
**Predicción de Riesgo Cardiovascular Explicable
mediante *Deep Learning***
**Explainable Cardiovascular Risk Prediction
using Deep Learning**



**Trabajo de Fin de Grado
Curso 2024–2025**

Autor
Noelia Barranco Godoy

Directores
Belén Díaz Agudo
Juan A. Recio García
María Ángeles Díaz Vicente

Doble Grado en Ingeniería Informática y Matemáticas
Facultad de Informática
Universidad Complutense de Madrid

Predicción de Riesgo Cardiovascular
Explicable mediante *Deep Learning*
Explainable Cardiovascular Risk
Prediction using Deep Learning

Trabajo de Fin de Grado en Ingeniería Informática

Autor
Noelia Barranco Godoy

Director
Belén Díaz Agudo
Juan A. Recio García
María Ángeles Díaz Vicente

Convocatoria: Febrero 2025

Doble Grado en Ingeniería Informática y Matemáticas
Facultad de Informática
Universidad Complutense de Madrid

21 de enero de 2025

Dedicatoria

A mi pareja, Ian, por ser mi refugio en cada instante de agobio y recordarme, con su presencia y cariño, que ninguna meta es inalcanzable cuando no se afronta sola.

Agradecimientos

Deseo expresar mi más sincero reconocimiento a los directores de mi TFG. A Juan Antonio por sus consejos a lo largo del desarrollo del trabajo, a Belén, por su constante apoyo y por haberme presentado a Marian, cuya presencia, más allá de los aspectos estrictamente académicos, fue esencial a nivel profesional y personal, ofreciéndome una dedicación, cercanía y orientación que marcaron profundamente mi proceso formativo. Estas atenciones, que superaron con creces el ámbito académico, han dejado una huella imborrable en mi trayectoria.

Asimismo, quiero agradecer también al profesor Mauro Buelga, del departamento de Cardiología del Hospital Universitario Ramón y Cajal y profesor de Enfermería de la Universidad de Alcalá de Henares, así como al doctor Gonzalo Alonso, del departamento de Cardiología del Hospital Universitario de Navarra, por haber revisado y valorado mi trabajo, brindando una valiosa validación experta. Sus observaciones han sido fundamentales para mejorar y afianzar la calidad de este proyecto.

También dentro del ámbito académico, no puedo no mencionar a los profesores del departamento de Arquitectura de Computadores y Automática de nuestra facultad, Joaquín Recas y Luis Piñel, que me han concedido acceso a una máquina de la facultad, y me han ayudado con la solución de problemas técnicos cuando ha sido necesario.

Por último, agradezco profundamente a Sergio González Cabeza, doctorando del departamento de Computadores y automática de nuestra facultad y Mario Sanz Guerrero, doctorando en la Universidad Johannes Gutenberg (Mainz, Alemania), cuyos trabajos relacionados y sus consejos han sido de gran ayuda en este trabajo.

Resumen

Predicción de Riesgo Cardiovascular Explicable mediante *Deep Learning*

Las enfermedades cardiovasculares son una de las principales causas de muerte según la OMS, para combatir esto, una de las mejores herramientas diagnósticas son los electrocardiogramas (ECGs). Recientemente se han desarrollado algunos modelos *Deep Learning* que permiten clasificar de manera bastante eficiente los ECGs, pero los médicos se han mostrado reacios a utilizarlos debido a que estos modelos no suelen ser explicables.

En este trabajo se modifica uno de los modelos más conocidos en la literatura (Ribeiro et al., 2020) para la predicción y clasificación de anomalías cardíacas a partir de señales de ECGs, integrando técnicas de explicabilidad que permiten justificar las decisiones del modelo. Utilizando la base de datos pública PTB-XL, se implementaron transformaciones de señal, como la Transformada de Fourier de Tiempo Reducido (STFT) y la Transformada de Onda Continua (CWT), con el objetivo de extraer características de tiempo y frecuencia para mejorar las predicciones.

Los resultados obtenidos indican que el modelo base, logra un desempeño destacado en la clasificación multietiqueta de anomalías cardíacas. Sin embargo, las transformaciones de señal presentan limitaciones cuando se utilizan con esta arquitectura. Para abordar la necesidad de transparencia en el ámbito médico, se emplearon *saliency maps*, validados por expertos médicos, quienes destacaron su potencial educativo y clínico, señalando áreas de mejora para futuras aplicaciones.

El estudio concluye que la combinación de técnicas de *Deep Learning* y métodos de explicabilidad puede contribuir significativamente a la adopción de herramientas de IA en entornos médicos, optimizando el diagnóstico temprano y apoyando la formación de profesionales de la salud.

Palabras clave

Deep Learning, explicabilidad, ECG, riesgo cardiovascular, STFT, CWT, redes neuronales convolucionales, clasificación multietiqueta.

Abstract

Explainable Cardiovascular Risk Prediction using Deep Learning

Cardiovascular diseases are one of the leading causes of death according to the WHO. The electrocardiogram (ECG) is one of the most effective diagnostic tools to address this issue. Recently, some *Deep Learning* models have been developed to efficiently classify ECG's, but medical professionals have been reluctant to adopt them due to the lack of explainability in these models.

In this study, one of the most well-known models in the literature (Ribeiro et al., 2020) for predicting and classifying cardiac anomalies from electrocardiogram (ECG) signals is modified by integrating explainability techniques to justify the model's decisions. Using the publicly available PTB-XL database, signal transformations such as the Short-Time Fourier Transform (STFT) and Continuous Wavelet Transform (CWT) were implemented to extract time and frequency features with the goal of improving predictions.

The results indicate that the base model achieves outstanding performance in the multi-label classification of cardiac anomalies. However, signal transformations demonstrate limitations when used with this architecture. To address the need for transparency in the medical field, *saliency maps* were employed and validated by medical experts, who highlighted their educational and clinical potential while identifying areas for improvement in future applications.

The study concludes that the combination of *Deep Learning* techniques and explainability methods can significantly contribute to the adoption of AI tools in medical settings, enhancing early diagnosis and supporting the education of healthcare professionals.

Keywords

Deep Learning, explainability, ECG, cardiovascular risk, STFT, CWT, convolutional neural networks, multilabel classification.

Índice

1	Introducción	1
1.1	Fundamentos del electrocardiograma	1
1.1.1	Derivaciones	2
1.1.2	Ondas y segmentos principales	4
1.1.3	Principales anomalías en un ECG	5
1.2	Motivación	6
1.3	Objetivos	7
1.4	Estructura del documento	7
2	Estado del Arte	9
2.1	Bases de datos	9
2.1.1	Bases de datos disponibles	9
2.1.2	PTB-XL	10
2.2	Modelos para la clasificación de ECG	13
2.2.1	Enfoques basados en técnicas tradicionales de aprendizaje automático	13
2.2.2	Modelos basados en redes neuronales	13
2.2.3	El modelo de Ribeiro	14
2.3	Explicabilidad en la IA	16
3	Metodología y Preparación de Datos	17
3.1	Análisis de los datos y distribución por clases	17
3.2	Preprocesamiento de datos	18
3.3	Métricas de evaluación	19
3.3.1	Métricas habituales	19

3.3.2	Métricas en clasificadores multietiqueta	20
3.3.3	Métricas para nuestro problema	21
3.4	Transformaciones	22
3.4.1	STFT	22
3.4.2	CWT	24
4	Entrenamiento y Resultados	33
4.1	Modelos modificados entrenados	33
4.2	Configuración del entrenamiento	34
4.2.1	<i>Hardware y software</i>	34
4.2.2	Parámetros de entrenamiento	34
4.3	Resultados cuantitativos	35
4.4	Análisis de los resultados	35
5	Explicabilidad y Validación Clínica	37
5.1	Método de explicabilidad utilizado	37
5.2	Resultados de explicabilidad	37
5.3	Validación con los expertos médicos	38
5.4	Modificaciones a partir del feedback	38
5.4.1	NORM	40
5.4.2	MI	40
5.4.3	STTC	41
5.4.4	CD	42
5.5	Comparación con otros resultados publicados	42
6	Conclusiones y Trabajo Futuro	45
6.1	Conclusiones	45
6.1.1	Conclusiones generales	45
6.1.2	Conclusiones sobre los modelos entrenados	46
6.1.3	Conclusiones sobre explicabilidad de los modelos	46
6.2	Trabajo futuro	47
6.3	Principales contribuciones personales	47
Introduction		49
6.4	Fundamentals of the Electrocardiogram	49
6.4.1	Leads	50

6.4.2	Main Waves and Segments	51
6.4.3	Main Anomalies in an ECG	53
6.5	Motivation	53
6.6	Objectives	54
6.7	Document Structure	55
7	Conclusions and Future Work	57
7.1	Conclusions	57
7.1.1	General Conclusions	57
7.1.2	Conclusions on the Trained Models	58
7.1.3	Conclusions on Model Explainability	58
7.2	Future Work	58
7.3	Main Personal Contributions	59
Bibliografía		61
A Código		65
B Librerías utilizadas		73
Lista de acrónimos		79

Índice de figuras

1.1	ECG de 12 derivaciones tomado a 100Hz (arriba) y 500Hz (abajo).	2
1.2	Interpretación del significado de las derivaciones de un electrocardiograma. Fuente: Wikimedia Commons (2015).	3
1.3	Ejemplo de ECG con etiquetas en P, QRS, T, ST y PR. Fuente: Wikimedia Commons (2023).	4
2.1	ECGs de todos los tipos de anomalías.	12
2.2	Arquitectura de la red neuronal que se emplea en el <i>gold standard</i> . Fuente: Ribeiro et al. (2020)	15
3.1	Transformadas STFT de la primera derivación de ECGs para cada clase.	25
3.1	Transformadas STFT de la primera derivación de ECGs para cada clase.	26
3.1	Transformadas STFT de la primera derivación de ECGs para cada clase.	27
3.2	Transformadas CWT de la primera derivación de ECGs para cada clase.	28
3.2	Transformadas CWT de la primera derivación de ECGs para cada clase.	29
3.2	Transformadas CWT de la primera derivación de ECGs para cada clase.	30
3.2	Transformadas CWT de la primera derivación de ECGs para cada clase.	31
3.2	Transformadas CWT de la primera derivación de ECGs para cada clase.	32
5.1	Explicaciones de la primera derivación de un ECG de las clases NORM, MI, STTC, CD (respectivamente)	39

5.2	Explicación de un ECG de clase NORM pintada sobre la librería <i>ecg_plot</i>	40
5.3	Explicación de un ECG de clase MI pintada sobre la librería <i>ecg_plot</i>	41
5.4	Explicación de un ECG de clase STTC pintada sobre la librería <i>ecg_plot</i>	41
5.5	Explicación de un ECG de clase CD pintada sobre la librería <i>ecg_plot</i>	42
5.6	Explicación de un infarto de miocardio en nuestro modelo (arriba) y en un artículo de investigación (abajo). Fuente: Gustafsson et al. (2022).	43
6.1	12-lead ECG recorded at 100Hz (top) and 500Hz (bottom).	50
6.2	Interpretation of the significance of electrocardiogram leads. Source: Wikimedia Commons (2015).	51
6.3	Example of an ECG with labeled P, QRS, T, ST, and PR segments. Source: Wikimedia Commons (2023)	52

Índice de tablas

3.1 Distribución de las superclases en PTB-XL	18
4.1 Resultados de los modelos (con tres dígitos decimales de precisión) . .	35

Capítulo 1

Introducción

RESUMEN: En este capítulo abordaremos la relevancia de la IA en el ámbito médico, describiremos los objetivos y el alcance del trabajo, y presentaremos la estructura general del documento.

1.1. Fundamentos del electrocardiograma

Un electrocardiograma (ECG) es una prueba diagnóstica no invasiva que registra la actividad eléctrica del corazón a lo largo del tiempo. Mediante la colocación de electrodos en puntos específicos del cuerpo, se captan variaciones en los potenciales eléctricos producidos por el músculo cardíaco (Ribeiro et al., 2020). Estas variaciones se reflejan en la forma de ondas y complejos (P, QRS, T), cuya interpretación permite identificar diversas patologías. Aunque estas mediciones pueden hacerse de diversas formas, lo habitual es medir la diferencia de potencial entre doce pares de electrodos. Cada una de estas mediciones genera una señal, que recibe el nombre de derivación. Podemos decir que las derivaciones nos dan una vista del funcionamiento del corazón desde distintos ángulos.

Dado que no es posible recoger una cantidad continua de mediciones en el tiempo, se toma una cantidad de mediciones finita a una frecuencia específica (usualmente de 100Hz a 500Hz, dependiendo de la precisión del aparato medidor). Por tanto un ECG puede almacenarse en doce vectores de igual tamaño (variable dependiendo de la duración y frecuencia de la prueba). En la Figura 1.1 podemos ver un ejemplo de electrocardiograma (extraido de Wagner et al. (2022) y dibujado mediante la librería de python *ecg_plot*).

La frecuencia a la que se mide un ECG es muy importante. Si tiene una mayor frecuencia, la onda se puede ver con una mayor resolución, lo que permite detectar mejor las posibles anomalías. No obstante, además de que un aparato con capacidad de medir a más resolución es más complejo y caro, hay otras desventajas de tener una frecuencia muy alta. Una mayor frecuencia implica que para almacenar un ECG



Figura 1.1: ECG de 12 derivaciones tomado a 100Hz (arriba) y 500Hz (abajo).

de la misma duración necesitamos más memoria, y será computacionalmente más costoso cualquier tipo de procesamiento o análisis que queramos realizar sobre este.

La relevancia clínica del ECG es enorme, además de ser una prueba de bajo coste y gran disponibilidad, el electrocardiograma constituye el primer paso para la detección de anomalías cardíacas en servicios de urgencias, consultas de atención primaria y especializadas. De acuerdo con la Organización Mundial de la Salud (OMS), las enfermedades cardiovasculares representan una de las principales causas de mortalidad a nivel global (World Health Organization, 2021); por ello, optimizar su diagnóstico temprano a través de métodos automatizados de análisis y clasificación puede tener un impacto significativo en la mejora de la salud pública.

En este contexto, se han desarrollado modelos de *Deep Learning* muy precisos para el diagnóstico interpretable de anomalías cardíacas (Lu et al., 2024). A lo largo de este capítulo se profundizará en aspectos esenciales de la señal, se expondrán los fundamentos de las ondas principales y se describirán las bases de datos de ECGs más utilizadas para la investigación.

1.1.1. Derivaciones

Un electrocardiograma (ECG) es una herramienta que registra la actividad eléctrica del corazón desde diferentes perspectivas, denominadas derivaciones. Cada de-

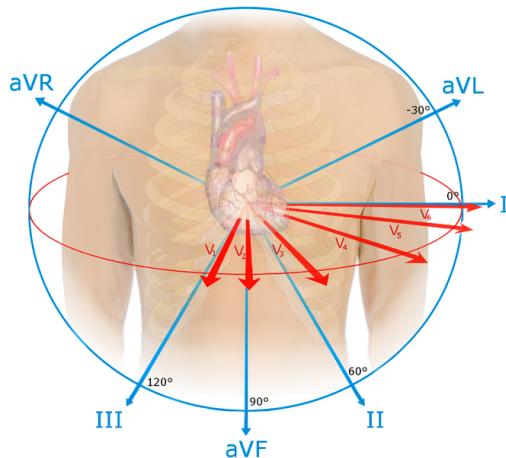


Figura 1.2: Interpretación del significado de las derivaciones de un electrocardiograma. Fuente: Wikimedia Commons (2015).

rivación ofrece una “vista” específica del corazón, facilitando la identificación de posibles anomalías en su funcionamiento.

Para obtener estas derivaciones, se colocan electrodos en puntos estratégicos del cuerpo. Aunque se utilizan 10 electrodos, el ECG estándar proporciona 12 derivaciones, divididas en dos grupos principales:

1. Derivaciones de las extremidades (plano frontal):

a) Derivaciones bipolares estándar:

- DI (I): Mide la diferencia de potencial eléctrico entre el brazo derecho y el brazo izquierdo.
- DII (II): Registra la diferencia entre el brazo derecho y la pierna izquierda.
- DIII (III): Capta la diferencia entre el brazo izquierdo y la pierna izquierda.

b) Derivaciones unipolares aumentadas:

- aVR: Observa el potencial del brazo derecho.
- aVL: Enfoca el potencial del brazo izquierdo.
- aVF: Se centra en el potencial de la pierna izquierda.

2. Derivaciones precordiales (plano horizontal):

- V1 a V6: Estos electrodos se colocan en posiciones específicas del tórax, proporcionando vistas detalladas de diferentes áreas del corazón.

Cada derivación ofrece una perspectiva única de la actividad eléctrica cardíaca (como podemos ver en la Figura 1.2), permitiendo a los profesionales de la salud evaluar de manera integral el estado del corazón y detectar posibles irregularidades en su funcionamiento.

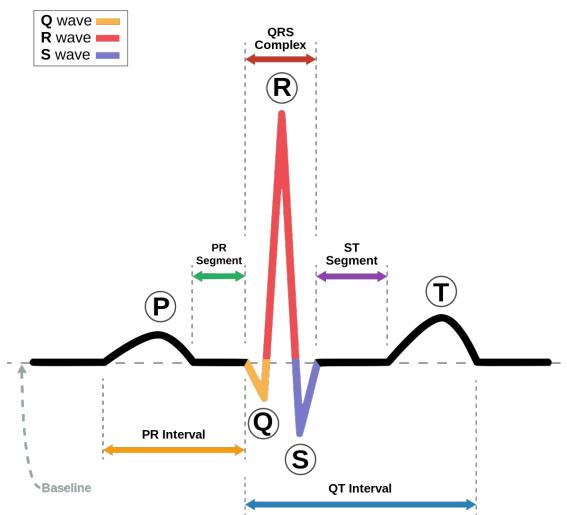


Figura 1.3: Ejemplo de ECG con etiquetas en P, QRS, T, ST y PR. Fuente: Wikimedia Commons (2023).

1.1.2. Ondas y segmentos principales

Cada una de las derivaciones del ECG está compuesta por una serie de ondas y segmentos que reflejan las distintas fases de la actividad cardíaca (MyEKG). En la Figura 1.3 se muestra un trazado típico de un ECG con etiquetas para cada onda y segmento.

1.1.2.1. Onda P

Representa la despolarización de las aurículas, es decir, la activación eléctrica que provoca su contracción. En un ECG normal, la onda P es positiva en la mayoría de las derivaciones, excepto en aVR, donde suele ser negativa. Su duración normal es menor de 0.10 segundos y su amplitud no supera los 2.5 milímetros.

1.1.2.2. Intervalo PR

Es el tiempo que transcurre desde el inicio de la onda P hasta el comienzo del complejo QRS. Este intervalo refleja el tiempo que tarda el impulso eléctrico en viajar desde las aurículas hasta los ventrículos, pasando por el nodo auriculoventricular. Su duración normal está entre 0.12 y 0.20 segundos.

1.1.2.3. Complejo QRS

Indica la despolarización de los ventrículos, lo que desencadena su contracción. Se compone de tres ondas:

- Onda Q: Primera deflexión negativa del complejo. No siempre está presente y, cuando lo está, suele ser de pequeña amplitud y duración.
- Onda R: Primera deflexión positiva del complejo. Es generalmente la onda de mayor amplitud.
- Onda S: Deflexión negativa que sigue a la onda R.

La duración total del complejo QRS es normalmente menor de 0.10 segundos.

1.1.2.4. Segmento ST

Es el tramo que va desde el final del complejo QRS hasta el inicio de la onda T. Representa el periodo en que los ventrículos están completamente despolarizados antes de comenzar su repolarización. En condiciones normales, este segmento es isoelectrónico, es decir, se mantiene en la línea base del ECG.

1.1.2.5. Onda T

Refleja la repolarización de los ventrículos, proceso que prepara al corazón para el siguiente ciclo de contracción. La onda T suele ser positiva en la mayoría de las derivaciones, excepto en aVR, donde es negativa. Su forma es asimétrica, con una ascensión más lenta y un descenso más rápido.

1.1.2.6. Intervalo QT

Comprende desde el inicio del complejo QRS hasta el final de la onda T. Este intervalo representa el tiempo total de despolarización y repolarización de los ventrículos. Su duración varía según la frecuencia cardíaca, pero generalmente se considera normal un QT entre 340 y 440 milisegundos en adultos jóvenes.

La correcta interpretación de estas ondas y segmentos es fundamental para evaluar la función cardíaca y detectar posibles anomalías en la conducción eléctrica del corazón.

1.1.3. Principales anomalías en un ECG

Existen numerosas anomalías en un ECG que pueden reflejar alteraciones en la función cardíaca. En este trabajo, agruparemos las anomalías de la misma forma en la que están agrupadas en la base de datos elegida, que podemos ver en la Sección 2.1.2.2.

1.2. Motivación

La Inteligencia Artificial (IA) ha transformado numerosos campos. En particular, la medicina ha experimentado avances significativos mediante la aplicación de la inteligencia artificial al diagnóstico y análisis de datos. Uno de los campos más prometedores es el de la interpretación automatizada de señales en un ECG, fundamentales para la detección temprana de enfermedades cardíacas. Las enfermedades cardiovasculares son una de las principales causas de muerte a nivel global (World Health Organization, 2021), lo que subraya la necesidad de métodos rápidos y precisos para su diagnóstico.

El análisis de ECGs ha sido históricamente una tarea realizada por profesionales de la salud, como cardiólogos, debido a la complejidad y variabilidad de las señales y de la interpretación de éstas. Sin embargo, este proceso es lento, costoso y depende en gran medida de la experiencia del especialista. Con la adopción de modelos de *Deep Learning*, como las redes neuronales convolucionales (CNN), surge la oportunidad de automatizar y mejorar este análisis, proporcionando resultados rápidos, y en muchos casos, comparables a los obtenidos por expertos humanos (Hannun et al., 2019). Este enfoque no solo promete aumentar la eficiencia del diagnóstico, sino también mejorar la precisión y reducir la carga de trabajo del personal médico.

Sin embargo, los médicos se han mostrado reacios a utilizar estas tecnologías debido a que en muchas ocasiones, por ser modelos predictivos que no proporcionan explicaciones (modelos de caja negra), no pueden comprender su funcionamiento. Para que este tipo de herramientas genere confianza en entornos clínicos y cumpla con los estándares de calidad y ética, es fundamental que sean explicables (Molnar, 2019). La explicabilidad de los modelos de IA hace posible comprender y justificar las decisiones tomadas durante el análisis de la señal cardíaca, un factor crítico en aplicaciones de alto impacto como la medicina, donde un error puede afectar directamente a la salud de los pacientes (Goodman y Flaxman, 2017).

Si bien se ha realizado una cantidad significativa de investigación en el campo de análisis y predicción de riesgos a partir de ECGs (por ejemplo Ribeiro et al. (2020)), no son tantos los trabajos que hay que ponen el foco en tener modelos explicables, lo que es muy importante (como señalábamos en el párrafo anterior) para que estos modelos sean realmente utilizados. La explicabilidad de modelos de clasificación de series temporales, como son las distintas derivaciones de un electrocardiograma, es un campo todavía en investigación. Sin embargo, se han producido más avances en la explicación basada por imágenes. Por este motivo, en este trabajo se analizarán algunas transformaciones de los datos de electrocardiogramas al dominio del tiempo/frecuencia con dos fines:

- Analizar si mejora la precisión de la clasificación al tener más características de la señal, tanto en el dominio del tiempo (ecg normal) como de la frecuencia.
- Representar estas transformaciones como imágenes para ver si así mejora su explicabilidad.

A lo largo de este trabajo, exploraremos diversas arquitecturas de redes neuro-

nales y técnicas de transformación de señales para clasificar y detectar anomalías en ECGs, con el objetivo de desarrollar y mejorar un modelo clasificador de ECGs en cuatro grupos de anomalías cardíacas. Además, incorporaremos métodos de explicabilidad para que los resultados del modelo puedan ser interpretados y validados por profesionales de la salud, lo que es esencial para la adopción clínica de estas tecnologías.

1.3. Objetivos

El principal objetivo de este trabajo es modificar, entrenar y evaluar el modelo propuesto originalmente por Ribeiro et al. (2020) (en adelante, '*gold standard*'), aplicándolo sobre una base de datos pública. Para lograrlo, introduciremos una modificación en la capa de entrada del modelo que permita la inclusión de transformadas matemáticas de las señales que permitan extraer características no solo en el dominio del tiempo, sino además en el de la frecuencia (por ejemplo STFT o CWT) a fin de explorar si la conversión de la señal en distintas representaciones puede mejorar el rendimiento. No obstante, no se realizan cambios en la arquitectura interna del modelo.

Con base en lo anterior, los objetivos específicos son:

1. **Entrenar el *gold standard*** con una base de datos pública.
2. **Modificar el *gold standard*** para que admita transformaciones de la señal, posibilitando el entrenamiento de versiones alternativas del modelo con datos transformados.
3. **Comparar el rendimiento** de cada variante con el *gold standard* mediante métricas como F1-Score, analizando si las transformaciones resultan o no beneficiosas.
4. **Aplicar un método de explicabilidad** para que un especialista pueda comprender qué partes de la señal de un ECG influyen en la predicción.

1.4. Estructura del documento

El presente documento se organiza en seis capítulos, cuyo contenido se describe a continuación:

- **Capítulo 2: Estado del Arte**

Se exponen los conceptos básicos sobre electrocardiogramas, resaltando la importancia de sus ondas (P, QRS, T) y las anomalías más comunes que se suelen detectar. Asimismo, se revisa la literatura relacionada con el uso de redes neuronales en el análisis de ECGs y se describen bases de datos relevantes (como PTB-XL).

- **Capítulo 3: Metodología y Preparación de Datos**

Se detalla el proceso de tratamiento de los ECGs, incluyendo la división en conjuntos de entrenamiento, validación y prueba. Además, se describen las métricas utilizadas para evaluar los modelos y las transformaciones empleadas.

- **Capítulo 4: Entrenamiento y Resultados**

Se explican las condiciones de entrenamiento de cada modelo (incluyendo las modificaciones del *gold standard*) y las librerías utilizadas y se presentan y comparan los resultados cuantitativos obtenidos por las métricas.

- **Capítulo 5: Explicabilidad y Validación Clínica**

Se describe el método de explicabilidad basado en *saliency maps* de gradientes aplicado al modelo de Ribeiro. Asimismo, se incluye la perspectiva de un médico especialista que analiza las explicaciones generadas y se reflexiona sobre el método y posibles mejoras.

- **Capítulo 6: Conclusiones y Trabajo Futuro**

Se integran las conclusiones derivadas de los resultados, valorando en qué medida se han cumplido los objetivos planteados. Además, se señalan las principales contribuciones de este trabajo y se proponen líneas de investigación futuras (como la inclusión de capas Conv2D).

Por último, se incluye un apartado de bibliografía, donde se recogen todas las fuentes consultadas a lo largo del documento, así como un anexo con el código utilizado¹ y otro con la totalidad de librerías instaladas en el entorno de *Python* sobre el que se ejecutó el código.

¹Todo el contenido de este trabajo puede encontrarse en este repositorio de github: <https://github.com/NotNoe/TFG-Info>.

Capítulo 2

Estado del Arte

RESUMEN: En este capítulo revisaremos las principales bases de datos disponibles y examinaremos los métodos de *Deep Learning* más relevantes para su análisis, destacando el modelo de Ribeiro et al. (2020) como punto de referencia en el estado del arte. Por último, revisaremos el significado de explicabilidad en el ámbito de la IA y mencionaremos algunos trabajos que tratan temas relacionados con este.

2.1. Bases de datos

Como ocurre con los sistemas de IA basados en datos, para entrenar modelos que permitan analizar automáticamente las señales de un ECG es esencial la disponibilidad de una gran cantidad de ECGs. En esta sección se revisan brevemente algunas colecciones de ECGs ampliamente utilizadas en el ámbito de la investigación y se presentan en detalle las características de PTB-XL (Wagner et al., 2022), la base de datos elegida para la realización de este trabajo.

2.1.1. Bases de datos disponibles

Como queremos comparar el rendimiento de varias modificaciones de un modelo ya existente, lo ideal sería poder trabajar con la misma base de datos con la que se entrenó el modelo de Ribeiro. Esta base de datos es CODE (Ribeiro et al., 2021b), pero no es pública, por lo que esto no es una opción.

Existe una versión pública reducida de esta base de datos llamada CODE-15 (Ribeiro et al., 2021a), que tiene alrededor de 350000 casos. No obstante, no podemos comparar un modelo entrenado con los datos de CODE-15 con otro entrenado con los de CODE, porque la ventaja en el conjunto de datos de entrenamiento haría imposible comparar la eficiencia de manera equitativa. Por ello, vamos a volver a entrenar el *gold standard* con una base de datos pública, PTB-XL (Wagner et al.,

2022). Esta base de datos cuenta con unos 22000 casos de prueba, pero todos estos casos están con el mismo formato, por lo que el procesamiento de los mismos será más sencillo. Adicionalmente, dado que el objetivo es comparar distintas modificaciones del modelo para evaluar si mejora el rendimiento, es mucho más importante emplear bases de datos uniformes y poder realizar más entrenamientos que usar una base de datos más grande, lo que incrementaría significativamente el tiempo de entrenamiento.

2.1.2. PTB-XL

La base de datos PTB-XL es un conjunto de registros de ECGs de 12 derivaciones que reúne información de miles de pacientes, con edades y condiciones de salud variadas (Wagner et al., 2020). Se caracteriza por:

2.1.2.1. Número de muestras

La base de datos incluye 21799 registros. Cada registro tiene una duración de 10 segundos y está disponible a una frecuencia de 100Hz y 500Hz. El hecho de que todos los registros sean uniformes facilita en gran medida el preprocesamiento de los datos.

2.1.2.2. Etiquetas clínicas

Se proporciona un conjunto de anotaciones diagnósticas que abarcan una amplia serie de condiciones. Adicionalmente, se incluye un archivo en formato .csv con información sobre cada condición.

Todas las posibles anomalías de un ECG se agrupan en cuatro clases diferentes, con una clase adicional para los ECGs categorizados como normales. Las etiquetas son las siguientes:

- **Normal (NORM):** Esta etiqueta significa que el ECG se categoriza como normal, y por tanto no tiene ninguna anomalía¹.
- **Infartos de miocardio (MI):** Las anomalías en esta categoría presentan elevaciones o descensos anómalos del segmento ST característicos de los infartos de miocardio. Dependiendo de la zona donde se produzca la interrupción en el flujo sanguíneo, pueden verse las alteraciones en diferentes derivaciones, siendo siempre necesario que se vean en dos derivaciones continuas.

¹Esto no significa que un ECG con esta etiqueta no pueda tener también otras etiquetas. Por cómo funciona el modelo, se decide para cada una de las 5 etiquetas la probabilidad de que pertenezca a esta, y luego (mediante un umbral de 0.5), se binariza la pertenencia a etiquetas. Esto hace que el modelo pueda etiquetar un mismo ECG como normal y con una anomalía, lo cuál no es intuitivo. Una posible línea de trabajo futuro es modificar esta etiqueta para que sea exclusiva, es decir, que solo se aplique si no coincide con ninguna anomalía.

- **Anomalías del segmento ST y onda T (STTC):** Esta categoría incluye otras alteraciones del segmento ST, siempre que no puedan ser clasificadas como infartos de miocardio.
- **Anomalías de la conducción (CD):** Los trastornos o anomalías de la conducción afectan la transmisión normal de los impulsos eléctricos a través del sistema de conducción cardíaco. A continuación, se describen de manera simplificada algunas de las principales anomalías de la conducción presentes en la base de datos PTB-XL:
 1. **Bloqueo de rama derecha (BRD):**

Ocurre cuando hay una interrupción o retraso en la conducción eléctrica a lo largo de la rama derecha del haz de His.

En el ECG se manifiesta como un ensanchamiento del complejo QRS con una morfología característica en las derivaciones pericordiales derechas (V1 y V2).
 2. **Bloqueo de rama izquierda (BRI):**

Se produce por una interrupción o retraso en la conducción a través de la rama izquierda del haz de His.

En el ECG, presenta un ensanchamiento del complejo QRS con patrones distintivos en las derivaciones precordiales izquierdas (V5 y V6).
 3. **Bloqueo auriculoventricular (AV):**

Implica un retraso o interrupción en la conducción de los impulsos eléctricos desde las aurículas hacia los ventrículos a través del nodo auriculoventricular. Se clasifica en diferentes grados:
 - a) **Primer grado:** Retraso en la conducción, evidenciado por un intervalo PR prolongado en el ECG.
 - b) **Segundo grado:** Conducción intermitente, donde algunos impulsos auriculares no logran conducir a los ventrículos.
 - c) **Tercer grado (completo):** Ausencia total de conducción entre aurículas y ventrículos, resultando en una disociación completa entre ambos ritmos.
- **Hipertrofia (HYP):** Esta categoría (que es la menos común) presenta alteraciones en la amplitud del complejo QRS o cambios en las ondas ST/T.

En la Figura 2.1 podemos ver ECGs que presentan cada tipo de anomalía. Para una descripción detallada y exhaustiva de las anomalías cardíacas y su clasificación, el lector puede referirse a **Braunwald's Heart Disease: A Textbook of Cardiovascular Medicine** Libby et al. (2021).

2.1.2.3. Formato de los datos

Los ECGs se almacenan en formato WFDB (*Waveform Database*), un estándar ampliamente utilizado en el ámbito médico que almacena señales biomédicas (como

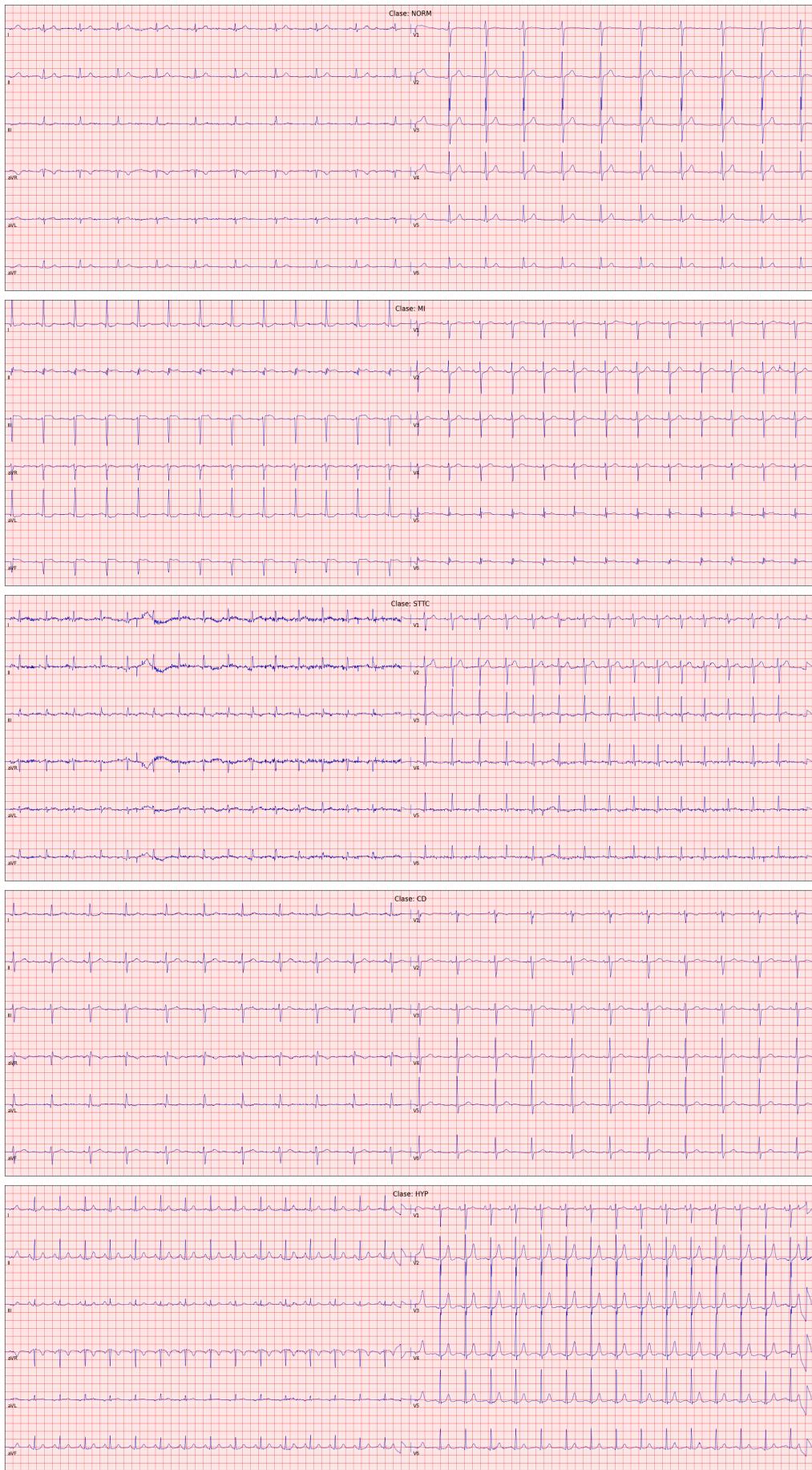


Figura 2.1: ECGs de todos los tipos de anomalías.

es el caso de un ECG) con diversas anotaciones, como por ejemplo comentarios médicos o especificaciones de la onda (el número de derivaciones, el orden de estas, la frecuencia de muestreo, etc.) (Xie et al., 2023).

2.1.2.4. Información de pacientes

Cada ECG está asociado a información demográfica (edad, sexo, etc.), un identificador para saber qué ECGs corresponden a un mismo paciente, la fecha de registro, etc.

En este trabajo únicamente tendremos en cuenta las doce derivaciones del ECG, así como las anotaciones que permiten clasificarlo en una de las cinco etiquetas contempladas en la Sección 2.1.2.2.

2.2. Modelos para la clasificación de ECG

En esta sección exploraremos las distintas aproximaciones que ha habido a la clasificación automática de ECG mediante el uso de IA a lo largo del tiempo.

2.2.1. Enfoques basados en técnicas tradicionales de aprendizaje automático

El desarrollo de clasificadores ha sido crucial en el campo de la IA enfocada a la detección de anomalías en señales de ECGs. Entre estos métodos, los árboles de decisión y el método k-NN han mostrado una eficiencia notable gracias a su sencillez y capacidad para modelar comportamientos complejos (Guo et al., 2023).

No obstante, estos enfoques presentan un problema, pues requieren seleccionar de manera manual una serie de características en la señal (como amplitud de la onda R, duración del complejo QRS, etc.), cosa que resulta extremadamente difícil ya que el análisis de electrocardiogramas es muy complejo. Aunque estos métodos resultan efectivos en escenarios con un volumen de datos moderado, su rendimiento depende en gran medida de las características anteriormente mencionadas (Acharya et al., 2017).

2.2.2. Modelos basados en redes neuronales

Las redes neuronales son un modelo computacional inspirado en las neuronas biológicas. Están formadas por capas de neuronas artificiales conectadas entre sí que aprenden a transformar los datos de las entradas en las salidas deseadas. Este aprendizaje se realiza ajustando los pesos de cada conexión, comúnmente mediante el algoritmo de *backpropagation* (Rumelhart et al., 1986), para minimizar un error definido (p.ej., la diferencia entre la salida prevista y la real). De esta forma, en lugar de depender de técnicas de extracción de características diseñadas manualmente, las

redes neuronales aprenden internamente los rasgos más relevantes a partir de los datos.

En particular, las CNNs (*Convolutional Neural Networks*, Redes Neuronales Convolucionales) se han popularizado en tareas de visión por computador y análisis de señales al emplear operaciones de convolución (ver siguiente apartado).

Aunque nacieron enfocadas al reconocimiento de imágenes, se han adaptado con éxito a otros tipos de datos, como ondas, usando convoluciones unidimensionales en lugar de bidimensionales. Esto permite a la red detectar características relevantes de la señal (como la forma o duración de los intervalos) sin necesidad de ser elegidas manualmente por un experto, lo que le otorga una gran capacidad de generalización (Hannun et al., 2019).

2.2.2.1. Convoluciones

Una convolución es una operación matemática que toma dos funciones o señales y produce una tercera, mostrando cómo una de ellas “filtra” a la otra. Por ejemplo, en el procesamiento de imágenes, la convolución se utiliza para aplicar filtros o detectores de bordes: se toma una matriz de píxeles (la imagen) y se combina con un kernel (también llamado filtro), multiplicando y sumando los valores de cada posición para generar un nuevo valor en la imagen resultante. Este proceso, repetido a lo largo de toda la matriz, ayuda a resaltar características como contornos o texturas específicas.

En redes neuronales, se utilizan convoluciones discretas (es decir, diseñadas para trabajar con una cantidad de datos discreta, lo que ocurre siempre ya que los datos son finitos). La convolución es la base para extraer características relevantes de datos de entrada como imágenes o señales (como es el caso de un ECG). Cada capa convolucional aprende pesos que responden a cierto patrón, de modo que, al aplicar esa capa sobre la entrada, se pueden detectar patrones específicos (por ejemplo, la presencia de bordes horizontales o verticales en una imagen). A través de varias capas de convolución, el modelo va construyendo representaciones cada vez más complejas, y eso permite una gran tasa de acierto en tareas de clasificación (como es el caso de los ECGs). Esta eficiencia y capacidad de extracción de características es lo que ha convertido a las convoluciones en una herramienta fundamental dentro de la informática moderna, especialmente en los campos de visión por computadora.

2.2.3. El modelo de Ribeiro

Dentro de los enfoques basados en redes convolucionales, destaca el modelo propuesto por Ribeiro et al. (2020), cuyo objetivo es la clasificación multiclase de ECGs de 12 derivaciones. La arquitectura (que podemos ver en la Figura 2.2) se caracteriza por una serie de capas convolucionales unidimensionales adaptadas a la naturaleza de la señal. Esto permite analizar el electrocardiograma como una señal, sin necesidad de representarla como una imagen.

Los distintos tipos de capas que utiliza el modelo son las siguientes (para ver más sobre los tipos de capas y su comportamiento completo, el lector puede referirse a

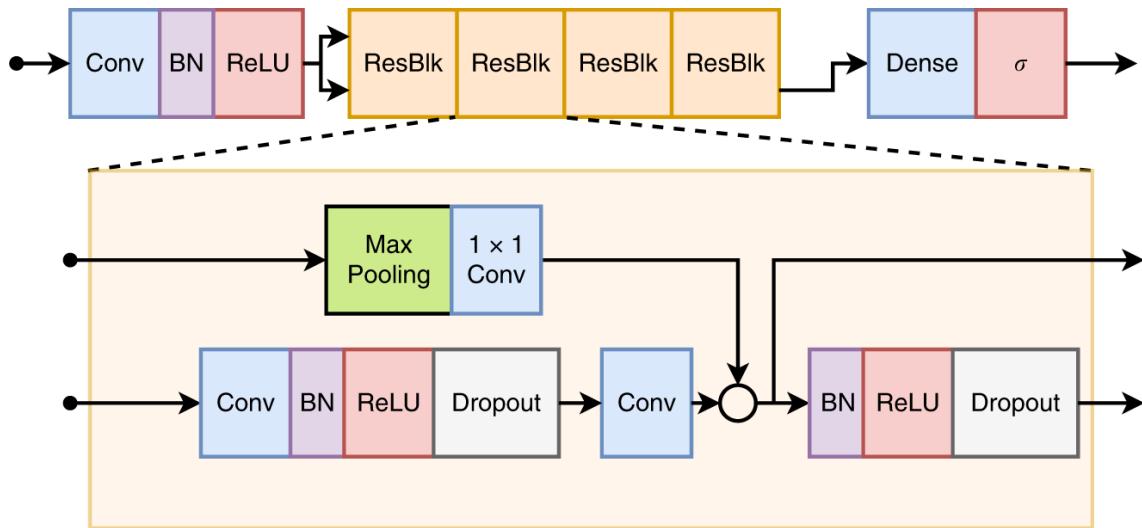


Figura 2.2: Arquitectura de la red neuronal que se emplea en el *gold standard*. Fuente: Ribeiro et al. (2020)

Deep Learning Goodfellow et al. (2016)):

1. **Conv**: Las capas convolucionales mencionadas anteriormente.
2. **BN**: Son capas de *batch normalization*, que mejoran y aceleran el entrenamiento.
3. **ReLU**: Capas de activación, que devuelven 0 si el valor de la entrada es negativo.
4. **Dropout**: Una capa que desactiva un porcentaje aleatorio de las neuronas en cada entrenamiento, para evitar ciertos problemas como la sobrespecialización.
5. **Max Pooling**: Se utiliza para reducir la dimensionalidad de la entrada.
6. **Dense**: Esta capa se suele utilizar antes de una función de activación para adecuar el número de salidas del problema
7. σ : Capas sigmoides, transforman la salida de las neuronas en un valor real entre 0 y 1. Son útiles para acotar la salida.

Como puede verse en el artículo de Ribeiro et al. (2020), el modelo demostró un alto desempeño en la detección de diversas anomalías. En este trabajo, adaptaremos ligeramente la capa de entrada de este modelo para poder tomar como entrada matrices de cualquier tamaño (originalmente el modelo requería de matrices de dimensión 12xN) para poder entrenarlo con transformaciones de la señal original a imágenes. Sin embargo, no modificaremos la estructura interna de las capas convolucionales.

2.3. Explicabilidad en la IA

A pesar de que los modelos de *Deep Learning* han demostrado su eficacia en tareas de diagnóstico clínico, es necesario contar con métodos que ofrezcan explicaciones y permitan interpretar sus predicciones, ya que la actual legislación europea (*GDPR*) exige poder dar una explicación comprensible de cualquier decisión tomada por un algoritmo o modelo predictivo. Esto es particularmente relevante en el ámbito médico, donde cualquier decisión que tome un algoritmo debe poder justificarse ante profesionales de la salud, ya que la salud de una persona podría verse afectada por esta decisión.

Algunos de los métodos más notables de explicación en la IA son los siguientes:

2.3.0.1. Métodos intrínsecos

Estos métodos consisten en analizar el código del modelo de IA para comprender qué hace exactamente un modelo por dentro. Estos métodos solo son aplicables cuando el modelo es intrínsecamente explicable, es decir, que hay una clara correlación entre el código del modelo y la salida del mismo, como es el caso de los árboles de decisión o regresiones lineales.

Las redes neuronales no entran dentro de este tipo de modelos, y por tanto no podemos aplicar este tipo de explicaciones en este trabajo.

2.3.0.2. Métodos basados en visualización

Estos métodos consisten en representar gráficamente qué partes de la entrada son más relevantes a la hora de tomar una decisión. Por esto, están estrechamente relacionados con el significado de los datos de entrada, y no tanto con su forma. Supongamos que tenemos como entrada una matriz de dos dimensiones. Esto puede interpretarse (entre otras cosas) como una imagen o como una serie de vectores de mediciones distintas (esto último se conoce como serie temporal) en el tiempo:

- Si los datos representan imágenes, existen métodos como Grad-CAM para dibujar mapas de calor sobre la imagen original que señalan las partes más relevantes de la imagen a la hora de tomar decisiones.
- Si tenemos una serie temporal, para explicar qué zonas son más importantes lo que necesitamos es señalar, en cada una de las mediciones, qué franjas temporales son más relevantes para la decisión. Esto es lo que hacen algunos métodos como *saliency maps*.

Si bien es cierto que en este trabajo tenemos tanto series temporales (los datos originales sin transformar) como imágenes (los datos transformados), en este trabajo utilizaremos exclusivamente *saliency maps*, por restricciones arquitectónicas que veremos en el Capítulo 5.

Capítulo 3

Metodología y Preparación de Datos

RESUMEN: En este capítulo describimos la metodología seguida para procesar y transformar las señales de ECG de la base de datos PTB-XL, detallamos la distribución de las clases, el preprocesamiento aplicado, las transformaciones realizadas y las métricas empleadas para evaluar los modelos.

3.1. Análisis de los datos y distribución por clases

La base de datos PTB-XL es un conjunto de registros formado por 21799 ECGs de 12 derivaciones, considerado uno de los mayores *datasets* públicos de ECGs disponibles en la actualidad (Wagner et al., 2020). Contiene muestras de ECGs de 10 segundos de duración anotadas con múltiples etiquetas diagnósticas. Para este trabajo, emplearemos la clasificación según las superclases definidas en la propia base de datos, que coinciden con las que presentamos en la Sección 2.1.2.2.

La Tabla 3.1 muestra la distribución de los datos de PTB-XL considerando estas cinco superclases. Se observa que la clase “NORM” es la más numerosa, mientras que la clase “HYP” tiene una cantidad considerablemente menor de datos. Este desbalanceo de clases puede causar varios problemas en el modelo, como por ejemplo:

1. **Sobreajuste hacia la clase mayoritaria:** Al haber bastantes más datos de entrenamiento de una clase (NORM) y menos de otra (HYP), el modelo puede aprender mejor los patrones que identifican las clases mayoritarias, haciendo que sepa distinguir peor las minoritarias, lo que en este caso podría reducir notablemente su capacidad de predicción de anomalías raras (He y Garcia, 2009).
2. **Métricas no representativas:** Las métricas más comunes, como la exactitud, pueden ser poco informativas cuando hay un desbalance en los datos de prueba, ya que un modelo que predice siempre la clase mayoritaria puede tener una

Número de registros	Superclase	Descripción	Porcentaje
9514	NORM	ECG Normal	43.64 %
5469	MI	Infarto de Miocardio	25.08 %
5235	STTC	Cambio ST/T	24.01 %
4898	CD	Transtorno de la conducción	22.46 %
2649	HYP	Hipertrofia	12.15 %

Tabla 3.1: Distribución de las superclases en PTB-XL

La información de esta tabla ha sido extraída directamente del repositorio de PTB-XL.

exactitud alta. Esto puede dificultar la evaluación real del rendimiento del modelo (Yanminsun et al., 2011).

3. **Dificultad en el entrenamiento:** Las redes neuronales profundas requieren de grandes cantidades de datos de entrenamiento para poder entender patrones complejos. Si una de las clases tiene muy pocos ejemplos, es muy probable que el modelo no sea capaz de predecirla correctamente (Leevy et al., 2018).

Para abordar estos problemas existen varias estrategias, como hacer *oversampling* o *undersampling*. El *oversampling* consiste en generar datos sintéticos a partir de los que ya tenemos para balancear las clases (He et al., 2008), pero esto no es una buena técnica cuándo los datos son complejos (como es el caso de un electrocardiograma), ya que no hay una técnica clara para crear datos sintéticos coherentes. Por otro lado, el *undersampling* hace que todas las clases se queden con el mismo número de candidatos que la clase minoritaria, lo que no es una técnica adecuada cuándo los datos de entrenamiento son reducidos desde un principio (Koziarski, 2019).

3.2. Preprocesamiento de datos

En cualquier desarrollo de IA, antes de poder utilizar los datos hay que hacer cierto procesamiento para asegurarnos que son adecuados. Lo primero que habría que hacer es quitar los datos repetidos, incompletos o corruptos, pero afortunadamente la base de datos que estamos utilizando ya ha sido revisada por sus creadores, por lo que podemos obviar este paso.

En procesamiento de señales biomédicas (especialmente en señales que son muy sensibles a determinadas perturbaciones, como es el caso de los ECGs) es muy importante aplicar determinados filtros antes de trabajar con las señales. En este trabajo utilizaremos los scripts que se utilizaron en el trabajo de González Cabeza (2024) (que nos han sido facilitados por el autor y están basados en el trabajo original de Ribeiro et al. (2020)). En concreto, los datos se preprocesan de la siguiente manera:

- Se reescalán todos para tener una frecuencia de 400Hz, que es con la que se entrenó al *gold standard*. Por tanto, tras hacer este procesamiento previo estaremos trabajando con vectores de 4096 elementos.

- Se elimina el desplazamiento de la línea base, que son las interferencias de baja frecuencia generadas por la respiración. Como podemos ver en Chouhan y Mehta (2007), es muy importante hacer esto antes de analizar un ECG.
- Se elimina la interferencia de la línea de alimentación, que es la interferencia generada por la corriente eléctrica del aparato medidor, lo que también es importante como podemos ver en González et al. (2005).

Por último, separamos los datos en tres conjuntos. Los autores de la base de datos presentan una estratificación (es decir, una división equilibrada de los datos) en diez clases (numeradas del uno al diez), y recomiendan utilizar el noveno estrato para validación, el décimo para pruebas y el resto para entrenamiento. Siguiendo esa clasificación, tendremos los siguientes conjuntos:

- **Entrenamiento (*train*):** El conjunto mayoritario (con un 80 % de los datos, es decir 17418 casos), que será usado para entrenar al modelo.
- **Validación (*validation*):** Este conjunto (que representa el 10 % de los datos, es decir 2183) se utilizará para ajustar los parámetros del modelo en el entrenamiento del mismo.
- **Pruebas (*test*):** Este conjunto (que está formado por el 10 % restante de los datos, es decir 2198 casos) es el que utilizaremos para obtener las diversas métricas de rendimiento del modelo.

3.3. Métricas de evaluación

En este apartado revisamos algunas de las métricas típicamente utilizadas y que usaremos para evaluar nuestros modelos.

3.3.1. Métricas habituales

Entre las métricas más habituales podemos encontrar la *F-β Score*, *precision* y *recall*.

3.3.1.1. Precision (Precisión)

La precisión es la proporción de predicciones positivas que son realmente positivas, o más concretamente:

$$\text{Precision} = \frac{\text{Verdaderos positivos}}{\text{Verdaderos positivos} + \text{Falsos positivos}}.$$

Un valor alto de esta métrica indica que el modelo es bueno minimizando falsos positivos, es decir, cuándo el modelo predice que un dato no pertenece a una clase, esa predicción es fiable.

3.3.1.2. Recall (Sensibilidad)

La sensibilidad mide la proporción de casos positivos que el modelo predice correctamente, o más concretamente:

$$\text{Recall} = \frac{\text{Verdaderos positivos}}{\text{Verdaderos positivos} + \text{Falsos negativos}}.$$

Un valor alto de esta métrica indica que el modelo es bueno minimizando falsos positivos, es decir, cuándo el modelo predice que un dato pertenece a una clase, esa predicción es fiable.

3.3.1.3. F- β Score

El F- β Score es una media entre la precisión y el recall, la fórmula concreta es:

$$F_\beta = (1 + \beta^2) \times \frac{\text{precision} \times \text{recall}}{\beta^2 \times \text{precision} + \text{recall}}.$$

El valor más habitual para esto es $\beta = 1$, que nos da la media armónica y permite valorar tanto la fiabilidad del modelo cuando predice positivo como negativo.

Esto es adecuado cuándo, por la naturaleza de un problema, el coste de los falsos positivos es similar al de los falsos negativos, pero no es nuestro caso. En modelos aplicados a la salud, es mucho más importante predecir las anomalías correctamente (ya que de esto puede depender la salud de una persona) que predecir correctamente la ausencia de anomalías.

Los valores de $\beta = 0,5$ y 2 hacen que tenga más peso la precisión y el recall respectivamente, por lo que la primera es más adecuada para cuándo los falsos positivos tienen un coste muy alto y la segunda para cuándo son los falsos negativos los que tienen el coste más alto.

3.3.2. Métricas en clasificadores multietiqueta

Todas las métricas que hemos listado anteriormente están definidas para clasificadores binarios, pero nuestro clasificador es multietiqueta, por lo que es necesario adaptarlas. Tres de los enfoques más habituales a este problema son el cálculo por clases, el promedio binario y el *micro average*.

3.3.2.1. Cálculo por clase

Este es el enfoque más sencillo de todos. Consideramos nuestro clasificador multietiqueta como uno binario para cada una de sus etiquetas, y calculamos las métricas para cada una de las clases.

Este enfoque permite ver el desempeño del modelo en cada una de sus clases, lo que permite entender mejor cuáles son sus debilidades y fortalezas. El principal problema que presenta este método es que no da un único valor para comparar modelos, por lo que puede ser difícil determinar qué modelo es el óptimo.

3.3.2.2. Promedio binario

El promedio binario (también conocido como *one-vs-rest*) consiste en tratar cada clase de forma independiente frente a las demás, calcular las métricas para cada clase de manera binaria y, finalmente, promediarlas. De este modo, cada clase se considera positivamente etiquetada en un escenario (con sus correspondientes verdaderos positivos, falsos positivos y falsos negativos) mientras que el resto de las clases se consideran negativamente etiquetadas.

Este método permite obtener un valor global de la métrica (por ejemplo, F1) que resume el desempeño del modelo en todas las clases, pero puede enmascarar diferencias importantes en la distribución de datos (por ejemplo, cuando el número de instancias de cada clase es muy desigual). Sin embargo, sigue siendo un enfoque útil si se desea una única medida para comparar el rendimiento de diferentes clasificadores.

3.3.2.3. Micro average

El *micro average* se basa en sumar las predicciones correctas e incorrectas de todas las clases antes de calcular las métricas. En lugar de promediar los resultados clase por clase, este método reúne todos los verdaderos positivos, falsos positivos y falsos negativos de manera global. A continuación, se calcula la métrica global a partir de estos valores agregados.

Esta aproximación proporciona un único valor general de desempeño del modelo, especialmente útil cuando se desea obtener una medida global en problemas de múltiples clases o etiquetas. El micro average, al considerar todos los ejemplos de forma conjunta, puede ofrecer una visión más equilibrada del rendimiento global, aunque a costa de no mostrar el detalle de cómo se comporta el modelo en cada clase individual.

3.3.3. Métricas para nuestro problema

Tras realizar el análisis de las posibles métricas que implementar, hemos decidido calcular y mostrar varias métricas para cada modelo, y elegir una que consideramos mejor para afirmar qué modelo es el mejor. Para hacer métricas globales, ya que nuestros datos presentan un importante desbalanceo en una de sus clases, utilizaremos *micro average*. Las métricas que mostraremos son las siguientes:

- Para cada una de las clases:
 - Precisión.
 - Recall.
 - F-1 Score.
- Precisión global calculada como *micro average*.

- Recall global calculado como *micro average*.
- F-1 Score global calculado como *micro average*.
- F Score ajustada, una métrica personalizada que definiremos a continuación.

Todas las métricas que mostraremos, salvo la personalizada, tienen el objetivo de entender mejor cómo funciona el modelo, no obstante, es útil escoger una sola métrica para poder comparar estrictamente qué modelo consideramos *mejor*. Esta métrica será la F Score ajustada

Dado que nuestro modelo es un clasificador en el que las etiquetas no tienen el mismo significado, ya que una de ellas representa un ECG normal mientras que las demás representan diversas anomalías, el coste de los falsos positivos o negativos varía dependiendo de la etiqueta. Nuestro objetivo es que el modelo identifique lo mejor posible las anomalías (cuándo las haya), por lo que el coste de los falsos negativos en las etiquetas de anomalías es muy alto, mientras que el coste de los falsos positivos en la etiqueta normal es muy alto.

Por ello, definiremos la F Score ajustada como la media de la F-0.5 Score de la clase normal y las F-2 Score del resto de etiquetas. Esto nos permite tener una métrica que tiene en cuenta tanto reducir falsos negativos como falsos positivos en todas las etiquetas, pero dando más peso a los falsos negativos o positivos dependiendo de la etiqueta concreta.

La fórmula concreta de la métrica sería:

$$\text{F Score ajustada} = \frac{F - 0,5(\text{NORM}) + \sum_{i \neq \text{NORM}} F - 2(i)}{\text{Número de clases}}$$

3.4. Transformaciones

Pueden aplicarse gran cantidad de transformaciones a una señal antes de procesarla. En nuestro caso concreto, aplicamos una transformada STFT y otra CWT, ambas en sus versiones discretas. En las siguientes secciones explicaremos la idea general de estas transformadas. Para una definición formal de estas, así como un estudio más detallado de sus propiedades y aplicaciones, pueden consultarse Oppenheim y Schafer (2009); Allen y Rabiner (1977).

3.4.1. STFT

La Transformada de Fourier en Tiempo Corto (STFT, del inglés *Short-Time Fourier Transform*) es una herramienta matemática utilizada para analizar la evolución espectral de una señal a lo largo del tiempo. A diferencia de la Transformada de Fourier convencional, que proporciona información global sobre la distribución de las frecuencias de una señal sin preservar la dimensión temporal, la STFT incorpora una ventana que permite observar cómo las frecuencias van cambiando localmente en el tiempo (Allen y Rabiner, 1977; Oppenheim y Schafer, 2009).

En nuestro contexto, la STFT resulta particularmente útil porque permite estudiar señales no estacionarias, es decir, aquellas cuyas características cambian a lo largo del tiempo. El ECG es un ejemplo de este tipo de señal, ya que su frecuencia varía debido factores como cambios en el ritmo, arritmias u otras alteraciones.

La STFT divide la señal en pequeños segmentos de tiempo (llamados ventanas) y aplica la Transformada de Fourier a cada uno de ellos, lo que permite obtener una representación en el dominio tiempo-frecuencia. Esto significa que podemos visualizar cómo cambian las diferentes frecuencias de la señal en el tiempo.

Esta capacidad de analizar variaciones temporales de la frecuencia hace que la STFT sea ideal para identificar eventos transitorios en el ECG, como la aparición repentina de una arritmia, alteraciones momentáneas en la forma de las ondas, o cambios en la frecuencia cardíaca, que suelen indicar anomalías médicas como las que estamos intentando detectar.

A la hora de implementar la STFT, existen varios parámetros importantes a ajustar:

- **Ventana (*window*):** Se utiliza una ventana de tipo Hann. La ventana Hann es una función que reduce el efecto de discontinuidades en los bordes del segmento.
- **Frecuencia de muestreo:** La señal original se encuentra muestreada a 400 Hz. Esta frecuencia de muestreo determina la resolución temporal que se logra.
- **Longitud del segmento:** Se utilizan 256 muestras. El segmento de datos tomado para cada ventana afecta la resolución en frecuencia. Un mayor número de puntos mejora la resolución en frecuencia, pero empeora la resolución temporal.
- **Solapamiento:** Se emplea 128 muestras. Esto significa que cada ventana se solapa con la mitad (128 muestras) de la ventana anterior. Un solapamiento mayor permite detectar eventos transitorios con mayor detalle temporal, a costa de un mayor costo computacional.

La elección de estos parámetros busca un equilibrio entre la resolución temporal y frecuencial, adecuado para el análisis de señales cardíacas. El tamaño de 256 muestras por segmento, junto con la frecuencia de muestreo de 400 Hz, ofrece una resolución en frecuencia suficiente para el rango de interés cardiológico; y el solapamiento de 128 muestras asegura una adecuada continuidad y capacidad de capturar eventos transitorios.

En la Figura 3.1 se presenta un ejemplo de la transformada STFT aplicada a la primera derivación de un ECG de cada clase diagnóstica con los parámetros que acabamos de elegir. La representación muestra cómo las frecuencias varían en función del tiempo, lo que permite ver eventos específicos como arritmias o cambios en las características de las ondas. Esta visualización puede ayudar a comprender y analizar los patrones complejos presentes en las señales del ECG.

3.4.2. CWT

La Transformada de Onda Continua (CWT, del inglés *Continuous Wavelet Transform*) es una técnica matemática utilizada para analizar señales no estacionarias. A diferencia de la Transformada de Fourier, la CWT descompone la señal en funciones llamadas *wavelets*, que se representan en función de tiempo-frecuencia. Esto (al igual que con la STFT) permite una mayor precisión al estudiar eventos transitorios, ya que las *wavelets* pueden ajustarse para capturar detalles de diferentes escalas temporales o frecuenciales.

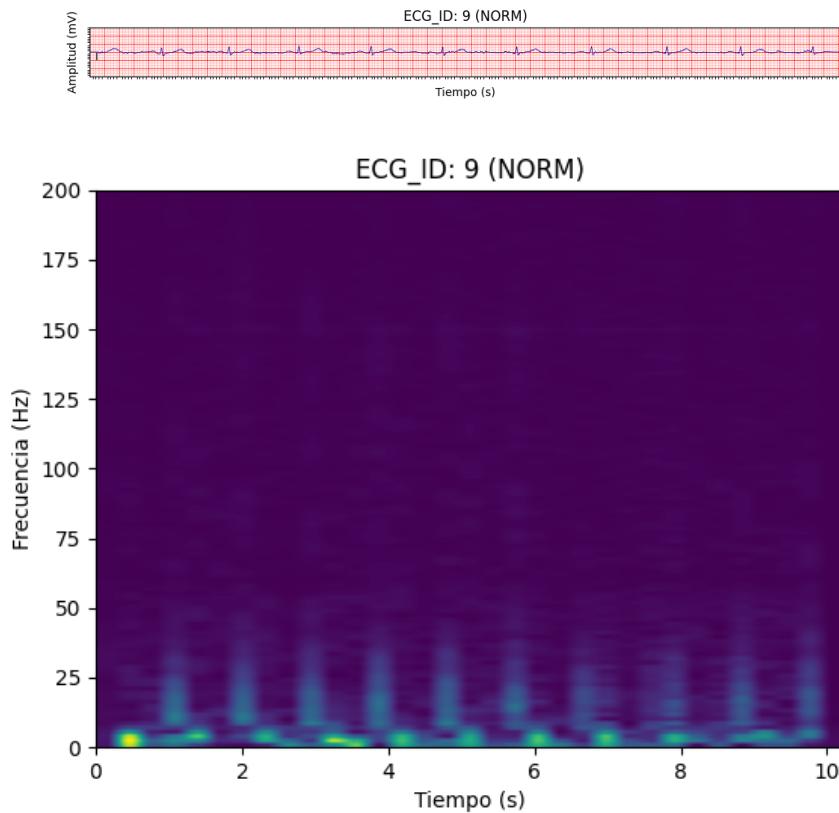
La CWT funciona mediante la correlación de la señal original con *wavelets* de diferentes escalas y ubicaciones temporales, lo que permite hacer una representación en el dominio tiempo-frecuencia. A diferencia de la STFT, donde las ventanas tienen un tamaño fijo, la CWT adapta la resolución automáticamente: se utilizan *wavelets* más cortas para capturar detalles de alta frecuencia, mientras que las más largas capturan las características de baja frecuencia. Esto permite un análisis detallado de las señales.

Al aplicar esta transformada, es necesario definir los siguientes parámetros:

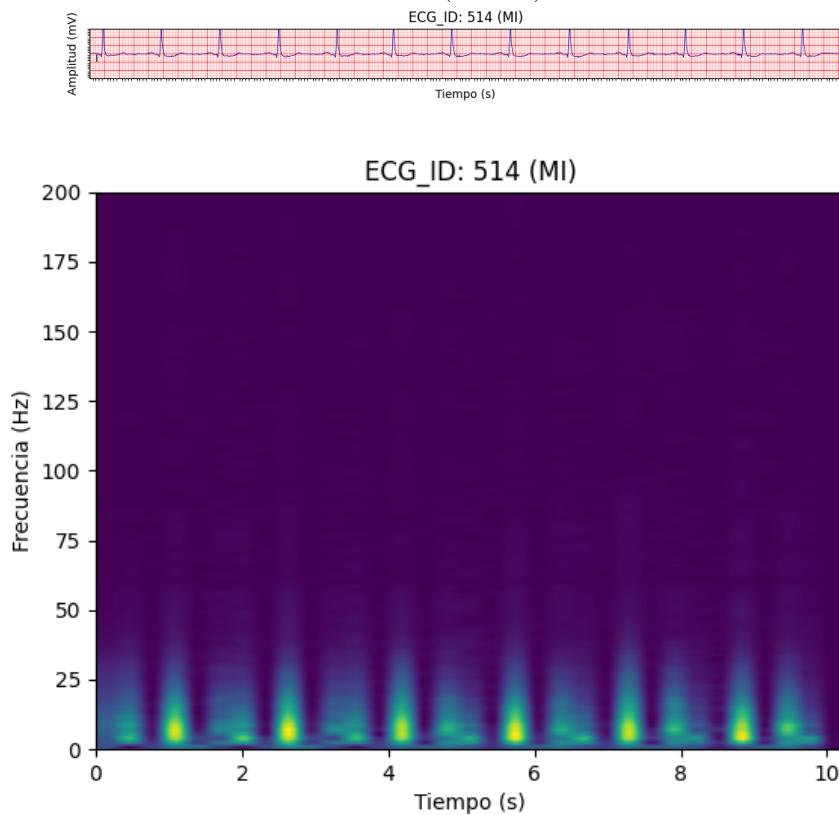
- **Wavelet:** Es la función base en la que se descompone la señal.
- **Escalas:** Son un conjunto de valores que determinan cómo se dilata (o compprime) la *wavelet* durante el análisis. Están relacionados con la frecuencia de la señal. Las escalas pequeñas corresponden a altas frecuencias mientras que las grandes corresponden a bajas frecuencias.

En este trabajo utilizaremos tanto la *wavelet* de Morlet como la de Ricker, ya que ofrecen un buen equilibrio entre localización temporal y frecuencial. Para ambas *wavelets* utilizamos cien valores de escalas, equiespaciadas desde 8 a 637 para la primera y desde 1 a 128 para la segunda.

En la Figura 3.2 se presentan dos ejemplos de transformadas con *wavelet* de Ricker y Morlet respectivamente, con los parámetros que acabamos de elegir. La representación muestra cómo las frecuencias varían a lo largo del tiempo, lo que permite ver eventos específicos como arritmias o cambios en las características de las ondas. Estas visualizaciones pueden ayudar a comprender y analizar los patrones complejos presentes en las señales del ECG con un nivel de detalle que no ofrece la Transformada de Fourier.

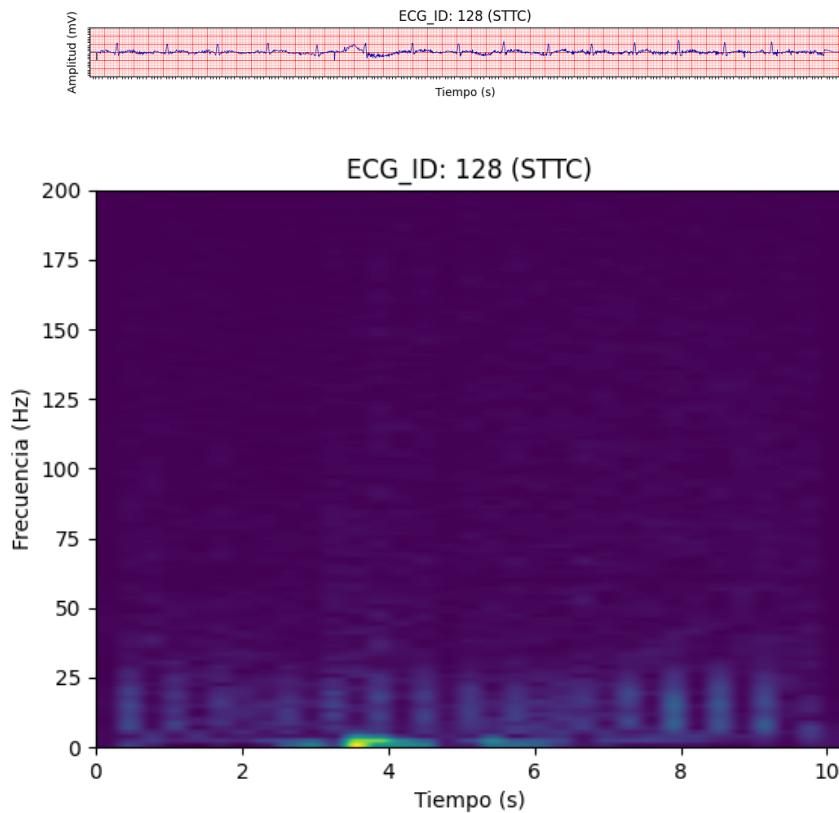


Primera derivación de un ECG normal (arriba) y su transformada STFT (abajo).

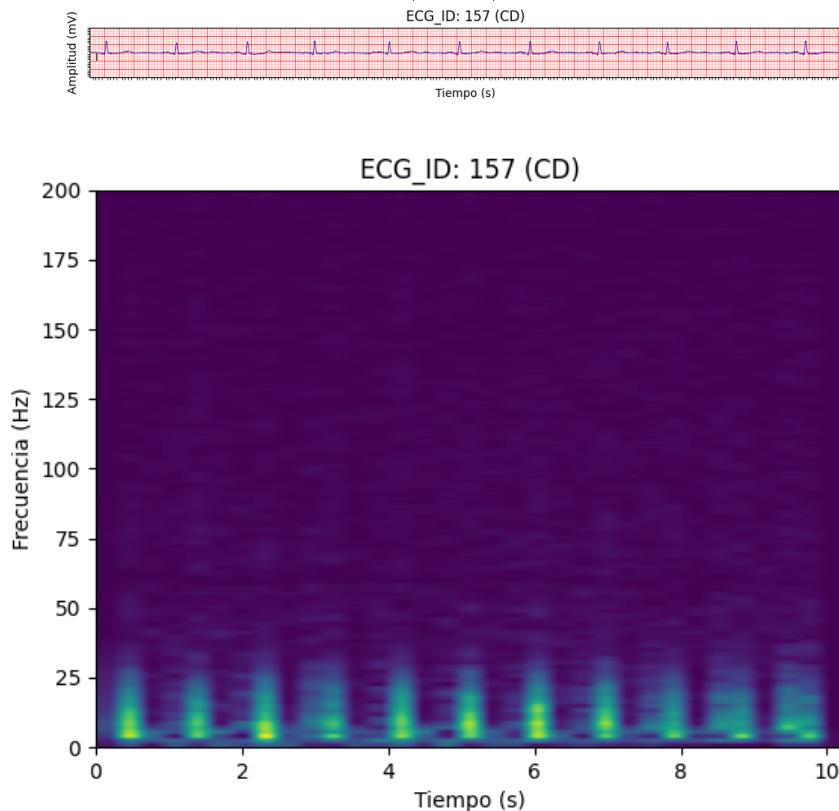


Primera derivación de un ECG de infarto de miocardio (arriba) y su transformada STFT (abajo).

Figura 3.1: Transformadas STFT de la primera derivación de ECGs para cada clase.

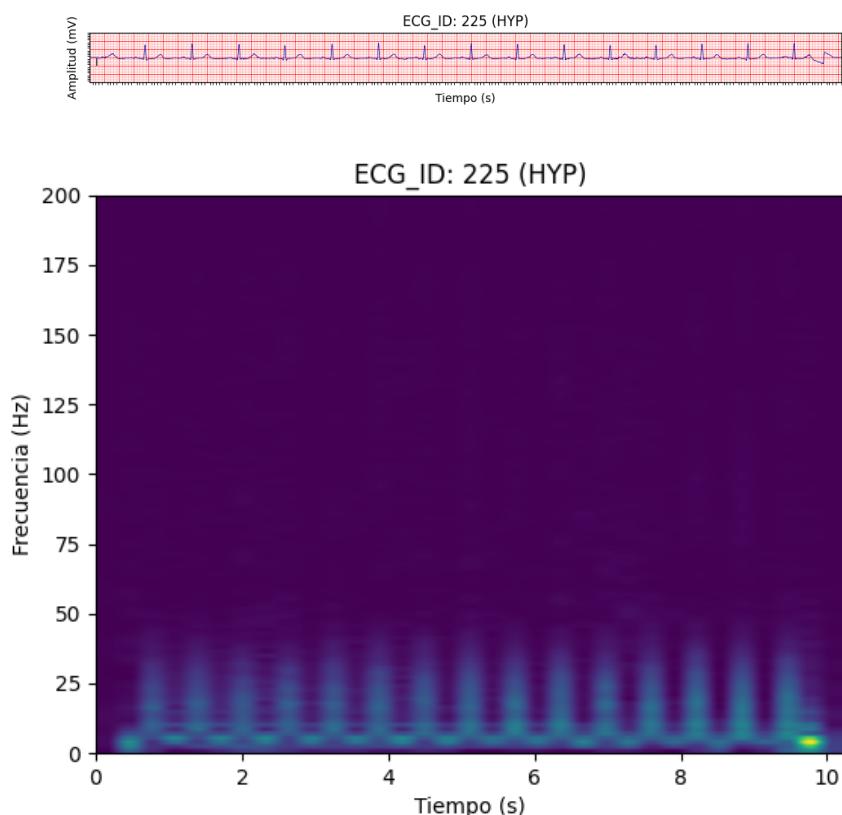


Primera derivación de un ECG de tipo STTC (arriba) y su transformada STFT (abajo).



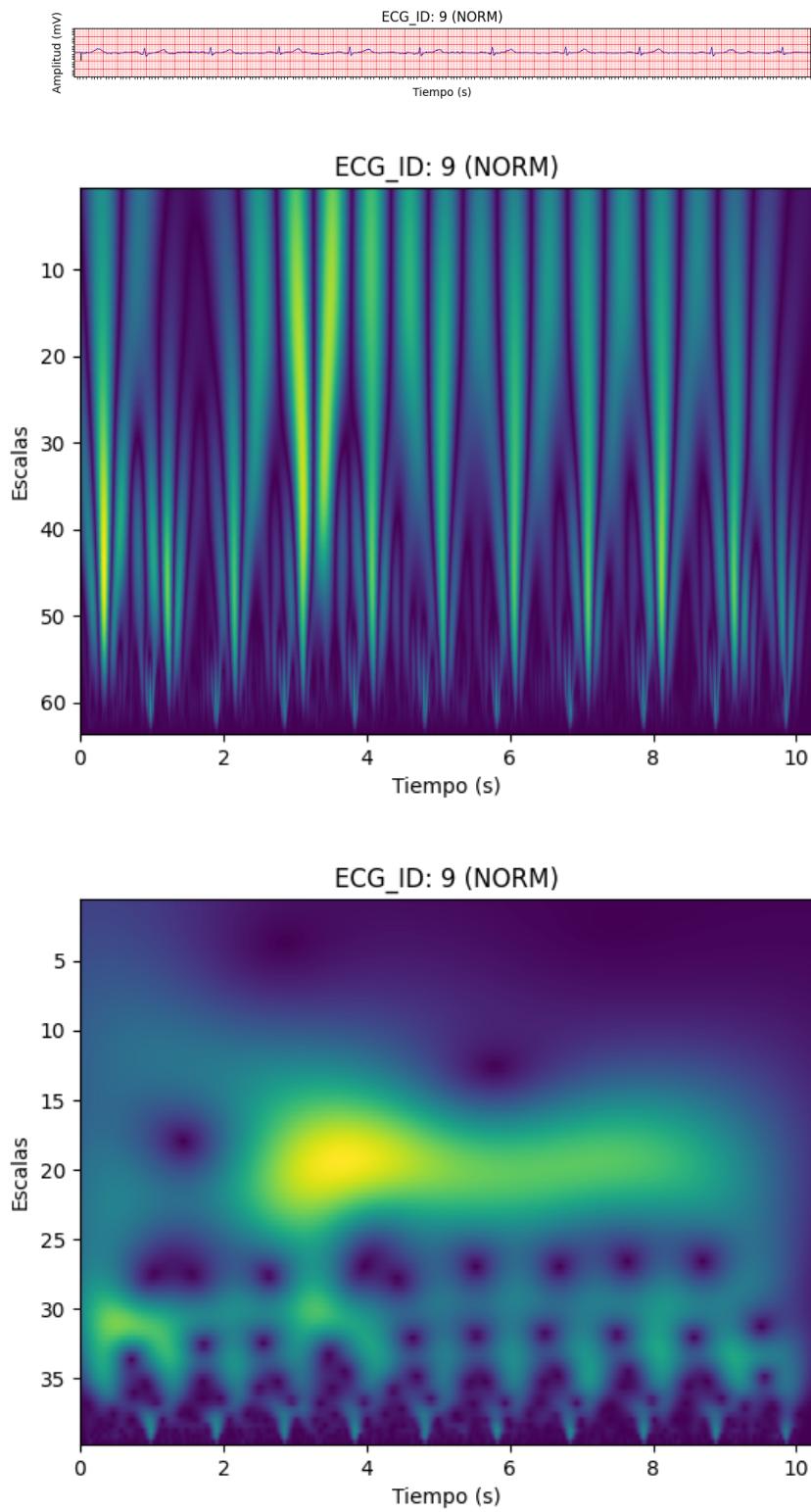
Primera derivación de un ECG de tipo CD (arriba) y su transformada STFT (abajo).

Figura 3.1: Transformadas STFT de la primera derivación de ECGs para cada clase.



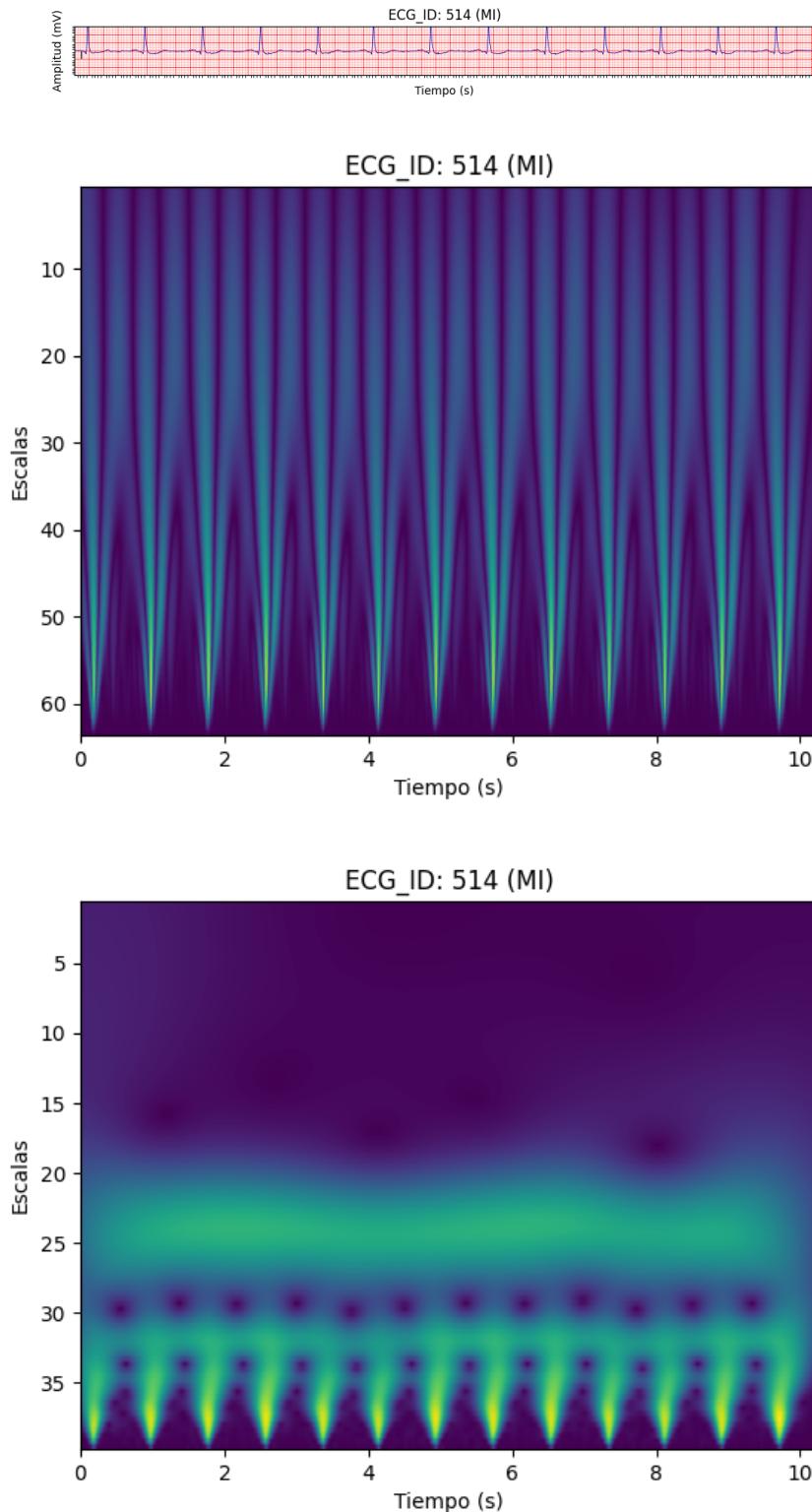
Primera derivación de un ECG con hipertrofia (arriba) y su transformada STFT (abajo).

Figura 3.1: Transformadas STFT de la primera derivación de ECGs para cada clase.



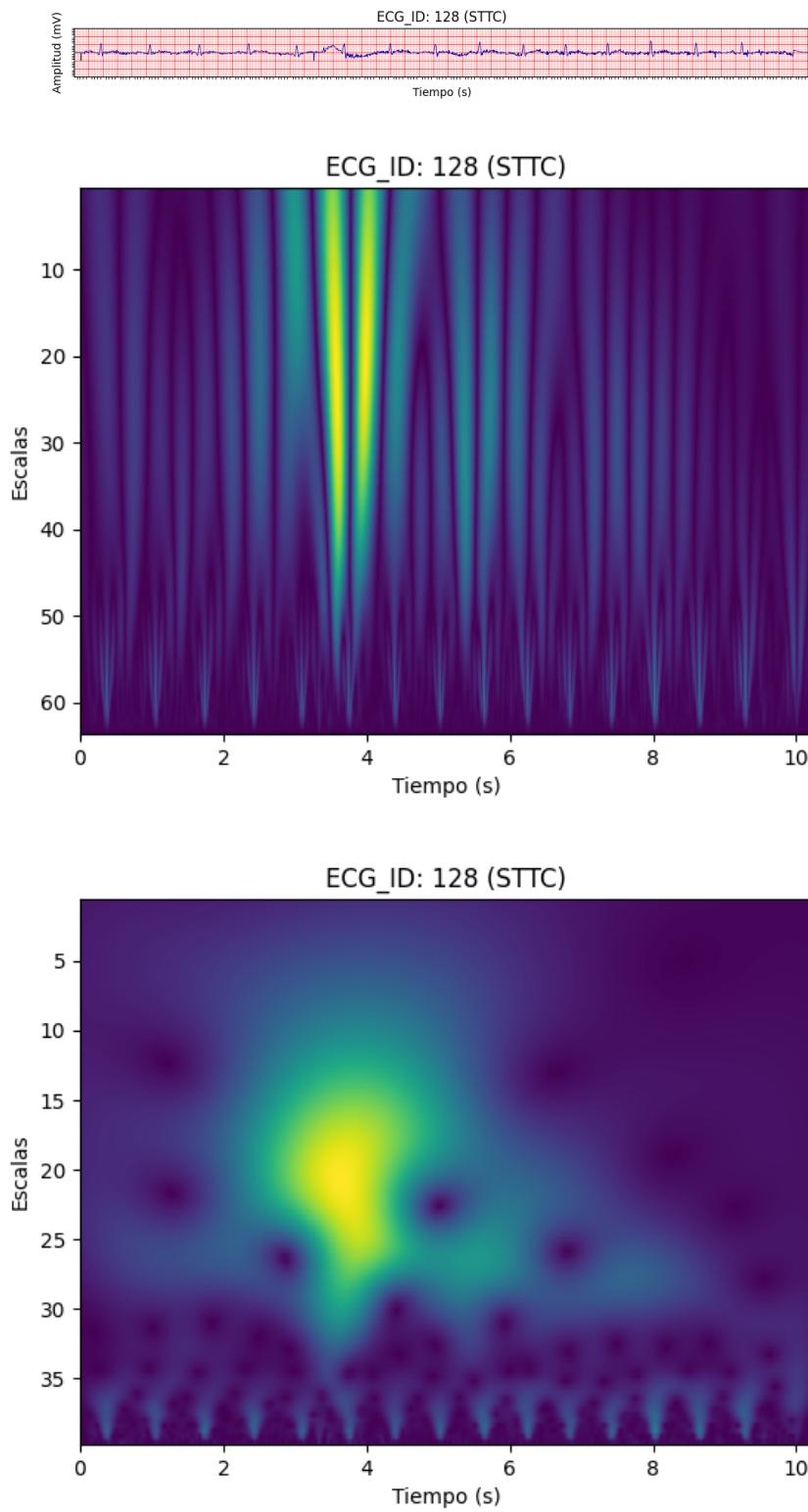
Primera derivación de un ECG normal (arriba) y su transformada CWT con *wavelet* de Ricker (izquierda) y Morlet (derecha).

Figura 3.2: Transformadas CWT de la primera derivación de ECGs para cada clase.



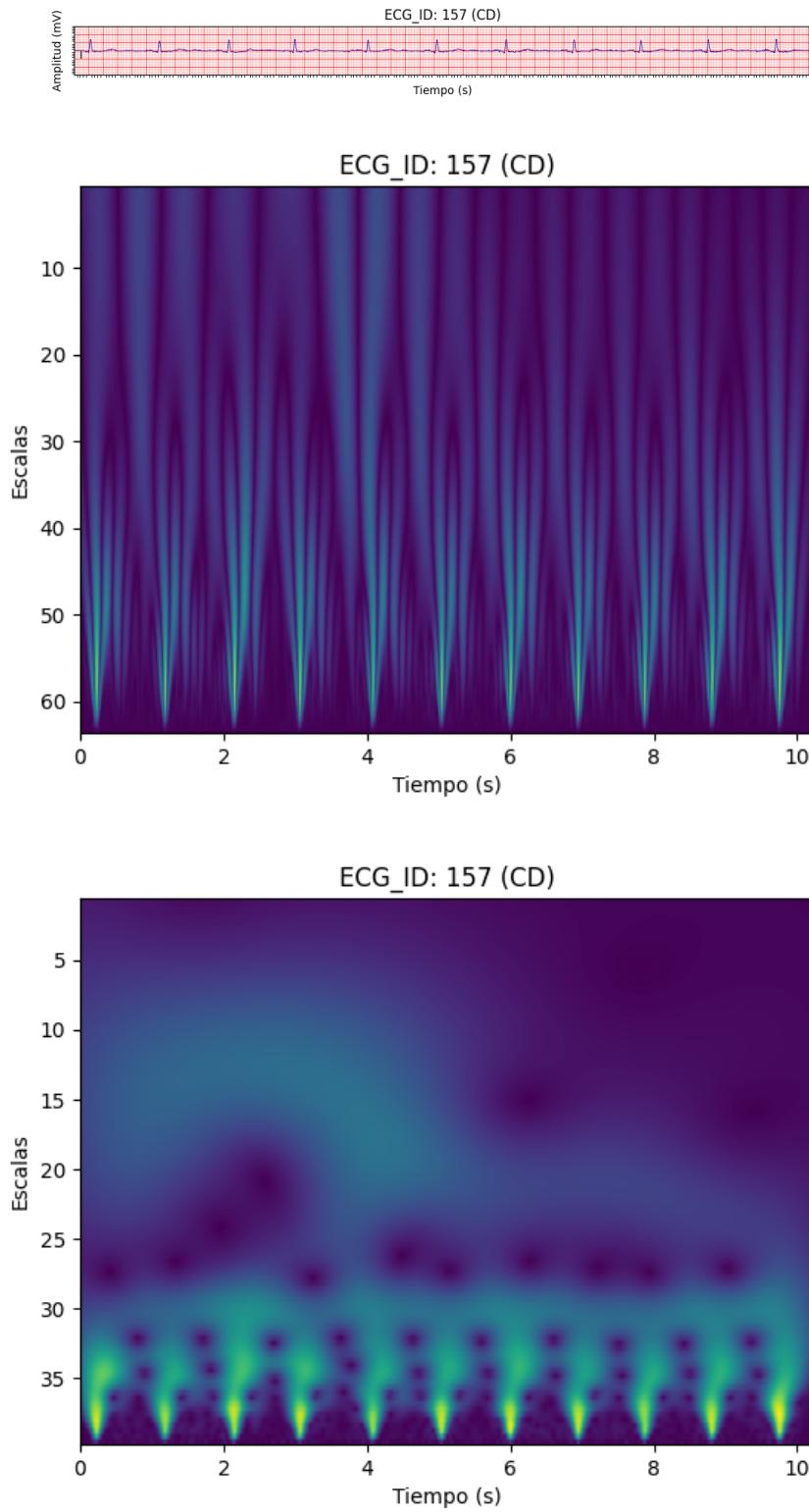
Primera derivación de un ECG de infarto de miocardio (arriba) y su transformada CWT con *wavelet* de Ricker (izquierda) y Morlet (derecha).

Figura 3.2: Transformadas CWT de la primera derivación de ECGs para cada clase.



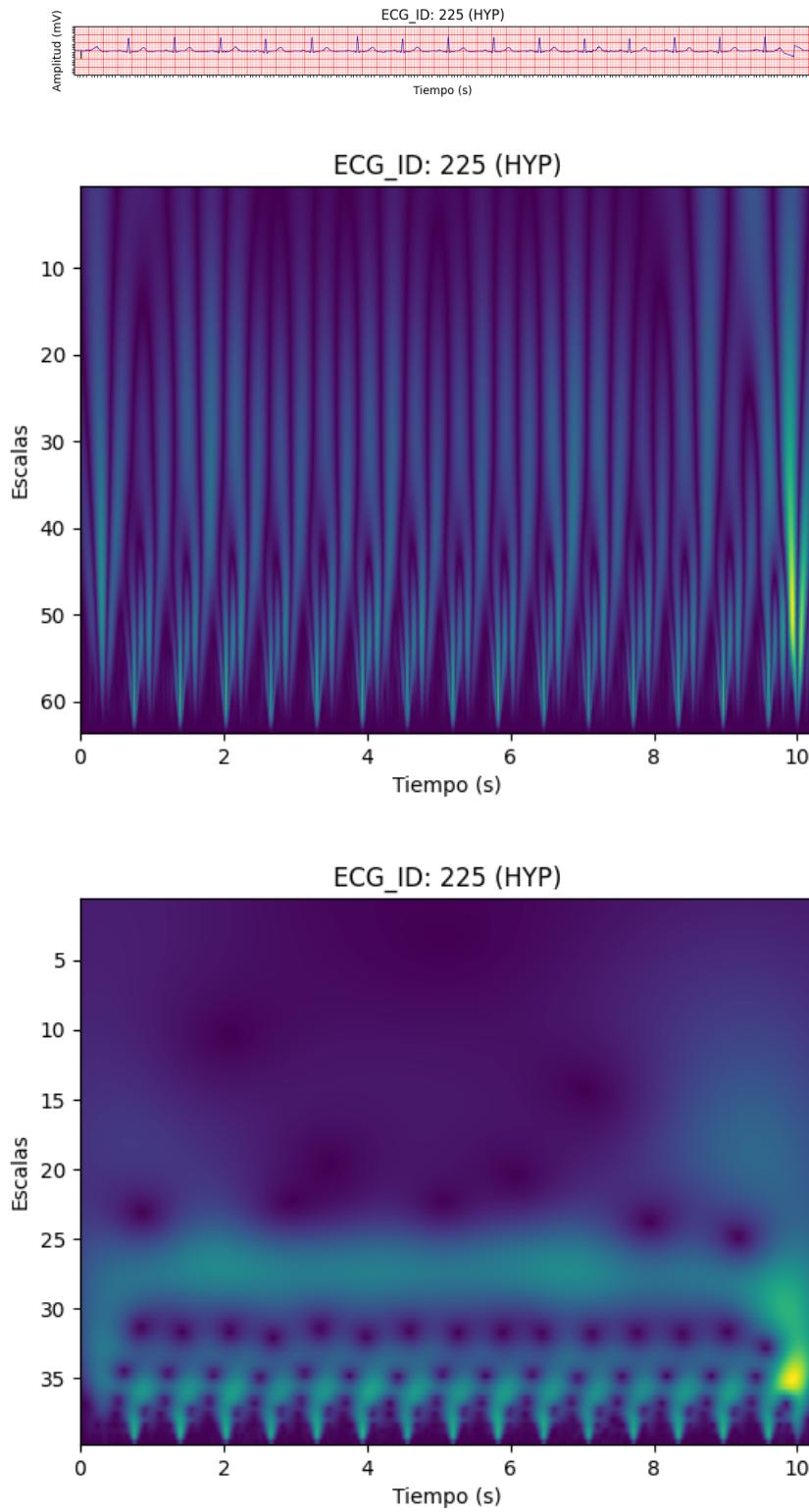
Primera derivación de un ECG de tipo STTC (arriba) y su transformada CWT con *wavelet* de Ricker (izquierda) y Morlet (derecha).

Figura 3.2: Transformadas CWT de la primera derivación de ECGs para cada clase.



Primera derivación de un ECG de tipo CD (arriba) y su transformada CWT con *wavelet* de Ricker (izquierda) y Morlet (derecha).

Figura 3.2: Transformadas CWT de la primera derivación de ECGs para cada clase.



Primera derivación de un ECG con hipertrofia (arriba) y su transformada CWT con *wavelet* de Ricker (izquierda) y Morlet (derecha).

Figura 3.2: Transformadas CWT de la primera derivación de ECGs para cada clase.

Capítulo 4

Entrenamiento y Resultados

RESUMEN: Este capítulo describe el proceso de entrenamiento de los modelos, detalla la configuración utilizada y presenta las métricas cuantitativas que permiten comparar su desempeño. Posteriormente se discuten los resultados obtenidos.

4.1. Modelos modificados entrenados

Modificamos la capa de entrada del *gold standard* para poder entrenar modelos con datos en matrices bidimensionales de cualquier tamaño, lo que nos permitió (con una modificación muy pequeña del *gold standard*) entrenar modelos con imágenes como datos de entrenamiento.

Al transformar cada una de las doce derivaciones obtenemos una matriz (de dimensiones dependientes exclusivamente del tamaño de la señal de entrada y de los parámetros de la transformada). Esto significa que al aplicar la misma transformada a las doce derivaciones de un ECG, obtenemos doce matrices de las mismas dimensiones, pero el modelo necesita solamente una matriz. Para abordar este problema, concatenamos las matrices en el eje de su altura, lo que nos permite utilizar el modelo modificado con los datos de las transformadas.

En concreto, los modelos modificados que entrenamos son los siguientes:

- **Modelo STFT:** Entrenado con los datos transformados utilizando los parámetros especificados en la subsección 3.4.1.
- **Modelo CWT Ricker:** Entrenado con los datos transformados utilizando los parámetros especificados en la subsección 3.4.2 con el *wavelet* de Ricker.
- **Modelo CWT Morlet:** Entrenado con los datos transformados utilizando los parámetros especificados en la subsección 3.4.2 con el *wavelet* de Morlet.

4.2. Configuración del entrenamiento

4.2.1. *Hardware* y *software*

Todos los entrenamientos se han realizado en *bujaruelo*, una máquina del departamento de arquitectura de computadores y automática de la facultad de informática de la universidad.

4.2.1.1. *Hardware*

- **Procesador:** Intel(R) Xeon(R) CPU E5-2695 v3 @2.3GHz
- **GPUs:** El equipo tiene en total cuatro gráficas, dos de cada uno de los modelos siguientes:
 - NVIDIA GeForce GTX 1080
 - NVIDIA GeForce GTX 980
- **Discos:** El equipo cuenta con dos discos de 1TB, además del disco donde está montado el sistema operativo, de 74.5GB.

4.2.1.2. *Software*

En el Anexo B puede encontrarse una tabla con la totalidad de las librerías instaladas en el entorno de python que hemos utilizado (así como sus respectivas versiones), pero las más destacables para el trabajo son las siguientes:

- **Python:** Utilizamos la versión 3.11.7.
- **TensorFlow:** Utilizamos la versión 2.15, por motivos de compatibilidad con el modelo de Ribeiro.
- **CUDA y cuDNN:** Estas librerías son las que utiliza TensorFlow para realizar los entrenamientos en las gráficas de NVIDIA. En nuestro caso utilizamos las versiones 12.2 y 8.9 respectivamente.

Además, utilizamos **Matplotlib** 3.8.2 y **Seaborn** 0.13.2 para generar los gráficos y **Pandas** 2.2.0 y **Scikit-learn** 1.3.0 para procesar y analizar los datos y generar métricas.

4.2.2. Parámetros de entrenamiento

Para entrenar a los modelos se ha utilizado el script de entrenamiento que aparece en el repositorio del *gold standard*, ligeramente modificado para que trabaje con los datos transformados, que tienen dimensiones diferentes a los datos originales.

4.3. Resultados cuantitativos

En la tabla 4.1 podemos ver los resultados de todos los entrenamientos.

Modelo	Métrica	NORM	MI	STTC	CD	HYP	Global
<i>Gold Standard</i>	F-1 Score	0.835	0.680	0.718	0.705	0.413	0.737
	Recall	0.849	0.603	0.665	0.647	0.284	0.687
	Precisión	0.822	0.778	0.780	0.773	0.759	0.796
	F Score ajustada	—	—	—	—	—	0.628
STFT	F-1 Score	0.826	0.536	0.687	0.672	0.466	0.706
	Recall	0.853	0.433	0.640	0.589	0.338	0.650
	Precisión	0.801	0.701	0.741	0.783	0.750	0.772
	F Score ajustada	—	—	—	—	—	0.587
CWT Morlet	F-1 Score	0.817	0.457	0.645	0.630	0.395	0.644
	Recall	0.857	0.342	0.636	0.548	0.270	0.623
	Precisión	0.780	0.688	0.654	0.741	0.732	0.734
	F Score ajustada	—	—	—	—	—	0.540
CWT Ricker	F-1 Score	0.775	0.316	0.629	0.587	0.426	0.628
	Recall	0.774	0.224	0.633	0.520	0.319	0.572
	Precisión	0.776	0.538	0.626	0.675	0.639	0.696
	F Score ajustada	—	—	—	—	—	0.512

Tabla 4.1: Resultados de los modelos (con tres dígitos decimales de precisión)

4.4. Análisis de los resultados

Los datos de la sección anterior muestran que al entrenar el *gold standard* con diversas transformadas arroja unos resultados destacablemente peores en casi todas las métricas. Esto puede deberse a:

- Una curva de aprendizaje más compleja para los modelos que utilizan transformadas, ya que generalizar patrones en estas podría ser más complejo para la red neuronal que hacerlo sobre las señales sin transformar. Esta hipótesis además se apoya en el hecho de que los modelos transformados hayan necesitado varios días para entrenarse, a diferencia del *gold standard* (que se pudo entrenar en tan solo unas horas).
- Una elección incorrecta de los parámetros para las transformadas, ya que, si bien estos se han elegido con el objetivo de cubrir los rangos de frecuencias donde aparecen las anomalías que estamos buscando, podrían elegirse una cantidad inmensa de configuraciones diferentes.
- Limitaciones de la arquitectura del *gold standard*, ya que utiliza redes Conv1D, que se especializan en detectar patrones en series temporales en lugar de en imágenes.

- Al transformar el modelo en un clasificador binario, se ha utilizado un *threshold* de 0.5 para decidir si el ECG entra dentro de cada una de las etiquetas. Es posible que utilizando diferentes valores se obtengan mejores predicciones.

Capítulo 5

Explicabilidad y Validación Clínica

RESUMEN: En este capítulo describimos cómo se generan y utilizan los *saliency maps* para explicar las decisiones del *gold standard*, detallamos la validación preliminar con un especialista y exponemos las mejoras introducidas a partir de su retroalimentación.

5.1. Método de explicabilidad utilizado

La única técnica de explicabilidad implementada en este trabajo son los *saliency maps* basados en gradientes, aplicados al *gold standard* exclusivamente. Esta decisión se debe a las siguientes razones:

1. **Desempeño superior del *gold standard*:** En el Capítulo 4 hemos visto que el *gold standard* obtiene mejores resultados en casi todas las métricas evaluadas.
2. **Imposibilidad de Grad-CAM en modelos transformados:** Como las redes transformadas mantienen la arquitectura con capas Conv1D, no se pueden aplicar métodos como Grad-CAM a la representación bidimensional. Además, los *saliency maps* no ofrecen una explicación clara sobre imágenes, ya que tratan cada una de las filas de la entrada como datos independientes.

5.2. Resultados de explicabilidad

Aplicando la librería de explicabilidad TSIinterpret¹, hemos explicado cinco casos del conjunto de test para cada clase que cumplen las siguientes condiciones:

¹Puede encontrarse su documentación en el siguiente enlace.

- El valor real de la etiqueta es solamente una condición, es decir, el valor esperado es 1.0 para una de las clases y 0.0 para el resto.
- La predicción es perfecta, es decir, la única etiqueta en la que el modelo clasifica el ECG es la que tiene un valor real de 1.0.
- La clase HYP no tiene ningún caso así, debido a que es la clase minoritaria, por lo que no hemos podido aplicar explicabilidad a esta clase.

Todas las explicaciones realizadas están en la carpeta /out/explanations del github del proyecto, pero en la Figura 5.1 podemos ver una imagen con la explicación de la primera derivación de un ECG de cada clase (excepto HYP).

5.3. Validación con los expertos médicos

Tras hacer las explicaciones, se las mostramos a varios médicos para obtener feedback de éstas. Su opinión general es que es una herramienta con potencial de ser muy útil, y que generalmente detecta correctamente zonas donde puede verse la anomalías, pero hizo los siguientes comentarios de cara a mejorarlo:

1. Heterogeneidad de la explicación

El experto señaló que, salvo excepciones, las anomalías cardíacas se presentan en todos los latidos, no solo en algunos, por lo que es extraño que la explicación señale únicamente las anomalías en algunos latidos.

2. Alternativa de presentación

El experto sugirió reducir las imágenes y mostrar únicamente dos latidos del corazón, ya que esto es una forma habitual de visualizar y explicar anomalías a los médicos en formación.

3. Conformidad con la práctica clínica habitual

El experto señaló que la presentación que hemos hecho del ECG se separa bastante de la presentación habitual de estos. Indica que si las imágenes estuvieran estandarizadas según la práctica clínica habitual, podría ser una herramienta muy útil para la enseñanza, pero que en su estado actual podría llevar a confusión a los alumnos.

En general los expertos tienen la sensación de que esta herramienta tiene el potencial de ser algo muy útil, principalmente en el ámbito de la enseñanza, pero que necesita implementar algunos cambios antes de ser utilizada.

5.4. Modificaciones a partir del feedback

Para poder hacer la explicación homogénea en cada latido o reducir la imagen a dos latidos representativos, necesitaríamos detectar donde empieza y acaba cada

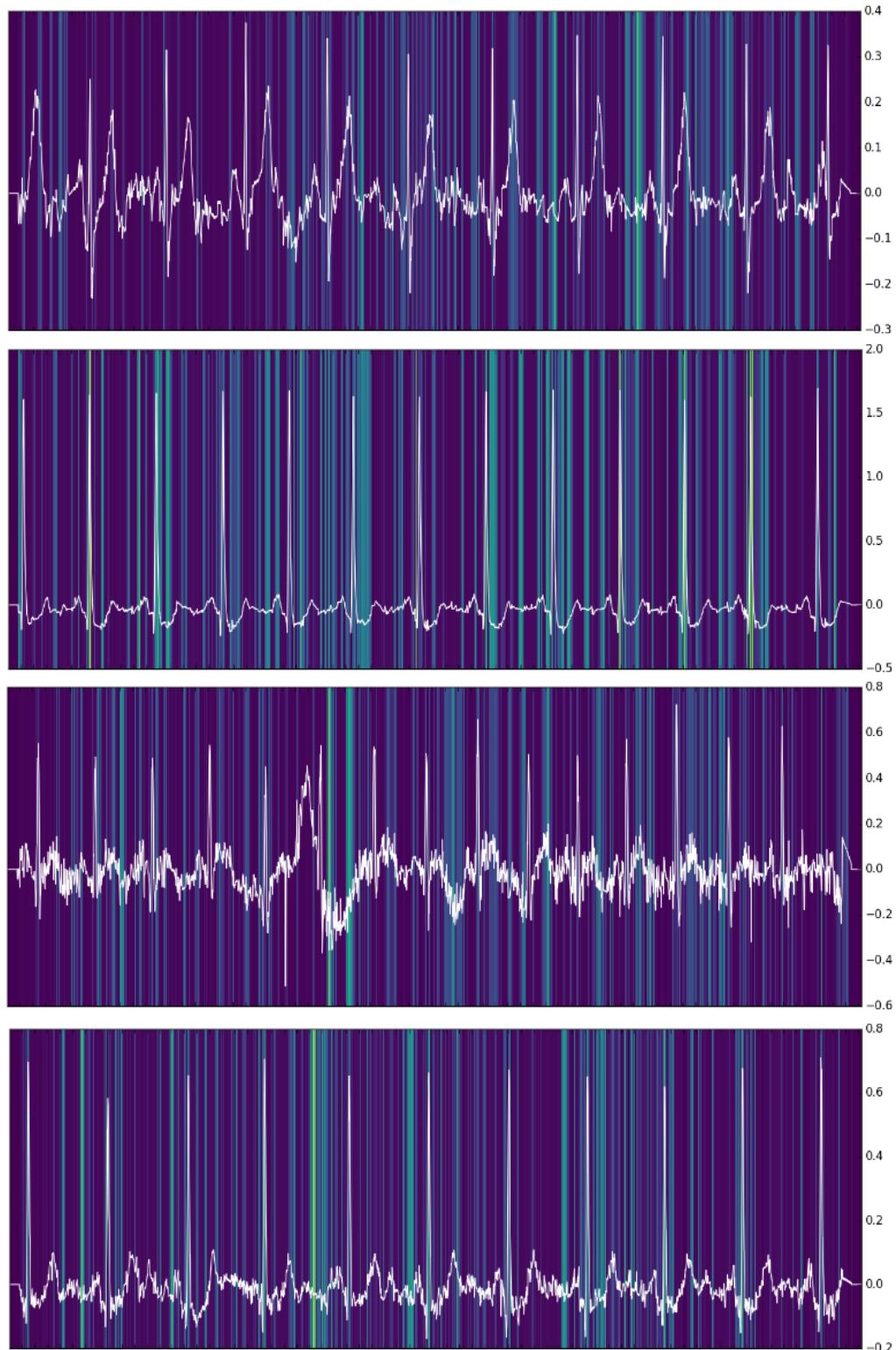


Figura 5.1: Explicaciones de la primera derivación de un ECG de las clases NORM, MI, STTC, CD (respectivamente)

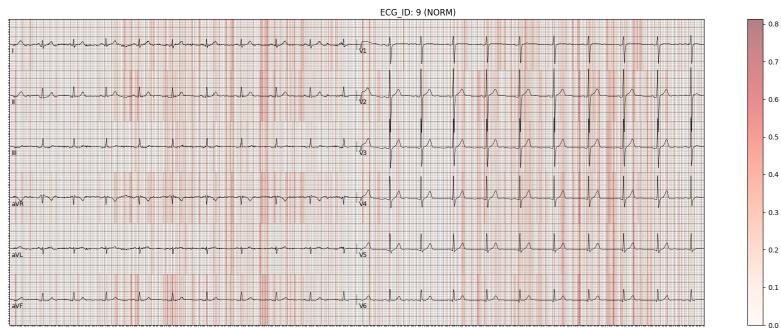


Figura 5.2: Explicación de un ECG de clase NORM pintada sobre la librería *ecg_plot*.

latido, así como las distintas partes de la señal. Si bien es cierto que esto es algo posible, se escapa del alcance de este trabajo, por lo que no lo implementaremos.

Respecto a la representación habitual del ECG, el motivo de que esté presentado de esta forma es que hemos utilizado la propia librería de explicabilidad para dibujar las gráficas, y ésta trabaja con series temporales en general. Para abordar este problema y mejorar la presentación, hemos modificado la imagen que genera librería *ecg_plot*, de manera que también pueda plasmar los colores relativos a la explicación por encima.

5.4.1. NORM

La explicación para la clase normal es un tanto especial, ya que es mucho más sencillo explicar los puntos donde pueden detectarse anomalías que la ausencia de estos.

Podemos ver que la explicación de la Figura 5.2 generalmente resalta las ondas P y T, así como el complejo QRS. Esto es coherente, ya que son las zonas donde suelen encontrarse las anomalías. Sin embargo, es destacable que en ocasiones resalta en el espacio isoelectrónico entre las ondas, lo que puede indicar que el modelo se fija en esas zonas para detectar las arritmias (o en este caso, la ausencia de ellas). En cualquier caso, esta explicación nos ayuda a entender cómo toma las decisiones el modelo.

5.4.2. MI

Según los expertos consultados, este caso se trata de un infarto inferior-lateral de miocardio. Se manifiesta principalmente con elevación del segmento ST en las derivaciones II, II, aVF y ascenso en las precordiales, sobretodo en V5 y V6.

Podemos ver en la Figura 5.3 como se resalta, por ejemplo, en la derivación III la zona del segmento ST donde se aprecia claramente esta elevación del segmento ST sobre la línea isoelectrónica. La explicación resalta también esa elevación presente en aVF y, en menor medida, en II. Recordemos que la derivación II es la resta de la I y la III. Del mismo modo, el modelo está señalando también el descenso de ST en las derivaciones I y aVL. Vemos que adicionalmente la explicación señala

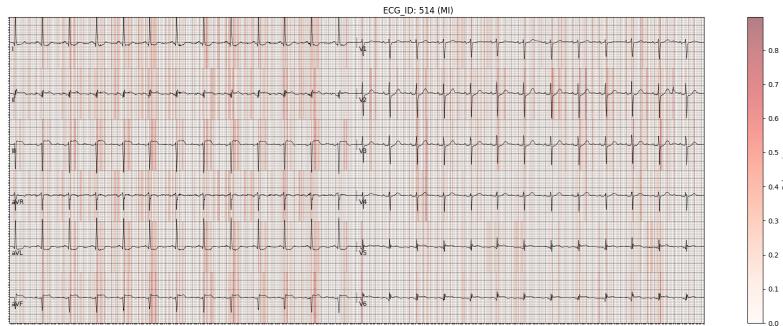


Figura 5.3: Explicación de un ECG de clase MI pintada sobre la librería *ecg_plot*.

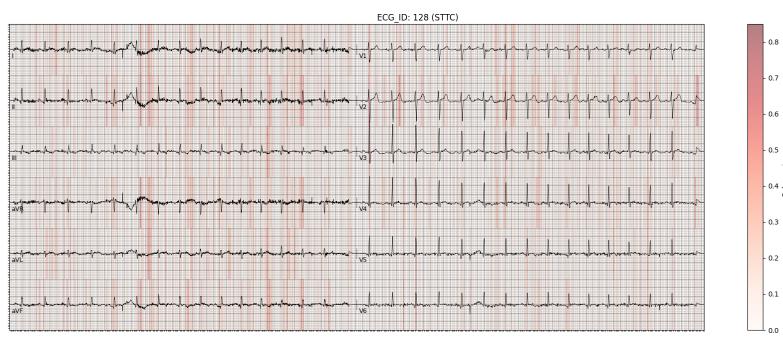


Figura 5.4: Explicación de un ECG de clase STTC pintada sobre la librería *ecg_plot*.

algunos latidos de las primeras derivaciones precordiales, sobretodo V2 y V3, donde se aprecia el descenso, siempre en la zona ddel segmento ST. Sin embargo se fija menos en V5 y V6, o lo hace en menos latidos.

Aunque no es una explicación lineal y homogénea en todos los latidos (como esperarían los médicos), claramente indica que el modelo se ha fijado en las principales características de la señal para clasificar correctamente este caso como infarto. También cabe destacar que la clase es genérica para infartos y seguramente no tiene el poder de discriminar qué tipo de infarto es, pero sí podemos concluir que parece fijarse en las zonas que miraría un médico para decidir.

5.4.3. STTC

En este caso, estamos ante alteraciones inespecíficas de la repolarización que se ven en la onda T, que aparece aplanada en las primeras derivaciones. Vemos que la explicación de la Figura 5.4 está resaltando principalmente las ondas T, de una manera más clara en las derivaciones II y V2 y en latidos aislados en otras derivaciones. No es una explicación tan clara como en el ejemplo anterior de infarto, pero sí parece razonable para los médicos consultados, que nos insisten en que es un caso inespecífico no patológico.

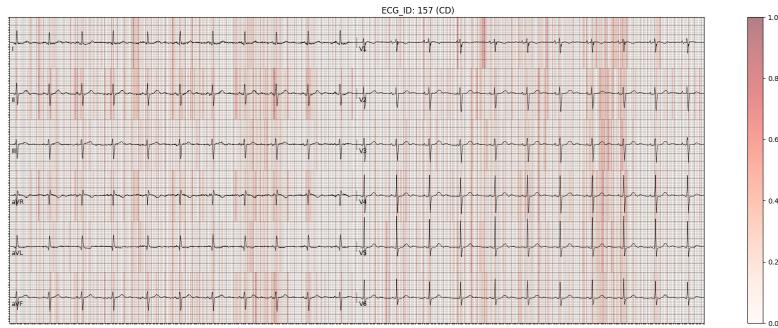


Figura 5.5: Explicación de un ECG de clase CD pintada sobre la librería *ecg_plot*.

5.4.4. CD

Este caso es una alteración de la conducción, en concreto un bloqueo de la rama izquierda (LRBB). Estos bloqueos se caracterizan por un ensanchamiento del complejo QRS, cuya duración se prolonga durante 120ms o más (2 cuadraditos pequeños en el ECG), alteraciones en V1 donde la R es pequeña y la S profunda y alteraciones características en V5 y V6 con onda R ancha y mellada. Vemos que la explicación de la Figura 5.5 está resaltando principalmente complejos QRS y ondas T, así como algunas ondas P, compatible con buscar en general trastornos de la conducción. Para este caso sí que resalta los complejos QRS en las derivaciones precordiales, por lo que los médicos dan la explicación por aceptable, aunque sería preferible que destacara cada uno de los complejos QRS.

Destaca que en la derivación II se está fijando en las ondas P, lo cuál puede servir para detectar, por ejemplo, fibrilación auricular, que es otro tipo de trastorno de la conducción que altera el ritmo. Sin embargo, el modelo clasifica adecuadamente el bloqueo de rama izquierda.

5.5. Comparación con otros resultados publicados

Recientemente, en la revista Nature se publicó un modelo explicable sobre una base de datos muy extensa (con datos de infartos exclusivamente) (Gustafsson et al., 2022). En la Figura 5.6 podemos ver una comparación entre la explicación del artículo y nuestra explicación de un infarto de miocardio. La explicación del artículo es, como inicialmente esperaban los médicos, uniforme en todos los latidos. Esto subraya nuestra sospecha de que un modelo binario para una patología concreta podría ser más explicable. Sin embargo, se puede comprobar que nuestras explicaciones parecen estar fijándose en las zonas correctas: el segmento ST.



Figura 5.6: Explicación de un infarto de miocardio en nuestro modelo (arriba) y en un artículo de investigación (abajo). Fuente: Gustafsson et al. (2022).

Conclusiones y Trabajo Futuro

6.1. Conclusiones

Gracias a la investigación y a las conversaciones con expertos en diversos campos que hemos realizado durante la realización de este trabajo, hemos podido obtener ciertas conclusiones respecto al análisis de ECGs, las posibilidades de los modelos de IA en la clasificación de estos y las posibles explicaciones de estos. En esta sección hacemos una recopilación de estas conclusiones.

6.1.1. Conclusiones generales

- Hemos descubierto que el análisis de ECGs es mucho más complejo de lo que creíamos en un principio, debido a la enorme cantidad de sutilezas que son necesarias comprender para clasificarlos correctamente.
- Es un punto clave preprocesar de manera adecuada las señales antes de trabajar con ellas. Esto conlleva también una serie de sutilezas, como elegir los tipos de filtros correctos y sus parámetros para que el ECG pueda verse con las mínimas interferencias posibles.
- Viendo los resultados de los mejores modelos estudiados en el estado del arte, está claro que a pesar de que estos modelos arrojan buenos resultados, estamos más lejos de lo que en un principio pensábamos de un modelo que permita automatizar la detección de anomalías en ECGs sin la necesidad de la intervención profesional.
- Como es habitual en modelos de aprendizaje automático, es necesario cientos de miles de casos de prueba para poder entrenar modelos confiables. En el caso de los ECGs, si bien es cierto que existen bases de datos relativamente grandes, suelen no ser públicas y con datos no uniformes, lo que complica mucho el trabajo de investigación. Para poder hacer avances significativos en este campo es de vital importancia la existencia de una base de datos pública

con los datos almacenados de manera uniforme (como es el caso de PTB-XL) con cientos de miles de casos.

- Los modelos no son eficaces en la clasificación de enfermedades poco comunes (como es el caso de la hipertrofia). Esto se debe a que estas anomalías, al ser menos comunes que el resto, tienen relativamente pocos casos de prueba en las bases de datos, lo que se agrava por el hecho de que estas bases de datos no son suficientemente grandes. Esto nos indica que probablemente nunca seamos capaces de tener modelos que puedan clasificar todas las anomalías de una forma eficaz y confiable, por lo que podrían considerarse la implementación de modelos híbridos, que detecten las enfermedades más comunes mediante aprendizaje automático y las menos comunes mediante otros métodos.

6.1.2. Conclusiones sobre los modelos entrenados

- Hemos demostrado que el modelo de Ribeiro funciona de manera adecuada aún entrenado con una base de datos relativamente pequeña, lo que señala que es una arquitectura bastante sólida y prometedora.
- A la hora de introducir transformaciones, hemos descubierto que la arquitectura del modelo de Ribeiro no es adecuada para manejar imágenes, como es el caso de las transformadas. Esto no significa que el modelo de Ribeiro no pueda ser adaptado para trabajar con imágenes de manera adecuada, ya que una arquitectura con unos resultados tan buenos es muy prometedora.
- Es importante señalar que el campo del *Deep Learning* aplicado a la medicina es relativamente nuevo, y se publican trabajos innovadores y avances muy frecuentemente, por lo que el uso de transformadas para el análisis de ECGs sigue siendo una posible dirección de avance.

6.1.3. Conclusiones sobre explicabilidad de los modelos

- Hemos descubierto, gracias a nuestras conversaciones con expertos en medicina, que mantener el formato habitual es algo imperativo a la hora de explicar ECGs, ya que los médicos están acostumbrados a ver las señales impresas sobre papel milimetrado a una determinada escala, lo que es necesario para poder analizarlos de manera eficiente y correcta.
- Los *saliency maps* basados en gradiente han demostrado ser un recurso prometedor para entender qué partes de la señal del ECG resultan relevantes para la red.
- Adicionalmente, las explicaciones mediante *saliency maps* han demostrado también su potencial como herramienta de aprendizaje y educación tanto para estudiantes como para profesionales sanitarios que quieran familiarizarse con la IA, no obstante, es necesario hacer más avances en esta dirección antes de

tener una versión de esta herramienta que realmente pueda ser utilizada en estos contextos.

6.2. Trabajo futuro

Dado el potencial del modelo y las limitaciones detectadas, se plantean varias líneas de trabajo para mejorar y ampliar los resultados:

1. Modificar la arquitectura para admitir capas Conv2D

Esta adaptación, si bien requeriría de un trabajo significativo, permitiría explotar mejor las ventajas de las transformaciones de la señal.

2. Utilizar modelos de análisis de imágenes para clasificar ECGs

Si bien es cierto que el modelo de Ribeiro es extremadamente prometedor para clasificar ECGs, es posible que a la hora analizar las señales transformadas, modelos diseñados originalmente para clasificar imágenes sean más eficientes que las posibles modificaciones del modelo de Ribeiro.

3. Explorar transformadas y parámetros adicionales

Siguiendo la línea anterior, una vez obtenida un modelo más adecuado para imágenes (ya sea una modificación de la arquitectura de Ribeiro u otro modelo), podrían probarse otros parámetros a la hora de hacer las transformadas y estudiar su impacto.

4. Especializar el modelo en una anomalía

Transformar el modelo en un clasificador binario permitiría elegir los parámetros de las transformadas de manera más precisa, ya que distintas anomalías se manifiestan en distintas frecuencias.

5. Cambiar los umbrales de detección

En el trabajo de Ribeiro, en lugar de utilizar el umbral de 0.5 para clasificar las etiquetas, se calcularon los umbrales que optimizaban los resultados sobre el conjunto de validación. Esto es algo que, si bien en este trabajo no hemos abordado, podría mejorar los resultados de todos los modelos que hemos entrenado.

6. Delineación de señales en la explicación

Como señaló el experto, mediante delineación de señales podrían reducirse las explicaciones a dos latidos y asegurarse de que estas son homogéneas en ambos latidos. Estas mejoras harían viable el uso de estas explicaciones en el ámbito docente.

6.3. Principales contribuciones personales

Las principales contribuciones al campo de clasificación explicable de ECGs mediante *Deep Learning* que hemos aportado son las siguientes:

1. Ajuste del modelo de Ribeiro

Hemos ajustado el modelo de Ribeiro para poder entrenarlo con datos de cualquier dimensionalidad, permitiendo así que este sea entrenado con imágenes de cualquier resolución.

2. Diseño de transformadas adecuadas para análisis de ECGs

Hemos diseñado transformadas matemáticas de las señales originales de un ECG de manera que capten las frecuencias en las que se pueden detectar las anomalías que estábamos intentando detectar.

3. Desarrollo de una métrica propia

Hemos desarrollado una métrica específica para este problema, que tiene en cuenta las distintas importancias sobre minimizar falsos negativos o positivos en cada clase.

4. Uso de TSInterpret para análisis de ECGs

Hemos utilizado exitosamente la librería TSInterpret para generar explicaciones adecuadas sobre la detección de anomalías en ECGs. Adicionalmente, hemos corroborado con expertos en medicina que estas explicaciones son coherentes y adecuadas.

5. Estandarización de explicaciones

Hemos aplicado las explicaciones anteriormente mencionadas de manera exitosa a un ECG con un formato estándar, haciendo que sean mucho más legibles (y por tanto útiles) para los profesionales sanitarios.

Introduction

6.4. Fundamentals of the Electrocardiogram

An electrocardiogram (ECG) is a non-invasive diagnostic test that records the heart's electrical activity over time. By placing electrodes at specific points on the body, it captures variations in electrical potentials generated by the cardiac muscle (Ribeiro et al., 2020). These variations manifest as waves and complexes (P, QRS, T), whose interpretation allows for the identification of various pathologies. Although measurements can be taken in multiple ways, the standard practice involves measuring the potential difference between twelve pairs of electrodes. Each of these measurements generates a signal, referred to as a lead. In essence, leads provide a perspective of the heart's function from different angles.

Since it is not possible to collect a continuous stream of measurements over time, a finite number of measurements are taken at a specific frequency (usually between 100Hz and 500Hz, depending on the precision of the measuring device). Consequently, an ECG can be stored in twelve vectors of equal size (variable depending on the duration and frequency of the test). Figure 6.1 presents an example of an electrocardiogram (sourced from Wagner et al. (2022) and visualized using the Python library *ecg_plot*).

The frequency at which an ECG is measured is critical. Higher frequencies provide greater resolution of the wave, facilitating the detection of potential abnormalities. However, a device capable of measuring at higher resolutions is more complex and expensive, and higher frequencies also have drawbacks. Increased frequency requires more memory to store an ECG of the same duration, and any type of processing or analysis becomes computationally more costly.

The clinical relevance of the ECG is enormous. In addition to being a low-cost and widely available test, it is a primary step in detecting cardiac anomalies in emergency services, primary care, and specialized consultations. According to the World Health Organization (WHO), cardiovascular diseases are one of the leading causes of mortality worldwide (World Health Organization, 2021). Therefore, optimizing early diagnosis through automated methods of analysis and classification can have a significant impact on improving public health.



Figure 6.1: 12-lead ECG recorded at 100Hz (top) and 500Hz (bottom).

In this context, highly accurate *Deep Learning* models have been developed for interpretable diagnosis of cardiac anomalies (Lu et al., 2024). This chapter will delve into essential aspects of the signal, present the fundamentals of the main waves, and describe the most widely used ECG databases for research.

6.4.1. Leads

An electrocardiogram (ECG) is a tool that records the heart's electrical activity from different perspectives, known as leads. Each lead provides a specific "view" of the heart, facilitating the identification of possible abnormalities in its function.

To obtain these leads, electrodes are strategically placed on the body. Although 10 electrodes are used, the standard ECG provides 12 leads, divided into two main groups:

1. Limb leads (frontal plane):

a) Standard bipolar leads:

- DI (I): Measures the electrical potential difference between the right arm and the left arm.

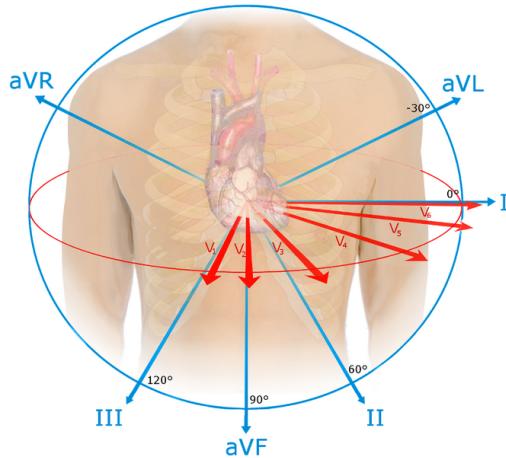


Figure 6.2: Interpretation of the significance of electrocardiogram leads. Source: Wikimedia Commons (2015).

- DII (II): Records the difference between the right arm and the left leg.
- DIII (III): Captures the difference between the left arm and the left leg.

b) Augmented unipolar leads:

- aVR: Observes the potential of the right arm.
- aVL: Focuses on the potential of the left arm.
- aVF: Targets the potential of the left leg.

2. Precordial leads (horizontal plane):

- V1 to V6: These electrodes are placed at specific positions on the chest, providing detailed views of different areas of the heart.

Each lead offers a unique perspective of the heart's electrical activity (as shown in Figure 6.2), allowing healthcare professionals to comprehensively assess the heart's condition and detect potential irregularities in its function.

6.4.2. Main Waves and Segments

Each ECG lead consists of a series of waves and segments that reflect different phases of cardiac activity (MyEKG). Figure 6.3 shows a typical ECG tracing with labels for each wave and segment.

6.4.2.1. P Wave

Represents atrial depolarization, the electrical activation that triggers their contraction. In a normal ECG, the P wave is positive in most leads, except in aVR, where it is typically negative. Its normal duration is less than 0.10 seconds, and its amplitude does not exceed 2.5 millimeters.

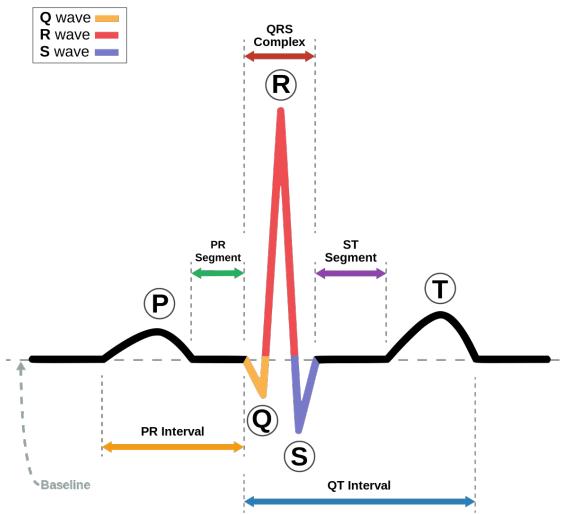


Figure 6.3: Example of an ECG with labeled P, QRS, T, ST, and PR segments.
Source: Wikimedia Commons (2023)

6.4.2.2. PR Interval

The time elapsed from the beginning of the P wave to the start of the QRS complex. This interval reflects the time taken for the electrical impulse to travel from the atria to the ventricles via the atrioventricular node. Its normal duration ranges between 0.12 and 0.20 seconds.

6.4.2.3. QRS Complex

Indicates ventricular depolarization, initiating their contraction. It consists of three waves:

- Q wave: The first negative deflection of the complex. It is not always present, and when it is, it is usually of small amplitude and duration.
- R wave: The first positive deflection of the complex. It is generally the wave with the highest amplitude.
- S wave: The negative deflection following the R wave.

The total duration of the QRS complex is typically less than 0.10 seconds.

6.4.2.4. ST Segment

The segment from the end of the QRS complex to the beginning of the T wave. It represents the period during which the ventricles are fully depolarized before beginning their repolarization. Under normal conditions, this segment is isoelectric, meaning it remains on the ECG baseline.

6.4.2.5. T Wave

Reflects ventricular repolarization, preparing the heart for the next contraction cycle. The T wave is usually positive in most leads, except in aVR, where it is negative. Its shape is asymmetric, with a slower rise and a faster descent.

6.4.2.6. QT Interval

Extends from the beginning of the QRS complex to the end of the T wave. This interval represents the total time of ventricular depolarization and repolarization. Its duration varies with heart rate, but a QT interval between 340 and 440 milliseconds is generally considered normal in young adults.

Proper interpretation of these waves and segments is essential for assessing cardiac function and detecting potential abnormalities in the heart's electrical conduction.

6.4.3. Main Anomalies in an ECG

Numerous anomalies in an ECG can indicate disturbances in cardiac function. In this work, anomalies will be grouped as they are categorized in the selected database, detailed in Section 2.1.2.2.

6.5. Motivation

Artificial Intelligence (AI) has transformed numerous fields. In particular, medicine has witnessed significant advancements through the application of artificial intelligence to diagnosis and data analysis. One of the most promising areas is the automated interpretation of ECG signals, which is critical for the early detection of heart diseases. Cardiovascular diseases are among the leading causes of death worldwide (World Health Organization, 2021), highlighting the need for fast and accurate diagnostic methods.

Historically, ECG analysis has been performed by healthcare professionals, such as cardiologists, due to the complexity and variability of the signals and their interpretation. However, this process is time-consuming, costly, and highly dependent on the expertise of the specialist. The adoption of *Deep Learning* models, such as convolutional neural networks (CNNs), presents an opportunity to automate and enhance this analysis, providing rapid results that, in many cases, are comparable to those achieved by human experts (Hannun et al., 2019). This approach promises not only to improve diagnostic efficiency but also to enhance accuracy and reduce the workload of medical personnel.

Nevertheless, medical professionals have shown reluctance to adopt these technologies, as many predictive models function as black boxes, offering no insight into their decision-making processes. To foster trust in clinical settings and meet quality and ethical standards, it is essential for these tools to be explainable (Molnar, 2019). The explainability of AI models enables the understanding and justification of the decisions made during the analysis of cardiac signals—a critical factor in high-stakes applications such as medicine, where errors can directly impact patient health (Goodman y Flaxman, 2017).

While significant research has been conducted on the analysis and risk prediction from ECG's (e.g., Ribeiro et al. (2020)), fewer studies have focused on developing explainable models. As noted above, explainability is crucial for ensuring these models' practical use. Explainability in time-series classification models, such as those applied to different ECG leads, remains an area of ongoing research. However, more progress has been made in image-based explainability methods. For this reason, this study will examine certain transformations of ECG data into the time/frequency domain for two purposes:

- To assess whether classification accuracy improves by extracting more signal features in both the time domain (standard ECG) and the frequency domain.
- To represent these transformations as images to evaluate whether this approach enhances explainability.

Throughout this work, we will explore various neural network architectures and signal transformation techniques to classify and detect anomalies in ECG's, aiming to develop and refine a classifier for ECG's across four categories of cardiac anomalies. Additionally, we will incorporate explainability methods to ensure that the model's results can be interpreted and validated by healthcare professionals, which is essential for the clinical adoption of these technologies.

6.6. Objectives

The primary objective of this work is to modify, train, and evaluate the model initially proposed by Ribeiro et al. (2020) (henceforth referred to as the "*gold standard*") using a public dataset. To achieve this, we will introduce a modification to the model's input layer to incorporate mathematical transformations of the signals, allowing the extraction of features not only in the time domain but also in the frequency domain (e.g., STFT or CWT). This aims to explore whether converting the signal into various representations can improve performance. However, no changes will be made to the model's internal architecture.

Based on the above, the specific objectives are:

1. **Train the *gold standard*** using a public dataset.
2. **Modify the *gold standard*** to support signal transformations, enabling the training of alternative versions of the model with transformed data.

3. **Compare the performance** of each variant with the *gold standard* using metrics such as F1-Score to determine whether the transformations are beneficial.
4. **Apply an explainability method** to enable a specialist to understand which parts of an ECG signal influence the predictions.

6.7. Document Structure

This document is organized into six chapters, described as follows:

- **Chapter 2: Estado del Arte**

Introduces the basic concepts of electrocardiograms, emphasizing the importance of their waves (P, QRS, T) and the most common anomalies detected. It also reviews the literature on the use of neural networks for ECG analysis and describes relevant databases (such as PTB-XL).

- **Chapter 3: Metodología y Preparación de Datos**

Details the process of processing ECG's, including the division into training, validation, and testing sets. Additionally, it describes the metrics used for model evaluation and the transformations applied.

- **Chapter 4: Entrenamiento y Resultados**

Explains the training conditions for each model (including modifications to the *gold standard*), the libraries used, and presents the quantitative results obtained through the metrics.

- **Chapter 5: Explicabilidad y Validación Clínica**

Describes the explainability method based on gradient saliency maps applied to Ribeiro's model. Additionally, it includes the perspective of a medical specialist who analyzes the generated explanations and reflects on the method and potential improvements.

- **Chapter 6: Conclusiones y Trabajo Futuro**

Synthesizes the conclusions drawn from the results, evaluating the extent to which the objectives were met. It also highlights the main contributions of this work and proposes future research directions (such as the inclusion of Conv2D layers).

Finally, a bibliography section lists all sources consulted throughout the document, along with an appendix containing the code used¹ and another appendix with the complete list of libraries installed in the *Python* environment used to execute the code.

¹All content from this work can be found in the following GitHub repository: <https://github.com/NotNoe/TFG-Info>.

Chapter 7

Conclusions and Future Work

7.1. Conclusions

Thanks to the research and discussions with experts from various fields conducted during this project, we have been able to draw certain conclusions regarding ECG analysis, the potential of AI models for ECG classification, and their explainability. This section compiles these conclusions.

7.1.1. General Conclusions

- We have discovered that ECG analysis is far more complex than initially anticipated due to the significant subtleties required to classify them correctly.
- Proper preprocessing of signals before working with them is a key step. This also involves several subtleties, such as selecting the appropriate filter types and parameters to ensure the ECG is as free from interference as possible.
- Based on the results of the best models studied in the state of the art, it is evident that, although these models produce good results, we are further away than initially thought from achieving a model capable of automating the detection of ECG anomalies without professional intervention.
- As is common in machine learning models, hundreds of thousands of test cases are necessary to train reliable models. In the case of ECG's, although relatively large datasets exist, they are often not publicly available and are inconsistently structured, which complicates research. Significant progress in this field requires a publicly available, uniformly structured dataset (such as PTB-XL) with hundreds of thousands of cases.
- Models are ineffective at classifying rare diseases (e.g., hypertrophy). This is because these anomalies, being less common, have relatively few test cases in the datasets. This issue is exacerbated by the fact that the datasets are not sufficiently large. This suggests that we may never develop models capable of

effectively and reliably classifying all anomalies. Hybrid models that detect common diseases using machine learning and rare conditions through other methods could be considered as a solution.

7.1.2. Conclusions on the Trained Models

- We have demonstrated that Ribeiro’s model performs adequately even when trained on a relatively small dataset, highlighting its robustness and promise.
- When introducing transformations, we discovered that Ribeiro’s model architecture is not suitable for handling images, such as transformed signals. This does not mean that Ribeiro’s model cannot be adapted to work effectively with images, as its excellent performance makes it a highly promising candidate.
- It is important to note that the field of *Deep Learning* applied to medicine is relatively new, with innovative research and advancements being published frequently. Thus, the use of transformations for ECG analysis remains a potential avenue for progress.

7.1.3. Conclusions on Model Explainability

- Based on discussions with medical experts, we found that maintaining the traditional format is imperative when explaining ECG’s, as medical professionals are accustomed to analyzing signals printed on graph paper at a specific scale, which is necessary for efficient and accurate analysis.
- Gradient-based *saliency maps* have proven to be a promising tool for understanding which parts of the ECG signal are relevant to the network’s predictions.
- Additionally, explanations through *saliency maps* have also shown potential as educational tools for both students and healthcare professionals seeking to familiarize themselves with AI. However, further advancements are needed before these tools can be fully utilized in educational contexts.

7.2. Future Work

Given the potential of the model and the identified limitations, several avenues for improvement and further exploration are proposed:

1. Modify the architecture to include Conv2D layers

While this adaptation would require significant effort, it would better leverage the advantages of signal transformations.

2. Use image analysis models for ECG classification

Although Ribeiro's model is highly promising for ECG classification, image classification models might be more efficient when analyzing transformed signals than potential modifications of Ribeiro's model.

3. Explore additional transformations and parameters

Following the previous point, once a more suitable model for images is established (either through a modification of Ribeiro's architecture or a different model), other parameters for transformations could be tested, and their impact analyzed.

4. Specialize the model in a specific anomaly

Transforming the model into a binary classifier would allow the transformation parameters to be more precisely tailored, as different anomalies manifest in different frequency ranges.

5. Adjust detection thresholds

In Ribeiro's work, instead of using a threshold of 0.5 for label classification, thresholds were calculated to optimize validation set performance. While this was not addressed in this study, it could improve the results of all the trained models.

6. Signal delineation for explanations

As noted by experts, signal delineation could reduce explanations to two heartbeats, ensuring homogeneity across both beats. These improvements would make these explanations viable for educational purposes.

7.3. Main Personal Contributions

The main contributions to the field of explainable ECG classification using *Deep Learning* presented in this work are as follows:

1. Adjustment of Ribeiro's model

We adjusted Ribeiro's model to enable training with data of any dimensionality, allowing it to be trained with images of any resolution.

2. Design of suitable transformations for ECG analysis

We designed mathematical transformations of the original ECG signals to capture the frequencies associated with the anomalies we aimed to detect.

3. Development of a custom metric

We developed a specific metric for this problem that accounts for the varying importance of minimizing false negatives or positives across different classes.

4. Use of TSInterpret for ECG analysis

We successfully used the TSInterpret library to generate suitable explanations for anomaly detection in ECG's. Additionally, we confirmed with medical experts that these explanations are coherent and appropriate.

5. Standardization of explanations

We applied the aforementioned explanations successfully to an ECG in a standard format, making them significantly more readable (and thus useful) for healthcare professionals.

Bibliografía

- ACHARYA, U. R., FUJITA, H., LIH, O. S., HAGIWARA, Y., TAN, J. H. y CHUA, C. Automated diagnosis of arrhythmia using different intervals of ECG segments with Convolutional Neural Network. *Information Sciences*, vol. 405, páginas 81–90, 2017.
- ALLEN, J. B. y RABINER, L. R. A unified approach to short-time fourier analysis and synthesis. *Proceedings of the IEEE*, vol. 65(11), páginas 1558–1564, 1977.
- CHOUHAN, V. y MEHTA, S. Total removal of baseline drift from ecg signal. En *2007 International Conference on Computing: Theory and Applications (ICCTA'07)*, páginas 512–515. 2007.
- GONZÁLEZ CABEZA, S. Are 12-lead ecgs necessary for detecting heart anomalies? a comparison between 12-lead and 3-lead ecg classification using deep learning, 2024. No Publicado.
- GONZÁLEZ, L. E. A., S., J. L. R. y CASTELLANOS, G. Métodos para la eliminación de interferencia ac en ecg. *Scientia et Technica*, vol. 3(29), páginas 49–53, 2005. ISSN 0122-1701. Accedido el 5 de diciembre de 2024.
- GOODFELLOW, I., BENGIO, Y. y COURVILLE, A. *Deep Learning*. MIT Press, 2016.
- GOODMAN, B. y FLAXMAN, S. European union regulations on algorithmic decision making and a “right to explanation”. *AI Magazine*, vol. 38(3), página 50–57, 2017. ISSN 2371-9621.
- GUO, C., AHMED, S. y ALOUINI, M.-S. Machine learning-based automatic cardiovascular disease diagnosis using two ecg leads. *arXiv preprint arXiv:2305.16055*, 2023.
- GUSTAFSSON, S., GEDON, D., LAMPA, E., RIBEIRO, A. H., HOLZMANN, M. J., SCHÖN, T. B. y SUNDSTRÖM, J. Development and validation of deep learning ecg-based prediction of myocardial infarction in emergency department patients. *Scientific Reports*, vol. 12, 2022.
- HANNUN, A. Y., RAJPURKAR, P., HAGHPANAH, M., TISON, G. H., BOURN, C., TURAKHIA, M. P. y NG, A. Y. Cardiologist-level arrhythmia detection

- and classification in ambulatory electrocardiograms using a deep neural network. *Nature Medicine*, vol. 25(1), páginas 65–69, 2019.
- HE, H., BAI, Y., GARCIA, E. A. y LI, S. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. En *Proceedings of the IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, páginas 1322–1328. IEEE, Hong Kong, China, 2008.
- HE, H. y GARCIA, E. A. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, vol. 21(9), páginas 1263–1284, 2009.
- KOZIARSKI, M. Radial-based undersampling for imbalanced data classification. *arXiv preprint arXiv:1906.00452*, 2019.
- LEEVY, J. L., KHOSHGOFTAAR, T. M., BAUDER, R. A. y SELIYA, N. A survey on addressing high-class imbalance in big data. *Journal of Big Data*, vol. 5(1), página 42, 2018.
- LIBBY, P., BONOW, R., MANN, D., TOMASELLI, G., BHATT, D., SOLOMON, S. y BRAUNWALD, E. *Braunwald's Heart Disease: A Textbook of Cardiovascular Medicine*. Elsevier, 2021. ISBN 9780323824712.
- LU, L., ZHU, T., RIBEIRO, A. H. ET AL. Decoding 2.3 million ecgs: interpretable deep learning for advancing cardiovascular diagnosis and mortality risk stratification. *European Heart Journal - Digital Health*, vol. 5(3), páginas 247–259, 2024.
- MOLNAR, C. *Interpretable Machine Learning*. Lulu.com, 2019. ISBN 9780244768522.
- MYEKG. Ondas del electrocardiograma. ???? Accedido: 2024-01-20, URL: <https://www.my-ekg.com/generalidades-ekg/ondas-electrocardiograma.html>.
- OPPENHEIM, A. V. y SCHAFER, R. W. *Discrete-Time Signal Processing*. Prentice Hall, Upper Saddle River, NJ, USA, 3rd edición, 2009.
- RIBEIRO, A. H., PAIXAO, G. M., LIMA, E. M., HORTA RIBEIRO, M., PINTO FILHO, M. M., GOMES, P. R., OLIVEIRA, D. M., MEIRA JR, W., SCHON, T. B. y RIBEIRO, A. L. P. CODE-15 %: a large scale annotated dataset of 12-lead ECGs. <https://doi.org/10.5281/zenodo.4916206>, 2021a.
- RIBEIRO, A. H., RIBEIRO, M. H., PAIXÃO, G. M. M., OLIVEIRA, D. M., GOMES, P. R., CANAZART, J. A., FERREIRA, M. P. S., ANDERSSON, C. R., MACFARLANE, P. W., MEIRA JR., W., SCHÖN, T. B. y RIBEIRO, A. L. P. Automatic diagnosis of the 12-lead ECG using a deep neural network. *Nature Communications*, vol. 11(1), página 1760, 2020. Aviable at <https://doi.org/10.1038/s41467-020-15432-4>.
- RIBEIRO, A. H., RIBEIRO, M. H., PAIXÃO, G. M., OLIVEIRA, D. M., GOMES, P. R., CANAZART, J. A., FERREIRA, M. P., ANDERSSON, C. R., MACFARLANE, P. W., JR., W. M., SCHÖN, T. B. y RIBEIRO, A. L. P. CODE dataset. https://figshare.scilifelab.se/articles/dataset/CODE_dataset/15169716, 2021b.

- RUMELHART, D. E., HINTON, G. E. y WILLIAMS, R. J. Learning representations by back-propagating errors. *Nature*, vol. 323(6088), páginas 533–536, 1986.
- WAGNER, P., STRODTHOFF, N., BOUSSELJOT, R.-D., SAMEK, W. y SCHAEFFTER, T. PTB-XL, a large publicly available electrocardiography dataset. <https://doi.org/10.1038/s41597-020-0495-6>, 2020. Último acceso: 28/09/2024.
- WAGNER, P., STRODTHOFF, N., BOUSSELJOT, R.-D., SAMEK, W. y SCHAEFFTER, T. PTB-XL, a large publicy available electrocardiography dataset. <https://doi.org/10.13026/kfzx-aw45>, 2022. (version 1.0.3). Último acceso: 28/09/2024.
- WIKIMEDIA COMMONS. Ekg leads. 2015. Accedido: 2024-01-20, licensed under CC BY-SA 3.0 URL: https://commons.wikimedia.org/wiki/File:EKG_leads.png#.
- WIKIMEDIA COMMONS. Sinus rhythm labels. 2023. Accedido: 2024-12-26, licenced under CC BY-SA 3.0 URL: <https://commons.wikimedia.org/wiki/File:SinusRhythmLabels.svg>.
- WORLD HEALTH ORGANIZATION. Cardiovascular diseases (cvds). [https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-\(cvds\)](https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds)), 2021. [Online; accessed 2023-10-10].
- XIE, C., MCCULLUM, L., JOHNSON, A., POLLARD, T., GOW, B. y MOODY, B. Waveform database software package (wfdb) for python (version 4.1.0). PhysioNet, 2023.
- YANMIN SUN, WONG, A. y KAMEL, M. S. Classification of imbalanced data: a review. *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 23, 2011.

Apéndice A

Código

```
1 import json
2 from math import ceil
3 import os
4 import numpy as np
5 import h5py
6 import ecg_plot
7 import matplotlib.pyplot as plt
8
9 def plot_ecg_with_explanations(N_ECG, explanation_path):
10     # Cargar el vector de numpy
11     explicaciones = np.load(os.path.join(explanation_path, "explanation.npy")).T
12     with h5py.File("./data/test.hdf5") as f:
13         ecg = f["tracings"][N_ECG-2].T
14         #El título contiene el id del ECG y la etiqueta
15         ecg_plot.plot(ecg, sample_rate=400, title=f"ECG_ID: {explanation_path.split('/')[-1]} ({explanation_path.split('/')[-2]})", style="bw")
16     fig = plt.gcf()
17     fig.subplots_adjust(
18         left = 0.1,
19         right=0.95,
20         bottom=0.1,
21         top=0.95
22     )
23     #Quitamos las etiquetas de los ejes, que se concentran
24     #demasiado
25     ax = fig.axes[0]
26     ax.tick_params(
27         labelbottom=False,
28         labelleft=False,
29         bottom=False,
30         left=False,
31         length=0
32     )
33     #En esencia replicamos el código que hace la función plot (que
34     #calcula offsets para saber donde dibujar)
35     #para luego poner el heatmap en la misma posición
```

```

34     sample_rate = 400
35     secs = ecg.shape[1] / sample_rate
36
37     columns = 2
38     leads = ecg.shape[0] #Debería de ser doce
39     rows = int(ceil(leads / columns))
40     row_height = 6
41     lead_order = list(range(12))
42
43     for c in range(columns):
44         for i in range(rows):
45             #Eso es para cada derivación
46             idx = c * rows + i
47             if idx >= leads:
48                 break
49
50             t_lead = lead_order[idx]
51
52             # === Mismos offsets que en plot
53             x_offset = secs * c
54             y_offset = -(row_height / 2) * (i % rows)
55
56             # Preparamos la imagen que vamos a pintar po encima
57             exp_lead = explicaciones[t_lead]           # (4096,)
58             exp_lead_2d = exp_lead.reshape(1, -1)      # (1, 4096)
59
60             # === Definimos el rectángulo (extent) donde se pintará
61             # esa franja ===
62             local_x_min = x_offset
63             local_x_max = x_offset + secs
64
65             # Escogemos el ancho de la franja para que quede bien.
66             local_y_min = y_offset - 1.5
67             local_y_max = y_offset + 1.5
68
69             # Dibujamos el heatmap con imshow
70             # zorder=-1 para que quede POR DEBAJO de la onda ECG
71             # alpha=0.5 (p.ej.) para dejar ver la línea
72             heatmap = ax.imshow(
73                 exp_lead_2d,
74                 extent=(local_x_min, local_x_max, local_y_min,
75                         local_y_max),
76                 cmap='Reds',
77                 origin='lower',
78                 aspect='auto',
79                 alpha=0.5, #Para que no tape el resto de cosas
80                 zorder=-1 #Para que quede por debajo de la onda
81             )
82
83             # Añadimos una barra de color
84             cbar = fig.colorbar(heatmap, ax=ax, shrink=1)
85             cbar.set_label("Relevancia")
86             plt.savefig(os.path.join(explanation_path, "better_explanation.
87             png"))

```

Listing A.1: Código empleado para pintar las explicaciones sobre la imagen generada por *ecg_plot*.

```

1 import json
2 import os
3 import numpy as np
4 import matplotlib.pyplot as plt
5 import tensorflow as tf
6 import h5py
7 import pandas as pd
8 import sys
9
10 #Nombres de las etiquetas
11 label_names = ["CD", "HYP", "MI", "NORM", "STTC"]
12
13 # --- Carga de datos ---
14 # Cargamos los datos de pruebas
15 test_x = h5py.File('./data/test.hdf5', 'r')["tracings"]
16
17 # Cargamos las etiquetas desde el CSV
18 df = pd.read_csv('./data/test_db.csv')
19 test_y = df.drop(columns=['ecg_id']).to_numpy()
20
21 # --- Ejecución del modelo ---
22
23 # Obtenemos las predicciones mediante el script original de Ribeiro
24 os.system("python ./ribeiro/predict.py data/test.hdf5 final_models/
    original_model.hdf5 --output_file ./tmp.npy --dataset_name
    tracings")
25
26 # Las cargamos
27 predictions = np.load("./tmp.npy")
28
29 # Y las convertimos a booleanos
30 predictions = predictions > 0.5
31
32 # --- Comparación de resultados ---
33
34 # Comprobamos que la forma de las predicciones sea correcta
35 if predictions.shape[0] != test_y.shape[0]:
36     print("The number of test samples and predictions do not match"
          )
37     sys.exit(1)
38 if predictions.shape[1] != test_y.shape[1] or predictions.shape[1]
39     != 5:
40     print("The number of classes is not 5")
41     sys.exit(1)
42
43 #Creamos un fichero out.txt donde se guardaran los resultados, si
44 # ya existe lo borramos
45 lista_de_indices = {}
46 with open("out.json", "w") as f:
47     # Para cada una de las etiquetas, generamos el vector que
48     # representa la predicción y valor real que necesita para ser

```

```

46     "adecuada" para explicarla
47     for label_idx in range(5):
48         #Desired_real es el vector que tiene 1.0 en la posicion
49         #label_idx y 0.0 en las demás
50         desired_real = np.zeros(5, dtype=np.float32)
51         desired_real[label_idx] = 1.0
52         desired_predictions = [False, False, False, False, False]
53         desired_predictions[label_idx] = True
54         desired_predictions = np.asarray(desired_predictions)
55         lista_de_indices[label_names[label_idx]] = []
56         for i in range(predictions.shape[0]):
57             # Para cada predicción, comprobamos si coincide con los
58             # arrays de valores deseados
59             if np.array_equal(test_y[i], desired_real) and np.
60                 array_equal(predictions[i], desired_predictions):
61                 # Si coincide, guardamos la línea del CSV donde est
62                 # á (que corresponde al índice en el dataset más
63                 # dos) y el id del ECG
64                 linea": i + 2, "ecg_id": int(df.iloc[i]['ecg_id'])
65             })
66         json.dump(lista_de_indices, f, indent=4)
67 os.remove("./tmp.npy")

```

Listing A.2: Código empleado para buscar los ECGs adecuados para explicaciones.

```

1  from scipy.signal import stft, cwt, ricker, morlet2
2  import numpy as np
3
4  FREQ = 400
5  #Creamos la clase para las transformaciones
6  class Transformaciones:
7      #El constructor recibe los datos, que son una cantidad de
8      #matrices de 4096x12
9      def __init__(self, ecgs):
10          self.ecgs = ecgs
11          self.n_cases = ecgs.shape[0]
12
13      def get_cwt_arrays(self, wavelet = ricker, scales = np.linspace
14          (1,128,100)):
15          #Calculamos las dimensiones de la matriz de salida usando
16          #la primera matriz de entrada
17          if wavelet == ricker:
18              f = FREQ / (2*np.pi*scales)
19          elif wavelet == morlet2:
20              f = (5 * FREQ) / (2 * np.pi * scales)
21          else:
22              raise NotImplementedError("Wavelet no reconocido")
23
24          t = np.arange(self.ecgs[0].shape[0]) / FREQ
25          return f,t
26
27      def cwt(self, wavelet = ricker, scales = np.linspace(1,128,
28          100)):
29          cwts = []

```

```

27     for i in range(self.n_cases):
28         ecg = self.ecgs[i]
29         cwt_i = np.zeros((len(scales) * ecg.shape[1], ecg.shape
30                           [0]))
31         for j in range(ecg.shape[1]): #Iteramos sobre las
32             derivaciones
33             Zxx = np.abs(cwt(ecg[:,j], wavelet=wavelet, widths=
34                           scales))
35             cwt_i[j * len(scales):(j + 1) * len(scales), :] =
36             Zxx
37             cwts.append(cwt_i)
38     return np.array(cwts)
39
40
41     def get_stft_arrays(self, nperseg=256, nooverlap=128):
42         #Calculamos las dimensiones de la matriz de salida usando
43         #la primera matriz de entrada
44         f, t, _ = stft(self.ecgs[0][:, 0], fs=FREQ, nperseg=nperseg,
45                         nooverlap=nooverlap)
46         return f, t
47
48     #Función que realiza la transformada STFT
49     def stft(self, nperseg=256, nooverlap=128):
50         #Calculamos las dimensiones de la matriz de salida usando
51         #la primera matriz de entrada
52         n_frequencies, n_times = stft(self.ecgs[0][:, 0], fs=FREQ,
53                                         nperseg=nperseg, nooverlap=nooverlap)[2].shape
54         #Creamos un array vacío para guardar los resultados
55         stfts = []
56         #Iteramos sobre los datos
57         for i in range(self.n_cases):
58             ecg = self.ecgs[i]
59             stft_i = np.zeros((n_frequencies * ecg.shape[1],
60                               n_times))
61             for j in range(ecg.shape[1]):
62                 _, _, Zxx = stft(ecg[:, j], fs=FREQ, nperseg=
63                                 nperseg, nooverlap=nooverlap)
64                 stft_i[j * n_frequencies:(j + 1) * n_frequencies,
65                         :] = np.abs(Zxx)
66             stfts.append(stft_i)
67     return np.array(stfts)

```

Listing A.3: Clase creada para hacer las transformaciones de los datos.

```

1 import numpy as np
2 import json
3 from sklearn.metrics import precision_recall_fscore_support,
4                             fbeta_score
5
6 class Metrics:
7     def __init__(self, y_true, y_pred, class_names=None):
8         """
9             Inicializa la clase Metrics con matrices de etiquetas
10            reales y predicciones.
11
12     Parameters:
13         y_true (np.ndarray): Array de valores reales (

```

```

    probabilidades entre 0 y 1).
12                               Dimensión: (n_samples, n_classes)
13 y_pred (np.ndarray): Array de predicciones (probabilidades
14                               entre 0 y 1).
15                               Dimensión: (n_samples, n_classes)
16 class_names (list, opcional): Lista con los nombres de las
17                               clases.
18                               Longitud: n_classes
19 """
20
21 self.y_true_prob = y_true
22 self.y_pred_prob = y_pred
23 self.class_names = class_names
24 self.n_classes = y_true.shape[1]
25
26
27 # Binarizar las etiquetas reales y predichas usando un
28 # umbral de 0.5
29 self.y_true = (self.y_true_prob >= 0.5).astype(int)
30 self.y_pred = (self.y_pred_prob >= 0.5).astype(int)
31
32
33 # Verificar si alguna clase no tiene representantes en
34 # y_true
35 self.classes_with_no_samples = []
36 for i in range(self.n_classes):
37     y_true_sum = np.sum(self.y_true[:, i])
38     if y_true_sum == 0:
39         if self.class_names:
40             class_name = self.class_names[i]
41         else:
42             class_name = f'Clase_{i}'
43         self.classes_with_no_samples.append(class_name)
44
45 if self.classes_with_no_samples:
46     print(f"Advertencia: Las siguientes clases no tienen
47           representantes en la muestra de test: {', '.join(
48           self.classes_with_no_samples)}")
49
50
51 def calculate_precision_recall_f1(self):
52 """
53     Calcula las métricas de precisión, recall y F1-score para
54     cada clase y de manera global.
55
56 Returns:
57 dict: Diccionario con las métricas calculadas.
58 """
59
60 precision_dict = {}
61 recall_dict = {}
62 f1_dict = {}
63
64 precision_list, recall_list, f1_list, _ =
65     precision_recall_fscore_support(self.y_true, self.y_pred,
66     zero_division=np.nan)
67 for i in range(self.n_classes): #Para cada clase
68     if self.class_names:
69         class_name = self.class_names[i]
70     else:
71         class_name = f'Clase_{i}'
```

```

58     precision_dict[class_name] = precision_list[i]
59     recall_dict[class_name] = recall_list[i]
60     f1_dict[class_name] = f1_list[i]
61
62
63     global_precision, global_recall, global_f1, _ =
64         precision_recall_fscore_support(self.y_true, self.y_pred,
65                                         average='micro', zero_division=np.nan)
66
67     metrics = {
68         'precision': {
69             'by_class': precision_dict,
70             'global_average': global_precision
71         },
72         'recall': {
73             'by_class': recall_dict,
74             'global_average': global_recall
75         },
76         'f1_score': {
77             'by_class': f1_dict,
78             'global_average': global_f1
79         }
80     }
81
82     return metrics
83
84 def calculate_adjusted_f_score(self):
85     """
86     Calcula la métrica personalizada 'adjusted_f_score',
87     combinando F0.5 para 'NORM' y F2 para las demás clases.
88
89     Returns:
90     float or None: Valor de la métrica personalizada.
91     """
92
93     if self.class_names is None:
94         clase_especial = 3
95         print("Warning: No se han proporcionado los nombres de
96              las clases. Se asumirá que la clase NORM es la clase
97              3.")
98     else:
99         clase_especial = self.class_names.index('NORM')
100
101     metricas = []
102     for cls in range(self.n_classes):
103         if cls == clase_especial:
104             beta = 0.5
105         else:
106             beta = 2
107             f_score = fbeta_score(self.y_true[:,cls], self.y_pred
108                                   [:,cls], beta=beta)
109             metricas.append(f_score)
110     adjusted_f_score = np.mean(metricas)
111
112     return adjusted_f_score
113
114
115     def dump_to_json(self, path):
116         """
117

```

```
108     Escribe las métricas calculadas en un archivo JSON.
109
110     Parameters:
111     path (str): Ruta al archivo JSON de salida.
112     """
113     # Crear un diccionario ordenado para asegurar el orden de
114     # las métricas
115     metrics = {}
116
116     # Calcular la métrica personalizada y agregarla primero
117     adjusted_f_score = self.calculate_adjusted_f_score()
118     metrics['adjusted_f_score'] = adjusted_f_score
119
120     # Calcular las demás métricas
121     prf_metrics = self.calculate_precision_recall_f1()
122     metrics.update(prf_metrics)
123
124     # Escribir en el archivo JSON
125     with open(path, 'w') as json_file:
126         json.dump(metrics, json_file, indent=4)
```

Listing A.4: Clase creada para generar métricas y guardarlas en un json.

Apéndice B

Librerías utilizadas

Librería	Versión
_libgcc_mutex	0.1=main
_openmp_mutex	5.1=1_gnu
bzip2	1.0.8=h7b6447c_0
ca-certificates	2023.12.12=h06a4308_0
ld_impl_linux-64	2.38=h1181459_1
libffi	3.4.4=h6a678d5_0
libgcc-ng	11.2.0=h1234567_1
libgomp	11.2.0=h1234567_1
libstdcxx-ng	11.2.0=h1234567_1
libuuid	1.41.5=h5eee18b_0
ncurses	6.4=h6a678d5_0
openssl	3.0.13=h7f8727e_0
pip	23.3.1=py311h06a4308_0
python	3.11.7=h955ad1f_0
readline	8.2=h5eee18b_0
setuptools	68.2.2=py311h06a4308_0
sqlite	3.41.2=h5eee18b_0
tk	8.6.12=h1ccaba5_0
wheel	0.41.2=py311h06a4308_0
xz	5.4.5=h5eee18b_0
zlib	1.2.13=h5eee18b_0
absl-py	2.1.0
aiohttp	3.9.3
aiosignal	1.3.1
asttokens	2.4.1
astunparse	1.6.3
attrs	23.2.0
autograd	1.7.0
beautifulsoup4	4.12.3

Librería	Versión
bleach	6.1.0
cachetools	5.3.2
captum	0.7.0
certifi	2024.2.2
cffi	1.16.0
charset-normalizer	3.3.2
cloudpickle	3.1.0
comm	0.2.1
contourpy	1.2.0
cycler	0.12.1
deap	1.4.1
debugpy	1.8.0
decorator	5.1.1
defusedxml	0.7.1
deprecated	1.2.13
ecg-plot	0.2.8
et-xmlfile	1.1.0
executing	2.0.1
fastjsonschema	2.19.1
filelock	3.13.4
flatbuffers	23.5.26
fonttools	4.48.1
frozenlist	1.4.1
fsspec	2024.3.1
gast	0.5.4
google-auth	2.27.0
google-auth-oauthlib	1.2.0
google-pasta	0.2.0
grpcio	1.60.1
h5py	3.10.0
idna	3.6
ipykernel	6.29.2
ipython	8.21.0
jedi	0.19.1
jinja2	3.1.3
joblib	1.3.2
jsonschema	4.21.1
jsonschema-specifications	2023.12.1
jupyter-client	8.6.0
jupyter-core	5.7.1
jupyterlab-pygments	0.3.0
keras	2.15.0
kiwisolver	1.4.5
libclang	16.0.6

Librería	Versión
lightning-utilities	0.11.2
llvmlite	0.43.0
locket	1.0.0
markdown	3.3.4
markupsafe	2.1.5
matplotlib	3.8.2
matplotlib-inline	0.1.6
mistune	3.0.2
ml-dtypes	0.2.0
mpmath	1.3.0
multidict	6.0.5
nbclient	0.9.0
nbconvert	7.16.1
nbformat	5.9.2
nest-asyncio	1.6.0
networkx	3.3
numba	0.60.0
numpy	1.26.4
nvidia-cublas-cu12	12.1.3.1
nvidia-cuda-cupti-cu12	12.1.105
nvidia-cuda-nvcc-cu12	12.2.140
nvidia-cuda-nvrtc-cu12	12.1.105
nvidia-cuda-runtime-cu12	12.1.105
nvidia-cudnn-cu12	8.9.2.26
nvidia-cufft-cu12	11.0.2.54
nvidia-curand-cu12	10.3.2.106
nvidia-cusolver-cu12	11.4.5.107
nvidia-cusparse-cu12	12.1.0.106
nvidia-nccl-cu12	2.19.3
nvidia-nvjitlink-cu12	12.2.140
nvidia-nvtx-cu12	12.1.105
oauthlib	3.2.2
opencv-python	4.6.0.66
openpyxl	3.1.2
opt-einsum	3.3.0
packaging	23.2
pandas	2.2.0
pandocfilters	1.5.1
parso	0.8.3
partd	1.2.0
patsy	1.0.1
pexpect	4.9.0
pillow	10.2.0
platformdirs	4.2.0

Librería	Versión
prompt-toolkit	3.0.43
protobuf	4.23.4
psutil	5.9.8
ptyprocess	0.7.0
pure-eval	0.2.2
pyaml	24.9.0
pyarrow	15.0.2
pyasn1	0.5.1
pyasn1-modules	0.3.0
pycparser	2.21
pygments	2.17.2
pymop	0.2.4
pyparsing	3.1.1
python-dateutil	2.8.2
pytorch-lightning	2.2.1
pyts	0.13.0
pytz	2024.1
pyyaml	6.0.1
pyzmq	25.1.2
referencing	0.33.0
requests	2.31.0
requests-oauthlib	1.3.1
rpds-py	0.18.0
rsa	4.9
scikit-base	0.11.0
scikit-learn	1.3.0
scikit-optimize	0.10.2
scipy	1.14.1
seaborn	0.13.2
shap	0.46.0
six	1.16.0
sktime	0.34.1
slicer	0.0.8
soundfile	0.12.1
soupsieve	2.5
stack-data	0.6.3
statsmodels	0.14.4
stumpy	1.13.0
sympy	1.12
tensorboard	2.15.1
tensorboard-data-server	0.7.2
tensorflow	2.15.0.post1
tensorflow-estimator	2.15.0
tensorflow-io-gcs-filesystem	0.36.0

Librería	Versión
termcolor	2.4.0
tf-explain	0.3.1
threadpoolctl	3.2.0
tinycc2	1.2.1
toolz	1.0.0
torch	2.2.2
torchcam	0.4.0
torchmetrics	1.3.2
tornado	6.4
tqdm	4.65.2
traitlets	5.14.1
triton	2.2.0
tsfresh	0.20.3
tsinterpret	0.4.7
tslearn	0.6.3
typing-extensions	4.9.0
tzdata	2023.4
urllib3	2.2.0
wcwidth	0.2.13
webencodings	0.5.1
werkzeug	3.0.1
wfdb	4.1.2
wrapt	1.14.1
xarray	2024.3.0
xmljson	0.2.1
yarl	1.9.4

Lista de acrónimos

STFT..... Transformada de Fourier de tiempo reducido (*Short-Time Fourier Transform*)

CWT..... Transformada de onda continua (*Continuous Wavelet Transform*)

ECG..... Electrocardiograma

IA..... Inteligencia Artificial

OMS..... Organización Mundial de la Salud

