overview of the ECG Waves, Deflections, Intervals, Durations. Accedido: 13 de agosto de 2024. Disponible en línea en: https://ecgwaves.com/overview-of-the-ecg-waves-deflections-intervals-durations/

[29] Stanford Children's Health. (s.f.). Anatomy and Function of the Electrical System. Accedido: 14 de agosto de 2024. Disponible en línea en: https://www.stanfordchildrens.org/es/topic/default?id=anatomy-and-function-of-the-electrical-system-90-P04865

[30] Burns, E. (s.f.). U Wave. Life in the Fast Lane • LITFL. Accedido: 14 de agosto de 2024. Disponible en línea en: https://litfl.com/u-wave-ecg-library/

[31] González-Cervantes, N., Espinoza-Valdez, A., & Salido-Ruiz, R. (2016). Potencial Eléctrico en el Corazón: Representación Mediante un Grafo. ReCIBE Revista Electrónica de Computación, Informática Biomédica y Electrónica, 5(3). Accedido: 14 de agosto de 2024. Disponible en línea en: https://www.redalyc.org/journal/5122/512253114012/html/

[32] Almost a Doctor. (s.f.). ECG Abnormalities. Accedido: 14 de agosto de 2024. Disponible en línea en: https://almostadoctor.co.uk/encyclopedia/ecg-abnormalities

[33] Horvath, S. (2013). DNA methylation age of human tissues and cell types. Genome Biology, 14(10), 3156. Disponible en línea en: https://doi.org/10.1186/gb-2013-14-10-r115

[34] Wikipedia. (2023, 14 de noviembre). Reloj epigenético. Accedido: 16 de agosto de 2024. Disponible en línea en: https://es.wikipedia.org/w/index.php?title=Reloj_epigen%C3%A9tico&oldid=155354551

[35] Tian, Y. E., Cropley, V., Maier, A. B., Lautenschlager, N. T., Breakspear, M., & Zalesky, A. (2023). Heterogeneous aging across multiple organ systems and prediction of chronic disease and mortality. Nature Medicine, 29(5), 1221-1231. Disponible en línea en: https://doi.org/10.1038/s41591-023-02296-6

[36] Rossiello, F., Jurk, D., Passos, J. F., & d'Adda di Fagagna, F. (2022). Telomere dysfunction in ageing and age-related diseases. Nature Cell Biology, 24(2), 135-147. Disponible en línea en: https://doi.org/10.1038/s41556-022-00842-x

[37] Actiage. (s.f.). ¿El estilo de vida puede ayudarnos a revertir la edad epigenética? Accedido: 3 de diciembre de 2024. Disponible en línea en: https://actiage.es/cuerpo/el-estilo-de-vida-puede-ayudarnos-a-revertir-la-edad-epigenetica

[38] Navarro-González, R., García-Azorín, D., Guerrero-Peral, Á. L., Planchuelo-Gómez, Á., Aja-Fernández, S., & de Luis-García, R. (2023). Increased MRI-based Brain Age in chronic migraine patients. The Journal of Headache and Pain, 24(1), 133. Disponible en línea en: https://doi.org/10.1186/s10194-023-01670-6

[39] Groenewegen, K., den Ruijter, H., Pasterkamp, G., Polak, J., Bots, M., & Peters, S. A. (2016). Vascular age to determine cardiovascular disease risk: A systematic review of its concepts, definitions, and clinical applications. European Journal of Preventive Cardiology, 23(3), 264-274. Disponible en línea en: https://doi.org/10.1177/2047487314566999

[40] Keyzer, J. M., Hoffmann, J. J., Ringoir, L., Nabbe, K. C., Widdershoven, J. W., & Pop, V. J. (2014). Age- and gender-specific brain natriuretic peptide (BNP) reference ranges in primary care. Clinical Chemistry and Laboratory Medicine, 52(9), 1341-1346. Disponible en línea en: https://doi.org/10.1515/cclm-2013-0791

[41] Starc, V., et al. (2012). Can functional cardiac age be predicted from the ECG in a normal healthy population? En 2012 Computing in Cardiology, 101-104. Accedido: 6 de diciembre de 2024. Disponible en línea en: https://ieeexplore.ieee.org/abstract/document/6420340

[42] Ball, R. L., Feiveson, A. H., Schlegel, T. T., Starc, V., & Dabney, A. R. (2014). Predicting "heart age" using electrocardiography. Journal of Personalized Medicine, 4(1), 65-78. Disponible en línea en: https://doi.org/10.3390/jpm4010065

[43] Perez, M. V., Dewey, F. E., Tan, S. Y., Myers, J., & Froelicher, V. F. (2009). Added Value of a Resting ECG Neural Network That Predicts Cardiovascular Mortality. Annals of Noninvasive Electrocardiology, 14(1), 26-34. Disponible en línea en: https://doi.org/10.1111/j.1542-474X.2008.00270.x

[44] Attia, Z. I., et al. (2019). Age and Sex Estimation Using Artificial Intelligence From Standard 12-Lead ECGs. Circulation: Arrhythmia and Electrophysiology, 12(9), e007284. Disponible en línea en: https://doi.org/10.1161/CIRCEP.119.007284

[45] Ladejobi, A. O., et al. (2021). The 12-lead electrocardiogram as a biomarker of biological age. European Heart Journal - Digital Health, 2(3), 379-389. Disponible en línea en: https://doi.org/10.1093/ehjdh/ztab043

[46] Strodthoff, N., Wagner, P., Schaeffter, T., & Samek, W. (2020). Deep Learning for ECG Analysis: Benchmarks and Insights from PTB-XL. arXiv. Disponible en línea en: https://doi.org/10.48550/arXiv.2004.13701

[47] Ribeiro, A. H., et al. (2020). Automatic diagnosis of the 12-lead ECG using a deep neural network. Nature Communications, 11(1), 1760. Disponible en línea en: https://doi.org/10.1038/s41467-020-15432-4

[48] Lima, E. M., et al. (2021). Deep neural network-estimated electrocardiographic age as a mortality predictor. Nature Communications, 12(1), 5117. Disponible en línea en: https://doi.org/10.1038/s41467-021-25351-7

[49] Park, H., et al. (2024). Artificial intelligence estimated electrocardiographic age as a recurrence predictor after atrial fibrillation catheter ablation. NPJ Digital Medicine, 7(1), 1-10. Disponible en línea en: https://doi.org/10.1038/s41746-024-01234-1

[50] Lin, C.-S., et al. (2020). A Deep-Learning Algorithm (ECG12Net) for Detecting Hypokalemia and Hyperkalemia by Electrocardiography: Algorithm Development. JMIR Medical Informatics, 8(3), e15931. Disponible en línea en: https://doi.org/10.2196/15931

[51] Chang, C.-H., Lin, C.-S., Luo, Y.-S., Lee, Y.-T., & Lin, C. (2022). Electrocardiogram-Based Heart Age Estimation by a Deep Learning Model Provides More Information on the Incidence of Cardiovascular Disorders. Frontiers in Cardiovascular Medicine, 9. Disponible en línea en: https://doi.org/10.3389/fcvm.2022.754909

[52] Ansari, M. Y., Qaraqe, M., Righetti, R., Serpedin, E., & Qaraqe, K. (2024). Enhancing ECG-based heart age: impact of acquisition parameters and generalization strategies for varying signal morphologies and corruptions. Frontiers in Cardiovascular Medicine, 11, 1424585. Disponible en línea en: https://doi.org/10.3389/fcvm.2024.1424585

[53] Baek, Y.-S., Lee, D.-H., Jo, Y., Lee, S.-C., Choi, W., & Kim, D.-H. (2023). Artificial intelligence-estimated biological heart age using a 12-lead electrocardiogram predicts mortality and cardiovascular outcomes. Frontiers in Cardiovascular Medicine, 10. Disponible en línea en: https://doi.org/10.3389/fcvm.2023.1137892

[54] Holmstrom, L., & Chugh, S. S. (2023). Can Artificial Intelligence Identify Physiologically “Old” Hearts? Mayo Clinic Proceedings, 98(3), 360-362. Disponible en línea en: https://doi.org/10.1016/j.mayocp.2023.01.012

[55] Papers with Code - CODE-15% Dataset. (n.d.). Accedido: 27 de septiembre de 2024. Disponible en línea en: https://paperswithcode.com/dataset/code-15

[56] Ribeiro, A. H., et al. (2021). CODE-15%: a large scale annotated dataset of 12-lead ECGs. Disponible en línea en: https://doi.org/10.5281/zenodo.4916206

[57] Ribeiro, A. L. P., et al. (2021). Sami-Trop: 12-lead ECG traces with age and mortality annotations. Zenodo. Disponible en línea en: https://doi.org/10.5281/zenodo.4905618

[58] CODE dataset. (2021). SciLifeLab. https://doi.org/10.17044/scilifelab.15169716

[59] Figure 1: An ECG signal before (a) and after (b) the detrending operation. (n.d.). ResearchGate. Accedido: 19 de octubre de 2024. Disponible en línea en: https://www.researchgate.net/figure/An-ECG-signal-before-a-and-after-b-the-detrending-operation_fig8_306083782

[60] Papers with Code - ImageNet Dataset. Accedido: 3 de noviembre de 2024. Disponible en línea en: https://paperswithcode.com/dataset/imagenet

[61] Papers with Code - The latest in Machine Learning. Accedido: 3 de noviembre de 2024. Disponible en línea en: https://paperswithcode.com/

[62] Papers with Code - ImageNet Benchmark (Image Classification). Accedido: 3 de noviembre de 2024. Disponible en línea en: https://paperswithcode.com/sota/image-classification-on-imagenet

[63] Huang, G., Liu, Z., van der Maaten, L., & Weinberger, K. Q. (2018). Densely Connected Convolutional Networks. arXiv. Disponible en línea en: https://doi.org/10.48550/arXiv.1608.06993

[64] Densenet, PyTorch. Accedido: 3 de noviembre de 2024. Disponible en línea en: https://pytorch.org/hub/pytorch_vision_densenet/

[65] Signal Processing Toolbox. Accedido: 4 de noviembre de 2024. Disponible en línea en: https://es.mathworks.com/products/signal.html

[66] «Image Processing Toolbox». Accedido: 4 de noviembre de 2024. Disponible en línea en: https://es.mathworks.com/products/image-processing.html

[67] «PyTorch», PyTorch. Accedido: 4 de noviembre de 2024. Disponible en línea en: https://pytorch.org/

[68] «Laboratorio de procesado de imagen | Image Processing Lab». Accedido: 4 de noviembre de 2024. Disponible en línea en: https://www.lpi.tel.uva.es/

[69] Ansari, M. Y., Qaraqe, M., Charafeddine, F., Serpedin, E., Righetti, R., & Qaraqe, K. (2023). Estimating age and gender from electrocardiogram signals: A comprehensive review of the past decade. Artificial Intelligence in Medicine, 146, 102690. Disponible en línea en: https://doi.org/10.1016/j.artmed.2023.102690

Anexo

El código final del mejor modelo empleado en este trabajo está disponible públicamente en GitHub:

https://github.com/AlbertoMF2/DenseNet-for-Spectrograms

A continuación se describirán las partes más importantes de este código:

1. SELECCIÓN DE GPU

Este fragmento se coloca al inicio del código, su propósito es explorar las diferentes tarjetas gráficas que hay disponibles en el servidor, seleccionar la de menor uso y usarla en CUDA. Para ello hace uso de la herramienta NVIDIA-SMI, disponible en el servidor.

```
def get_gpu_memory_usage():
    output = subprocess.check_output(['nvidia-smi', '--query-gpu=memory.used', '--format=csv,nounits,noheader'])
    memory_usage = [int(x) for x in output.decode('utf-8').strip().split('\n')]
    return memory_usage

def get_gpu_with_least_memory():
    memory_usage = get_gpu_memory_usage()
    return memory_usage.index(min(memory_usage))

gpu_index = get_gpu_with_least_memory()
print(f"Using GPU {gpu_index} with the least memory usage.")
#Configurar el dispositivo
device = torch.device(f'cuda:{gpu_index}' if torch.cuda.is_available() else 'cpu')
```

2. PARÁMETROS DEL MODELO

Después del bloque anterior se definen los valores de diferentes hiperparámetros empleados en el modelo.

```
# Hyperparameters
learning_rate = 0.0002 #0.0005. LO BAJO PARA MAYOR SUAVIDAD
batch_size = 32 #64. LO BAJO PARA MENOR SOBREAJUSTE
num_epochs = 25 #100. LO BAJO PQ NO MEJORABA A PARTIR DE CIERTAS ÉPOCAS
weight_decay = 1e-3 #0.1 #0.01
beta1 = 0.9
beta2 = 0.999
epsilon = 1e-8
delta = 1.0
```

3. CREACIÓN DE LOS CONJUNTOS DE DATOS

Mediante el siguiente fragmento se cargan las imágenes en el código, se convierten en tensor y se normalizan. Del nombre del fichero de las imágenes saca el *target* que debe predecir el modelo, es decir, la edad.

```
class CustomDataset(torch.utils.data.Dataset):
    def __init__(self, root_dir, transform=None):
        self.root_dir = root_dir
        self.transform = transform
        self.train_image_files = []
        self.valid_image_files = []

        pattern = re.compile(r'spectrogram_no_patologia_ecg_normal_no_repe(\d+)_canal1\b')
        for subdir, _, files in os.walk(root_dir):
            subfolder_name = os.path.basename(subdir)
            match = pattern.match(subfolder_name)
            if match:
                folder_num = int(match.group(1))
                for filename in files:
                    if filename.endswith(".tiff") and os.path.isfile(os.path.join(subdir, filename)
                        file_path = os.path.join(subdir, filename)
                        if 0 <= folder_num <= 11:
                            self.train_image_files.append(file_path)
                        elif 12 <= folder_num <= 13:
                            self.valid_image_files.append(file_path)

        print(f"Total de imágenes de entrenamiento: {len(self.train_image_files)}")
        print(f"Total de imágenes de validación: {len(self.valid_image_files)}")

        self.train_targets = [self.extract_age(filename) for filename in self.train_image_files]
        self.valid_targets = [self.extract_age(filename) for filename in self.valid_image_files]

        if self.transform is None:
            self.calculate_global_stats()

        # Variable para controlar si ya hemos impreso el primer archivo
        self.first_file_printed = False


    def __len__(self):
        return len(self.train_image_files) + len(self.valid_image_files)

    def __getitem__(self, idx):
        if idx < len(self.train_image_files):
            img_path = self.train_image_files[idx]
            target = self.train_targets[idx]
```

En este mismo bloque se hace la división entre los conjuntos de entrenamiento y validación.

Justo después se instancia la clase y se crean los conjuntos de datos:

```
dataset = CustomDataset(root_dir="/datos/work/ECG/CODE15/espectrogramas")
train_dataset = torch.utils.data.Subset(dataset,
range(len(dataset.train_image_files)))
valid_dataset = torch.utils.data.Subset(dataset,
range(len(dataset.train_image_files), len(dataset)))
train_loader = DataLoader(train_dataset, batch_size=batch_size, shuffle=True,
num_workers=4)
valid_loader = DataLoader(valid_dataset, batch_size=batch_size, shuffle=False,
num_workers=4)
print(f"Size of train_loader: {len(train_loader.dataset)}")
print(f"Size of valid_loader: {len(valid_loader.dataset)}")
```

4. CREACIÓN DE LA RED

La siguiente parte del código es el montaje de la nueva red, para ello se emplea una red DenseNet121, disponible en las librerías importadas. Esta red está diseñada para tareas de clasificación por lo que tiene varias salidas, así que sustituye la última capa por un bloque creado especialmente para tareas de regresión.

```
def modify_densenet_for_regression(model):
    model.features.conv0 = nn.Conv2d(1, 64, kernel_size=7, stride=2,
padding=3, bias=False)
    # Get the number of features in the last layer
    num_features = model.classifier.in_features

    # Create a new classifier
    new_classifier = nn.Sequential(
        nn.Dropout(0.5),
        nn.Linear(num_features, 512),
        nn.BatchNorm1d(512),
        nn.ReLU(),
        nn.Dropout(0.5),
        nn.Linear(512, 256),
        nn.BatchNorm1d(256),
        nn.ReLU(),
        nn.Dropout(0.5),
        nn.Linear(256, 1),
        nn.Sigmoid()
    )

    # Replace the classifier
    model.classifier = new_classifier

    return model

# Load the pre-trained DenseNet121 model
densenet121 = models.densenet121(weights='DEFAULT')

# Modify the model for regression
densenet121 = modify_densenet_for_regression(densenet121)
```

5. DEFINICIÓN DE HIPERPARÁMETROS

En esta sección del código se define que función de pérdida, optimizador y programador de tasa de aprendizaje usará el programa.

```
# Define la función de pérdida
criterion = nn.HuberLoss(delta=delta)

# Define el optimizador AdamW
optimizer = optim.AdamW(densenet121.parameters(), lr=learning_rate,
weight_decay=weight_decay, betas=(beta1, beta2), eps=epsilon)

# Define el programador de tasa de aprendizaje
scheduler = optim.lr_scheduler.ReduceLROnPlateau(optimizer,
mode='min', factor=0.1, patience=5, verbose=True)
```

6. ETAPAS DE ENTRENAMIENTO Y VALIDACIÓN

En el siguiente código se define lo que sucederá en cada época, en la que pasa los datos al modelo y lo entrena. Usa una barra de progreso para poder ver las métricas según sucede el entrenamiento. En la sección 3.5.1 del documento se explican diferentes herramientas empleadas en este apartado.

```
for epoch in range(num_epochs):
    epoch_train_loss = 0.0
    epoch_valid_loss = 0.0

    # ENTRENAMIENTO

    densenet121.train()

    # Listas para ver alguna predicción y su target real
    train_preds = []
    train_targs = []
    train_files = []

    with tqdm(total=len(train_loader), desc=f'Epoch {epoch + 1}/{num_epochs}', unit='batch') as pbar:
        for batch_idx, (data, targets, img_names) in enumerate(train_loader):
            data = data.to(device)
            targets = targets.unsqueeze(1).float().to(device)

            optimizer.zero_grad()

            with torch.cuda.amp.autocast():
                # Forward pass
                scores = 10 + 85*densenet121(data)
                loss = criterion(scores, targets)

            scaler.scale(loss).backward()

            torch.nn.utils.clip_grad_norm_(densenet121.parameters(), max_norm=1.0)

            scaler.step(optimizer)
            scaler.update()

            epoch_train_loss += loss.item()

            # Update the training progress bar
            pbar.set_postfix({'Train Loss': epoch_train_loss / (batch_idx + 1)}, update=True)  # Update the me
progress bar
            pbar.update(1)  # Increment the progress counter

            train_preds.extend(scores.detach().cpu().numpy())
            train_targs.extend(targets.cpu().numpy())
```

Justo después del fragmento anterior se define la etapa de validación.

```
densenet121.eval()

    # Listas para ver alguna predicción y su target real
    valid_preds = []
    valid_targs = []
    valid_files = []

    with torch.no_grad():
        for data, targets, img_names in valid_loader:
            data = data.to(device=device)
            targets = targets.unsqueeze(1).float().to(device)

            # Forward pass
            scores = 10 + 85*densenet121(data)
            loss = criterion(scores, targets)

            epoch_valid_loss += loss.item()

            valid_preds.extend(scores.cpu().numpy())
            valid_targs.extend(targets.cpu().numpy())
            valid_files.extend(img_names)

    # Calcular la media entre las predicciones y los targets reales para validación
    valid_mean_diff = np.mean(np.abs(np.array(valid_preds) - np.array(valid_targs)))
```

La diferencia principal con respecto a la etapa de entramiento es que esta vez los gradientes no se van a modificar gracias a llamar a método *eval()*.

En la misma época se almacenan las pérdidas según la función de pérdida y el error real (MAE) de ambas etapas, además se actualiza la tasa de aprendizaje.

Finalmente se compara si el error obtenido en la época es mejor que el obtenido en las anteriores épocas y si se da el caso se guardan los pesos del modelo en la época actual.

```
print(f"Epoch [{epoch + 1}/{num_epochs}], Train Mean Diff:
{train_mean_diff:.2f}")
    print(f"Epoch [{epoch + 1}/{num_epochs}], Valid Mean Diff:
{valid_mean_diff:.2f}")

    # Almacena la pérdida de entrenamiento y de validación de la época
actual
    train_losses.append(epoch_train_loss / len(train_loader))
    valid_losses.append(epoch_valid_loss / len(valid_loader))

    print(f'Epoch [{epoch + 1}/{num_epochs}], Train Loss:
{train_losses[-1]:.4f}')
    print(f'Epoch [{epoch + 1}/{num_epochs}], Validation Loss:
{valid_losses[-1]:.4f}')

    scheduler.step(valid_losses[-1])

     # Guardar el mejor modelo
    if epoch_valid_loss / len(valid_loader) < best_valid_loss:
        best_valid_loss = epoch_valid_loss / len(valid_loader)
        torch.save(densenet121.state_dict(), best_model_path)
        print(f"New best model saved with validation loss:
{best_valid_loss}")

        # Guardar el nombre de la imagen, las predicciones y los
targets de la mejor época
        best_train_preds = train_preds
        best_train_targs = train_targs
        best_train_files = train_files
        best_valid_preds = valid_preds
        best_valid_targs = valid_targs
        best_valid_files = valid_files
```

7. EVALUACIÓN DE LOS RESULTADOS

El siguiente bloque de código calcula las métricas de interés para la evaluación de los resultados que más tarde se imprimirán en un documento.

```
def evaluate_model(loader, model, criterion, device):
    model.eval()
    total_loss = 0.0
    total_samples = 0

    true_ages = []
    predicted_ages = []

    with torch.no_grad():
        for data, targets, img_names in loader:
            data = data.to(device)
            targets = targets.unsqueeze(1).float().to(device)
            true_ages.extend(targets.cpu().detach().numpy())

            scores = 10 + 85 * model(data)
            loss = criterion(scores, targets)

            total_loss += loss.item() * data.size(0)
            total_samples += data.size(0)

            predicted_ages.extend(scores.cpu().numpy().flatten())

    total_loss /= total_samples

    true_ages = np.array(true_ages).squeeze()
    predicted_ages = np.array(predicted_ages).squeeze()

    mae = mean_absolute_error(true_ages, predicted_ages)
    mse = mean_squared_error(true_ages, predicted_ages)
    r, _ = pearsonr(true_ages, predicted_ages)
    R2 = r2_score(true_ages, predicted_ages)

    return total_loss, mae, mse, r, R2, predicted_ages, true_ages

train_loss, train_mae, train_mse, train_r, train_R2,
train_predictions, train_targets  = evaluate_model(train_loader,
densenet121, criterion, device)
valid_loss, valid_mae, valid_mse, valid_r, valid_R2,
valid_predictions, valid_targets  = evaluate_model(valid_loader,
densenet121, criterion, device)
```

El código termina con la generación de los ficheros de salida y las imágenes que permiten observar la pérdida a través de todas las épocas.