PROYECTO FINAL INTEGRADOR

Redes neuronales convolucionales para la detección de eventos epileptogénicos en señales de electroencefalografía intracraneal

Estudiante
María Belén Bächli

Legajo
CyT 7967

Ingeniería Biomédica

Escuela de Ciencia y Tecnología
Universidad Nacional de San Martín

Supervisores
Enzo Ferrante
Victoria Peterson

Lugar de trabajo
Instituto de Investigación en Señales, Sistemas e Inteligencia Computacional sinc(i) - Universidad Nacional del Litoral

Fecha
Octubre 2022

"Declara el pasado, diagnostica el presente, pronostica el futuro."

- Hipócrates

Agradecimientos

Quiero aprovechar esta oportunidad para agradecer a todos los que hicieron que hoy esté presentando este proyecto final de la carrera.

A la universidad, y a los docentes que tuve, por darme la oportunidad de formarme y de aprender en todo momento. Quiero hacer un especial agradecimiento a todo el personal de la UNSAM que hizo que disfrute tanto de la facultad, de su campus y de cada momento en estos años, incluso en virtualidad.

A mis supervisores, que confiaron ciegamente en mí y me ofrecieron un tema de proyecto que me atrapó y disfruté mucho desarrollando. Gracias por todas las reuniones, las charlas, las correcciones y los consejos. También quiero agradecer a la gente del laboratorio del sinc, que siempre estuvieron para ayudarme con cualquier problema que tuve con el servidor que utilice durante el proyecto y a la gente del laboratorio *Brain Modulation Lab*, que me permitieron acceso a los datos que fueron la base del proyecto.

A Agustina, quien me acompañó y me apoyó durante este último tramo de la carrera, animándome a confiar en mis decisiones y en mis ideas. Gracias Agus por todos esos viernes que estuviste para escucharme con tu paciencia infinita.

A mis amigos, que sin ellos hubiera sido imposible atravesar todos estos años. Mención especial a Andy, fiel compañera de mates, viajes en el 21, resúmenes o trabajos prácticos, y que sin ella no hubiera podido disfrutar tanto mi época de facultad.

A Juli, que me vio alegrarme y caerme en gran parte de la carrera. Que estuvo como compañero de estudio, de mates y de ruta, y que me banca en todas. Sin nuestras tardes de estudio no estaba hoy presentando este informe.

Por último y más importante a mi familia, que me apoyó durante toda la carrera y siempre me ayudó con lo que necesitara para alcanzar mis objetivos. A mis papás que nunca me dejaron bajar los brazos y que me dieron todo lo que tenían a su alcance para que yo pudiera avanzar. Gracias por confiar en mí.

Resumen

La epilepsia es una enfermedad neurológica que afecta a más de 50 millones de personas según la Organización Mundial de la Salud. Las actividades de la vida diaria de las personas diagnosticadas con este desorden pueden resultar muy restringidas, dado que el mismo se caracteriza por episodios convulsivos que pueden implicar movimientos involuntarios y pérdida de consciencia. Aunque es posible acceder a tratamientos farmacológicos o quirúrgicos, existen muchos casos en los que estos tratamientos no resultan exitosos o no son posibles. Para estos pacientes se han desarrollado terapias basadas en neuroestimulación, en las que los pacientes son implantados con dispositivos que generan descargas eléctricas para intentar revertir o prevenir el estado cerebral convulsivo.

Dentro de estos dispositivos de neuroestimulación se encuentra el sistema de neuroestimulación responsiva. Este dispositivo permite generar una terapia que responde al estado de actividad neuronal, produciendo descargas cuando detecta señales específicas en la zona en la que está implantado, y también permite obtener mediciones de electroencefalografía de la actividad neuronal del paciente en situaciones de la vida diaria. Para obtener una respuesta que resulte beneficiosa para el tratamiento de la patología, el dispositivo necesita ser personalizado para cada paciente, tarea que realizan los neurofisiólogos que realizan el seguimiento clínico a los mismos. Esto implica un proceso rutinario que requiere la visualización y análisis manual de las señales de electroencefalografía invasiva tomadas con el sistema de neuroestimulación, el cual debe ser llevado a cabo por especialistas en la patología, y que puede resultar ineficiente, costoso y extenso.

Es en este contexto que se propone el análisis de rendimiento y factibilidad del desarrollo de un método de inteligencia artificial que permita la detección automática de los episodios convulsivos, y sea capaz de predecir el tiempo de inicio de los patrones convulsivos sobre las señales de electroencefalografía intracraneal. Para cumplir con este objetivo se pensó en utilizar espectrogramas generados a partir de las señales y analizar los mismos mediante modelos de aprendizaje profundo. Específicamente, se buscó utilizar redes neuronales totalmente convolucionales, basadas en la arquitectura UNet, para aprender a evaluar los datos y obtener la información deseada de los mismos.

Como resultado de los experimentos realizados se encontró que es factible aplicar las técnicas propuestas para la detección de patrones convulsivos sobre los espectrogramas de señales de electroencefalografía intracraneal. Con la realización del proyecto se obtuvo un modelo optimizado con una capacidad de clasificar las muestras según

la presencia o no de patrones convulsivos con un 81.3 % de exactitud balanceada y se logró estimar el tiempo de inicio de los patrones convulsivos con un error absoluto medio de 6.2 segundos, obteniendo ambos valores en escenarios de simulación clínica.

Además, se logró obtener un método de detección que al ser totalmente convolucional permite el análisis de muestras de cualquier longitud temporal, permitiendo que el procesamiento previo se reduzca a la generación de los espectrogramas a partir de los datos crudos de electroencefalografía. Por otro lado, los resultados alcanzados fueron comparables con los encontrados en la bibliografía, lo cual permitió validar los desafíos y resultados hallados en este proyecto.

Palabras clave: Epilepsia, Electroencefalografía intracraneal, Patrones ictales, Redes neuronales artificiales, Redes neuronales convolucionales, Arquitectura UNet

Índice general

1. Introducción **1**
1.1. Aprendizaje automático e ingeniería biomédica 1
1.1.1. Aplicaciones biomédicas del aprendizaje automático 3
1.1.2. Aprendizaje profundo en electroencefalografía 4
1.2. Objetivos del proyecto 5
1.2.1. Motivación 6
1.3. Estructura del informe 7

2. Epilepsia y electroencefalografía **9**
2.1. Epilepsia 9
2.1.1. Mecanismos básicos convulsivos 10
2.1.2. Clasificación de la epilepsia 11
2.1.3. Tratamientos posibles 12
2.2. Sistemas de neuroestimulación 14
2.2.1. Estimulación del nervio vagal 15
2.2.2. Estimulación del núcleo anterior del tálamo 16
2.2.3. Estimulación responsiva 16
2.3. Electroencefalografía en el análisis de epilepsia 19
2.3.1. Generalidades de la señal de EEG 19
2.3.2. Registros en el EEG durante episodios convulsivos 20
2.3.3. Registros de EEG intracraneal 21
2.3.4. Desafíos en el análisis de patrones ictales en EEG 22
2.3.5. Espectrogramas de registros de EEG 22
2.4. Resumen del capítulo 23

3. Técnicas de aprendizaje automático **26**
3.1. Introducción 26
3.2. Aprendizaje profundo 27
3.2.1. Redes neuronales artificiales 28
3.2.2. Función de pérdida y optimización 30
3.2.3. Funciones de activación 32
3.2.4. Testeo y validación del modelo 34
3.2.5. Técnicas para evitar el sobreajuste 35
3.2.6. Redes neuronales convolucionales 37
3.3. Segmentación de imágenes 39
3.3.1. Arquitectura UNet para la segmentación de imágenes médicas . 40

3.3.2. Funciones de pérdida para evaluación de la segmentación 42
3.4. Resumen del capítulo 44

4. Datos, herramientas y métricas 46
4.1. Herramientas 46
4.1.1. Anaconda y Spyder 46
4.1.2. Pytorch y CUDA 47
4.1.3. Git y GitLab 47
4.2. Base de datos 48
4.2.1. Generación de espectrogramas 49
4.2.2. Desbalance en la base de datos 49
4.2.3. Armado de los conjuntos de datos 50
4.3. Métricas de evaluación de los resultados 52
4.3.1. Clasificación de las señales 52
4.3.2. Curvas Operador-Receptor 55
4.3.3. Cálculo del tiempo de inicio ictal 56
4.4. Resumen del capítulo 57

5. Diseño y optimización del modelo 59
5.1. Diseño y desarrollo experimental 59
5.1.1. Arquitectura utilizada 60
5.1.2. Datos y etiquetas 61
5.1.3. Post-procesamiento de las máscaras predichas 64
5.1.4. Evaluaciones realizadas durante el diseño 65
5.1.5. Definición final de la arquitectura 73
5.2. Optimización de la red diseñada 74
5.2.1. Aumento de datos de entrenamiento 74
5.2.2. Optimización de hiperparámetros 78
5.3. Desempeño del modelo final 80
5.3.1. Resultados del entrenamiento y la validación 82
5.4. Resumen del capítulo 84

6. Validación y análisis comparativo 87
6.1. Evaluación del modelo sobre datos de testeo 87
6.2. Validación cruzada *Leave One Subject Out* 88
6.2.1. Resultados de la validación cruzada 88
6.3. Evaluación del modelo con datos de duración real 90
6.3.1. Evaluación del modelo sobre datos de testeo con datos de duración real 92
6.3.2. Resultados de la validación cruzada con datos de duración real . 94
6.4. Comparación con otras técnicas de aprendizaje automático 95
6.4.1. Desarrollo y desempeño del modelo de ML 96
6.4.2. Resultados del modelo de base 98
6.4.3. Análisis y comparación 99
6.5. Comparación con modelos de bibliografía 100
6.6. Resumen del capítulo 104

7. Discusión y conclusiones 106

7.1. Discusión 106
7.1.1. Trabajos futuros 109
7.1.2. Relevancia del trabajo realizado 110
7.2. Conclusiones 110

Glosario **112**

Bibliografía **115**

Anexos **120**

A. Descripción de arquitectura utilizada **122**

B. Descripción de los datos **125**

C. Bases teóricas del algoritmo XGBoost **128**

Capítulo 1

Introducción

En este capítulo se indica el contexto en el cual se llevó a cabo este proyecto. Inicialmente se plantea la razón de aplicar técnicas del campo de la inteligencia artificial en el área de la ingeniería biomédica, incluyendo las distintas aplicaciones posibles que surgen de la interrelación de ambos campos. Luego, dado que este proyecto se enmarcó dentro de una investigación sobre el análisis en señales de electroencefalografía, se desarrolla sobre la relevancia del aprendizaje automático en el procesamiento de este tipo de datos. Dentro de las aplicaciones en electroencefalografía se realiza una mención especial al caso de la investigación en epilepsia, dado que es la patología sobre la que se centra este trabajo final.

Por otro lado, se listan los objetivos del proyecto y el contexto en el que se plantearon los mismos. Finalmente, se indica la manera en la que se estructuró el trabajo.

1.1. Aprendizaje automático e ingeniería biomédica

La Inteligencia Artificial (IA) es una disciplina que tiene como objetivo interpretar la inteligencia humana y construir sistemas inteligentes en base a ello. Es decir, es posible entender a la inteligencia artificial como la habilidad de los sistemas digitales de resolver problemas que están normalmente asociados con las altas capacidades intelectuales de procesamiento de los humanos. Sin embargo, existen hoy en día diversos enfoques para lograr este objetivo, por lo que la inteligencia artificial engloba múltiples subdisciplinas [1].

El aprendizaje automático, en inglés *Machine Learning* (ML), es una rama de la inteligencia artificial, centrada en el aprendizaje de modelos a partir de datos [1]. Mediante estas técnicas, tareas realizadas por los humanos pueden ser automatizadas, y, además, existen indicios que sugieren que en diversas aplicaciones estos algoritmos pueden ser capaces de reconocer patrones que van más allá de la percepción humana. En los últimos años, la rápida y considerable evolución de los algoritmos de aprendizaje automático ha llevado a su uso en diferentes campos de la vida diaria.

Específicamente, dentro del ámbito de la salud, su aplicación para el análisis de imágenes y señales biomédicas ha mostrado ser de gran interés, dado que el uso de estas

herramientas puede resultar en numerosas ventajas potenciales. Dentro de las mismas encontramos la posibilidad de brindar soporte a los profesionales para la interpretación de hallazgos clínicos, permitiendo así reducir los tiempos de análisis, y colaborando en los procesos de toma de decisiones y diagnóstico. Esta posibilidad puede asistir en la reducción potencial de costos directos e indirectos, la mejora de la precisión diagnóstica y el incremento en el acceso a la información clínica para personas que no podrían hacerlo debido a restricciones de distinta índole, por ejemplo económicas o geográficas [2] [1]. En los últimas años, estos algoritmos han sido utilizados para varias tareas que pueden resultar desafiantes, entre las cuales se pueden mencionar la detección de cáncer de mama a partir de imágenes de mamografía, la segmentación de tumores cerebrales en imágenes de resonancia magnética, la detección de los estados cognitivos del cerebro con resonancia magnética funcional en el diagnóstico de enfermedades neuronales (como el Alzheimer), entre otros [3] [1].

Dentro del aprendizaje automático encontramos el aprendizaje profundo, en inglés *Deep Learning* (DL), el cual comprende un conjunto de algoritmos particulares que incluyen redes neuronales con muchas capas interconectadas y combinan módulos simples generando redes más complejas y profundas. Entrenar estos modelos es hoy posible gracias al desarrollo de tecnología que permitió realizar cálculos en paralelo con unidades de procesamiento gráficas, en inglés *Graphic Processing Unit* (GPU). Distintos tipos de modelos de DL fueron creados para varios propósitos, como por ejemplo, para la detección automática de objetos, la segmentación de imágenes, el reconocimiento de voz automático, y la clasificación de enfermedades en bioinformática [2].

Actualmente estos algoritmos están atrayendo el interés en investigadores en el campo de la medicina y la salud, ya que al aplicar modelos de este tipo con datos médicos es posible incrementar la precisión en los diagnósticos y tratamientos médicos. En particular está siendo muy elegido para distintas investigaciones por las siguientes razones [1]:

- El DL es capaz de obtener una comprensión de fenómenos muy complejos en casos donde los datos disponibles tienen alta dimensionalidad. Su rendimiento se puede beneficiar en gran medida por la cantidad de datos con los que se entrena, y actualmente, con la digitalización de los datos en el ámbito de la salud, este punto viene tomando cada vez mas relevancia.

- Se caracteriza por un alto grado de flexibilidad. Los datos médicos incluyen distintos tipos de datos estructurados, como imágenes, señales, expresiones genéticas, y datos de texto. Debido a la complejidad de sus arquitecturas, el DL se puede beneficiar de esta heterogeneidad, logrando altos niveles de abstracción en el análisis de datos.

- Se tiene un nivel de automatización alcanzable, por lo que no es necesario extraer y calcular características de los datos de entrada previo al análisis de la red. Así es posible desarrollar estos modelos sin realizar un procesamiento previo extensivo de los datos ni tener conocimiento de técnicas alternativas.

1.1.1. Aplicaciones biomédicas del aprendizaje automático

En la mayoría de los casos, el uso del aprendizaje automático en aplicaciones dentro de la ingeniería biomédica puede estructurarse en 3 orientaciones principales [4]:

1. Diagnóstico asistido por computadora, con el objetivo de asistir a los profesionales de la salud en la obtención de un diagnóstico eficiente y temprano, con una mayor armonía entre profesionales y menores diagnósticos contradictorios.

2. Cuidado médico de pacientes con terapias más personalizadas, permitiendo analizar las características especificas de cada caso.

3. Análisis poblacionales, para comprender nuevas maneras de incrementar la calidad de vida de las personas, por ejemplo permitiendo el desarrollo de interfaces persona-máquina, o generando métricas sobre las características de la población evaluada.

Para alcanzar estos 3 objetivos se puede segmentar al área de en diferentes subáreas [4]:

- Ómica: genómica y proteómica, las cuales incluyen el estudio del ADN y ARN, incluyendo sus secuencias, las interacciones proteicas y la predicción de estructuras.

- Bioimágenes: el estudio de la célula biológica y los tejidos al analizar la histopatología o imágenes inmunohistoquímicas

- Imágenes médicas: el estudio de órganos humanos, obtenidos por técnicas basadas en rayos x, resonancia, ultrasonido, etc.

- Interfaces persona-máquina: el estudio del cerebro y el cuerpo humanos, decodificando interfaces con la máquina y analizando diferentes bioseñales como el electroencefalograma, electrocardiograma, etc.

- Gestión de salud pública y médica: el estudio de datos médicos para desarrollar e impulsar las decisiones de cuidado de la salud

- Desarrollo de medicamentos: se busca reducir el tiempo de los ensayos clínicos para la aprobación de nuevas drogas [5]

Dentro del área vinculada a las interfaces persona-máquina específicamente, encontramos dos aplicaciones principales: decodificación de la actividad cerebral para lograr el control de una máquina o para el monitoreo de pacientes con distintas patologías neurológicas. En la primera, las señales son utilizadas para controlar computadoras o dispositivos externos, como pueden ser un brazo robótico o una silla de ruedas [6], mientras que en la segunda, el análisis de las señales permite obtener información sobre la patología del paciente, permitiendo realizar diagnósticos, monitorizar la evolución de la enfermedad y detectar anormalidades [4].

1.1.2. Aprendizaje profundo en electroencefalografía

La Electroencefalogafía (EEG), es decir, la medición de los campos eléctricos generados por la actividad cerebral, es una técnica de mapeo cerebral y de neuroimagen ampliamente utilizada dentro y fuera del dominio clínico [7]. Específicamente, el EEG captura las diferencias de potencial eléctrico, en el orden de 10μV, que alcanzan la corteza cerebral cuando pequeños potenciales postsinápticos excitatorios, producidos por las neuronas piramidales en las capas corticales del cerebro, se suman juntas. Los potenciales medidos reflejan entonces la actividad neuronal y pueden ser utilizados para estudiar una amplia variedad de procesos cerebrales [8]. El estudio mas comúnmente conocido es el EEG superficial, que incluye la medición por medio de múltiples electrodos o sensores colocados directamente en el cuero cabelludo, los cuales con ayuda de interfaces obtienen las señales en forma digital para ser almacenadas en una computadora. Este procedimiento es considerado como el más común para el registro de señales cerebrales, ya que tiene una alta resolución temporal permitiendo tomar mediciones de actividad cerebral a nivel de milisegundos. Además, es cómodo y seguro de usar al no ser invasivo [7]. También es posible utilizar métodos de medición de EEG con electrodos invasivos intracraneales, los cuales son colocados quirúrgicamente. Esto es conocido como Electroencefalogafía Invasiva (iEEG) y permite una medición mas localizada, con mejor resolución espacial y con menor ruido.

Sin embargo, al momento de analizar una señal de electroencefalografía surgen distintos desafíos que dificultan obtener resultados objetivos, entre los que se encuentran:

- El EEG tiene una relación señal-ruido baja: la actividad cerebral medida se encuentra generalmente escondida debajo de múltiples fuentes de ruido ambiental, fisiológico o específico de la actividad, que puede tener una amplitud similar o mayor que la señal, y que se conoce como artefacto. Varias técnicas de filtrado y reducción de ruido se usan para minimizar el impacto de estas fuentes ruidosas y extraer la señal cerebral real de las señales registradas. En el caso de la electroencefalografía intracreaneal este fenómeno decrece pero se encuentra presente de igual forma [8].

- El EEG es una señal no estacionaria: sus estadísticas varían a través del tiempo. Como resultado, un clasificador entrenado en una cantidad limitada de datos a lo largo del tiempo puede generalizar de manera pobre a datos registrados en distintos momentos del mismo individuo [8] [6].

- Existe una alta variabilidad entre sujetos: este fenómeno se debe a diferencias fisiológicas entre individuos, las cuales varían en magnitud, pero pueden afectar el rendimiento de modelos que buscan generalizar entre distintos sujetos [8] [9].

Para resolver estos desafíos, resulta necesario abordar el procesamiento de EEG con técnicas que tengan mejores capacidades de generalizar y puedan ser aplicadas en situaciones más flexibles. En particular, el aprendizaje profundo ha sido recientemente utilizado para el procesamiento automatizado de señales de EEG en diferentes contextos, tanto como interfaces cerebro-computadora [9], evaluaciones del sueño [10] o predicción de eventos convulsivos [11]. Esto se corresponde con la gran capacidad que tienen estos algoritmos para reconocer patrones complejos a partir de datos crudos y

por su buen rendimiento en tareas de reconocimiento visual.

En el caso de la epilepsia, los desarrollos existentes generalmente han buscado resolver el desafió de la detección de patrones ictales, siendo los mismos los patrones que se visualizan en una señal de iEEG durante un evento convulsivo. Las bandas de frecuencias características del cerebro han llevado tradicionalmente a realizar análisis de las frecuencias de los registros de EEG para resolver este desafío, incluyendo modelos basados en la extracción de características relacionadas con el tiempo o la frecuencia. Estos modelos han incluido técnicas que involucran el análisis en el dominio de la frecuencia a partir de la transformada de Fourier o la transformada ondita, o técnicas que utilicen análisis de componentes principales, entre otros [12]. Estas características, calculadas manualmente, se utilizan generalmente como entradas en algún clasificador que sea capaz de distinguir entre patrones ictales y actividad normal en un iEEG. Una de las limitaciones de usar métodos basados en este tipo de extractores de características es que resultan susceptibles a variaciones en los patrones convulsivos, las cuales son generadas por los artefactos presentes durante la adquisición de la señal de EEG y la naturaleza no estacionaria de la misma, lo que hace que los componentes estadísticos de las señales cambien según el sujeto y el tiempo. Las redes neuronales convolucionales permiten que la extracción de características de las señales no se deba realizar de forma manual, sino que la misma red aprende las transformaciones que se deben realizar sobre la señal para poder clasificarla [12].

Aunque se ha demostrado eficacia en estudios de procesamiento de EEG con aprendizaje profundo, pocos estudios han abordado la detección de episodios ictales por iEEG, en parte, dado el pequeño número de convulsiones por paciente típicamente disponibles en investigaciones con este tipo de señales. Hoy en día, la disponibilidad de datos está creciendo, por lo que surge la necesidad de obtener algoritmos que permitan procesarlos y analizarlos de manera automática y eficaz [11].

1.2. Objetivos del proyecto

Este proyecto busca desarrollar y analizar el rendimiento de un modelo basado en redes neuronales convolucionales para poder automatizar la detección de patrones ictales en señales de pacientes con epilepsia focal refractiva, entrenándolo a partir de una base de datos obtenida mediante un sistema de neuroestimulación cerebral. Este tipo de sistema permite el registro de datos electroencefalográficos en grandes cantidades y a lo largo de meses y años, sin la necesidad de que el paciente concurra a los centros clínicos. Con estos registros, y considerando el modo en que se los obtiene, el poder contar con una herramienta que sea capaz de anotar automáticamente los eventos epilépticos resulta de gran ventaja para aliviar el trabajo de los expertos en neurofisiología, quienes deben manualmente realizar dichas anotaciones para el correcto monitoreo de sus pacientes. Además, este modelo podría permitir un mejor seguimiento de la evolución de los pacientes en el tiempo, y, por ende, una mejor personalización de los tratamientos.

Para lograr este objetivo general se implementan arquitecturas *encoder-decoder* de redes neuronales convolucionales, siendo las mismas pertenecientes al DL [13], para

abordar el problema de patrones ictales. En lugar de utilizar la señales crudas obtenidas del sistema de neuroestimulación, se elige utilizarlas habiendo sido convertidas previamente al formato de espectrogramas, pudiendo así obtener imágenes que representen las señales medidas en el tiempo pero también en el dominio frecuencial. Considerando esta base de datos formada por espectrogramas, se utiliza como arquitectura de base una red existente para el análisis de imágenes, la arquitectura UNet [14], originalmente diseñada para la segmentación semántica de imágenes biomédicas. La primera parte de la red sigue un esquema codificador, formado por varias capas que permiten extraer las características de las imágenes de entrada en múltiples niveles mientras que la segunda parte de la red tiene la función de decodificar las características encontradas, combinándolas con la imagen original de entrada, para obtener un mapa de segmentación a la salida.

Esta red es adaptada para el procesamiento de señales de iEEG, con el objetivo último de estimar la probabilidad de haber obtenido un inicio de un evento convulsivo en los distintos instantes medidos. Por un lado, en la entrada de la red se usan señales de iEEG en formato de espectrogramas 3D, siendo las dimensiones de los espectrogramas el tiempo, la frecuencia y el canal de monitoreo. La tercera dimensión está dada por las señales obtenidas en un mismo momento desde los 4 electrodos implantados en el paciente. Por el otro, se toman los datos a la salida de la arquitectura estándar y se agregan capas para obtener los vectores de probabilidad buscados. Luego, se aplica una técnica de postprocesamiento sobre los vectores de probabilidad para clasificar la muestra y obtener el tiempo estimado de inicio de los patrones ictales. Finalmente, los resultados obtenidos a partir de la red desarrollada y probada se comparan con resultados obtenidos de bibliografía y los obtenidos mediante el análisis con otros algoritmos de aprendizaje automático.

1.2.1. Motivación

La propuesta del proyecto surge a partir de dos investigadores del Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET), Enzo Ferrante y Victoria Peterson, quienes son además docentes en la Universidad Nacional del Litoral (UNL) y supervisaron este proyecto final, para aportar a su línea de investigación. Ellos se encontraban colaborando con el *Brain Modulation Lab* afiliado al *Massachusetts General Hospital y Harvard Medical School* en Boston, y como parte de esta investigación plantearon evaluar los resultados obtenidos al entrenar un modelo de red neuronal basado en una arquitectura específica, utilizando las señales de electroencefalografía intracraneal que venían trabajando en sus ensayos. Además de supervisar el proyecto, ellos proveyeron los datos y las herramientas necesarias para poder llevarlo a cabo.

Por otro lado, el interés por llevar a cabo un proyecto que relaciona tecnologías nuevas como el aprendizaje profundo con señales biomédicas surge a partir del gran crecimiento que esta disciplina está experimentando actualmente, y de la potencial importancia que estas técnicas pueden tener para la ingeniería biomédica en los años futuros. El aprendizaje de estas técnicas innovadoras viene resultando cada vez más relevante en la formación de ingenieros biomédicos, y es por esto que se plantea su aprendizaje y utilización para desarrollar el proyecto.

1.3. Estructura del informe

En los primeros capítulos del informe se desarrollan los conceptos necesarios para introducir al problema abordado y las técnicas utilizadas durante el proyecto. En el Capítulo 2 se desarrolla sobre la epilepsia, los tratamientos posibles y su relevancia en la mejora de la calidad de vida de las personas que la padecen, los sistemas de neuroestimulación y la importancia del análisis de EEG durante el diagnóstico y el tratamiento. En el Capítulo 3 por su parte, se exponen las técnicas de aprendizaje automático relevantes para el proyecto, incluyendo una descripción sobre los problemas de segmentación y una presentación de la arquitectura utilizada.

Luego se describe el protocolo experimental llevado a cabo durante el desarrollo del proyecto. En el Capítulo 4 se presenta la base de datos utilizada, las métricas de evaluación y las herramientas que permitieron llevar a cabo los experimentos. En el Capítulo 5 se describen los experimentos realizados para llegar al modelo final y los resultados obtenidos en cada etapa de la experimentación, además de los métodos de postprocesamiento y evaluación. Más adelante, en el Capítulo 6 se incluyen los experimentos llevados a cabo para validar el funcionamiento del modelo final y se realizan comparaciones con modelos de base y modelos encontrados en la bibliografía.

En el Capítulo 7 se desarrolla la discusión sobre el modelo desarrollado, su significancia clínica, y los resultados de validación. Por último, cierra con una conclusión sobre el proyecto realizado, incluyendo su relevancia y evaluando el desempeño a partir de los resultados obtenidos. Además, se listan los trabajos futuros a incorporar como parte de la misma linea de investigación.

Capítulo 2

Epilepsia y electroencefalografía

En este capítulo se desarrolla sobre la epilepsia, la patología sobre la que se basa el proyecto, describiendo sus síntomas, sus problemas asociados y sus tratamientos. Aquí se incluyen los sistemas de neuroestimulación, particularmente el sistema de neuroestimulación con el cual se obtuvo la base de datos en la que se basa esta tesis.

Adicionalmente, se describe la relación entre la electroencefalografía y la epilepsia, explicando la forma en que este método puede diagnosticar la misma. En específico, se describe el análisis que se puede realizar sobre la patología utilizando las señales de electroencefalografía convertidas en espectrogramas, y la utilidad que los mismos tienen en el diagnóstico.

2.1. Epilepsia

La epilepsia es una enfermedad cerebral crónica y no transmisible, que afecta a personas de todas las edades. Se estima que la tasa de incidencia se encuentra alrededor de un 1 % de la población mundial, representando aproximadamente 50 millones de personas, de las cuales cerca de 5 millones viven en Latinoamérica y el Caribe [15]. Si bien las causas son diversas, y en su mayoría desconocidas, el daño cerebral, malformaciones congénitas, traumatismos craneoencefálicos graves, accidentes cerebro-vasculares, infecciones cerebrales, tumores, pueden ser causantes de descargas epileptogénicas [16].

Este desorden neurológico se caracteriza por la presencia de convulsiones recurrentes y repentinas. Las convulsiones se definen como episodios breves de movimiento involuntario de una parte o de todo el cuerpo, y suelen acompañarse por pérdida de conciencia y de la función intestinal. Las convulsiones se deben a descargas eléctricas excesivas e hipersincrónicas de grupos de neuronas que pueden producirse en diferentes partes del cerebro, conocidas como áreas epileptogénicas [16].

El hecho de que una persona haya sufrido un evento convulsivo no significa que sea epiléptica, incluso hasta el 10 % de las personas de todo el mundo tiene una convulsión a lo largo de la vida. Las convulsiones provocadas por causas agudas son aquellas que ocurren en el contexto de un desorden sistémico o un trauma cerebral agudo, mientras que las convulsiones que ocurren en ausencia de un evento que las haya provocado se

consideran como ocurrencias no provocadas o esporádicas [7]. La epilepsia se define por dos o más convulsiones no provocadas, separadas por un mínimo de 24hs [7]. Esto debe estar acompañado por un contexto de características clínicas que ocurren consistentemente de manera conjunta, con tipos de convulsiones similares, una edad de inicio, señales específicas en el EEG, factores de desencadenamiento de episodios, y respuesta a tratamientos [17].

Los eventos convulsivos conllevan síntomas agudos, como pueden ser la pérdida del conocimiento o la conciencia, y alteraciones del movimiento o de los sentidos (incluyendo visión, audición y gusto). Otros síntomas relacionados a un evento convulsivo son cambios en el estado de ánimo, trastornos en las funciones cognitivas, y problemas físicos, como fracturas y hematomas, consecuencia de los movimientos involuntarios presentes durante el episodio [16]. Las convulsiones pueden ir desde episodios muy breves de ausencia o de contracciones musculares, hasta convulsiones prolongadas y graves. Su frecuencia también puede variar desde menos de una al año hasta varias al día. Esto puede llevar a que los pacientes puedan estar altamente restringidos en sus actividades de la vida diaria [15]. Asimismo, el riesgo de muerte prematura en las personas epilépticas es hasta tres veces mayor que en la población general, registrándose las tasas más altas en los países de ingresos bajos y medianos, y en las zonas rurales [16].

2.1.1. Mecanismos básicos convulsivos

La corteza cerebral consiste en 3 a 6 capas de neuronas que recubren la superficie cerebral. Esta incluye dos tipos de neuronas, las principales o piramidales, que proyectan o envían la información a neuronas que están localizadas en áreas distantes del cerebro, y las interneuronas, que son consideradas como neuronas de circuito local y solo influencian la actividad de otras neuronas cercanas [18]. La forma en que se genera la comunicación entre neuronas es a través de los potenciales de acción, los cuales consisten en potenciales eléctricos que ocurren a partir de la despolarización de la membrana celular en una forma "todo o nada"(lo cual significa que si se disparan siguen un patrón único). Estos cambios en el potencial de membrana están ligados a la activación de canales de iones que permiten flujos iónicos hacia adentro y fuera de la membrana neuronal. La activación de estos flujos iónicos causan la generación de los potenciales postsinápticos, los cuales permiten que se de la comunicación neuronal. Los potenciales sinápticos pueden ser excitatorios o inhibitorios [19], y la forma de comunicación entre neuronas se da generalmente por neurotransmisores, cuya liberación es inducida por los potenciales de acción, y ellos son los responsables de generar los potenciales en neuronas vecinas [18]. Hay diversos mecanismos cerebrales que controlan la actividad neuronal y mantienen estable el nivel de activación eléctrica en las distintas regiones cerebrales [18].

En el caso de una actividad neuronal convulsiva, se generan descargas hipersincrónicas que pueden iniciar en distintas zonas de la corteza y luego propagarse a zonas vecinas. La iniciación de una convulsión se caracteriza por dos eventos concurrentes: ráfagas de alta frecuencia de los potenciales de acción e hipersincronización de la población neuronal [18].

A nivel de neuronas individuales la actividad epileptiforme consiste en una despolarización neuronal sostenida, resultando en ráfagas de potenciales de acción, seguida de una despolarización sostenida, y luego una rápida repolarización seguida por una hiperpolarización. Esta secuencia se puede observar y comparar con un potencial de acción normal en la Figura 2.1, y se conoce como el desplazamiento de la despolarización paroxística, en inglés Paroxysmal Depolarizing Shift (PDS) [18] [19]. La propagación convulsiva es un proceso que ocurre cuando se esparce la convulsión a través del cerebro, lo cual ocurre cuando hay suficiente activación para reclutar a neuronas vecinas. En condiciones normales esto puede ser prevenido por las neuronas inhibitorias vecinas, pero durante una convulsión hay una falla en los mecanismos inhibitorios que no pueden dar una respuesta adecuada, bloqueando la presencia de una hiperpolarización luego del PDS, y, con una activación suficiente, generando la propagación de la actividad convulsiva [18].

Las descargas ictales no siempre consisten en puntas repetitivas u ondas agudas, sino que las descargas varían en morfología, frecuencia y distribución. Frecuentemente las descargas ictales muestran patrones evolutivos con cambios en la frecuencia, amplitud, área afectada o morfología durante el curso de la convulsión [19].

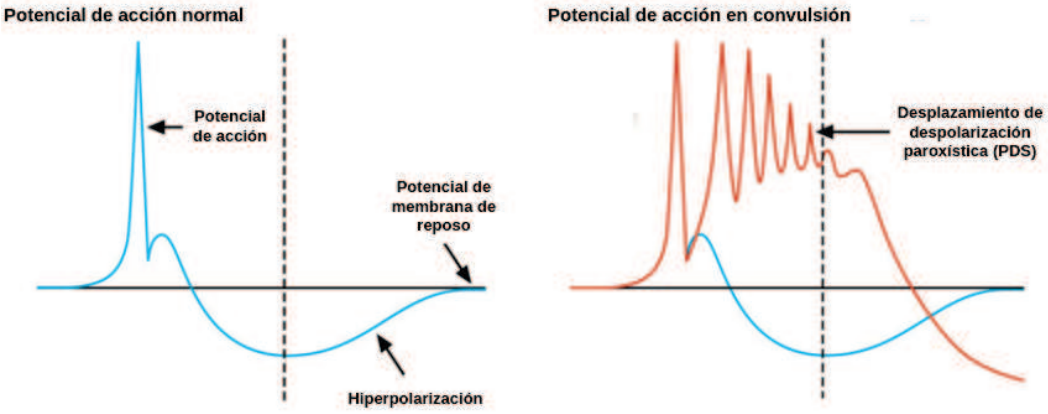


Figura 2.1: *Comparación de un potencial de acción e una neurona sana con un potencial de acción con la secuencia PDS. Figura adaptada de [20].*

2.1.2. Clasificación de la epilepsia

La clasificación del tipo de epilepsia es importante ya que permite la identificación de la región del cerebro donde se origina la misma y guía el diagnóstico clínico inicial. Sus características son variables, dependiendo de la región del cerebro afectada y de la forma de propagación donde se generen las convulsiones. Se pueden diferenciar distintos niveles de clasificación, según el tipo de convulsión, el tipo de epilepsia y el síndrome epiléptico, los cuales se encuentran esquematizados en la Figura 2.2 [7].

En un primer nivel es posible clasificar el tipo de convulsión en 3 grandes grupos, de inicio focal o parcial, de inicio general o de inicio desconocido, según dónde y cómo se generan las descargas eléctricas que causan las convulsiones [17].

- Convulsiones de inicio focal: se refieren a convulsiones que se originan en redes neuronales limitadas a un único hemisferio, pero que rápidamente puede involucrar a ambos hemisferios [7].
- Convulsiones de inicio generalizado: se define como aquellas que se originan en algún punto y rápidamente involucran redes neuronales distribuidas bilateralmente [7].
- Convulsiones de inicio desconocido: son aquellas convulsiones que no pueden ser caracterizadas como de inicio focal o generalizado [17].

Una convulsión puede comenzar focalmente y luego generalizarse. También puede originarse en la corteza del cerebro o en estructuras subcorticales. En un segundo nivel es posible clasificar el tipo de epilepsia, en focal, generalizada, combinada o desconocida, según donde afecten generalmente las convulsiones [7]. Con la historia clínica del paciente, los eventos reconocidos en el EEG, y otra información disponible, los neurofisiólogos pueden categorizar el tipo de epilepsia, con el cual luego se formula un diagnóstico apropiado y se propone un plan de tratamiento [17].

En el último nivel se encuentra el síndrome epiléptico, una entidad clínica distintiva que puede ser identificada por un conjunto de características clínicas y eléctricas, entre las que se incluyen la edad de inicio, los tipos de epilepsia, las características del EEG (tanto la señal de base como la patológica), etiología, etc. Estos síndromes generalmente tienen implicaciones claras con respecto a la etiología, tratamiento y prognosis [7].

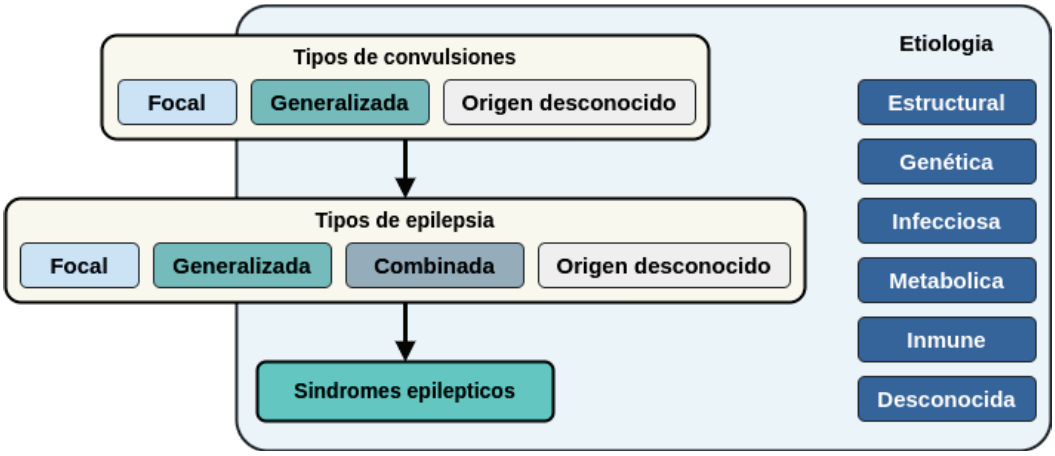


Figura 2.2: *Clasificación de la epilepsia. Figura adaptada de [7].*

2.1.3. Tratamientos posibles

Este desorden neurológico es tratable, e incluso, con un tratamiento adecuado las crisis pueden ser completamente controladas en alrededor del 70 % de los pacientes. El éxito de estos tratamientos depende de diversos factores, entre los que se incluyen el tipo de epilepsia, el diagnóstico correcto, la elección del tratamiento y el cumplimiento del mismo por parte del paciente [21].

En general el tratamiento mayormente utilizado es el farmacológico, el cual incluye la toma de medicamentos según prescripción médica. El fármaco y la dosis aplicada dependen del paciente y su respuesta al tratamiento, considerando la eficacia en el control de los episodios convulsivos y los efectos adversos que se presenten [21]. Muchas personas experimentan efectos secundarios leves o que desaparecen pocos días después de empezar a tomar el tratamiento. Hay distintos fármacos y combinaciones según el tipo de epilepsia que se haya diagnosticado [21]. En argentina, la ley 25.404 establece que las obras sociales deben cubrir las prestaciones de salud de las personas diagnosticadas con epilepsia, y que si una persona no tiene acceso a una obra social, el Ministerio de Salud es quien debe asegurar el acceso a los medicamentos necesarios [22].

Puede estudiarse la posibilidad de suspender la medicación anticonvulsiva después de 2 años sin convulsiones; para ello, deben tenerse en cuenta los factores clínicos, sociales y personales pertinentes. Una etiología documentada de las crisis epilépticas y un patrón electroencefalográfico anormal son los dos factores predictivos más consistentes de la recurrencia de convulsiones [16].

Tratamientos quirúrgicos son también posibles. La cirugía para el control de la epilepsia conlleva un procedimiento complejo que sólo debe realizarse en centros con experiencia y recursos adecuados. El requisito principal para un tratamiento quirúrgico es que la epilepsia sea focal, es decir, que las descargas eléctricas anormales comiencen en un lugar localizado del cerebro. El procedimiento quirúrgico entonces consiste en la extirpación total o parcial de la lesión focal o tejido anormal del cerebro responsable de las crisis [21].

Hay dos tipos esenciales de cirugía de la epilepsia, entre los que se encuentran los procedimientos paliativos, que pueden ayudar a mejorar, pero no necesariamente curar como la callostomía (corte del cuerpo calloso, que comunica los hemisferios cerebrales), y los potencialmente curativos como las lobectomías (resección de un lóbulo o parte de un lóbulo) y las lesionectomías (resección de lesiones visibles en la resonancia magnética). Para poder realizar este tipo de cirugías la zona epileptogénica, la que genera las convulsiones, debe ser correctamente determinada con el uso del EEG, imágenes de resonancia magnética de cerebro y, en ocasiones, estudios más invasivos, como puede ser la realización de un estudio de iEEG [21].

A pesar de la posibilidad de realizar estos tratamientos y de la disponibilidad de muchas drogas antiepilépticas aprobadas por la Administración de Alimentos y Medicamentos, en inglés *Food and Drug Administration* (FDA), alrededor del 40 % de los pacientes con epilepsia de inicio focal aún presentan convulsiones tomando la medicación [23]. Para muchos de estos pacientes la cirugía neurológica para extirpar el foco de las convulsiones ofrece la mejor oportunidad para el tratamiento. Sin embargo, no todos los pacientes son candidatos o acceden a estos procedimientos quirúrgicos por los riesgos y las consecuencias que los mismos pueden generar [24]. Para estos casos se han propuesto múltiples alternativas, y una de ellas es el uso de dispositivos de neuroestimulación, que se exponen en la siguiente sección, entre los cuales se encuentra el *Responsive Neurostimulation System* (RNS) (NeuroPace, Mountain View, California, EEUU) [24], que motivó la realización de este proyecto.

2.2. Sistemas de neuroestimulación

La neuroestimulación en epilepsia se entiende como la aplicación de pulsos eléctricos buscando afectar un grupo de neuronas o una zona del cerebro, con el objetivo de disminuir la frecuencia de los eventos convulsivos y su severidad. La neuroestimulación no debería considerarse como una primera opción para el tratamiento de la epilepsia focal, sino que surge como una alternativa para aquellos pacientes que no logran controlar sus episodios convulsivos mediante tratamiento farmacológico, y que no resultan compatibles con un tratamiento quirúrgico. Este tipo de tratamiento presenta múltiples ventajas, incluyendo la disminución de la frecuencia y duración de las crisis. Es un tratamiento menos invasivo que el quirúrgico, además de ser reversible y personalizado, entre otras ventajas. Todas estas características permiten reducir el riesgo de muerte del paciente y mejorar su calidad de vida [25]. La neuromodulación puede ser a nivel focal, e incluso se han identificado distintos focos a lo largo de la historia con resultados variados, entre los que se encuentran el hipocampo, el subtálamo, el tálamo, y el neocórtex [25].

Sin embargo, la neuromodulación es un tratamiento, sin llegar a ser una cura, por lo que precisa un ajuste frecuente. Además presenta otras desventajas que se derivan de consistir en un dispositivo implantable, entre las que se incluye la presencia de un generador de pulsos interno, posibles complicaciones en la biocompatibilidad entre la carcasa y el tejido (como infecciones o daño en los cables), y un alto costo monetario. Además, la mayoría de los dispositivos pueden presentar efectos secundarios, como dolor de cabeza o cuello, parestesia, o hemorragias [25].

En general, se utilizan dos mecanismos de neuroestimulación, el sistema a lazo abierto y el sistema a lazo cerrado. La estimulación a lazo abierto tiene un modo de funcionamiento en donde los parámetros de configuración son preprogramados, por lo que no responden automáticamente a los cambios en el estado clínico del paciente. Mientras que este tipo de terapia suele ser efectiva, presenta varias limitaciones en el uso, ya que siempre da la misma respuesta sin considerar el estado del paciente [26] [24]. Por otro lado, los sistemas a lazo cerrado son dispositivos que surgieron de forma más reciente, los cuales sensan continuamente las señales fisiológicas del paciente, y, al detectar cambios fisiológicos especificados con anterioridad, ajustan la terapia en respuesta a la señal detectada. Entre las ventajas de estos sistemas se incluye la mejora de la eficacia de la estimulación, permitiendo así obtener mayores beneficios clínicos y reducir los efectos adversos [26] [24].

La estimulación cerebral profunda, en ingles *Deep Brain Stimulation* (DBS), y la estimulación del nervio vagal, en ingles *Vagus Nerve Stimulation* (VNS), son ejemplos de sistemas de lazo abierto para el tratamiento de epilepsia, mientras que la estimulación responsiva (RNS) es un sistema de estimulación a lazo cerrado [25]. Los distintos dispositivos mencionados se muestran en la Figura 2.3, junto con sus modos de funcionamiento y la localización de los electrodos en cada caso.

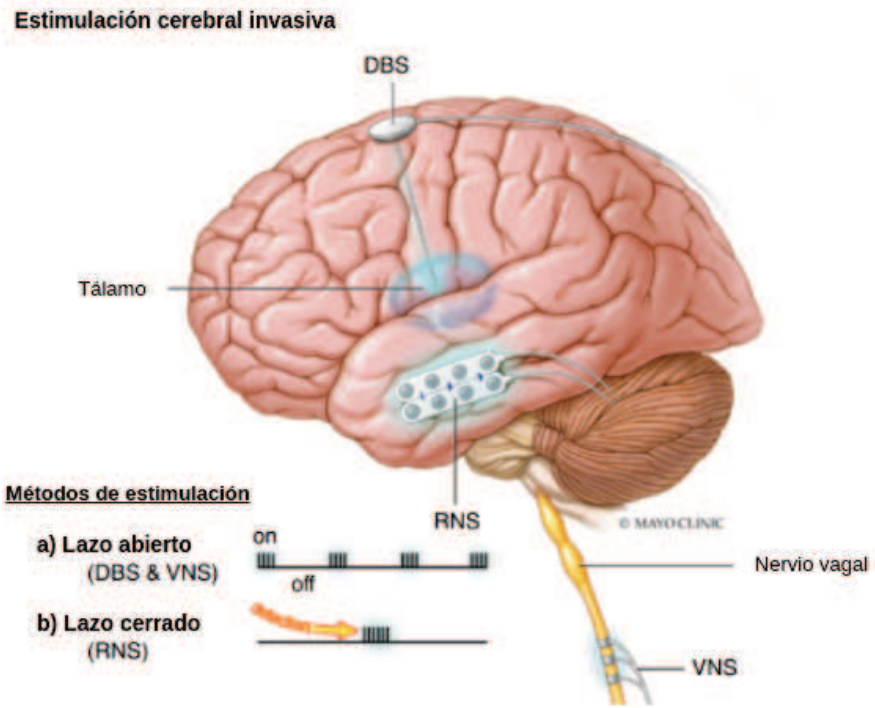


Figura 2.3: *Dispositivos de neuroestimulación. Figura adaptada de [7].*

2.2.1. Estimulación del nervio vagal

El VNS es uno de los métodos de neuroestimulación más estudiados para el tratamiento de epilepsia, el cual fue aprobado por la FDA en 1997 como una terapia adjunta para reducir la frecuencia de las convulsiones en pacientes con epilepsia refractaria de inicio focal [26] [23]. En la técnica resulta uno de los sistemas menos complejos, y consiste en un estimulador colocado alrededor del nervio vagal izquierdo y conectado al generador de pulsos interno debajo de la clavícula. Suelen ser sistemas a lazo abierto, pero existe un modelo reciente que trabaja a lazo cerrado, y que fue diseñado para evitar daño cardíaco [27].

Hay algunas hipótesis que explican el mecanismo de funcionamiento de este tipo de dispositivos, aunque el verdadero motivo se mantiene desconocido. Una de estas teorías es la teoría de sincronización, la cual plantea que este sistema puede inducir una desincronización electroencefalográfica, lo cual permite alterar el estado hipersincronizado de las neuronas corticales y del tálamo durante una convulsión. Otras teorías están relacionadas con el efecto que podría tener la neuroestimulación de este tipo sobre la producción de neurotransmisores, y con la alteración del flujo sanguíneo en ciertas estructuras subcorticales, lo que podría llevar a la activación de mecanismos inhibitorios [25]. Los efectos adversos relacionados con la estimulación del nervio vago incluyeron alteraciones en la voz, tos fuerte, faringitis, parestesias, disnea, dispepsia, náusea y laringismos [23].

2.2.2. Estimulación del núcleo anterior del tálamo

La estimulación cerebral profunda (DBS) en el núcleo del tálamo anterior es un sistema que involucra el implante de electrodos bilaterales en el núcleo anterior del tálamo, los cuales se conectan a un generador de pulsos interno infraclavicular. Se trata de un sistema de lazo abierto, por lo que no tiene la habilidad de detectar actividad convulsiva [23].

No es completamente claro cómo los sistemas DBS modulan la actividad convulsiva. En general permiten llevar a la reducción de la frecuencia en la que ocurren los episodios al causar excitación o inhibición a nivel de la red neuronal, lo cual altera la propagación de la convulsión o aumenta el umbral de las mismas. El núcleo anterior del tálamo pertenece a uno de los dos circuitos esenciales involucrados en la generación de actividad ictal, por lo que su estimulación puede generar la interrupción de la hipersincronización patológica entre neuronas. [25]. Sin embargo, al ser una región del cerebro central y altamente conectada, la principal desventaja de estimularla sin conocer el estado de actividad neuronal es la posibilidad potencial de interrumpir la actividad cerebral normal. Consistente con esto, una parte de los pacientes en las pruebas clínicas reportaron efectos adversos relacionados con la memoria y otros tuvieron efectos adversos relacionados con la depresión [23].

2.2.3. Estimulación responsiva

El dispositivo RNS consiste en un neuroestimulador implantable con respuesta de estimulación frente a la detección de patrones ictales. Este fue aprobado por la FDA en 2013 en Estados Unidos como una terapia adjunta en adultos con epilepsia focal, localizada en una o dos áreas epileptogénicas. Este dispositivo provee un tratamiento individualizado y dirigido al foco de la actividad convulsiva [26] [24].

Al contrario de las terapias a lazo abierto en neuromodulación, en dónde la terapia se da continuamente o en un cronograma fijo, este sistema es a lazo cerrado, por lo que monitorea de forma continua la actividad neuronal en un foco convulsivo, mediante mediciones de iEEG, y responde con estimulación sólo cuando se detecta actividad epileptiforme [24]. La aplicación de corriente eléctrica sobre el área de comienzo de la convulsión puede afectar el progreso de los patrones patológicos detectados o generar cambios beneficiosos en el largo plazo. Este efecto puede ser causado por bloqueo de la despolarización, inhibición sináptica, o modulación de la red [24]. La razón original para desarrollar este sistema consistió en generar una especie de marcapasos cerebral. Esta idea fue basada en la naturaleza de las convulsiones, las cuales resultan del disparo sincrónico anormal de poblaciones neuronales, y de la hipótesis que la estimulación intervente podría interrumpir dicha sincronización devolviendo la actividad neuronal a su estado basal [28].

El sistema, que se puede encontrar esquematizado en la Figura 2.4, se conforma principalmente de un neuroestimulador, los cables que conectan al paciente, un monitor remoto y un imán. El neuroestimulador consiste en un pequeño dispositivo que funciona con batería y que está programado para enviar pulsos eléctricos cortos al cerebro en ciertas oportunidades. Los electrodos pueden usarse tanto para la estimulación

como para el sensado. De forma remota, el monitor recopila los datos registrados por el neuroestimulador. Además, el médico interactúa con este monitor para programar el sistema y observar las mediciones. Adicionalmente, es posible utilizar un imán para generar una señal al neuroestimulador para que registre las señales de electroencefalografía intracraneal en algún momento específico o para detener temporalmente la estimulación [29].

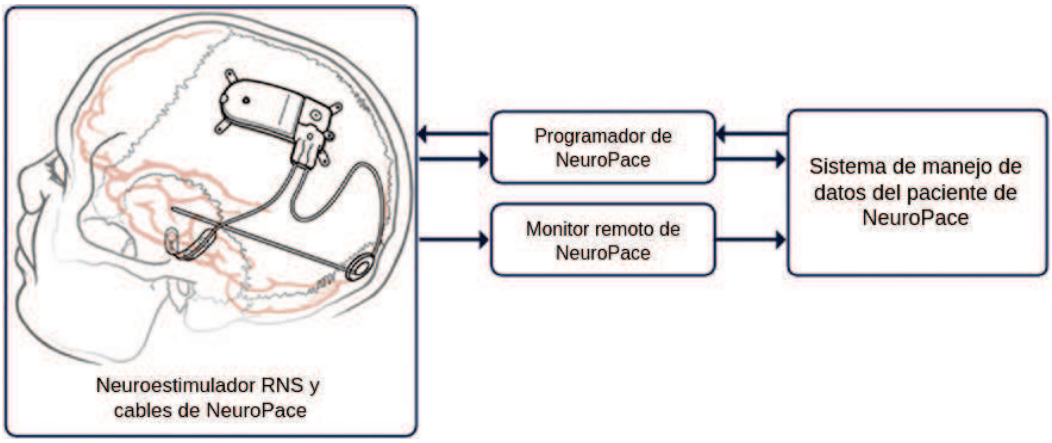


Figura 2.4: *Esquema del sistema RNS. Figura adaptada de[29]*

El neuroestimulador está programado para almacenar segmentos del iEEG cuando algunos eventos específicos ocurren, y este registro se puede dar de tres maneras en los pacientes implantados. El primer tipo consiste en el pase de un imán, y lo puede realizar el paciente para indicar que ha ocurrido un evento. Además, se pueden dar registros planeados, los cuales son generados a partir del reloj del dispositivo en ciertos horarios del día. Estos pueden ocurrir cada 12 o 24 horas y ofrecen un muestreo continuo de la actividad neurofisiológica espontánea. Por otro lado, el sistema esta configurado para recolectar datos de iEEG cuando se detectan ciertas condiciones que podrían significar un evento convulsivo, como puede ser la detección de saturación (un segmento de iEEG de gran amplitud) y de episodio largo (cambios en el trazado del iEEG sostenidos en el tiempo) [29]. El médico selecciona los tipos de eventos que generan el disparo para la captura de datos. Al iniciar la terapia los pacientes son instruidos para descargar los datos registrados de forma diaria a una computadora local, desde la cual se actualiza la información registrada en la base de datos de Neuropace [29].

2.2.3.1. Protocolo de programación del sistema RNS

La detección de la actividad anormal es programada para cada paciente por el médico, usando un conjunto de algoritmos de detección personalizable, luego de que el mismo revise la respuesta a la estimulación del paciente. Esto puede incluir la configuración de la forma de las señales, como trenes de picos, ondas lentas, cambios rítmicos en la frecuencia, u otros patrones que caracterizan eventos electrográficos significativos en el paciente [24]. El médico también ajusta la estimulación de respuesta para cada paciente, y configura los parámetros de forma personalizada dentro de los rangos recomendados según los reportes del paciente y las respuestas clínicas, como así también los datos cuantitativos electrográficos que el equipo provee [23].

En una primera fase, inmediatamente luego de ser implantados con el neuroestimulador, el dispositivo es configurado para estar en modo de registro pasivo por un período corto, de 1 mes aproximadamente. Durante este período, el protocolo de NeuroPace indica que no se debe estimular al paciente, de forma que resulte posible registrar la actividad electroencefalográfica de base del mismo. Para esto es posible seleccionar uno de los 3 programas de detección básicos que presenta el dispositivo para obtener una colección inicial de registros de iEEG. Una vez que se revisa esta actividad de base en una consulta con un médico neurólogo, se configuran y se activan los parámetros de estimulación para cada paciente de manera personalizada. El sistema provee un punto de inicio para establecer un conjunto de detección específico para el paciente, pero es el médico quien toma la decisión final [30].

Durante el resto del período post-implantación el equipo entrega una terapia basada en la estimulación a partir de la detección de episodios. Los parámetros de configuración son modificados periódicamente en visitas clínicas posteriores, en las cuales el neurofisiólogo se basa en la evolución del paciente con respecto a los episodios convulsivos y a la información registrada en el dispositivo. Los intervalos de tiempo entre estos cambios de configuración se conocen como épocas de programación. Este proceso se recomienda hacerlo una vez cada 3 meses, determinando en cada visita si es necesario modificar los parámetros de detección y/o estimulación [30].

Esta programación requiere un análisis visual detallado de los registros de iEEG de cada paciente, el cual es llevado a cabo por neurofisiólogos experimentados en cada visita, quienes identifican los patrones de las actividades interictales e ictales. Por la forma y la extensión de los registros de iEEG necesarios, el análisis de las señales puede resultar muy costoso, ineficiente y consumidor de tiempo. Además, la inspección visual de los datos puede generar discrepancias entre los especialistas para un mismo registro. Similarmente, los patrones del iEEG que caracterizan una convulsión pueden perderse o confundirse como ruido de fondo o artefactos. Por último, también existe la variabilidad entre pacientes tanto en actividad convulsiva como no convulsiva [31]. La exactitud promedio para el detector alcanzada en un grupo de varios pacientes es del 85 % aproximadamente, para la detección de patrones ictales. Sin embargo, la exactitud es dependiente de la programación y refinación apropiadas [28].

Estas dificultades en el análisis de los datos, generadas tanto por el tiempo requerido como por las diferencias existentes en los patrones esperados y la naturaleza de las señales de iEEG, permiten visualizar la necesidad de desarrollar un sistema que sea capaz de evaluar las señales de forma automatizada y que sirva de apoyo a la decisión clínica. Siendo una herramienta de apoyo, el sistema requerido debería ser capaz de analizar las señales de manera previa a que lo haga el médico, y así dar soporte durante la revisión clínica, minimizando el tiempo requerido y los errores posibles. Es en este punto que surge la motivación para llevar a cabo este proyecto, buscando una herramienta que pueda servir como soporte en las visitas de los pacientes para programar el dispositivo RNS.

2.3. Electroencefalografía en el análisis de epilepsia

Las señales de electroencefalografía son generadas por potenciales de las neuronas postsinápticas corticales y proveen una evaluación de tiempo real de la función fisiológica del cerebro, como se mencionó en la Sección 1.1.2. El análisis de los resultados del EEG, en particular de las convulsiones y de las descargas interictales, tiene un rol fundamental en el diagnóstico de la epilepsia. Además estos análisis son esenciales para clasificar los tipos de episodios convulsivos y los síndromes epilépticos [19].

2.3.1. Generalidades de la señal de EEG

En una primera instancia, la actividad eléctrica espontánea detectada en el EEG aparenta una naturaleza aleatoria, sin un sentido determinado. Sin embargo, existe una cierta organización y ritmo en la actividad que dependen del nivel de alerta y la edad de cada persona. Las ondas en la señal pueden ser caracterizadas basándose en su localización, amplitud, frecuencia, morfología, continuidad (rítmica, intermitente o continua), sincronía, simetría y su reacción a estímulos externos. Sin embargo, la forma más utilizada para clasificar las formas de onda es la frecuencia [18].

Desde el punto de vista frecuencial el EEG puede estudiarse en bandas de frecuencia definidas por la rapidez de las oscilaciones presentes, pudiendo relacionarse cada banda con distintos estados de actividad mental. Las formas de onda más estudiadas normalmente incluyen el ritmo alpha (rango frecuencial de 8 a 12Hz) que aparece normalmente en sujetos despiertos con los ojos cerrados [32], beta (rango frecuencial entre 13 y 30Hz), que aparece en estados de concentración profunda [32] o de somnolencia y sueño ligero [7], delta (entre 0.5 y 4Hz) y theta (de 4 a 7Hz), características del sueño [7], y gamma (con frecuencias mayores a 30Hz), relacionada con la función cognitiva, el procesamiento de información y la memoria [7]. Cada uno de estos ritmos se suele ver en localizaciones y momentos específicos, los cuales se resumen en la Tabla 2.1. Adicionalmente, las formas de los ritmos presentados se pueden observar en la Figura 2.5 [18] [7] [32].

Tabla 2.1

Ritmos eléctricos cerebrales [7]

Ritmo	Frecuencia (Hz)	Amplitud	Ocurrencia normal
Delta	0.5 - 4	Alta	Fase 2 y sueño de onda lenta
Theta	4 - 8	Media	Sueño, somnolencia e hiperventilación
Alpha	8 - 13	Menor a 50uV	Ritmo posterior dominante
Beta	13 - 30	Pequeña y simétrica	Somnolencia y sueño ligero
Gamma	30 - 80	Pequeña	Movimiento motor voluntario y tareas como el lenguaje

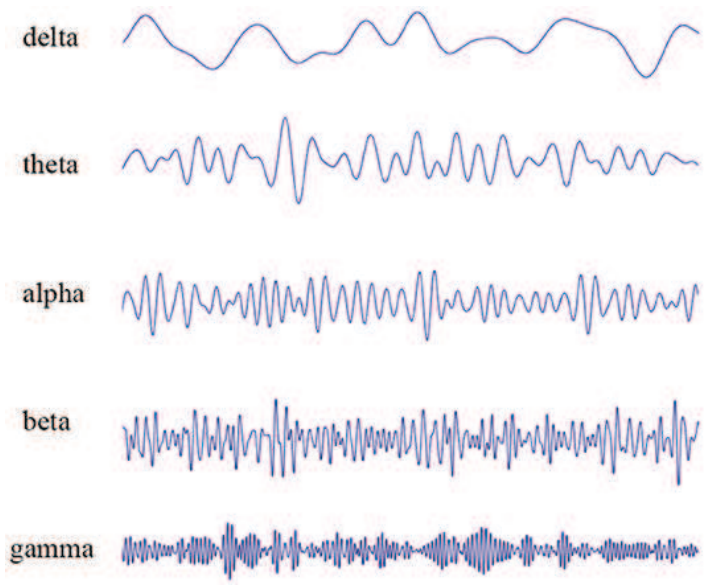


Figura 2.5: *Ritmos eléctricos cerebrales. Figura extraída de [32]*

2.3.2. Registros en el EEG durante episodios convulsivos

Una convulsión epiléptica se define como una ocurrencia transiente de signos o síntomas que se deben a una actividad neuronal excesiva o sincrónica anormal en el cerebro, tal como se definió en la Sección 2.1. Los neurólogos clínicos en la práctica diaria normalmente examinan registros cortos (usualmente de 20 minutos) de registros de EEG, de forma de poder descifrar los patrones que indican una posible ocurrencia de un episodio convulsivo [19] [18].

La señal de EEG de una persona que sufre epilepsia muestra distintas categorías de actividad anormal. Las señales en estos casos se dividen en períodos según el momento del episodio [33], como se puede observar en la Figura 2.6, siguiendo las siguientes denominaciones:

- Preictal: actividad observada previamente a la convulsión. Es la transición del período interictal al ictal, y sus síntomas, duración y tiempo de aparición son difíciles de determinar.
- Ictal: la actividad registrada durante un evento epiléptico. Varía en largo, sus síntomas son muy específicos y son identificados por especialistas. Un patrón de EEG distintivo se observa durante el período ictal, el cual consiste en formas de onda rítmicas con una variedad amplia de frecuencias.
- Postictal: actividad que sigue al punto de finalización de la convulsión, con el paciente recuperándose.
- Interictal: actividad anormal registrada entre convulsiones que precede el estado preictal. Las señales de EEG en esta fase no están asociadas con síntomas epilépticos. En la señal las pendientes de la actividad interictal epileptiforme son en general asimétricas con el componente inicial negativo y agudo, seguido por un componente positivo más lento. Estos patrones están presentes en la mayoría

de los pacientes con epilepsia, y por esta razón la detección de eventos interictales juega un rol vital en el diagnóstico de la patología [19].

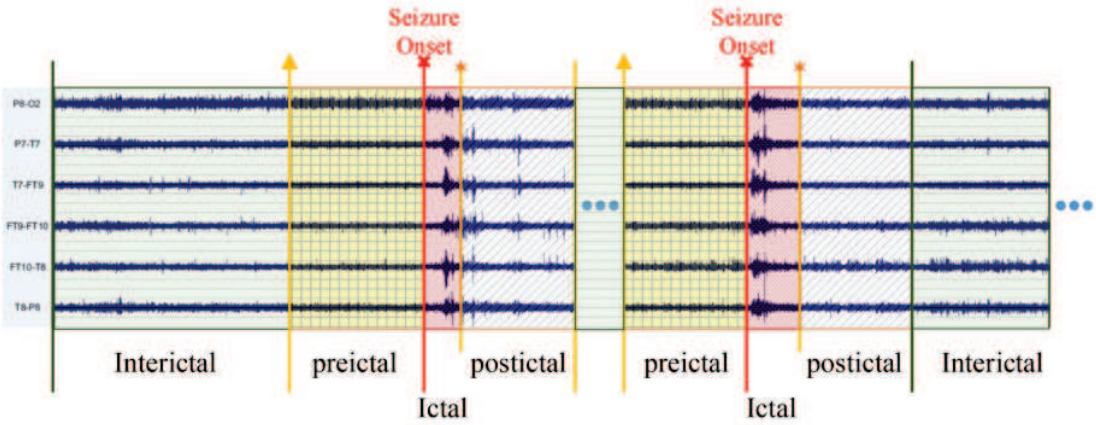


Figura 2.6: *Señales de electroencefalografía correspondientes a diferentes periodos de actividad ictal. Figura extraída de [33]*

Por otro lado, como se desarrolló en la Sección 2.1.2, según sus inicios clínicos y electrográficos las convulsiones se clasifican como focales, generalizadas o desconocidas. Las convulsiones repetitivas en la epilepsia focal tienden a tener un patrón consistente de inicio y evolución tanto clínica como electrográficamente. Por otro lado, en las convulsiones generalizadas, los cambios ictales en el EEG no se lateralizan e indican involucramiento del cerebro de forma bilateral, aunque se pueden detectar puntos de inicio asimétricos en algunos casos [19].

2.3.3. Registros de EEG intracraneal

Los registros de EEG intracraneales se utilizan ampliamente para la localización de las convulsiones y para realizar un mapeo funcional como parte de una evaluación prequirúrgica en pacientes. Estos registros evitan la atenuación de la señal causada por los efectos de filtro del cráneo y de la piel, los cuales se dan en el registro de superficie [7]. Es por esto que las puntas registradas por electrodos intracraneales pueden no aparecer en el EEG de superficie. Los electrodos intracraneales están en contacto directo con el tejido cerebral, incluyendo estructuras cerebrales profundas, por lo cual brindan una gran resolución espacial. Además, estas señales no son susceptibles a los artefactos electromiográficos, lo cual ofrece una mayor ventaja, especialmente en convulsiones hipermotoras. Sin embargo, los electrodos de iEEG pueden ser implantados solo en una región confinada, lo cual hace necesario que la implantación se base en hipótesis prequirúrgicas sólidas sobre la localización esperada del foco convulsivo.

Las convulsiones registradas intracranealmente difieren de aquellas registradas mediante EEG de superficie, ya que un patrón ictal en este estudio aparece únicamente luego de que la descarga ictal se haya sincronizado sobre un área relativamente extensa del cerebro. Es por esto que las señales eléctricas causantes de la convulsión suelen encontrarse ya dispersas cuando son detectadas por el EEG de superficie, y no es posible localizarlas en su punto de inicio. Los contactos intracraneales, por otra parte,

permiten la detección temprana de las descargas convulsivas si se colocan apropiadamente en la zona de inicio ictal, por lo que es más probable poder determinarlo con este método [7].

2.3.4. Desafíos en el análisis de patrones ictales en EEG

En general, neurólogos expertos analizan los registros visualmente, lo que puede resultaren un proceso extenso y tedioso en el tiempo [34] [35]. En particular, las características ruidosas de los registros de EEG hacen que sea difícil separar convulsiones de artefactos con patrones de tiempo - frecuencia similares [34].

La detección visual de convulsiones no ha probado ser muy eficiente [36] [35], y es por esto que se busca desarrollar esquemas de detección automática y eficiente de convulsiones, que faciliten el diagnóstico de la epilepsia y mejoren el manejo de registros de EEG de largo plazo. Desde el principio del análisis automático de señales de EEG epilépticas se han aplicado representaciones basadas en transformada de Fourier y métodos paramétricos [34]. Estos acercamientos se basan en observaciones anteriores que indican que las convulsiones generan cambios en ciertas bandas de frecuencias, como delta, theta, alpha y beta, descriptas anteriormente, pero dado que la naturaleza de las señales de EEG epilépticas es no estacionaria y depende de múltiples variables, estos métodos no siempre resultan adecuados para la descomposición en frecuencia de estas señales [35]. Además, dado que la evolución de las convulsiones es típicamente un proceso dinámico y no estacionario, sumado a que las señales se componen de frecuencias múltiples, los métodos visuales y basados en frecuencia convencionales tienen claras limitaciones [35]. En este sentido, los métodos basados en tiempo - frecuencia demostraron mejor rendimiento que los métodos convencionales de análisis frecuencial.

Como resumen, se puede considerar que:

- La señal de EEG es una señal del tipo no-estacionaria y por ende, sus características estadísticas cambian en los distintos sujetos y a través del tiempo para un mismo sujeto.
- La adquisición de datos de EEG es un sistema susceptible a un gran rango de artefactos que afectan negativamente el rendimiento de los clasificadores.
- La detección requiere la evaluación de un neurólogo directa y una gran cantidad de tiempo y esfuerzo humano.
- Los expertos con distintos niveles de experiencia diagnóstica reportan opiniones discrepantes sobre los resultados y posibles diagnósticos.

2.3.5. Espectrogramas de registros de EEG

La naturaleza dinámica del fenómeno epiléptico genera que las señales de EEG muestren un comportamiento estocástico y no estacionario [36]. Es por esto que no pueden ser estudiadas fácilmente en el dominio frecuencial, siendo que este análisis no provee información sobre la localización temporal de dicho cambio frecuencial. Sin

embargo, es posible analizar este tipo de señales, las cuales varían su contenido en frecuencia con el tiempo, mediante la aplicación de distribuciones de tiempo-frecuencia. Estas son potencialmente útiles para detectar y analizar los EEG con epilepsia no estacionaria, y entre los mismos se encuentran los espectrogramas [36].

Los espectrogramas consisten en gráficos de tiempo-frecuencia en los que se muestra la potencia de una señal en el tiempo correspondiente a distintas frecuencias. Para construir los espectrogramas se usa la transformada de tiempo corto de Fourier, en inglés Short Time Fourier Transform (STFT) [37] [35]. Para esto, una señal en el dominio del tiempo es dividida en pequeños fragmentos de igual duración, y sobre cada fragmento se realiza un análisis de Fourier, obteniendo la frecuencias encontradas para ese segmento. La totalidad de los espectros permite indicar cómo varía la señal en su contenido frecuencial a lo largo del tiempo. La forma de definir los segmentos en los que se analiza la señal es mediante ventanas de muestreo [35].

Para el cálculo de la STFT, una señal denotada por $x(t)$ es pre-ventaneada alrededor de un instante de tiempo t_0, en dónde se calcula la transformada de Fourier según la Ecuación 2.1.

$$STFT(t, f) = \int_{-\infty}^{+\infty} x(\tau)h(\tau - t)e^{(-i\omega\tau)}\, d\tau, \tag{2.1}$$

donde $h(t)$ es una ventana de tiempo corto, que debe ser elegida según el problema. Existen muchas ventanas posibles, entre las cuales se encuentra la ventana de Kaiser-Bessel, que permite un balance entre la precisión de la amplitud, distancia y altura del lóbulo lateral con respecto a otro tipo de ventanas. La STFT sufre de compensación entre el largo de la ventana y la resolución en frecuencia. Usando un análisis tiempo-frecuencia se calcula la densidad de potencia espectral de la señal, la cual representa la distribución de energía de la señal en el plano tiempo-frecuencia [35].

Los espectrogramas entonces son gráficos 2D en los que en el eje x se muestra el tiempo, en el y la frecuencia y se representa la potencia correspondiente a cada punto (x, y). Es así como el espectrograma se puede interpretar como una proyección en 2 dimensiones de una sucesión de transformadas de Fourier de tramas consecutivas, donde la energía y el contenido frecuencial de la señal va variando a lo largo del tiempo. La potencia se representa en el plano mediante un color representativo. Las escalas de colores en general van de un tono de baja amplitud, como puede ser el azul, a un tono de alta amplitud, como pueden ser el amarillo o el rojo. Este mapa de colores permite identificar señales fuertes y observar la variación de la frecuencia en el tiempo, tal como se observa en la Figura 2.7.

2.4. Resumen del capítulo

En este capítulo se describió la patología epiléptica, siendo la misma la patología sobre la que se basa el proyecto. Se buscó entender las bases fisiológicas y las consecuencias que tiene la misma sobre la calidad de vida de las personas que la sufren. A partir de esto se explicaron los distintos tratamientos existentes, introduciendo así los

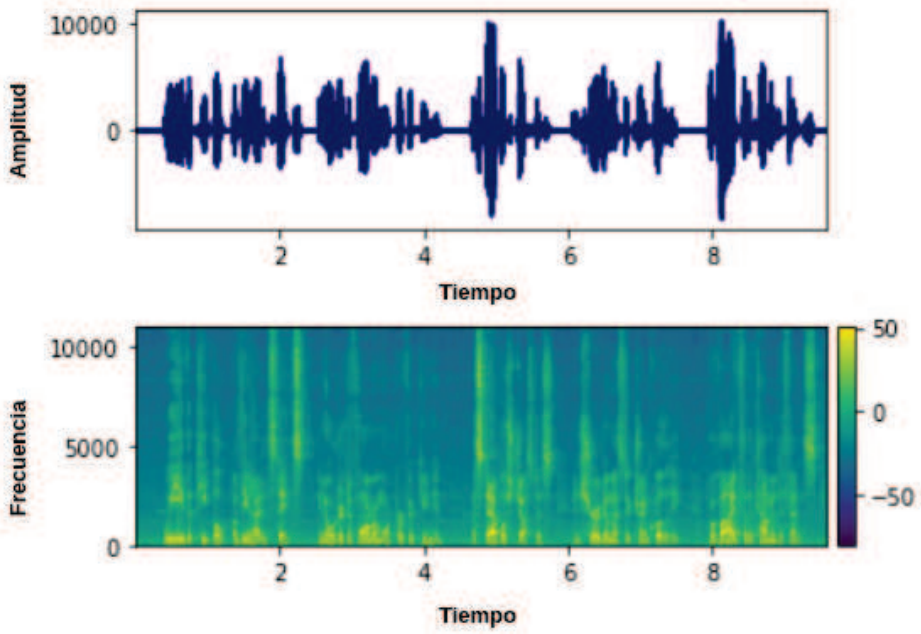


Figura 2.7: *Ejemplo de espectrograma, construido a partir de una señal de audio. Arriba, señal en el dominio temporal, abajo su correspondiente espectrograma. La barra de colores indica la densidad de potencia espectral para cada punto* $(tiempo, frecuencia)$

sistemas de tratamiento por neuroestimulación. Así se presentó el sistema de neuroestimulación responsiva, sistema que permitió obtener los datos para desarrollar este proyecto. Luego, se desarrolló sobre la importancia de los estudios de EEG e iEEG en el diagnóstico y seguimiento de la epilepsia. Aquí se introdujeron los desafíos existentes en el análisis de estas señales y se presentó la posibilidad de utilizar espectrogramas para resolver estos problemas. Habiendo desarrollado este contexto, en el siguiente capítulo se introducirán las técnicas de aprendizaje automático que se plantean utilizar para poder automatizar el análisis de este tipo de datos.

Capítulo 3

Técnicas de aprendizaje automático

En este capítulo se introducen los conceptos básicos sobre aprendizaje automático, redes neuronales artificiales, aprendizaje profundo y redes neuronales convolucionales que serán aplicados durante el desarrollo de este proyecto final.

Además, se describe la segmentación como una tarea dentro del campo del aprendizaje automático, y en específico se menciona su relación con las imágenes médicas. En este contexto se describe la arquitectura UNet y su utilidad.

3.1. Introducción

Actualmente, la inteligencia artificial (IA) se utiliza sobre todo como un término amplio que abarca todas las formas de inteligencia basada en cómputo. En líneas generales, se aplica a cualquier sistema que imita los procesos humanos de aprendizaje o toma de decisiones en respuesta a información, análisis de datos, reconocimiento de patrones o desarrollo de estrategias. Dentro de este campo se encuentra el aprendizaje automático (ML), que a su vez incluye un subcampo conocido como aprendizaje profundo (DL). Estos dos campos, en especial el último mencionado, se hallan en auge hoy en día, y hay una gran variedad de investigaciones sobre el alcance y las posibles aplicaciones de ambos [38].

Los métodos de ML son capaces de aprender a realizar tareas a partir de datos, analizándolos en busca de patrones o estructuras, al contrario de los métodos clásicos, donde los sistemas se codifican para que tomen decisiones según un conjunto de reglas determinadas por el programador. Los algoritmos de ML pueden potencialmente mejorar su desempeño con el transcurso del tiempo, gracias a su capacidad de aprender a partir de más datos. Este aprendizaje a partir de los datos se puede dar de dos formas principales, encontrándose por un lado el aprendizaje supervisado donde un grupo de datos con sus etiquetas don utilizados para entrenar un algoritmo, y por el otro el aprendizaje no supervisado, donde se utilizan los datos disponibles buscando encontrar patrones ocultos en los mismos, sin contar en este caso con etiquetas. En este trabajo se utilizaron métodos del aprendizaje supervisado.

Este tipo de algoritmos permiten construir sistemas complejos que son difíciles de diseñar a través del análisis manual. Hay muchos tipos de algoritmos de aprendizaje automático, pero los casos de uso más comunes incluyen:

- Clasificación: permiten la predicción de la categoría a la cual pertenecen las entradas, pudiendo clasificar los datos evaluados en dos (binaria) o más (multiclase) categorías o clases.
- Regresión: estos algoritmos predicen el valor continuo de un punto a partir de una entrada nueva en función de datos históricos.
- *Clustering*: consiste en descubrir aquellos grupos o subconjuntos de similares características dentro de un conjunto de datos.

Al igual que en el proceso de aprendizaje humano, estos modelos evalúan los datos según los elementos reconocidos de cada uno. El índice de éxito de la tarea realizada puede mejorar con el transcurso del tiempo a través de información aportada por datos, lo que ayuda a que el sistema aprenda y discierna las decisiones correctas de las incorrectas, con la mirada puesta en la máxima eficiencia y en una mayor exactitud [38].

Por su lado, el DL se enfoca en un subconjunto de algoritmos y modelos que están inspirados en la manera en que el cerebro humano reconoce y recuerda la información. Los algoritmos de DL utilizan redes neuronales profundas para aprender a partir de grandes conjuntos de datos [38]. En la Figura 3.1 se presenta la relación entre los tres campos presentados: inteligencia artificial, aprendizaje automático y aprendizaje profundo.

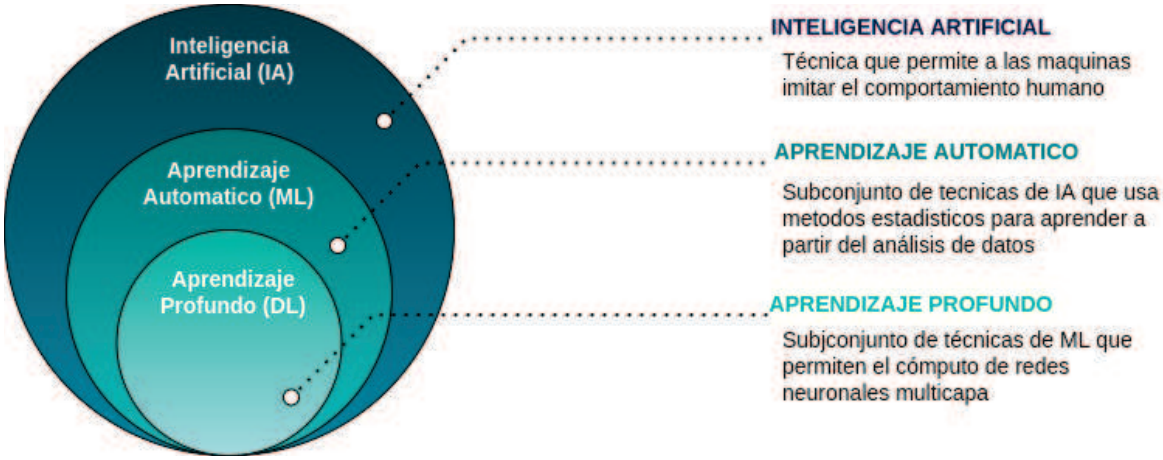


Figura 3.1: *Relación entre la inteligencia artificial (IA), el aprendizaje automático (ML) y el aprendizaje profundo (DL)*

3.2. Aprendizaje profundo

El aprendizaje profundo incluye un conjunto de algoritmos particulares donde los modelos son capaces de aprender representaciones de los datos en múltiples niveles

de abstracción. Cada capa incluida en estas arquitecturas está compuesta de módulos simples, que se aplican sucesivamente para poder extraer características más complejas.

Una particularidad de los métodos de DL en comparación a otras técnicas clásicas de ML, es que en dichas técnicas resulta necesario realizar la extracción de características para resolver la tarea en cuestión de forma artesanal. Estos vectores de características se extraen siguiendo distintos criterios a tomar, y buscando que los mismos permitan obtener información de los datos. Son luego esos vectores los que se someten al análisis en el algoritmo de ML y los que son evaluados para hacer una predicción. Estos métodos de extracción de características son desarrollados por expertos, quienes deben evaluar los datos para determinar la mejor caracterización. Esto resulta tedioso porque para cada problema en específico se tienen que extraer características específicas.

Aquí es donde el aprendizaje profundo marca una diferencia importante, dado que en este tipo de algoritmos la fase de extracción de características se elimina del proceso de desarrollo, ya que las características importantes de los datos son obtenidas por el modelo utilizado a partir de la evaluación de grandes volúmenes de datos. Este tipo de algoritmos minimiza el trabajo de procesamiento previo de los datos para extraer las características que permitan diferenciarlos de manera efectiva [38], como se puede observar en el esquema de la Figura 3.2. Con el *deep learning* se intenta entender el problema en términos de una jerarquía de conceptos, donde cada concepto se construye en base a los demás. Los conceptos en los niveles más bajos de la red codifican representaciones básicas del problema, mientras que los niveles más altos de la red usan los conceptos de los niveles anteriores para formar conceptos más abstractos.

Esta diferencia entre los algoritmos de ML y DL ha permitido abordar una mayor variedad de problemas, en específico en el tratamiento de datos complejos, como pueden ser las imágenes. Existen muchos problemas de análisis de imágenes que pueden ser tratados por medio de estas técnicas, como la segmentación de imágenes o la localización de objetos.

Sin embargo, una de las desventajas de estos modelos es la gran cantidad de parámetros que utilizan. En la actualidad, es posible entrenar estos modelos gracias al desarrollo de herramientas que poseen la potencia necesaria para manejar muchas líneas de cálculo paralelas, como las unidades de procesamiento gráfico (GPU) [38]. Una GPU es un procesador especializado con memoria dedicada que permite realizar operaciones de punto flotante, necesarias generalmente para renderizar gráficos, permitiendo dejar libre a la unidad de procesamiento central, en inglés *Central Processing Unit* (CPU), para otras tareas. En DL las GPU resultan óptimas dado que son capaces de procesar múltiples cálculos computacionales en simultáneo. Tienen una gran cantidad de núcleos de procesamiento, los cuales permiten trabajar con múltiples procesos paralelos.

3.2.1. Redes neuronales artificiales

Uno de los conceptos fundamentales en la formulación de modelos de aprendizaje profundo son las redes neuronales. Estas redes permiten representar funciones comple-

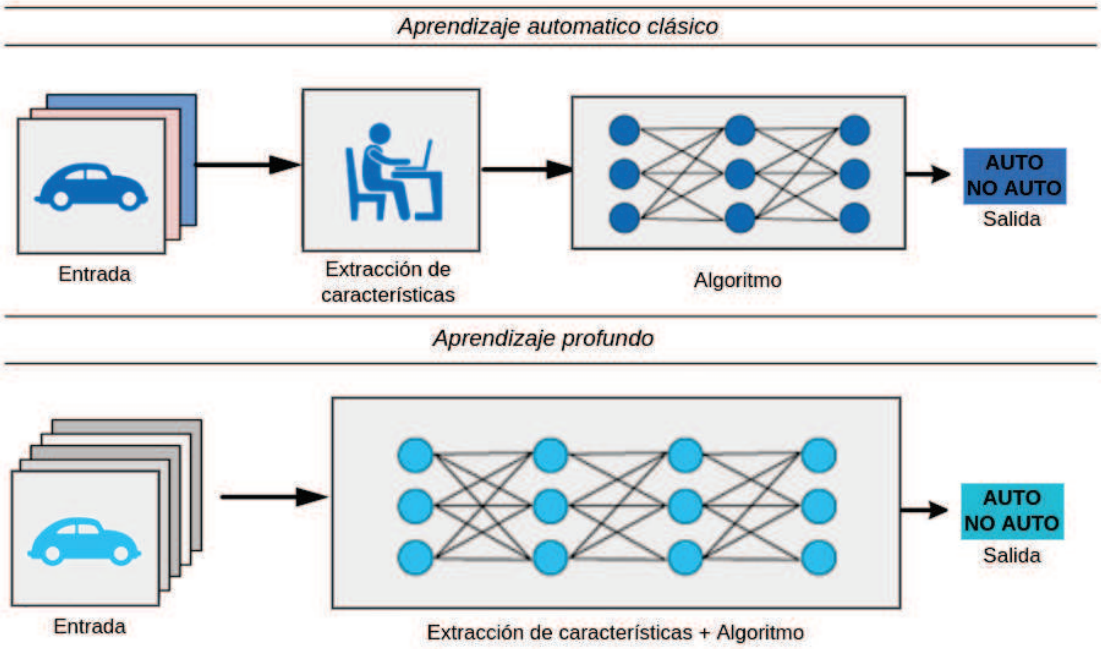


Figura 3.2: *Diferencias en el trabajo con aprendizaje automático y aprendizaje profundo. Figura adaptada de [39]*

jas y no lineales $y = f(x;\theta)$ que asocian una entrada x con una salida y, controladas por un conjunto θ de parámetros, los cuales pueden ser modificados. La idea principal de las redes se basa en el funcionamiento de la neurona biológica, en el que todas las entradas a la neurona, o unidad de procesamiento, se suman de forma ponderada y luego esa suma es evaluada por una función de activación que determina la salida. Esta idea sobre el funcionamiento de una neurona se puede resumir en la Ecuación 3.1 [38], donde w_i son los pesos de la red, x_i son las entradas y b es un término de bias.

$$y = f(\sum_{i=1}^{d} w_i x_i - b). \tag{3.1}$$

Uno de los modelos de redes neuronales más populares es el perceptrón multicapa, el cual está compuesto de múltiples unidades de procesamiento interconectadas y organizadas en capas. Este modelo se puede observar en la Figura 3.3. Las capas se dividen en tres tipos: la capa de entrada, la capa de salida y las capas ocultas. Los datos se muestran a la red en la capa de entrada y se transfieren a una o más capas ocultas mediante conexiones entre neuronas de cada capa. A través de las conexiones, la información se procesa utilizando operaciones ponderadas, alcanzando finalmente la capa de salida, en donde se obtiene el valor final. Los pesos de cada conexión son actualizados de forma iterativa durante el entrenamiento de la red, por lo que se consideran parámetros entrenables [38]. Entrenar una red neuronal consiste en ajustar cada uno de los pesos de las entradas de todas las neuronas para que las respuestas de la capa de salida se ajusten lo más posible a los datos conocidos [38].

El perceptrón es un ejemplo de red totalmente conectada, es decir, una red en la cual todas las neuronas de una capa se conectan con todas las neuronas de la capa

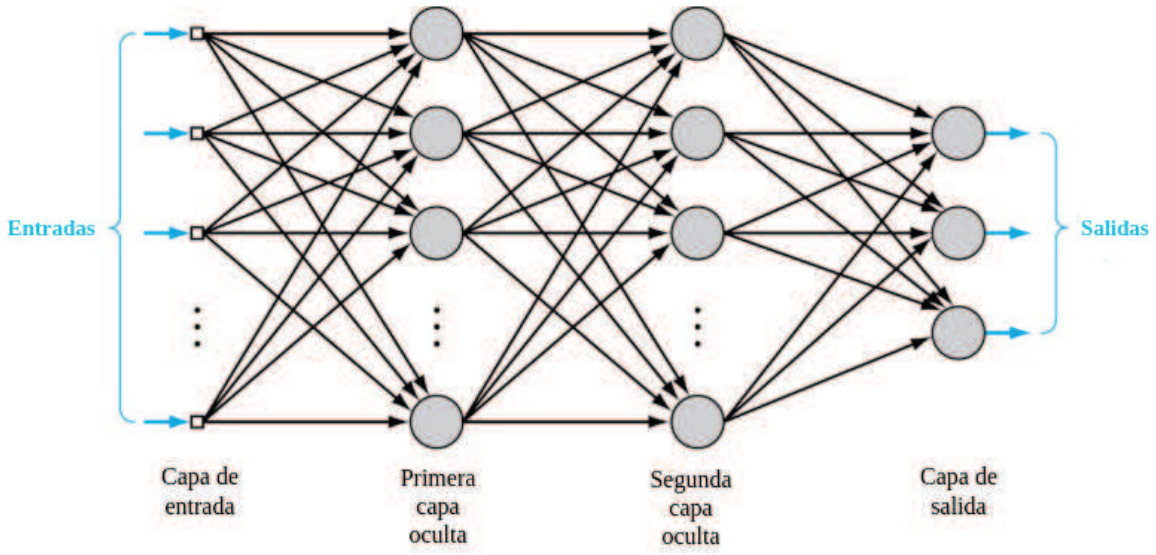


Figura 3.3: *Esquema del perceptrón multicapa. Figura adaptada de [40]*

siguiente. Esto genera que este tipo de redes cuenten con una gran cantidad de parámetros, debiéndose ajustar todos los pesos de las conexiones durante el entrenamiento. Además de las redes totalmente conectadas existen otro tipo de arquitecturas posibles como lo son las redes convolucionales, las totalmente convolucionales, las recurrentes, entre otras.

3.2.2. Función de pérdida y optimización

Durante el entrenamiento de los parámetros de la red neuronal en un esquema de aprendizaje supervisado, se busca que las salidas de la misma frente a entradas determinadas sean consistentes con el *ground truth* (es decir, las etiquetas) de los datos de entrenamiento. Los parámetros posibles para la red se encuentran dentro de un determinado espacio de parámetros, el cual consiste en el espacio de posibles valores de parámetros que definen al modelo. Para lograr esta consistencia se utiliza una función de pérdida, que va a dar un valor alto si las predicciones son malas y un valor bajo si las diferencias entre las salidas y las etiquetas son pequeñas [38].

De forma simplificada, la función de pérdida cuantifica la capacidad de un predictor determinado para aprender de los datos de entrenamiento, cuantificando la diferencia entre la salida del predictor y las etiquetas. A menor pérdida, mejor es el trabajo que está realizando el algoritmo al modelar la relación entre los datos de entrada y las etiquetas de salida. De igual manera, a mayor pérdida, mayor es el trabajo que necesita hacerse para mejorar la precisión del algoritmo [38]. Durante el entrenamiento, es necesario ajustar los parámetros de las distintas capas de la red, buscando disminuir el valor de la pérdida a medida que se avanza en el ajuste. Según el tipo de problema abordado existen distintas funciones de pérdida que se pueden utilizar [38]. Para un problema de regresión por ejemplo, es posible usar el error cuadrático medio como función de pérdida (el cual se calcula como el promedio de la diferencia al cuadrado entre las predicciones y las etiquetas), y para un problema de clasificación se puede

utilizar la entropía cruzada binaria (la cual permite comparar la probabilidad de cada clase con la clase real).

El rol de un optimizador es el de actualizar los pesos de cada capa de la red, de manera de minimizar la función de pérdida con la que se evalúa el rendimiento. El optimizador evalúa la función de pérdida y busca la dirección dentro del espacio de parámetros en la que la misma disminuye [41] [42].

Los objetivos de la optimización se pueden resumir en tres grupos. En primer lugar el optimizador busca que el algoritmo sea capaz de aprender de los datos y de encontrar la convergencia hacia una solución razonable. El segundo objetivo de la optimización consiste en que el algoritmo pueda converger lo más rápido posible. Por ultimo, el optimizador busca asegurar que el algoritmo llegue a converger a una solución con un valor objetivo bajo, idealmente siendo el mínimo global de la función de pérdida [42].

Una gran parte de los métodos están basados en descenso por gradiente, en inglés *Gradient Descent* (GD) [42]. La forma básica de esta técnica se puede consultar en la Ecuación 3.2, donde se observa la forma en la que se actualizan los parámetros que conforman el modelo a entrenar, identificados con θ_t.

$$\theta_{t+1} = \theta_t - \eta_t \nabla F(\theta_t), \tag{3.2}$$

donde η_t es la tasa de aprendizaje, también conocida como *learning rate*, y $\nabla F(\theta_t)$ es el gradiente de la función de pérdida en la iteración número t. El *learning rate* decide el valor de las actualizaciones de los pesos [41]. En el descenso por gradiente la actualización de pesos se da por épocas, siendo cada época un ciclo en el que cada uno de los datos de entrenamiento fueron evaluados y usados para calcular el gradiente. La cantidad de épocas es un hiperparámetro que debe ser optimizado durante la elección del modelo.

El algoritmo básico de descenso por gradiente tiene sus limitaciones de cálculo, por lo que existen diversas técnicas de optimización que se incorporan en la implementación para superar estos desafíos [41]. Una modificación al descenso por gradiente se da con el descenso por gradiente estocástico. Este es una modificación que computa el gradiente y actualiza la matriz de pesos en pequeños *batches* o grupos de datos de entrenamiento, en vez de con todo el conjunto disponible, los cuales son formados al azar en cada época. Mientras que se pueden generar actualizaciones más ruidosas, esto permite tomar más pasos para optimizar el gradiente (se da un paso por *batch* en vez de un paso por época), lo que lleva a una convergencia más rápida [38]. La cantidad de *batches* en los que se divide el conjunto de datos es un hiperparámetro que debe ser elegido durante la elección del modelo.

Uno de los métodos de optimización basados en descenso por gradiente de mayor popularidad es el método *Adaptive Moment Estimation* (Adam) [41]. Este es uno de los métodos más utilizados actualmente [42], ya que resulta eficiente para trabajar con problemas complejos que involucran muchos datos y grandes cantidades de parámetros.

Adam trabaja combinando dos tipos de metodologías de descenso por gradiente, Momentum y *Root Mean Square Propagation* (RMSProp):

- Momentum: este algoritmo permite acelerar el descenso por gradiente tomando en cuenta el promedio pesado exponencialmente de los gradientes, lo que permite hacer que el algoritmo converja al mínimo a un ritmo más veloz [42]. El objetivo es que a medida que se vaya entrenando, el algoritmo va tomando momento y obtiene menor pérdida en menos épocas. El término de momento debería incrementar la fuerza de las actualizaciones para aquellas dimensiones donde el gradiente va en la misma dirección y disminuir la fuerza de las actualizaciones de aquellas dimensiones cuyos gradientes vayan en distintas direcciones [38].
- RMSProp: este algoritmo utiliza diferentes tasas de aprendizaje para las variables teniendo en cuenta el gradiente acumulado en cada una de ellas. A diferencia de otros algoritmos que usan esta misma técnica, en RMSProp se utiliza el concepto de "ventana" para considerar solo los gradientes más recientes [42].

Para poder computar los gradientes con respecto a los pesos de las neuronas en la red se utiliza un método conocido como *backpropagation*. El algoritmo consiste en dos fases: una fase de propagación hacia adelante donde las entradas son evaluadas por la red y se obtienen las predicciones a la salida, y una fase de propagación hacia atrás, de donde deriva el nombre del algoritmo, donde se computa el gradiente de la función de pérdida respecto a cada peso de la red, por medio de la aplicación recursiva de la regla de la cadena [38]. Aplicando la regla de la cadena entonces es posible ir de atrás hacia delante calculando las derivadas parciales de la función de costo con respecto a todos los pesos en la red. Este algoritmo es el que permite el cálculo del gradiente de la función de pérdida, y es utilizado durante la actualización de los pesos por el método de gradiente descendente [38].

3.2.3. Funciones de activación

Las funciones de activación permiten transformar la salida de una neurona, antes de que la misma sea entregada como entrada a la siguiente capa (ya sea la de salida o una capa intermedia). Cada neurona suma la información de entrada y luego aplica una función de activación a su salida. El desempeño de las redes puede estar influenciado en gran medida por el tipo de función de activación que se utiliza [43].

Si no se utiliza una función de activación entonces la salida de la capa consistiría en una función lineal (dada por el producto interno entre el vector de entradas y el vector de parámetros), que puede resultar simple y fácil de resolver pero con capacidad limitada para representar problemas más complejos. Es por esto que se utilizan las funciones de activación entre capas, y por lo que las mismas generalmente son funciones no lineales. Agregando la no linealidad al modelo, el mismo es capaz de alcanzar relaciones entre los datos de entrada y salida arbitrariamente complejos, de gran dimensionalidad y capaces de modelar fronteras de decisión no lineales [43]. Existen distintas funciones de activación que pueden ser utilizadas, entre las cuales se encuentran:

- Función escalón o binaria: esta función es intuitiva pero no es diferenciable, por lo que se pueden generar problemas cuando se aplica el descenso por gradiente y se entrena la red [43].

- Función sigmoidea: esta función suele ser una buena opción dado que es continua y diferenciable en todo el dominio, es simétrica alrededor del eje y alcanza sus valores de saturación de forma asintomática. Sin embargo las neuronas saturadas (es decir, cuya salida es muy grande o muy pequeña) terminan produciendo gradiente nulo, ya el mismo tiende a cero cuando la entrada tiende a infinito (positivo o negativo). Esto resulta en un problema conocido como el gradiente que se desvanece, que dificulta el entrenamiento de las redes neuronales [43].

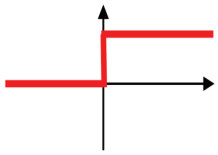
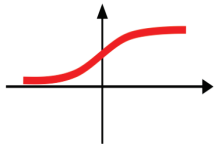
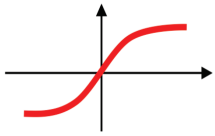
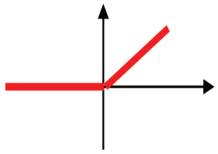
- Función tangente hiperbólica (en inglés *Hyperbolic Tangent* (TanH)): esta función tiene una forma similar a la sigmoidea pero resulta simétrica alrededor del origen. Esto genera que las salidas se encuentren en el rango de -1 a 1, pero está centrada en el cero y tiene un gradiente más pronunciado [43].

- Función unidad lineal rectificada (en inglés *Rectified Linear Unit* (ReLU)): también se conoce como función rampa. La función es cero para entradas negativas y luego incrementa linealmente para valores positivos. No es saturable y es computacionalmente eficiente [43].

La Tabla 3.1 muestra la expresión matemática y el gráfico asociado a cada una de las funciones de activación antes mencionadas.

Tabla 3.1

Funciones de activación [43]

Función	Expresión	Gráfico
Escalón	$f(x) = \begin{cases} 1 \text{ si } x > 0 \\ 0 \text{ sino} \end{cases}$	
Sigmoidea	$f(x) = \dfrac{1}{1 + exp(-x)}$	
TanH	$f(x) = \dfrac{2}{e + exp(-2x)} - 1$	
ReLU	$f(x) = \max(0, x)$	

3.2.4. Testeo y validación del modelo

Durante el aprendizaje del modelo se utilizan datos, denominados datos de entrenamiento, que permiten ajustar los parámetros de la red. Una vez que se ha entrenado el modelo con este conjunto de datos, es necesario evaluar su desempeño sobre datos desconocidos, de forma de poder discriminar si el modelo realmente es capaz de descifrar patrones sobre los datos o si durante el entrenamiento ha logrado únicamente "memorizar" los datos de entrenamiento. Es por esto que el conjunto de datos disponible no se utiliza completo para entrenar el modelo, sino que una parte de los datos se utiliza como un conjunto independiente para evaluar el desempeño final del modelo y su capacidad de procesar datos nunca vistos. Este conjunto independiente de datos se suele identificar como conjunto de testeo, y es usual que el mismo sea aproximadamente un 20 % de los datos disponibles totales [38] [44].

Como estos datos están pensados para probar el desempeño del modelo desarrollado no se deben utilizar los mismos durante la elección del modelo, ya que de esta forma no sería posible conocer el error real cometido por el mismo en el contexto real de uso [44]. Es por esto que generalmente, además del conjunto de datos de testeo, se utiliza un tercer conjunto de datos que se separa del conjunto de entrenamiento, conocido como conjunto de validación. Este conjunto permite optimizar el modelo y elegir los hiperparámetros, es decir aquellos valores configurables que definirán las condiciones de entrenamiento del modelo, sin caer en el sobreajuste a datos de entrenamiento [38].

Los datos de validación generalmente se utilizan durante el entrenamiento de los pesos del modelo, evaluando con la función de pérdida el error cometido en los mismos. De esta forma es posible monitorear el desempeño del modelo en datos que no se utilizaron para entrenar, y se puede comparar la curva de pérdida de validación con la de entrenamiento, para asegurar que el aprendizaje evolucione de forma pareja en ambos conjuntos.

3.2.4.1. Validación cruzada

La validación cruzada es una técnica generalmente utilizada para evaluar la variabilidad de un conjunto de datos y la confiabilidad de cualquier modelo entrenado con ellos. Esto se utiliza especialmente en los casos en que se entrena con conjuntos de datos chicos, donde es posible que no se pueda contar con un conjunto de validación que pueda cumplir correctamente con su función de controlar el error de generalización [44]. La forma de llevar a cabo esta técnica comienza por dividir el conjunto de datos en varios subconjuntos, para luego crear un modelo entrenado con cada uno y, a continuación, calcular las pérdidas y las métricas de rendimiento de cada modelo. Al comparar las métricas es posible interpretar la calidad del conjunto de datos [44]. El objetivo principal de la validación cruzada consiste en proveer una estimación imparcial del rendimiento del modelo. En comparación con el uso de un conjunto de validación para monitorear durante el entrenamiento, este acercamiento suele permitir una estimación más precisa del error de generalización.

Generalmente en los conjuntos de datos médicos suele haber pocos datos, siendo los mismos caracterizados por una disponibilidad limitada e inclusión de una canti-

dad de sujetos limitada con mediciones repetidas, donde cada sujeto genera múltiples muestras, como lo es en el caso de este proyecto. Al ser conjuntos de datos pequeños, puede suceder que la gran mayoría de los datos se usen en la fase de entrenamiento, cayendo en el riesgo de que el modelo no logre generalizar bien. Por esto es necesario estimar el rendimiento del modelo en configuraciones del tipo validación cruzada, que permitan estimar el error cometido por el modelo en escenarios que simulen la realidad clínica [44].

3.2.5. Técnicas para evitar el sobreajuste

Los métodos de regularización buscan mejorar la capacidad de generalización de los modelos, evitando el sobreajuste a los datos de entrenamiento. La regularización debe ser controlada ya que su aplicación desmedida puede generar subajuste, y el modelo podría no rendir bien ni llegar a modelar la relación entre los datos de entrada y las salidas [38] [44]. A continuación se listan distintas técnicas que se utilizan para evitar el sobreajuste a los datos de entrenamiento durante la fase de aprendizaje del modelo.

3.2.5.1. *Weight decay*

La técnica de regularización por *weight decay* es una técnica popular que ha demostrado su capacidad de mejorar el proceso de entrenamiento de distintos modelos y de generar modelos con mayor capacidad de generalización. La idea de la regularización por *weight decay* es que las redes con pesos de menor valor tienden a sobreajustar menos a los datos de entrenamiento y pueden generalizar mejor [44].

Weight decay consiste en aplicar un término de penalización a la función de pérdida buscando achicar los pesos en las capas. La forma de hacer esto es sumando un término con el cuadrado de la suma de los pesos multiplicado por un factor determinado, que se selecciona manualmente. El valor del coeficiente de *weight decay* define la importancia de minimizar la función de pérdida original y encontrar pesos pequeños [45].

3.2.5.2. Normalización por batch

El entrenamiento con métodos de descenso por gradiente en redes de gran profundidad puede resultar dificultoso conforme se incrementa la cantidad de capas. Como las entradas de cada capa se encuentran afectadas por los parámetros de todas las capas precedentes, los pequeños cambios en los parámetros de la red se amplifican a medida que la red se vuelve mas profunda. Esto hace que las capas se estén adaptando continuamente a nuevas distribuciones, experimentando cambios en la covarianza. La eliminación de este fenómeno en el entrenamiento puede generar un entrenamiento más veloz y estable, y esto es lo que se busca generar con el mecanismo de normalización por *batch* [46].

Las capas de *batch normalization* se usan para normalizar las activaciones de una entrada antes de pasar a la próxima capa de la red. Esto ha demostrado ser efectivo

para reducir el número de épocas que lleva entrenar una red neuronal y también ha demostrado su capacidad para estabilizar el entrenamiento [46]. La normalización por *batches* permite regularizar el modelo, disminuyendo la necesidad de aplicar otras técnicas de regularización. Por otro lado, este método tiene un efecto positivo sobre el cálculo del gradiente, disminuyendo su dependencia sobre la escala de los parámetros y sus valores iniciales. También permite usar mayores valores de *learning rate* sin caer en el riesgo de divergencia de la pérdida [46].

3.2.5.3. Early stopping

En el caso del *early stopping*, en vez de limitar los valores permitidos para los pesos, se limita el número de épocas durante el entrenamiento. El criterio mas conocido para determinar el punto de parada consiste en monitorear la pérdida en validación durante el entrenamiento, calculándolo en cada época, y luego frenar el entrenamiento cuando la validación no ha disminuido en una cantidad pequeña luego de una determinada cantidad de épocas. Esta cantidad de épocas generalmente se conoce como criterio de paciencia [44]. Al permitir la elección del modelo que mejor se ajusta a los datos de validación esta técnica resulta la forma más eficiente de selección de modelo para la evaluación en datos nunca vistos.

Además del potencial que tiene esta técnica para llevar a una mejor generalización, otro beneficio que tiene es que permite ahorrar tiempo. Una vez que se finaliza el período definido por la paciencia, el entrenamiento se corta y no se continúa con la modificación de los parámetros. En los casos en que los datos son muy ruidosos o hay una variabilidad intrínseca en las etiquetas, esta técnica resulta crucial [44].

3.2.5.4. Aumento de datos

El aumento de datos es una técnica que busca evitar el sobreajuste, generando nuevos datos artificiales a partir del conjunto de datos inicial. Esta técnica busca abordar el problema del sobreajuste generando modificaciones en el conjunto de datos de entrenamiento, asumiendo que es posible obtener más información de los datos aplicando técnicas de aumento sobre los mismos [47].

Los aumentos redundan en un aumento del tamaño del conjunto de datos de entrenamiento, agregando datos modificados al conjunto. En el caso de las imágenes, entre las modificaciones posibles se encuentran el reflejado, cambio de características en las imágenes como brillo o contraste, transformaciones lineales y no lineales, entre otros. Durante la elección de la técnica de aumento es importante entender el contexto del problema, y evaluar aquellos detalles que agregan complejidad al problema que se quiere resolver [47].

En el caso de los modelos de DL desarrollados para analizar imágenes médicas es usual encontrar que se han aplicado técnicas de aumento durante el entrenamiento. Esto es consecuencia de la disponibilidad limitada de conjuntos de datos de tamaño adecuado y correctamente etiquetados. Resulta especialmente complicado construir conjuntos grandes dada la rareza de ciertas patologías, la sensibilidad de los datos

médicos, el requerimiento de profesionales experimentados para etiquetar los datos, y los recursos necesarios para llevar a cabo procesos médicos [47].

3.2.6. Redes neuronales convolucionales

Las redes neuronales tradicionales suelen ser del tipo completamente conectadas, donde cada neurona en la capa de entrada se conecta a cada neurona de salida de la capa anterior (como la red que se puede observar en la Figura 3.3). Una de las grandes limitaciones de este tipo de redes es que tienden a presentar problemas con la complejidad computacional al procesar datos de entrada de gran tamaño, como sucede en el caso de las imágenes. Cuando se trata de imágenes de dimensiones pequeñas las redes completamente conectadas pueden funcionar. Sin embargo, al aumentar el tamaño de las imágenes, aumenta rápidamente la cantidad de parámetros de la primer capa de la red. Este aumento en los parámetros genera que la red se vuelva muy compleja y difícil de entrenar [38].

Las redes neuronales convolucionales, en inglés *Convolutional Neural Networks* (CNN), son una clase de red neuronal que se ha vuelto dominante en las tareas de visión computacional en los últimos años. Las CNN son un tipo de modelos de aprendizaje profundo que permiten procesar datos con patrones de grilla, como lo son las imágenes. Dichas redes están inspiradas en la organizaciones de la corteza visual de los animales y están diseñadas para aprender automática y adaptativamente las jerarquías espaciales de las características de los datos, desde las bajas resoluciones a las altas [48].

En la práctica las CNN tienen dos beneficios [38]:

- Invarianza traslacional: estas redes permiten localizar determinados objetos dentro de una imagen sin importar el espacio de la misma donde aparece el objeto.

- Composicionalidad: cada filtro consiste en un conjunto local de características de bajo nivel que luego permiten generar representaciones en mayores niveles de complejidad, similar a la composición de funciones matemáticas.

Las CNN son redes que típicamente se componen de dos tipos de capas: convolucionales y *pooling*. Estos dos tipos de capas se van combinando junto a otras operaciones adicionales para construir la arquitectura de red completa, como son las activaciones no lineales o la normalización de *batches*. Las capas iniciales aprenden los conceptos de niveles más bajos, como los bordes y los colores, y las siguientes capas aprenden los conceptos de mayor nivel, como puede ser la diferenciación de objetos. En un nivel más bajo las neuronas contienen información de una región pequeña de la imagen, mientras que a niveles mayores las neuronas contienen información de regiones más extensas. Por esto, al agregar más capas, el tamaño de la imagen decrece y el número de canales se incrementa. La disminución de tamaño a medida que se avanza en la red se hace mediante capas de *pooling*.

3.2.6.1. Capa convolucional

Una capa convolucional es el componente fundamental de las CNN, que permite realizar la extracción de características de las entradas, aplicando una combinación de operaciones lineales y no lineales. La capa convolucional consiste en un conjunto de una cantidad K de filtros entrenables, donde cada filtro tiene un ancho y un alto, siendo los mismos cuadrados generalmente. Estos filtros son pequeños pero se extienden en toda la profundidad del volumen. Para las entradas de la CNN, la profundidad es el número de canales que componen la imagen, pero para volúmenes más profundos correspondientes a las salidas intermedias de la red, la profundidad va a ser el número de filtros aplicado en la capa previa [38]. En la Figura 3.4 se puede observar un esquemático de los filtros utilizados en este tipo de capas.

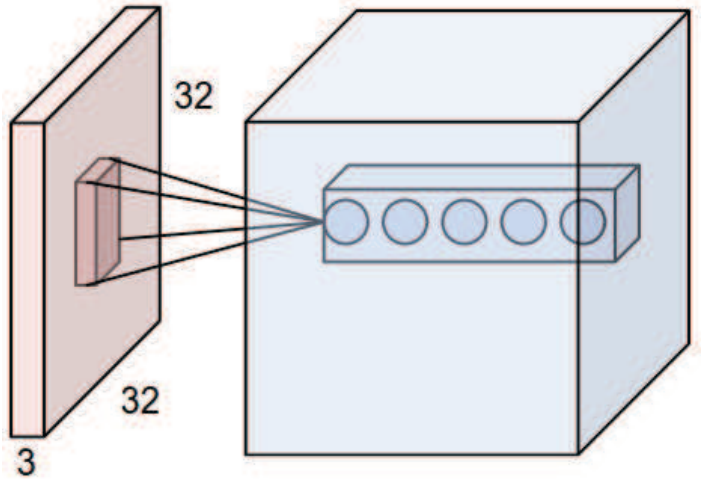


Figura 3.4: *Esquema de aplicación de un kernel en una capa convolucional. Figura extraída de [49]*

La red aprende durante el entrenamiento los valores de los filtros que se activan cuando reconocen un determinado tipo de característica en una localización espacial en el volumen de entrada. El concepto de activación se relaciona con la excitación de las neuronas cuando se presenta un determinado patrón en la imagen de entrada [38].

Hay varios hiperparámetros que permiten moldear la forma en que va a trabajar la capa convolucional, y que en general determinan la relación entre el volumen de salida y el de entrada.

- Tamaño del filtro: es el tamaño del filtro F, produciendo un kernel de $F \times F$ (en el caso de filtros cuadrados) que se convoluciona con el volumen de entrada [38].
- Profundidad: se relaciona con el volumen de la capa, es decir, representa el numero de filtros que se incluyen en una capa. Según la profundidad la capa tendrá una cantidad de filtros determinada que conectarán a una región local del volumen de entrada y generaran una salida con un volumen determinado. Cada filtro en la capa produce un mapa de activación que se activa en presencia de una cierta forma o patrón en los datos [38].
- *Stride:* el *stride* S está relacionado con el corrimiento de la ventana. Para cada

paso el filtro avanza una cantidad de píxeles o unidades correspondientes con el valor del *stride*, y luego se convoluciona cada uno de los filtros K con la región, generando la salida a almacenar. Cuando se crean las capas convolucionales generalmente se usa un paso de *stride* de 1 o 2 [38].

- *Padding:* consiste en aumentar los bordes de la imagen por medio del agregado de ceros o valores predefinidos. El objetivo es retener el tamaño original de la imagen cuando se aplica una convolución. La cantidad de *padding* aplicado es controlado por el parámetro P. Sin esto, las dimensiones espaciales de la entrada decrecen muy rápidamente y los volúmenes pueden terminar siendo muy pequeños como para entrenar [38].

Se puede computar el tamaño del volumen de salida como una función del volumen de entrada, el tamaño del filtro F (considerando filtros cuadrados), el *stride* S, y la cantidad de *padding* P, según la Ecuación 3.3.

$$O = \frac{W - F + 2P}{S} + 1, \tag{3.3}$$

donde W es el tamaño de la entrada, F el tamaño del filtro, P el *padding*, S el valor del *stride*, y O el tamaño de la salida.

Luego de cada capa de convolución se aplica una función de activación no lineal, como las descriptas en la Sección 3.2.3. La salida de esta mantiene la misma dimensión.

3.2.6.2. Capas de *Pooling*

La función principal de las capas *pooling* es la de reducir progresivamente el tamaño espacial del volumen de entrada. Esto nos permite reducir la cantidad de parámetros y la computación de la red, y además permite controlar el sobreajuste [38].

Este tipo de capas operan en cada uno de los cortes de profundidad de una entrada de forma independiente usando una función de agregación (máximo o promedio, resultando en una capa de *max pooling* o de *average pooling*, respectivamente) [38]. En la Figura 3.5 se pueden observar los distintos efectos que tiene capa tipo de capa de *pooling* sobre la imagen o matriz original.

Existe un tipo de capas que realizan la operación inversa al *maxpooling*, que se conocen como capas de *upsampling*. Esta capas permiten aumentar el tamaño de una cierta entrada, permitiendo así recuperar el tamaño original de una entrada luego de haber aplicado operaciones de convolución y de *maxpooling*. Esto se puede hacer mediante diferentes operaciones de interpolación, entre las cuales se encuentran vecino más cercano, bilineal, bicubica y trilineal [50].

3.3. Segmentación de imágenes

La segmentación semántica de imágenes es la tarea de clasificar cada píxel en una imagen a partir de un conjunto predefinido de clases [51]. Dada una imagen,

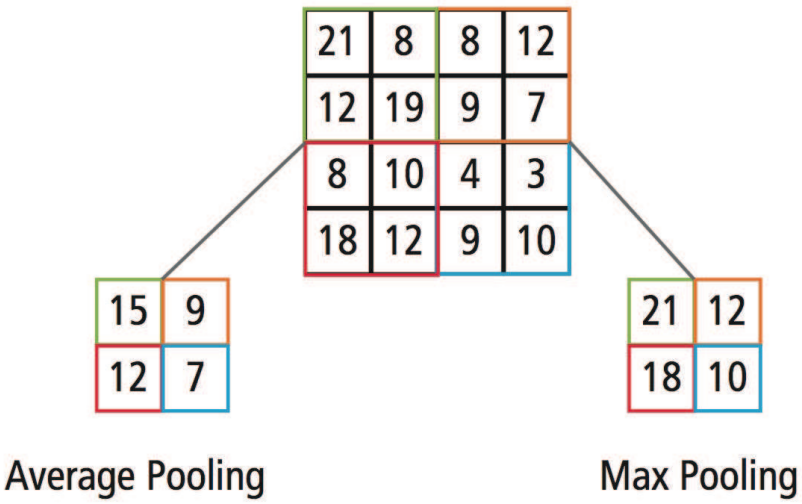


Figura 3.5: *Representación de los efectos de dos diferentes capas de pooling. A la derecha, average pooling, a la izquierda máx pooling. Los colores de los cuadrados muestran el efecto al tomar un filtro de tamaño* $2x2$*. Figura extraída de [49]*

un algoritmo de segmentación semántica etiqueta con el mismo valor numérico a los píxeles de una imagen que poseen coherencia semántica (es decir, pertenecen al mismo objeto o estructura). En particular, el objetivo es tomar una imagen que tiene un alto y un ancho, y una determinada cantidad de canales distintos que la componen (como es el caso de las imágenes RGB que se componen de 3 canales para generar el color), y generar una matriz del mismo alto y ancho que la imagen original conteniendo las etiquetas (generalmente números enteros) de las clases predichas para cada píxel. El algoritmo debe poder entender cuáles son los objetos presentes en una imagen y cuales son los píxeles que corresponden a cada objeto.

En el campo de la visión computacional, especialmente para tareas de segmentación, es usual encontrar el uso de redes neuronales convolucionales. Dentro de las CNN, un tipo de arquitectura muy popular es el de la redes completamente convolucionales. Este tipo de redes incluyen únicamente capas convolucionales y de *pooling* (evitando las capas totalmente conectadas), lo cual es muy útil para tareas de segmentación, ya que generan a la salida una imagen del mismo tamaño que la imagen de entrada [51] [52].

3.3.1. Arquitectura UNet para la segmentación de imágenes médicas

Dentro de las arquitecturas totalmente convolucionales, un tipo muy popular son las arquitecturas *encoder-decoder*, las cuales se caracterizan por tener dos partes: una que codifica la imagen de entrada, el *encoder*, y la otra que toma esa información y decodifica o reconstruye la imagen hasta llegar al mapa de segmentación deseado, el *decoder* [51]. El *encoder* reduce la dimensionalidad de la imagen de entrada hasta llegar a un vector de características de baja dimensionalidad. Por su parte, el *decoder* utiliza ese vector de características como entrada y aumenta poco a poco su dimensionalidad

hasta llegar a la imagen de salida, que será la correspondiente a la segmentación. Dentro de este grupo de arquitecturas, encontramos la red UNet [14], la cual sirvió como base para este proyecto.

La arquitectura de red neuronal U-Net fue desarrollada por Olaf Ronneberger y su equipo en el año 2015, con el objetivo principal de ser aplicada en la segmentación de imágenes biomédicas [14]. Esta arquitectura es una red del tipo completamente convolucional, lo que habilita que la red sea utilizable con imágenes de tamaños variables, es decir, a la entrada se pueden presentar imágenes de tamaños diversos, aunque los mismos deban cumplir con ciertas restricciones dadas por las transformaciones presentes en la arquitectura.

La arquitectura contiene dos ramas, como se puede observar en la Figura 3.6, siendo la primera una rama de contracción (que funciona como el *encoder*), y la segunda una rama de expansión (que toma el lugar del *decoder*).

En la primera parte la imagen es evaluada y se obtienen mapas de características de la imagen con distintos niveles de resolución a medida que se avanza por las capas de la red. La misma es utilizada para captar el contexto de la imagen, y consiste en un conjunto de capas de convolución y *maxpooling*. Este camino es similar a una red convolucional regular y es la que provee la información requerida para clasificar cada píxel. Esta parte se compone de la aplicación repetida de 2 convoluciones con filtros de tamaño $3x3$ seguidas por una activación ReLU y una operación de *maxpooling*. En cada operación de *maxpooling* se duplica la cantidad de canales de características.

La segunda etapa combina la imagen reconstruida y los mapas de características en la primera parte de la red [14]. Este camino es el que permite la localización precisa de las características de la imagen, y genera el aumento de la resolución de la imagen a la salida, lo que luego puede pasar a una capa convolucional final para crear una imagen completamente segmentada. Cada paso de esta parte consiste de una capa de *upsampling* seguido por una convolución con filtro de tamaño $2x2$ que disminuye a la mitad la cantidad de canales de características, y luego una concatenación del mapa en el paso anterior, la cual se obtiene a partir de una conexión cruzada entre los dos caminos (*skip connections*) seguido por dos convoluciones con filtros de tamaño $3x3$, cada una seguida por una activación ReLU. Estas conexiones cruzadas permiten que la red no solo obtenga la información codificada en la capa anterior sino que también se utilice la información correspondiente al mismo nivel de resolución del camino codificador, logrando así obtener una mejor localización de los componentes de las imágenes evaluadas. Al completar todos los niveles de la red se aplica una convolución de filtro $1x1$ para generar el mapa de segmentación final para cada clase [14].

En total la arquitectura cuenta con 23 capas convolucionales. La red resultante es casi simétrica, lo cual le permite tomar una forma de U para ser representada, forma de la que deriva su nombre. Lo que hace a esta red especial, es su capacidad de crear mapas de segmentación altamente detallados, utilizando muestras limitadas, lo cual se debe principalmente al uso de las *skip connections* en la red. Esta disponibilidad limitada de muestras suele ser normal en los datos médicos, ya que no es fácil conseguir una gran cantidad de datos anotados, y es por esto que esta red es popular en este

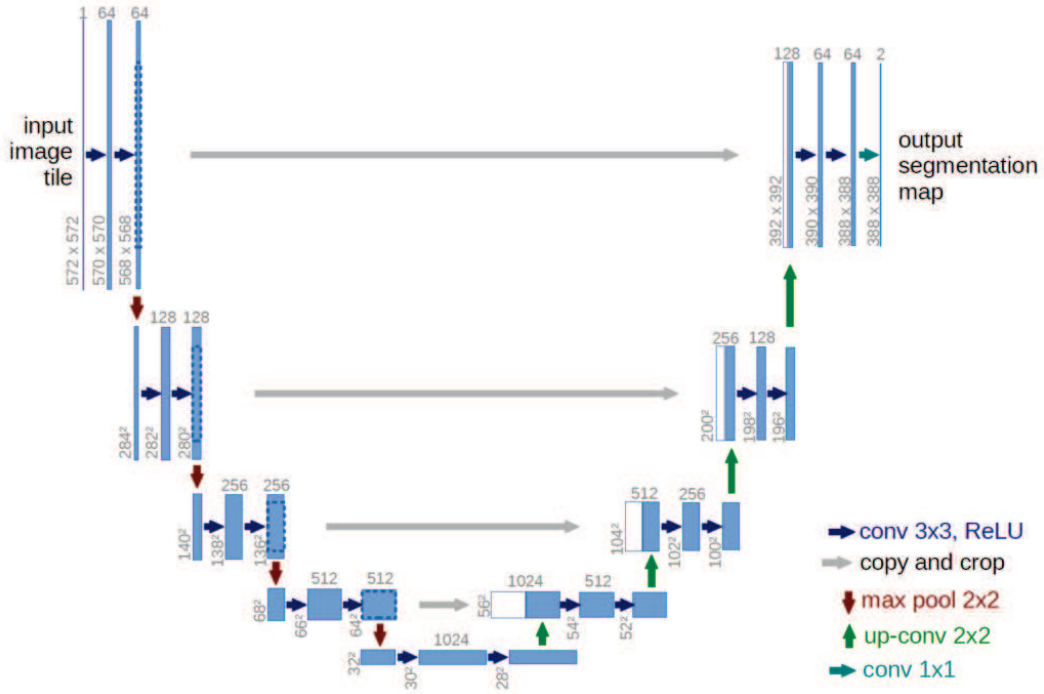


Figura 3.6: *Arquitectura UNet, extraído de [14]*

tipo de aplicaciones. Durante el entrenamiento de la red, en general se utilizan técnicas de aumento de datos basadas en deformaciones elásticas y aleatorias sobre los datos de entrenamiento, lo cual permite a la red aprender estas variaciones sin necesidad de tener una mayor cantidad de datos anotados [53].

Desde su creación la arquitectura UNet ha sido utilizada en gran medida en la segmentación de imágenes médicas, pero además se han dado diferentes avances en la arquitectura, dado que diversos investigadores han agregado nuevos métodos en su aplicación [52]. Dentro de las variantes se ha desarrollada una UNet 3D, otra en formato de red recurrente y otra con ventanas de atención [53]. También se han desarrollado redes con una cantidad menor de niveles de resolución o menores cantidades de parámetros, dado que, a pesar de ser una red totalmente convolucional, llega a niveles de resolución muy altos en los mapas de características, por lo que según el tamaño de las imágenes puede llegar a tener una gran cantidad de parámetros. En el caso de este proyecto en particular, y considerando que esta arquitectura fue creada para el procesamiento de imágenes pero que cuenta con múltiples variantes, se utilizó la misma como inspiración para generar un modelo que pueda procesar directamente espectrogramas de EEG.

3.3.2. Funciones de pérdida para evaluación de la segmentación

La elección de la función de pérdida u objetivo es muy importante durante el diseño de modelos de aprendizaje profundo, como se mencionó en la Sección 3.2.2. En las tareas de segmentación es posible aplicar distintas funciones de pérdida [54].

Específicamente, existen tres funciones de pérdida comúnmente utilizadas para esta tarea, las cuales fueron evaluadas en este proyecto. Estas funciones incluyen la entropía cruzada binaria, la entropía cruzada binaria pesada, y la perdida *soft Dice*, las cuales se listan a continuación.

3.3.2.1. Entropía cruzada binaria

La función de pérdida más comúnmente utilizada en la segmentación de imágenes es la pérdida de entropía cruzada píxel por píxel, en inglés *Binary Cross Entropy* (BCE) [51]. La función de pérdida examina cada píxel por separado y compara la predicción de clase (el vector de píxel en la dirección de profundidad) con el vector objetivo. La función de pérdida de entropía cruzada evalúa por separado la predicción de clase de cada vector de píxeles y luego promedia todos los píxeles, por lo que es posible pensar que los píxeles de la imagen contribuyen de igual manera, independientemente de la clase a la que correspondan. Sin embargo, el problema del desequilibrio de clases ocurre a menudo, por lo que el entrenamiento estará dominado por clases con más píxeles y es difícil aprender las características de los objetos más pequeños, lo que reduce la efectividad de la red [54].

Para el caso de la segmentación binaria se puede considerar que $P(Y = 0) = p$ y $P(Y = 1) = 1 - p$, y si las predicciones están dadas como $\hat{p}$, la ecuación para definir el valor de BCE se define en la Ecuación 3.4 [51].

$$BCE(p, \hat{p}) = -\left[plog(\hat{p}) + (1 - p)log(1 - \hat{p})\right] \tag{3.4}$$

3.3.2.2. Entropía cruzada pesada

La perdida de entropía cruzada evalúa las predicciones para cada clase de cada píxel de forma individual y luego hace el promedio de todos los píxeles, lo cual implica que se aprende cada píxel de igual manera. Esto puede generar problemas si las clases tienen representaciones desbalanceadas en la imagen, ya que la clase con mayor cantidad de píxeles puede dominar el entrenamiento, afectando al aprendizaje de la clase con menor presencia. Para esto se puede usar un valor de desbalance que puede contraatacar el desbalance en las imágenes, llegando a la entropía binaria cruzada pesada, en ingles *Weighted Binary Cross Entropy* (WBCE). La misma se define en la Ecuación 3.5, donde p representa los píxeles correspondientes a la clase *target*, $\hat{p}$ se refiere al valor de las predicciones y β representa el peso para la clase *target* [51] [55].

$$WBCE(p, \hat{p}) = -\left[\beta p\, log(\hat{p}) + (1 - p)\, log(1 - \hat{p})\right]. \tag{3.5}$$

De esta forma, para disminuir el número de falsos negativos, β se define en un valor mayor a 1, mientras que para disminuir los falsos positivos β se define en un valor menor a 1. Esto se conoce como un peso para valores positivos [55].

3.3.2.3. Pérdida *soft Dice*

Una de las funciones de pérdida más utilizadas en el caso de segmentación de imágenes es la pérdida basada en el coeficiente Dice [54]. Esta pérdida permite conocer el grado de solapamiento que hay entre las predicciones y las etiquetas. El coeficiente Dice tiene un rango de valores entre 0 y 1, donde 1 indica un solapamiento total[51]. El coeficiente Dice se puede calcular tal como indica la Ecuación 3.6, donde X representa el área de la clase *target* indicada en las etiquetas y Y representa el área de la clase *target* resultante en las predicciones.

$$DC = \frac{2|X \cap Y|}{|X| + |Y|}. \tag{3.6}$$

Considerando dos volúmenes binarios, el coeficiente Dice puede ser escrito como se muestra en la Ecuación 3.7, según:

$$DC = \frac{2\sum_i^N p_i g_i}{\sum_i^N p_i^2 + \sum_i^N g_i^2}, \tag{3.7}$$

donde la suma se realiza sobre la totalidad de N voxels, tanto del volumen de la segmentación binaria predicha p_i y el volumen binario con la etiqueta g_i.

En la función de pérdida *soft Dice* se usan las probabilidades predichas por el modelo para cada clase. Para poder minimizar al máximo posible esta función se usa $(1 - Dice)$ para definir la función de pérdida [51].

3.4. Resumen del capítulo

Durante el desarrollo del capítulo se desarrollaron los conceptos importantes para comprender las técnicas de aprendizaje automático, haciendo énfasis sobre las técnicas de aprendizaje profundo y las redes neuronales convolucionales. Estos conceptos resultaron importantes durante el proyecto para el desarrollo del sistema de procesamiento propuesto. Además se introdujo la tarea de segmentación semántica y se presentó la arquitectura UNet, la cual fue utilizada como inspiración para desarrollar el modelo utilizado en el proyecto. Adicionalmente se presentaron las funciones de pérdida que fueron utilizadas para entrenar dicho modelo. En el siguiente capítulo se pasará a describir la base de datos utilizada, junto con las herramientas que permitieron llevar a cabo el proyecto y las métricas que se usaron para poder definir el desempeño del modelo.

Capítulo 4

Datos, herramientas y métricas

En este capítulo se describen los datos y herramientas utilizados durante el diseño y desarrollo del proyecto. Además se indican las métricas utilizadas para evaluar el rendimiento de los modelos generados en cada experimento.

4.1. Herramientas

Durante el proyecto se trabajó con diferentes herramientas y entornos de desarrollo. Se utilizó el entorno de desarrollo Spyder para escribir y corregir el código generado para llevar a cabo los experimentos, el cual fue desarrollado en el lenguaje Python, utilizando distintas librerías. Se utilizó el sistema operativo Ubuntu Linux.

Además se utilizó una GPU para el entrenamiento de los modelos y el postprocesamiento de los resultados, la cual poseía la capacidad de procesar más datos al mismo tiempo (con respecto al uso de una CPU), permitiendo así un análisis más rápido y con mejor rendimiento. La forma de acceder a la GPU fue a través de los servidores del Instituto de Investigación en Señales, Sistemas e Inteligencia Computacional (sinc(i)), CONICET, UNL, siendo este el laboratorio donde se desarrolló el proyecto, ingresando a los mismos desde la terminal de Linux. El código se desarrolló y modificó en el entorno Spyder y luego se ejecutó por comandos en terminal habiendo conectado previamente a la GPU. La GPU utilizada era modelo NVIDIA GeForce RTX 3090 [56].

Por otro lado, para el manejo de las versiones del código desarrollado y la organización del repositorio se utilizó un gestor de versiones. En este caso se usó GitLab como plataforma, la cual permitió tener todo el código en un mismo espacio y acceder a una misma versión desde distintos puntos, simplificando el uso del servidor.

4.1.1. Anaconda y Spyder

Anaconda [57] es una plataforma libre y gratuita para los lenguajes Python y R utilizada principalmente en ciencia de datos y aprendizaje automático, la cual puede

ser utilizada en Windows, macOS o Linux. Su principal ventaja consiste en simplificar la gestión e implementación de paquetes. Incluye un conjunto de paquetes que permite ejecutar y desarrollar programas en Python. Anaconda también incluye un gestor de paquetes llamado conda, el cual hace que sea fácil instalar y actualizar paquetes adicionales.

Como entorno de desarrollo se utilizó Spyder [57]. El mimso consiste en un entorno de desarrollo integrado y multiplataforma de código abierto, en inglés *Integrated Development Environment* (IDE), para programación científica en el lenguaje Python. Permite escribir código en Python e interactuar con el mismo desde una interfaz sencilla de utilizar. La razón principal para utilizar Spyder consistió en poder desarrollar *scripts* que luego puedan ser ejecutados por comandos desde una terminal. Esta aplicación posee una consola integrada que puede ser usada de forma interactiva para evaluar el código editado en simultáneo. También incluye una integración para visualizar figuras, documentación relacionada con el proyecto, y un explorador de variables, que permite visualizar las variables creadas durante la ejecución de un *script*.

4.1.2. Pytorch y CUDA

Pytorch [58] es un entorno de trabajo *open source* que contiene todas las funciones que permiten al usuario desarrollar cálculos computacionales con tensores, lo que permite principalmente acelerar entrenamiento de aprendizaje profundo mediante el uso de GPU. Pytorch es muy utilizado en los desarrollos de aprendizaje profundo ya que posee una interfaz simple y flexible, que resulta fácil de implementar.

Pytorch puede ser utilizado en combinación con el lenguaje conocido en inglés como *Compute Unified Device Architecture* (CUDA) [59]. Este es un lenguaje de programación y un kit de herramientas desarrollado por NVidia, que es utilizado para realizar operaciones computacionales complejas, como por ejemplo multiplicaciones matriciales, de manera más rápida al paralelizar tareas a través de los núcleos de la GPU.

4.1.3. Git y GitLab

Git es un sistema de control de versiones, un *software* que sirve básicamente para gestionar las versiones por las que va pasando el código de los proyectos. Git es el más popular de los sistemas de control de versiones en la actualidad y una herramienta indispensable para el desarrollo de proyectos.

GitLab es una plataforma en la nube que ofrece un servicio web de control de versiones y desarrollo de *software* colaborativo basado en Git. Además de ser un gestor de repositorios, la plataforma ofrece distintas herramientas para facilitar el manejo y desarrollo de proyectos.

4.2. Base de datos

En este proyecto se trabajó con datos provistos por los directores, quienes ya habían realizado experimentos previos con los mismos. Los datos crudos se obtuvieron de forma previa al proyecto. Corresponden a una base de datos generada a partir de mediciones en 24 pacientes con epilepsia focal resistente a medicación implantados con el sistema RNS, quienes fueron monitoreados entre los años 2015 y 2020. Cada uno de estos pacientes se identifica con un código particular, siguiendo la estructura PIT-RNS#ID, pudiendo así identificar correctamente los datos correspondientes a cada uno. Los datos relacionados con el contexto de las mediciones, incluyendo los registros y las configuraciones de detección y estimulación, fueron obtenidos del Sistema de Manejo de Datos de Neuropace junto con los datos electroencefalográficos.

Los pacientes involucrados en la base de datos pertenecían a distintas nominaciones o grupos según la zona y localización de los electrodos implantados bajo el protocolo RNS. La localización de los electrodos implantados varia entre pacientes y según esta localización los mismos fueron agrupados en Neocortex, Hipocampo, Neocortex-Hipocampo o Displasia Migracional. Los grupos correspondientes a cada paciente se pueden consultar en el Anexo B. En el proyecto se entrenó el modelo sin diferenciar los datos por grupo de implantación del paciente al que pertenecían, de forma de que el mismo lograra adaptarse a todos los grupos por igual y funcionar de manera independiente a la zona de implantación.

Los registros crudos a partir de los cuales se construyeron los espectrogramas utilizados consistían en señales de electroencefalografía intracraneal de 90 segundos en 4 canales, es decir con un canal por electrodo del sistema RNS. Estas señales fueron filtradas de manera *online* en 4-125 Hz, muestreadas en 250Hz y digitalizadas por un filtro conversor analógico-digital de 10 bits. Las derivaciones del iEEG se colocaron de modo que fueran bipolares entre electrodos vecinos, conectados a masa a la carcasa del neuroestimulador, resultando en impedancias menores a 1KΩ para todas las mediciones.

En concordancia con la práctica clínica establecida, los patrones ictales en el iEEG fueron visualmente identificados por un neurofisiólogo especialista en cirugía de epilepsia y confirmados por un experto en epilepsia certificado. De esta manera se logró un acuerdo mutuo del 99.8 %, y sólo se mantuvieron aquellos datos con concordancia entre expertos. El punto de inicio de los patrones ictales fue anotado por un marcador sobre los registros crudos. Se consideró siempre el concepto de patrón ictal en vez del de convulsión, ya que el sistema no provee información alguna sobre la manifestación clínica del episodio, únicamente muestra los registros electroencefalográficos.

De esta manera se llegó a contar con el detalle de los datos crudos de las señales de electroencefalografía invasiva con las anotaciones sobre la muestra. Estas anotaciones incluían en primer lugar la indicación sobre la presencia o no de un patrón ictal en la señal que indicara un posible patrón ictal. En segundo lugar se indicaba el tiempo de inicio del patrón ictal sobre el episodio, en aquellos casos en que se observaba un episodio.

4.2.1. Generación de espectrogramas

Los datos con los que se desarrolló este proyecto final no consistieron en los datos crudos de las señales de iEEG que se obtuvieron del sistema RNS sino que directamente se trabajó con espectrogramas derivados de estas señales. Previo al desarrollo del proyecto, cada segmento de iEEG de 90 segundos con sus 4 canales se tipificó y luego se transformó al espacio tiempo-frecuencia. Dichos espectrogramas fueron construidos con una ventana de Kaiser de 0.5 segundos.

Por cada muestra se evaluaron períodos de 90 segundos, los cuales eran evaluados por ventanas de medio segundo, resultando en un total de 180 muestras temporales por segmento. Por otro lado, las frecuencias analizadas en los espectrogramas se encontraban en un rango de 0 a 60Hz, con una resolución de 0.5Hz, por lo que en total se tenían 120 píxeles por espectrograma en el eje frecuencial. La potencia de cada espectrograma se escaló a decibeles restringiendo el máximo a 40dB.

Los segmentos que resultaban mayores a 90 segundos fueron cortados para extraer muestras no solapadas de 90 segundos y aquellos que resultaban menores a 90 segundos y no llegaban a completar un evento fueron ampliados con una técnica de *zero-padding* de manera previa a la construcción de los espectrogramas. Durante el procesamiento no se aplicó ninguna técnica de remoción de artefactos ni de preprocesamiento adicional de los datos crudos.

Al final de la transformación se contó con una base de datos con muestras de tamaño 4x120x181 (en el orden Canales x Frecuencia x Tiempo). Cada conjunto de 4 espectrogramas se considera una muestra o dato, pudiendo ser el mismo perteneciente a la clase positiva (es decir que fue identificado un patrón ictal con su respectivo *onset* durante el análisis de los expertos), o a la clase negativa (sin episodio ictal, es decir, datos que se obtuvieron de segmentos temporales durante el período interictal o actividad de base del paciente). Un ejemplo de una muestra se puede observar en la Figura 4.1, siendo esta una muestra de clase positiva, donde el inicio del patrón ictal está demarcado por una línea punteada roja.

4.2.2. Desbalance en la base de datos

Para el proyecto se contó con 36293 muestras preprocesadas, de las cuales únicamente 8693 correspondían a muestras con patrón ictal. Esto permite ver que el conjunto de datos tiene un desbalance, siendo un 23.95 % de los datos de clase positiva, o con episodio ictal. Durante el desarrollo del proyecto se debió considerar este desbalance en las clases dado que el mismo tenía el potencial de dificultar el aprendizaje de la red, ya que se contaba con una cantidad sustancialmente menor de datos con presencia de patrones ictales.

En la Figura 4.2 es posible observar el desbalance de los datos por paciente, tendiendo a ser en general una relación de datos por clase baja. En el Anexo B se puede consultar la distribución de datos por sujeto, y es posible encontrar que el porcentaje de datos positivos generalmente se encuentra por debajo del 50 % para toda la base de datos, con solo unas pocas excepciones que se acercan o superan levemente este valor.

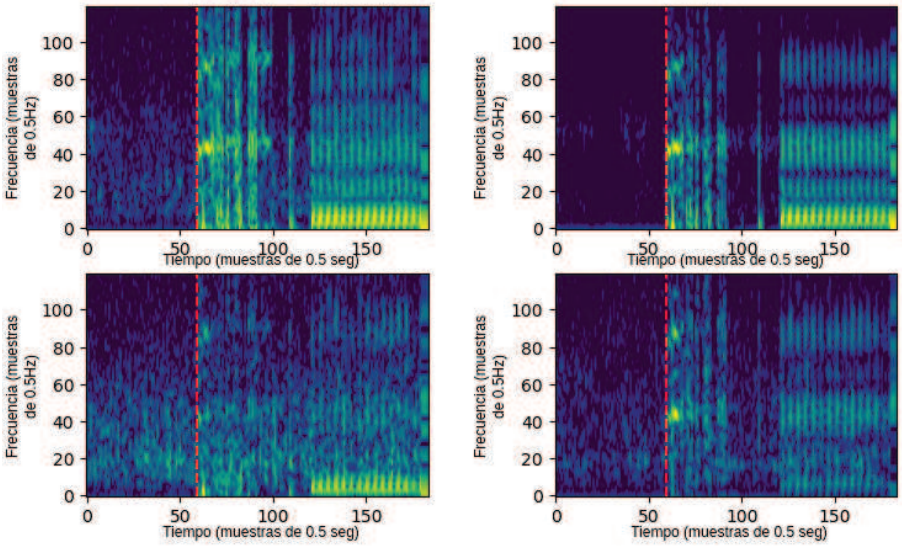


Figura 4.1: *Espectrogramas obtenidos en los 4 electrodos en un mismo momento que conforman una muestra de los datos de entrada. En esta muestra se puede observar un patrón ictal, cuyo inicio es marcado por una linea punteada roja. Se puede observar que al ser una muestra conformada de señales tomadas por electrodos distintos las mismas difieren entre sí pero muestran al mismo tiempo el cambio en los patrones de los espectrogramas.*

El desbalance de datos se traduce en que el resultado del entrenamiento del modelo puede no entregar información discriminativa para el problema en cuestión, y no tener una capacidad de predicción que sirva para su uso posterior. Es por esto que este desbalance fue tomado en cuenta en gran medida durante el desarrollo del modelo, influyendo especialmente en la elección de las métricas de evaluación.

4.2.3. Armado de los conjuntos de datos

En los distintos experimentos realizados en el proyecto se agruparon los datos con distintos criterios, los cuales se ajustaban a lo buscado en cada prueba. Es por esta razón que se pueden diferenciar dos formas en las que se agruparon los datos.

4.2.3.1. Partición de datos para diseño de la red

Para todos los experimentos que se realizaron para la elección del modelo se trabajó con un conjunto de datos de entrenamiento, uno de validación y uno de testeo. Los conjuntos de datos se divieron por paciente, de forma de tener una determinada cantidad de pacientes en cada uno, y que no se mezclaran los datos de cada sujeto entre los distintos conjuntos. Esto se hizo con el fin de que el modelo resultara aplicable incluso sobre sujetos cuyos datos no fueron involucrados dentro de estos experimentos, favoreciendo que el mismo fuera utilizable en contextos clínicos para soporte al profesional médico. Además, se buscó que los conjuntos de validación y testeo estuvieron compuestos por pacientes pertenecientes a distintos grupos según su zona de

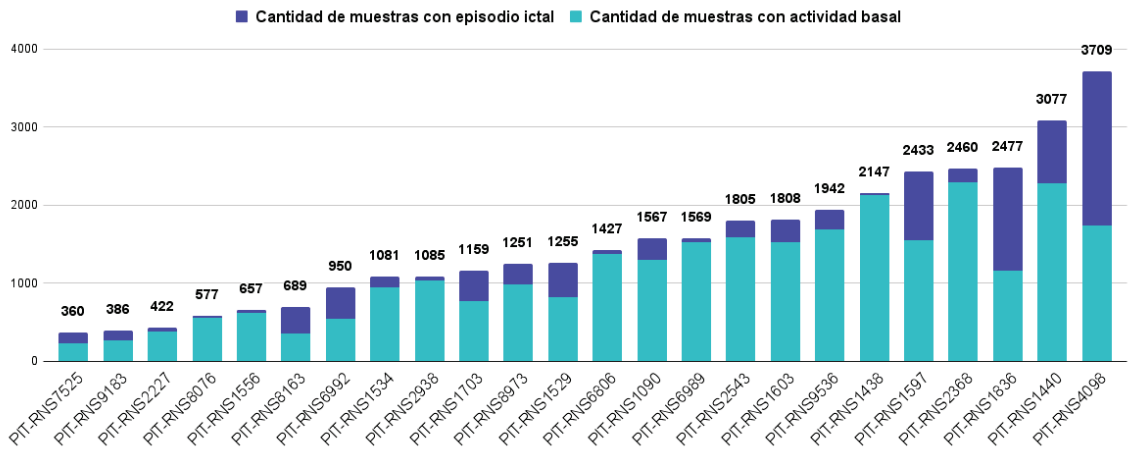


Figura 4.2: *Datos disponibles por paciente ordenados según la cantidad de muestras disponibles para cada sujeto. Con la diferencia de colores se indica la cantidad de muestras correspondientes a cada sujeto por clase. Note que PIT-RNS#ID corresponde al identificador de cada paciente.*

implantación de electrodos del RNS, de forma de evitar que se dieran diferencias en la evaluación de espectrogramas de sujetos pertenecientes a distintos grupos.

La cantidad de datos para cada conjunto se puede consultar en la Tabla 4.1. Además, en los datos de entrenamiento generalmente se trabajó con técnicas de aumento de datos, de forma de incrementar la cantidad de datos disponibles para mejorar el aprendizaje de la red. Estas técnicas se presentarán más adelante en la Sección 5.2.1.

Se puede observar que se trabajó con un 85 % de los datos para entrenamiento y validación, es decir para el desarrollo y elección del modelo, mientras que aproximadamente un 15 % de los datos disponibles se utilizaron para testear los distintos modelos y evaluar su capacidad de generalización. Estas proporciones se encuentran dentro de los valores normales utilizados en los desarrollos de modelos de aprendizaje automático [38]. Luego, dentro del 85 % de los datos usados para desarrollar y optimizar el modelo, aproximadamente un 22 % (un 19 % del total) se uso para validación y un 78 % (un 66 % del total) para entrenar los parámetros del modelo.

También es posible observar, como se mencionó en la descripción de la base de datos, que la cantidad de datos de clase positiva (con episodio ictal) resulta muy baja en todos los conjuntos. El conjunto de datos con mayor cantidad de muestras positivas es el de entrenamiento, lo cual permitió que el modelo aprenda y pueda ver datos con episodio durante el aprendizaje de los parámetros.

Los cuatro pacientes que componen el conjunto de datos de testeo pertenecen cada uno a un grupo de implantación distinto, es decir, cada uno tiene electrodos del sistema RNS implantados en distintas ubicaciones anatómicas. Se realizó esta prueba para poder analizar el rendimiento del modelo en distintos grupos evaluados, de forma de poder realizar un análisis de desempeño que estuviera ligado a la clínica de cada uno de estos grupos.

Tabla 4.1

Distribución de datos para el desarrollo del modelo

Dataset	Cant. de pacientes	Totales	Clase 1	Clase 0	% sobre el total	% de positividad
Entrenamiento	15	24048	7274	16774	66.26	30.25
Validación	5	6920	724	6196	19.07	10.46
Testeo	4	5325	695	4630	14.67	13.05
Total	24	36293	8693	27600		

4.2.3.2. Partición de datos en el contexto clínico

Para evaluar el rendimiento del modelo elegido en un contexto comparable con uno clínico, se realizó una validación cruzada del modelo elegido, siguiendo la idea presentada en la Sección 3.2.4.1. La forma de trabajar con este tipo de validación consiste en evaluar el rendimiento de los modelos en pequeños conjuntos o cuando se retiene parte de los datos para realizar el análisis.

Para realizar la validación cruzada en este proyecto se utilizó una exclusión de datos por grupos de muestras, en la que cada grupo se determinó por paciente, por lo que la cantidad de muestras de entrenamiento y testeo dependió del paciente evaluado. Como la exclusión de datos se realiza por paciente esta forma de validación cruzada se conoce como *Leave One Subject Out* (LOSO). Esto permite que el modelo no conozca ningún dato del paciente antes de evaluar en el mismo, por lo que se elimina la posibilidad de que resulte imparcial durante la evaluación. Adicionalmente, de manera que los experimentos en la validación cruzada resulten comparables a los realizados durante la elección del modelo, se utilizaron 3 pacientes, distintos al excluido, para usar como datos de validación durante el entrenamiento.

4.3. Métricas de evaluación de los resultados

De forma de poder evaluar los resultados obtenidos en cada experimento se eligieron determinadas métricas para cada resultado a evaluar. Con el fin de generar resultados comparables se utilizaron las mismas métricas en todos los experimentos.

4.3.1. Clasificación de las señales

La predicción de convulsiones puede analizarse como un problema de clasificación binaria, donde una clase consiste en las muestras con características preictales o ictales, es decir, las que se dan en el momento previo o durante un patrón ictal, y la otra clase consiste en las muestras con iEEG con actividad normal. Identificar las características que pueden diferenciar o discriminar el estado preictal o ictal del estado interictal es la clave para la predicción de eventos.

Una matriz de confusión es una métrica muy popular utilizada en los problemas

de clasificación [60]. Las matrices de confusión representan las cuentas de casos entre valores predichos y reales, como se indica en la Figura 4.3. El grupo de verdaderos negativos, en inglés *True Negatives* (TN), representa la cantidad de negativos reales, es decir, aquellos valores que son negativos y fueron predichos correctamente. Los verdaderos positivos, en inglés *True Positives* (TP), son los valores positivos que fueron predichos como positivos correctamente. El valor de falsos positivos, en inglés *False Positives* (FP), indica la cantidad de valores reales negativos que fueron clasificados como positivos. Asimismo el valor de falsos negativos, en inglés *False Negatives* (FN), indica los valores positivos que fueron clasificados como negativos. En una clasificación se va a buscar que el modelo pueda obtener la mayor cantidad de verdaderos negativos y verdaderos positivos, con una cantidad baja de falsos positivos y falsos negativos.

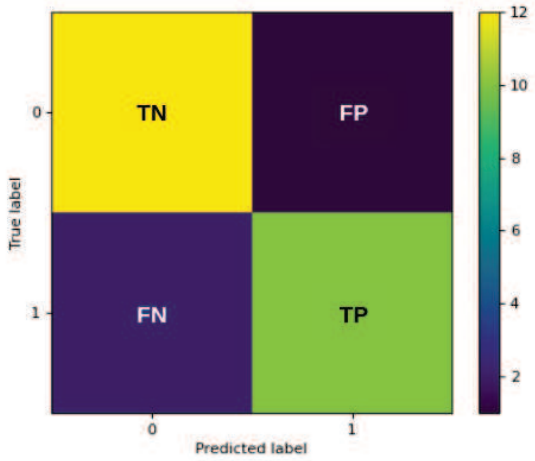


Figura 4.3: *Esquema de una matriz de confusión. FN denota la cantidad de falsos positivos en la clasificación y TN denota los falsos negativos, mientras que TN indica los verdaderos negativos y TP indica los verdaderos positivos.*

Una de las métricas más comunes para evaluar el rendimiento de un clasificador es la exactitud, en inglés *Accuracy* (ACC). La exactitud de un modelo es calculada según la Ecuación 4.1 [60].

$$ACC = \frac{TN + TP}{TN + FP + FN + TP} \tag{4.1}$$

En conjuntos de datos desbalanceados, es decir, aquellos en que las clases no tienen cantidades equitativas de datos, la exactitud puede resultar un indicador poco fiable. Esto se puede ver en los casos en que el modelo resulta muy capaz para predecir los casos de la clase mayoritaria, donde el cálculo de la exactitud tenderá a ignorar las predicciones sobre la clase minoritaria. En el caso de los datos médicos en general se trabaja con conjuntos desbalanceados, y en este caso, tal como se mostró en la Sección 4.2.2, este conjunto de datos no es una excepción. Para evitar resultados engañosos, existen dos métricas que resultan útiles, la sensibilidad y la especificidad.

El *recall* positivo o sensibilidad es la relación entre la cantidad de datos clasificados correctamente como positivos contra la cantidad que realmente son positivos

[60]. En este caso la sensibilidad representa la capacidad del modelo de detectar un patrón epiléptico y predecirlo correctamente como positivo. Un buen desempeño en esta métrica es de importancia en este problema, dado que una clasificación incorrecta de un caso positivo generaría que se pierda un episodio epiléptico sufrido por el paciente durante el intervalo estudiado. Esta métrica está definida en la Ecuación 4.2.

$$Sensibilidad = \frac{TP}{TP + FN} \tag{4.2}$$

La especificidad, por otro lado, mide la cantidad de valores predichos correctamente como negativos frente a la cantidad total de muestras negativas [60]. En los datos evaluados se puede interpretar como la capacidad del modelo de detectar un caso negativo o sin patrón ictal considerando todos los negativos que se presentan al mismo. Su expresión de cálculo se puede consultar en la Ecuación 4.3. Generalmente se busca que un modelo sea específico cuando hay especial interés en que las clases negativas se clasifiquen correctamente. En el caso del anotado de señales de iEEG resulta importante que el modelo, además de detectar los casos con patrones ictales, tenga también la capacidad de diferenciar los casos que no poseen un patrón ictal, de forma de que el médico pueda beneficiarse del modelo, disminuyendo la cantidad de datos que debe revisar manualmente.

$$Especificidad = \frac{TN}{TN + FP} \tag{4.3}$$

Con la sensibilidad y la especificidad es posible obtener la exactitud balanceada, en inglés *Balanced Accuracy* (BACC), la cual se puede calcular como el promedio de ambas, según la Ecuación 4.4. Esta métrica permite mostrar la exactitud general del modelo dándole el mismo peso a las dos clases involucrados, y es por esto que en reemplazo de la exactitud se suele utilizar la exactitud balanceada en la evaluación con conjuntos de datos desbalanceados.

$$BACC = \frac{Sens. + Espec.}{2} = \frac{1}{2}\left(\frac{TP}{TP + FN} + \frac{TN}{TN + FP}\right) \tag{4.4}$$

Otras métrica utilizada para evaluar un modelo de clasificación es la precisión, la cual consiste en la relación entre la cantidad de datos realmente positivos y los datos clasificados como positivos [60]. Su expresión de cálculo se puede consultar en la Ecuación 4.5. En este caso la precisión va a indicar la proporción de muestras que se clasificaron correctamente como positivas, con episodio ictal, sobre el total de casos que se clasificaron como con episodio ictal, incluyendo también aquellos que no poseen un patrón ictal y fueron clasificados incorrectamente. En este proyecto interesó que el modelo resulte preciso, pero se prefirió que los casos resulten clasificados como positivos ante la duda, evitando que se pierdan episodio ictales en la clasificación negativa.

$$Precision = \frac{TP}{TP + FP} \tag{4.5}$$

También se utilizó como métrica de evaluación el F1, que es la media armónica entre la precisión y la sensibilidad [60]. Su expresión de cálculo se puede consultar en la Ecuación 4.6. El F1 logra establecer un balance entre la sensibilidad, que hace que el modelo genere una mayor cantidad de predicciones positivas, y la precisión, que tiene el mismo para solamente clasificar a los verdaderos positivos como positivos. De esta forma un F1 alto indica que gran parte de los datos que son positivos se han clasificado correctamente, pero que además no se han clasificado erróneamente una gran cantidad de falsos positivos. Esta métrica es útil en los conjuntos desbalanceados, siendo la clase positiva la minoritaria. El F1 no toma en cuenta la cantidad de verdaderos negativos que son clasificados, sino que toda la relevancia de la métrica se basa en la clasificación de los positivos [60].

$$F1 = 2 * \frac{precision * sensibilidad}{precision + sensibilidad} \tag{4.6}$$

En la Tabla 4.2 se resumen las métricas utilizadas para el desarrollo del proyecto y la significancia clínica de cada una. En el proyecto se ha dado prioridad a la exactitud balanceada (BACC) y al valor del F1, dado que se quiso establecer una buena exactitud en ambas clases, con y sin patrón ictal, y se le dio importancia a la capacidad del modelo para diferenciar correctamente una gran parte de los datos que contenían patrones ictales, ya que estas muestras son las que resultan de mayor interés para los neurofisiologos tratantes.

4.3.2. Curvas Operador-Receptor

Las salidas de los clasificadores binarios pueden interpretarse como probabilidades a posteriori condicionales. La asignación de una probabilidad a una u otra clase se determina por la selección de un umbral de decisión, el cual transforma la salida continua, dada por las probabilidades, en una categórica, es decir la clase correspondiente. El valor del umbral se puede definir en cada problema en particular, pero generalmente se toma un valor de 0.5, de manera que las probabilidades mayores a 0.5 son asignadas a una clase positiva, y las menores son asignadas a una clase negativa.

Una evaluación útil en estos casos es la comparación de la sensibilidad, o la tasa de verdaderos positivos, en inglés *True Positive Rate* (TPR), relativa a la tasa de falsos positivos, en inglés *False Positive Rate* (FPR), siendo la misma $FPR = 1 - Espec.$. La relación entre los verdaderos positivos y las detecciones de falsos positivos va a depender de la frecuencia convulsiva de cada sujeto. Un concepto importante en los modelos de clasificación es la curva Operador-Receptor, en inglés *Receiver Operator Curve* (ROC), ya que la misma muestra el desempeño de un modelo en distintos puntos del umbral de decisión que se usa para determinar si una muestra es positiva o negativa [61]. Esta curva es un gráfico de la sensibilidad relativa al FPR, y se genera determinando la relación entre ambos para distintos umbrales variables o puntos de decisión. En la Figura 4.4 se pueden observar las curvas ROC correspondientes a distintos modelos, cada uno con distintas capacidad de clasificación. Por ejemplo, un experimento puede ser hecho más sensible al mover el umbral para aumentar la tasa de resultados positivos, lo cual podría incrementar los verdaderos positivos pero

Tabla 4.2

Métricas de evaluación para la clasificación de las muestras

Métrica	Ecuación	Descripción
Sensibilidad	$Sens = \frac{TP}{FN + TP}$	Proporción de muestras que han sido correctamente identificadas como con episodio ictal
Especificidad	$Spec = \frac{TN}{FP + TN}$	Proporción de muestras que han sido correctamente identificadas como sin episodio
Precision	$Prec = \frac{TP}{TP + FP}$	Proporción de muestras clasificadas correctamente como positivas dentro de las clasificadas positivas
BACC	$BACC = \frac{1}{2}\left(\frac{TP}{TP + FN} + \frac{TN}{TN + FP}\right)$	Combinación de sensibilidad y especificidad
F1	$F1 = \frac{2TP}{2TP + FP + FN}$	Combinación de sensibilidad y precisión

también la tasa de falsos positivos [61]. El área bajo la curva es una medida efectiva de exactitud, dado que provee una medida del desempeño independientemente del umbral de decisión [61].

4.3.3. Cálculo del tiempo de inicio ictal

Como se mencionó previamente, durante el desarrollo también se buscó que el modelo generado lograra establecer el tiempo de inicio del patrón ictal en aquellos episodios etiquetados como de clase positiva o con episodio. Para poder evaluar la precisión con la que el modelo y el proceso de postprocesamiento determinan el tiempo de inicio ictal resultó necesario contar con una métrica que permitiera determinar el error en la predicción.

Para esta evaluación se eligió el error absoluto medio, en inglés *Mean Squared Error* (MAE), el cual permite determinar la diferencia que existe entre un valor estimado y el valor real de una cantidad que se quiere medir o calcular. El error absoluto medio representa el promedio de todos los errores de predicción individuales sobre todas las instancias en las que se evaluó. Cada error de predicción se calcula a partir de la diferencia entre el valor real y el valor predicho para cada instancia determinada. El mismo se puede calcular según la Ecuación 4.7, donde N es la cantidad de muestras, y_i es la cantidad real y $\widehat{y}_i$ es el valor estimado.

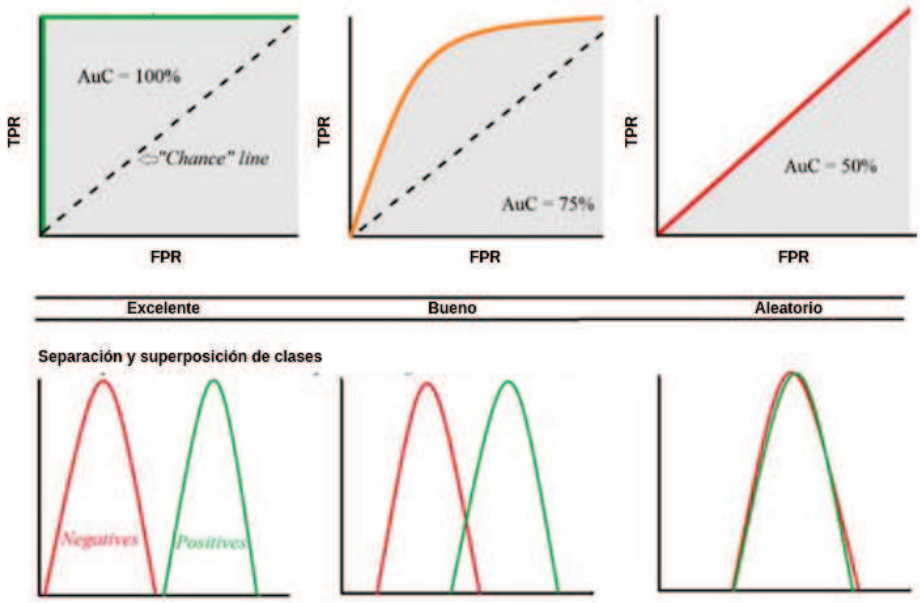


Figura 4.4: *Curvas de relación para distintos modelos. El modelo de la derecha es un modelo ideal, con la curva ROC pegándose a los valores de las esquina. El modelo del medio es un modelo bueno, en el que hay superposición de las clases pero se logra un separación alto. El tercer modelo es un modelo sin capacidad de clasificar, dado que las muestras pueden ser clasificadas errónea o correctamente con la misma probabilidad. Figura adaptada de [62]*

$$MAE = \frac{1}{N}\sum_{i=1}^{N} |(y_i - \widehat{y}_i)| \tag{4.7}$$

Estas evaluaciones se realizaron únicamente sobre los datos clasificados correctamente como positivos, de forma de evitar trasladar los errores de la clasificación a la evaluación del tiempo de onset. Además, su evaluación en datos que no pertenecieran a los verdaderos positivos no tendría sentido, dado que no estaría aportando ninguna información sobre la capacidad del modelo de establecer correctamente el punto de inicio ictal.

4.4. Resumen del capítulo

Este capítulo permitió conocer la base de datos utilizada, describiendo además las características de la misma, y la forma en que fue organizada para poder entrenar y testear el modelo propuesto. Específicamente se observó que se trata de una base desbalanceada, característica que debió ser considerada durante el diseño, elección y validación del modelo en los capítulos siguientes. Además, se presentaron las herramientas que permitieron llevar adelante el proyecto, y que debieron ser incorporadas con este fin. Por último se presentaron las métricas que fueron utilizadas durante todo el proyecto para determinar el rendimiento del modelo propuesto y poder realizar comparaciones en los distintos experimentos. Los experimentos en los que se diseñó y se optimizó el modelo elegido finalmente se presentan en el siguiente capítulo.

Capítulo 5

Diseño y optimización del modelo

En este capítulo se indica la forma en que se planificaron los experimentos y las pruebas que se llevaron a cabo para desarrollar y validar el diseño del modelo. También se indican los métodos que se utilizaron para realizar las comparaciones necesarias para evaluar el modelo final.

En la primera parte del capítulo se define el esquema experimental y se indican las decisiones que se tomaron para llegar al diseño de la red final. Durante esta parte del proyecto se definió la arquitectura neuronal a utilizar en los experimentos, donde se buscó que la misma tenga una cantidad óptima de parámetros (Sección 5.1.1). Luego evaluó el impacto del uso de diferentes etiquetas como elementos de entrada al modelo (Sección 5.1.2). Se trabajó con 4 etiquetas de diferentes características, y se evaluó el rendimiento de cada una. Adicionalmente, se definió un método de postprocesamiento cuyo propósito consistió en tomar las salidas de la red y calcular a partir de ellas la clase del segmento y el tiempo de inicio del patrón ictal, aplicando este último únicamente al caso en que la muestra fuera clasificada como positiva (Sección 5.1.3).

Una vez definida la arquitectura, la forma de las máscaras y el método de procesamiento se pasó a optimizar el entrenamiento del mismo, buscando por un lado las técnicas de aumento que podían ser aplicadas sobre los datos de entrenamiento (Sección 5.2.1), y por otro, los hiperparámetros del modelo (Sección 5.2.2). Esta última búsqueda se hizo llevando a cabo un estudio de ablación, modificando cada vez distintos hiperparámetros de la red, con el objetivo de encontrar los puntos de rendimiento óptimo. Como resultado de todo este proceso, se consiguió definir un modelo dado para probar las predicciones en escenarios de utilidad clínica simulados.

5.1. Diseño y desarrollo experimental

El modelo de aprendizaje profundo se desarrolló considerando al problema a resolver como uno de segmentación. El motivo de usar una arquitectura desarrollada para tareas de segmentación surgió a partir de la hipótesis de que resultaba posible considerar a los espectrogramas como imágenes. De esta forma se planteó que en los espectrogramas se podía detectar la variación en la energía en cada instante de la señal

evaluada como un cambio de colores e intensidades en la imagen.

Para desarrollar el modelo se plantearon algunos objetivos que el mismo debía cumplir. Entre estos objetivos se incluyó que el modelo tuviera la capacidad de predecir la clase a la que pertenece cada segmento de iEEG, es decir, determinar si los patrones de las muestras se corresponden con un patrones ictal o no, y por otro lado que pudiera determinar el tiempo de inicio del patrón ictal en aquellos casos que presentan un episodio ictal. El esquema utilizado se puede consultar en la Figura 5.1. Con este esquema se logró conseguir un sistema que incorpora las señales ya transformadas en espectrogramas y obtener la clase y el tiempo de inicio ictal.

Para esto se utilizó una estructura totalmente convolucional basada en la arquitectura UNet (la cual fue introducida en el Capítulo 3), que interpreta los espectrogramas como si fueran imágenes y genera un mapa de segmentación que indica cambios en el mapa de frecuencias, a partir del cual de puede determinar el inicio de un patrón ictal. Una vez obtenido el mapa de segmentación con dichas probabilidades, se pensó en generar un método que pudiera descifrar la clase y el tiempo de *onset* realizando operaciones matemáticas predefinidas. Esto se realizó como una etapa de postprocesamiento y no como parte de la red ya que se buscaba que la red fuera totalmente convolucional, y para obtener dos resultados específicos resultaba necesario incorporar capas totalmente conectadas. De esta manera se debieron utilizar métricas de evaluación tanto para la forma en que la red genera las etiquetas como para la precisión en las clases y tiempos calculados.

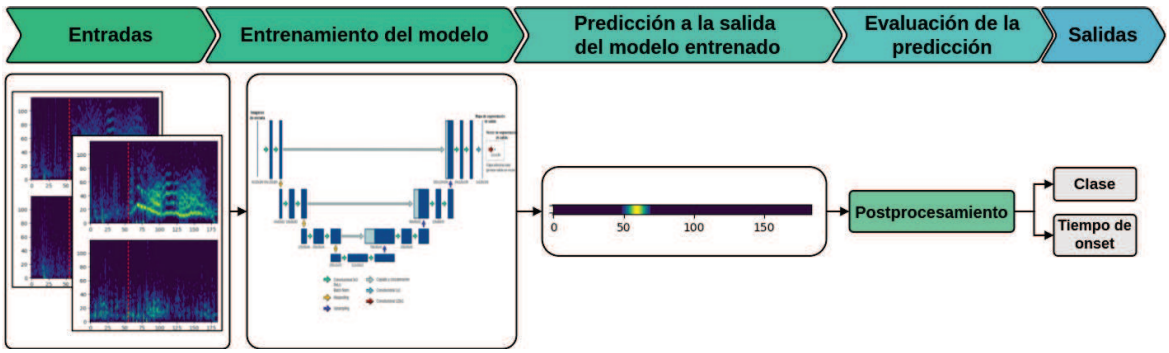


Figura 5.1: *Esquema de procesamiento utilizado en los experimentos*

5.1.1. Arquitectura utilizada

Durante el desarrollo se buscó obtener una red que tenga la menor cantidad de parámetros entrenables posible sin llegar a perder rendimiento. Sin embargo, la arquitectura original de UNet consiste en una red con una gran cantidad de parámetros, dado que la misma llega a niveles de resolución de la imagen muy altos. Es por esto que durante este desarrollo no se utilizó la arquitectura original sino que la misma fue modificada con el objetivo de lograr una arquitectura que resultara lo suficientemente compacta sin pérdida de desempeño en la detección de etiquetas.

En esta búsqueda se planteó como requerimiento inicial que el modelo no tenga grandes cantidades de parámetros, principalmente para evitar caer en el sobreajuste

del modelo a los datos de entrenamiento, y por otro lado, para evitar generar un modelo que resulte demasiado pesado para entrenar. La cantidad de parámetros en un modelo afecta generalmente a la posibilidad de sobreajuste, dado que, a mayor cantidad de parámetros, mayor es la cantidad de combinaciones posibles, por lo que es más probable que se encuentre alguna combinación de parámetros en el espacio disponible que ajuste perfectamente a los datos de entrenamiento, pero que no logre generalizar sobre distintos datos. Sin embargo, el modelo debía también resultar lo suficientemente robusto para no perder capacidad de aprendizaje de los datos.

Todas las arquitecturas evaluadas fueron entrenadas utilizando tanto las funciones de pérdida de Entropía cruzada binaria y la pérdida Dice (ambas descriptas en la Sección 3.3.2). Se utilizaron estas funciones de pérdida dado que se encuentran entre las más populares en los problemas de segmentación de imágenes, siendo Dice específicamente muy utilizada en aplicaciones con imágenes médicas. Además, se optimizó el entrenamiento usando el algoritmo Adam (presentado en la Sección 3.2.2), siendo el mismo también uno de los métodos más utilizados hoy en día para entrenar redes profundas.

Entre otras diferencias que se tomaron con respecto a la arquitectura original se incluye la forma de realizar la operación de *upsampling*. En la UNet se realiza una operación de *upsampling* de los datos que duplica el tamaño de los mapas y luego se realiza una convolución con un filtro de tamaño $2x2$ que disminuye a la mitad la cantidad de mapas. En la red utilizada en el proyecto se realiza una operación de *upsampling* que aumenta el tamaño del mapa pero que no modifica la cantidad de mapas de características. Luego, el resultado de esta operación se concatena con una copia del mapa resultante en el nivel correspondiente de la parte contractiva de la red, y los datos resultantes de esta operación se ingresan a una capa convolucional, la cual genera la disminución de los mapas de características. Por otro lado, en la red original las capas convolucionales no utilizan *padding*, lo que genera que los tamaños de los mapas disminuyan en cada convolución. En este proyecto, para simplificar los tamaños manejados por la red, se utilizan capas convolucionales con un filtro de tamaño $3x3$ y *padding* de valor 1, para mantener el tamaño de los mapas en sus longitudes temporal y frecuencial luego de cada convolución.

5.1.2. Datos y etiquetas

La base de datos utilizada para entrenar y validar el modelo estaba compuesta por los espectrogramas correspondientes a cada segmento de iEEG y el *ground truth* que contenía la etiqueta de clase y tiempo de inicio ictal para el caso de clases positivas. Al pensar a este problema como un problema de segmentación de imágenes resultaba necesario contar con máscaras que pudieran ser utilizadas como etiquetas durante el entrenamiento. Sin embargo, en la base de datos estas máscaras no se encontraban generadas, por lo que debieron ser construidas y seleccionadas en la etapa de diseño del modelo.

Para el manejo de los datos y las máscaras de segmentación se contó con una función que permitió una carga y manejo ordenado de los datos, de forma que la red pudiera acceder a los mismos de manera correcta durante su entrenamiento. Esta

función además permitía el armado de las máscaras *on the fly*, es decir, en simultáneo con la carga de los datos, lo que permitió que las mismas no ocupen memoria y que sean fácilmente modificables. A continuación se desarrollan los análisis realizados durante la exploración del formato de los datos y las máscaras en la fase de diseño del modelo.

5.1.2.1. Datos

Los datos a la entrada consistieron en los espectrogramas de cada muestra de tamaño $4x120x181$, incluyendo en una misma muestra las 4 señales que surgen de las señales recogidas por cada electrodo en simultáneo, como se describió en la Sección 4.2 y en la Figura 4.1. Para poder ser procesadas por la red, las muestras debieron ser modificadas en su tamaño ya que el tamaño inicial generaba incongruencias durante el procesamiento, por lo que no podían ser analizadas. Estas incongruencias se generaron por la forma en que se realizan los cálculos sobre las imágenes en la red UNet, a causa de las operaciones de recorte, copia y concatenación que se dan en la conexión entre la parte de codificación y la de decodificación.

Si se analiza la arquitectura de la red UNet utilizada es posible ver que las uniones entre caminos se dan 3 veces. Así es como, para poder ser analizadas por la red, las imágenes debían cumplir con un tamaño que fuera divisible por 2^3. Esto resultaba necesario ya que en cada capa de *max pooling* el tamaño de las entradas se divide a la mitad, tanto en ancho como en alto, y luego al hacer el *upsampling* el tamaño original se recupera. Si el resultado de la división no es un número entero entonces se achica el tamaño al valor más pequeño y luego, cuando se intenta reconstruir el tamaño original, el mismo queda incompleto.

La forma de resolver este problema consistió en tomar la última columna de cada espectrograma y copiarla en 3 columnas adicionales, llegando a un tamaño total de $4x120x184$. Este tamaño permitió que tanto la dimensión de frecuencia como la de tiempo fueran divisibles por 2^3 según lo requerido. Se decidió modificar el tamaño de los espectros con este método ya que este se consideró que generaba modificaciones menores sobre las imágenes. Además, como lo que se quería detectar en los espectros era el cambio de energía en el tiempo, era importante que el método de ampliación no generara cambios que pudieran resultar interpretados erróneamente por la red.

5.1.2.2. Máscaras

Durante el desarrollo del modelo se encontró que el modelo podía aprender y predecir diversos mapas de segmentación. Dado que este problema no contaba con máscaras de segmentación previamente generadas, sino que los únicos datos de anotaciones eran la clase de cada dato y el tiempo de inicio ictal, un paso importante en el desarrollo fue determinar la forma preferente para las máscaras de segmentación. Para esto se plantearon distintas opciones, considerando un total de 4 formas, con 2 variantes según dimensiones, y 2 variantes según la función representada en la máscara.

- Según sus dimensiones:

- Imagen: estos mapas de segmentación se definieron con las mismas dimensiones que uno de los 4 espectros de la entrada. De esta forma se respetó el tamaño de salida de la UNet original, siendo la salida de tamaño $120x184$.
- Vector: estos mapas permitieron que las salidas resulten más acotadas. Se respetó la dimensión temporal pero se redujo la dimensión frecuencial, obteniendo un mapa de tamaño $184x1$. Para este tipo de máscaras (de forma vectorial) se debió modificar el modelo de UNet original para que la salida pueda tener una forma vectorial, agregando una capa convolucional adicional con un filtro de tamaño rectangular de $120x1$.

- Según la función representada:
  - Función escalón: en este tipo de máscaras se colocan columnas de píxeles de valor igual a 1 desde el *sample* en el que se inicia el patrón ictal en el espectrograma hasta el final del mismo. De esta forma se marca el punto de inicio del episodio convulsivo separando en dos el mapa de segmentación. Conceptualmente, en el caso de la función escalón, se intenta determinar un momento fijo con el que se pueda separar al espectro en un intervalo sin patrón convulsivo, es decir un período pre-ictal o interictal, de un período ictal. Sin embargo, esta máscara solo puede ser aplicada a intervalos cortos de tiempo y es desconocido el dato de la duración de los patrones ictales, por lo que no se puede determinar un tiempo de fin.
  - Función Gaussiana: este mapa de segmentación está representado con un valor de 1 en la posición de mayor probabilidad del punto de inicio convulsivo, teniendo en cuenta el tiempo anotado previamente. En los píxeles vecinos al tiempo de inicio ictal la probabilidad (y así también la intensidad del píxel) disminuye. Conceptualmente, en el caso de la función Gaussiana, lo que se intenta es que el modelo pueda determinar la probabilidad de que se haya experimentado un cambio en los patrones del espectrograma. Si se experimenta un cambio de energía de la señal entonces la máscara la representa con una función Gaussiana, que establece la probabilidad de que se haya iniciado un período ictal o convulsivo en el paciente. De esta forma la función Gaussiana permite establecer la probabilidad de un inicio convulsivo en un determinado momento, observando el cambio en la amplitud de las frecuencias activas en las señales de electroencefalografía del paciente.

Considerando estas variaciones se forman 4 tipos de máscaras distintas, como las que se observan en la Figura 5.2. El primer tipo de máscara corresponde a la de imagen con escalón, correspondiente con la máscara a) de la Figura 5.2. Este tipo de máscaras consiste en una matriz del mismo tamaño que los espectros analizados por muestra con un solo canal de salida, y, al ser una función escalón, las máscaras se completan con ceros y se colocan columnas con unos desde el tiempo correspondiente al inicio ictal y en todas las columnas que le siguen a la misma.

El segundo tipo de máscara es la imagen con función Gaussiana, correspondiente con la máscara b) de la Figura 5.2. Lo que se buscó en este caso fue asignar a un grupo de columnas de ancho predefinido una distribución de probabilidad del tipo Gaussiana con valores entre 0 y 1, de forma que el valor máximo se encuentre en el

sample correspondiente al punto de inicio ictal.

Luego se encuentra la máscara tipo vector con función escalón, correspondiente con la máscara c) de la Figura 5.2. Esta máscara consiste en vectores del largo de la señal temporal que contienen unos a partir del momento del inicio ictal epiléptico marcado manualmente. Esta forma lo que permite es disminuir el número de píxeles totales a comparar entre la máscara real y la predicha, lo cual puede mejorar el rendimiento del modelo. Por último, se analizó la máscara tipo vector con función Gaussiana, correspondiente con la máscara d) de la Figura 5.2. Esta opción buscó combinar las anteriores, utilizando una máscara tipo vector pero siguiendo una distribución Gaussiana en la intensidad de los píxeles.

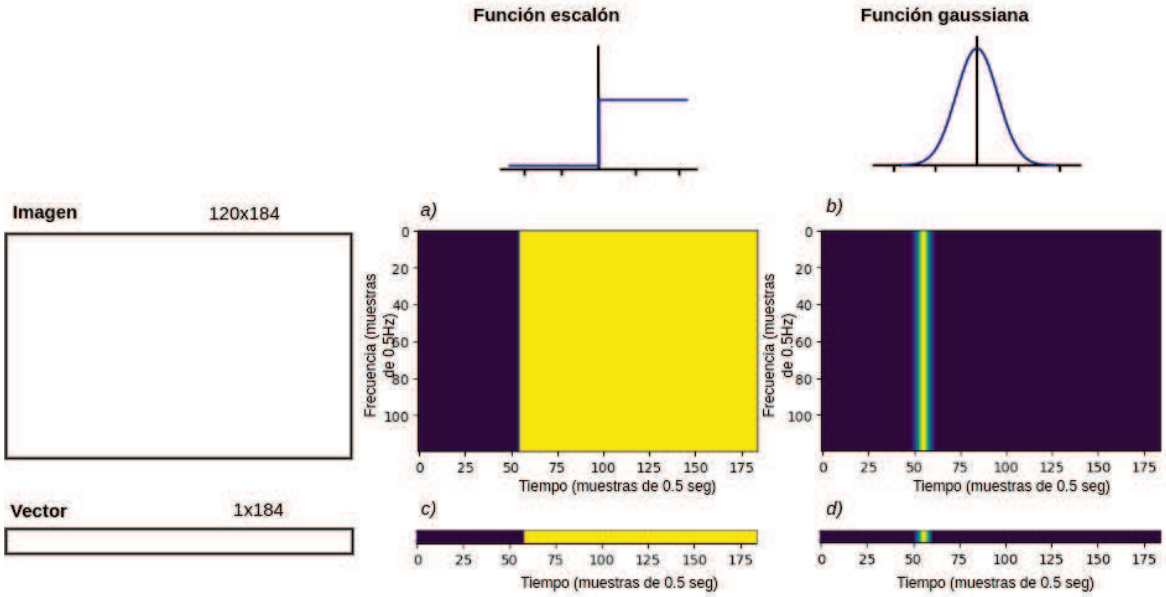


Figura 5.2: *Ilustración de las diferentes configuraciones de máscaras evaluadas. a) Máscara tipo imagen con función escalón. b) Máscara tipo imagen con función Gaussiana. c) Máscara tipo vector con función escalón. d) Máscara tipo vector con función Gaussiana.*

5.1.3. Post-procesamiento de las máscaras predichas

Durante el postprocesamiento de los datos de salida de la red se buscó un método sencillo que pudiera detectar el instante de tiempo con probabilidad máxima de ocurrencia, la cual debía superar un valor determinado, para establecerlo como tiempo de inicio del patrón ictal. Se buscó que esta etapa de procesamiento de los datos permita no alterar significativamente las etiquetas resultantes de la red, por lo que se debió evaluar la forma en que este proceso se mantuviera simple y agregara el menor ruido posible a los valores de desempeño del modelo. En el postprocesamiento se debió evitar que las métricas de evaluación resultaran afectadas por el cálculo realizado en él, de manera que se pudiera distinguir efectivamente el desempeño del modelo evaluado.

Los pasos para el postprocesamiento fueron esencialmente 2: umbralado y detección de picos. El primer paso, de umbralizado, consistió en aplicar un filtrado por umbral de las máscaras de salida de la red. Este umbral se debió optimizar experimentalmente,

en base a los resultados obtenidos en el conjunto de validación. Esta evaluación se presenta más adelante, en la Sección 5.1.4.3, pero en principio se tomó un valor de 0.2.

Con las máscaras ya umbralizadas, se pasó a la realizar la detección de picos. En esta parte se buscó si la máscara predicha tenía un valor mayor a cero y, en caso de existir, se calculó la posición del máximo valor en la misma. Si se encontraba un máximo mayor a cero entonces se consideraba que la muestra pertenecía a la clase positiva o con episodio y si no se encontraba dicho valor se clasificaba como negativa o sin episodio.

Los cálculos para encontrar la posición del máximo se definieron de distinta forma para cada tipo de máscara. Durante el postprocesamiento de las máscaras con función escalón lo que se buscó fue el valor máximo en la matriz binarizada, luego de haber sido filtrada la misma con un umbral predeterminado. Si el máximo encontrado resultaba mayor a cero, entonces se definía la posición de la columna correspondiente como el tiempo de inicio del período ictal. Se debió binarizar las máscaras luego de umbralizarlas dado que el máximo no siempre coincidía con el punto de inicio del *onset*, siendo que los bordes en general resultaban de valores menores que en el centro del área segmentada. Por otro lado, la forma de encontrar el inicio ictal durante el postprocesamiento de máscaras con función gaussiana se realizó buscando el valor máximo de la distribución Gaussiana, es decir, buscando el punto de mayor intensidad en la matriz. Luego ese punto permitía definir la columna correspondiente al inicio ictal, determinando así el tiempo buscado.

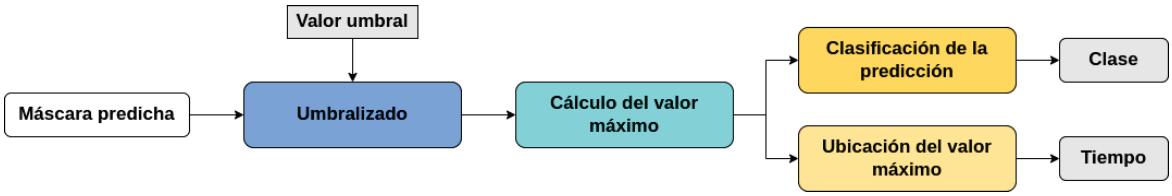


Figura 5.3: *Esquema del postprocesamiento aplicado al label*

5.1.4. Evaluaciones realizadas durante el diseño

Las distintas decisiones que se debieron tomar durante el diseño se basaron en resultados experimentales. Para cada uno de los componentes del sistema de procesamiento propuesto en la Sección 5.1 se debieron evaluar distintas opciones. A continuación se indican los criterios que se tomaron para definir cada opción, y hacia el final de la sección se describe el funcionamiento del sistema resultante.

5.1.4.1. Estudio de ablación sobre la cantidad de parámetros del modelo

Los modelos evaluados se definieron a partir de distintas variaciones que se aplicaron sobre la arquitectura original de UNet. Para simplificar las comparaciones todos los experimentos se realizaron utilizando la máscara tipo vector con función Gaussiana. En un primer intento se utilizó la cantidad de niveles de resolución de la arquitectura

original, lo que dio lugar a una arquitectura con 31 millones de parámetros aproximadamente, generando así que esta sea la red con mayor cantidad de parámetros dentro de las redes evaluadas. En una segunda prueba, se quitó un nivel de resolución y se pasó así a una red de 7.7 millones de parámetros. En esta red se contaba con un nivel de resolución menos, pero las modificaciones no afectaron la cantidad de filtros presentes en cada capa convolucional. Las siguientes dos arquitecturas evaluadas, en cambio, modificaban adicionalmente la cantidad de filtros en cada nivel de resolución, generando que luego de cada capa convolucional disminuyera la cantidad de mapas de características que se obtenían. Esto permitió obtener dos redes adicionales, una que usaba la mitad de los filtros de la arquitectura anterior, con un total de 1.9 millones de parámetros, y otra que usaba un cuarto de los filtros, llegando a contar con un total de apenas 488.970 parámetros.

En la Figura 5.4 es posible observar el desempeño de cada modelo. El modelo con mayor cantidad de niveles (31 M) tiene un muy buen rendimiento en la fase de entrenamiento, con una exactitud balanceada media del 99 %, un F1 de 0.98 y un MAE de 0.36 segundos. Sin embargo en los datos de validación el desempeño de este modelo pasa a ser de una exactitud balanceada del 83 % y un F1 de 0.80 con un MAE de 4.81 segundos. Esto indica un posible sobreajuste, dado que el modelo se ajusta muy bien a los datos de entrenamiento, pero no logra reproducir este desempeño para los datos de validación.

El modelo con menor cantidad de parámetros (0.5 M), por otro lado, tiene un menor valor de BACC en entrenamiento y validación, con 90 % y 78 % respectivamente, y el valor del F1 empeora en ambos conjuntos de datos, lo cual indica que el modelo pierde mucha sensibilidad y precisión en el análisis de los mismos. Al analizar el valor del MAE se puede observar que es el modelo con el mayor error en el cálculo del tiempo de inicio en datos de entrenamiento, pero resulta mejor en validación. Esto se podría explicar, a partir de lo evaluado empíricamente por la disminución en la exactitud balanceada del modelo, a partir de la cual se espera que los datos que pasan por el cálculo del tiempo de inicio sean datos más fáciles de clasificar y analizar para el modelo.

El caso con mejores resultados en validación, considerando tanto los valores de BACC como de F1, resulta ser el de 7.7 millones de parámetros (7.7 M), es decir, aquel modelo que cuenta con un nivel de resolución menos que la arquitectura de UNet original, pero que mantiene la cantidad de filtros en cada capa. Con respecto a los puntajes de BACC, este modelo tiene un valor de 96 % en entrenamiento y 82 % en validación. Además, logra un valor de MAE de 0.88 segundos en entrenamiento, y uno de 5.21 segundos en validación. Este valor resulta el más alto del conjunto de validación, aunque no se aleja demasiado del modelo con 1.9 millones de parámetros, que tiene un MAE de 4.98 segundos pero con una distribución más amplia y un F1 menor.

Se puede ver que el modelo con mayor cantidad de parámetros (de 31 millones) tiende a sobreajustar a los datos de entrenamiento, según la diferencia en los valores de F1 y BACC en entrenamiento y validación, perdiendo la capacidad de generalizar en datos diferentes a los conocidos. También es posible analizar que a medida que disminuyen los parámetros de la red, el desempeño de los modelos en entrenamiento

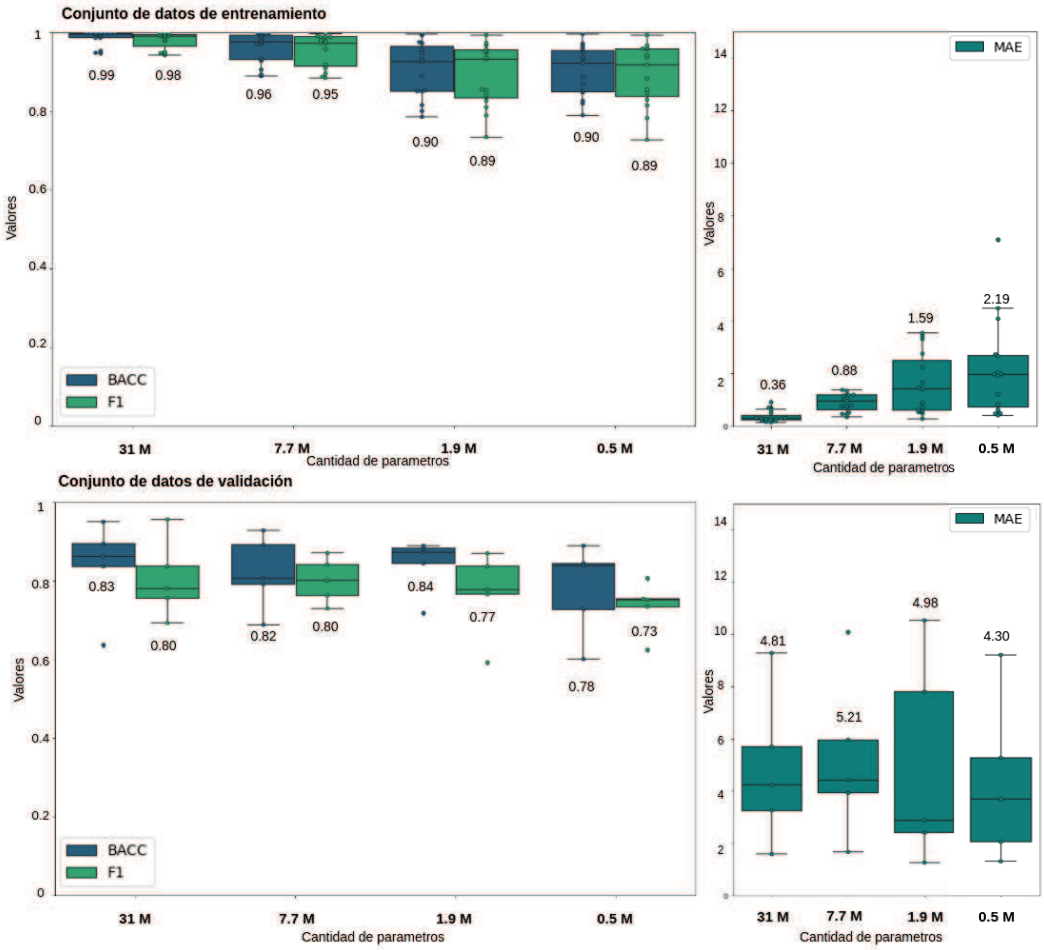


Figura 5.4: *Estudio de ablación sobre la cantidad de parámetros de la red. Cada punto en el gráfico corresponde a la evaluación sobre un paciente. En la primera fila se pueden consultar los resultados de F1, BACC y MAE para los datos de entrenamiento, tanto para los modelos de 31 millones (31 M), de 7.7 millones (7.7 M), de 1.9 millones (1.9 M), y de 0.5 millones (0.5 M) de parámetros. En la segunda fila se pueden consultar los mismos resultados para los datos de validación.*

disminuye a la par. En un principio el desempeño en validación mejora, ya que mejora la capacidad de generalización del modelo, pero luego comienzan a disminuir al igual que en entrenamiento, ya que el modelo comienza a perder capacidad de aprendizaje. A partir de este análisis se eligió la red con 7.7 millones de parámetros aproximadamente como arquitectura final.

5.1.4.2. Selección del tipo de máscara de salida

Como segunda fase de los experimentos llevados a cabo se procedió entonces a evaluar el rendimiento del modelo con los distintos tipos de máscaras propuestas. En esta fase de los experimentos se evaluó el rendimiento utilizando los valores de las pérdidas alcanzadas durante entrenamiento, tanto en los conjuntos de entrenamiento como de validación, y también considerando la posibilidad de obtener correctamente la clase y el tiempo de inicio ictal de cada muestra realizando un postprocesamiento sobre las

máscaras predichas por el modelo. Todos los experimentos se realizaron sin aumento de datos y con condiciones similares de entrenamiento, para garantizar que los resultados sean comparables y las diferencias se generen por el cambio en la representación de las máscaras y no por otros factores.

Se utilizaron como funciones de pérdida la Entropía cruzada binaria (BCE) y la pérdida Dice para generar los modelos con las distintas máscaras (ambas descriptas en la Sección 3.3.2) resultando en un total de 8 modelos finales. Inicialmente, en el análisis conceptual, se intuía que la pérdida Dice funcionaría mejor en las máscaras de tamaño imagen, ya que esta pérdida prioriza obtener buenas predicciones en la clase positiva.

Imagen con función escalón

En este caso, al tratarse de máscaras que se componían de dos zonas bien determinadas y separadas, se esperó que la mejor elección resulté con la función de pérdida Dice, ya que la misma evalúa los puntos en el mapa de segmentación considerándolo como un conjunto, y no individualmente como lo hace BCE. Al entrenar los modelos con este tipo de máscaras se llegó a un modelo con pérdida Dice de 0.05 en entrenamiento y 0.478 en validación. Se puede observar que hay una gran diferencia en ambos conjuntos de datos, siendo la misma del 850 % del valor de pérdida en entrenamiento. Al entrenar con pérdida BCE se logró una pérdida de 0.124 en entrenamiento y de 0.157 en validación, siendo en este caso las pérdidas más parejas entre los conjuntos de datos (con una diferencia del 26 % de la pérdida de entrenamiento).

Durante el entrenamiento se encontró que efectivamente la pérdida Dice permitió generar modelos con menor error de cálculo del tiempo de inicio ictal y mayor capacidad de generalización. En la Figura 5.5 se pueden observar las predicciones de las máscaras para datos de validación en modelos con los dos tipos de pérdida. Se puede observar que la pérdida Dice permite predecir máscaras con valores más similares a la máscara usada como etiqueta, mientras que BCE no logra reconstruir la función escalón de forma tan marcada.

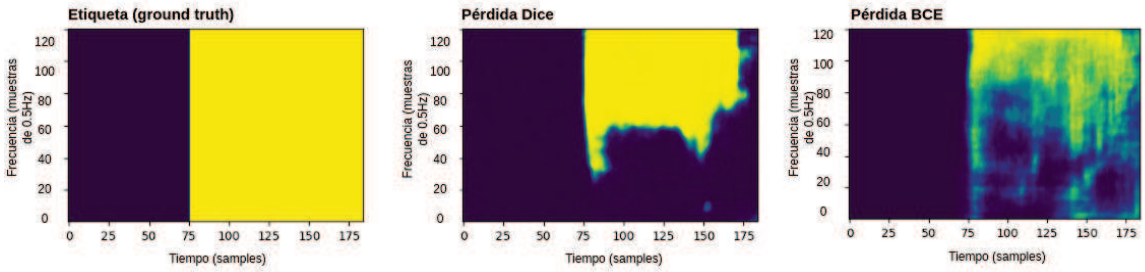


Figura 5.5: *Entrenamiento de modelos con la máscara tipo imagen escalón con distintas funciones de pérdida. A la izquierda de la figura se puede ver la máscara utilizada como etiqueta, en el centro, la máscara predicha con el modelo entrenado con pérdida Dice, y a la derecha, la máscara predicha con el modelo entrenado con pérdida BCE.*

Imagen con función Gaussiana

El entrenamiento con este tipo de máscaras buscó que se generara un modelo que pudiera segmentar el punto donde se encuentra un cambio de energía en el espectrograma, pudiendo determinar luego el tiempo de la señal con mayor probabilidad de presentar un patrón de inicio ictal. Al entrenar los modelos con este tipo de máscaras se llegó a un modelo con pérdida Dice con un valor de 0.217 en entrenamiento y 0.622 en validación, con una diferencia del 186 % sobre la pérdida de entrenamiento entre ambos conjuntos. Al entrenar con pérdida BCE se logró una pérdida de 0.033 en entrenamiento y de 0.031 en validación, siendo los valores similares (con una diferencia del 4.63 % sobre la pérdida de entrenamiento) y menores que en el caso de la imagen con función escalón. Esta disminución con respecto al caso anterior se puede deber a que la proporción entre píxeles de valor 0 con respecto a los positivos es mucho mayor que en el caso del escalón, donde según la máscara las proporciones entre ambos grupos de píxeles puede resultar más balanceada.

En la Figura 5.6 se pueden ver las predicciones de los modelos entrenados con distintos tipos de funciones de pérdida. Se puede observar que la pérdida Dice intentó asignar un valor de 1 al conjunto de puntos segmentados con probabilidad mayor a cero. Esto surge por una propiedad conocida de la función de pérdida Dice, la cual es aprovechada en los problemas de segmentación binaria, que consiste en que esta pérdida produce bordes de segmentación nítidos, lo que dificulta las predicciones de valores no binarios [63]. Por otro lado, la pérdida BCE generó una predicción más parecida a la etiqueta generando máscaras que respetaron la distribución de probabilidad planteada.

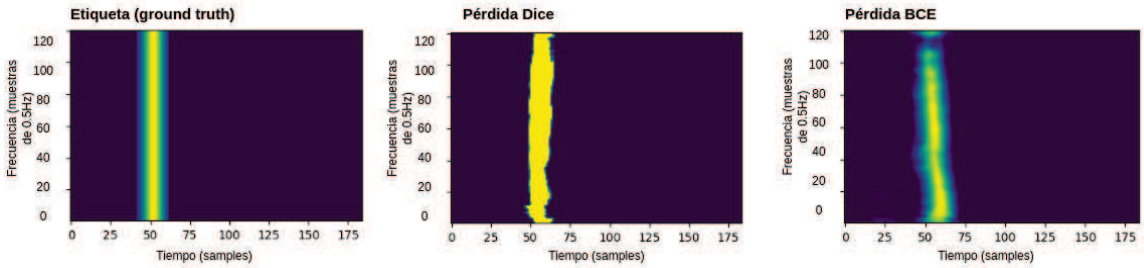


Figura 5.6: *Entrenamiento de modelos con la máscara tipo imagen Gaussiana con distintas funciones de pérdida. A la izquierda de la figura se puede ver la máscara utilizada como etiqueta, en el centro, la máscara predicha con el modelo entrenado con pérdida Dice, y a la derecha, la máscara predicha con el modelo entrenado con pérdida BCE.*

Vector con función escalón

En las máscaras del tipo vector se esperó que en las predicciones se generen dos zonas bien diferenciadas, logrando reconstruir el escalón de la etiqueta. Sin embargo, con los dos tipos de pérdida utilizadas para entrenar se generaron máscaras con distintos segmentos positivos separados por intervalos de ceros, como se puede observar en la Figura 5.7. Tal como en el caso de la máscara de imagen con función escalón, se esperó que la mejor elección resulte con la función de pérdida Dice pero, en este caso el entrenamiento con ambos tipos de pérdidas generó predicciones similares.

En el caso del modelo entrenado con pérdida Dice se logró una pérdida de 0.055 en entrenamiento y de 0.431 en validación. Nuevamente la pérdida en validación con Dice resultó bastante mayor que en el conjunto de datos de entrenamiento, con un diferencia del 690 % sobre el valor de pérdida de entrenamiento. Por otro lado, con la pérdida BCE se llegó a un modelo con un valor de pérdida de 0.102 en entrenamiento y de 0.179 en validación (dando una diferencia del 74 % sobre el valor de pérdida de entrenamiento).

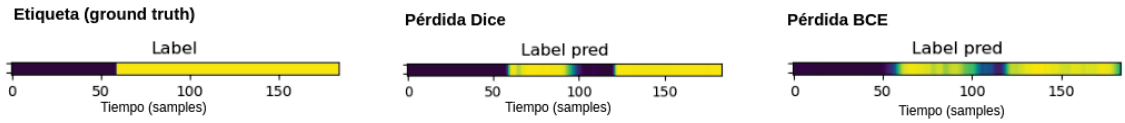


Figura 5.7: *Entrenamiento de modelos con la máscara tipo vector escalón con distintas funciones de pérdida. A la izquierda de la figura se puede ver la máscara utilizada como etiqueta, en el centro, la máscara predicha con el modelo entrenado con pérdida Dice, y a la derecha, la máscara predicha con el modelo entrenado con pérdida BCE.*

Vector con función Gaussiana

En el modelo entrenado con pérdida Dice se obtuvieron máscaras que no diferenciaban las distintas probabilidades de la Gaussiana propuesta, sino que generaba predicciones con conjuntos de píxeles con valor de uno, tal como se puede observar en la Figura 5.8. Sin embargo, la pérdida BCE logró generar un modelo que predice las máscaras con las probabilidades correspondientes con la etiqueta del *ground truth*. Se puede observar cualitativamente que este modelo fue el que permitió generar las salidas con mayor similaridad a las etiquetas propuestas.

En este caso se obtuvo un modelo con pérdida Dice de 0.23 en entrenamiento y 0.58 en validación (con una diferencia del 154 % sobre el valor de entrenamiento). En el modelo entrenado con BCE se llegó a una pérdida de 0.04 en entrenamiento y de 0.025 en validación, siendo menor la pérdida en validación que en entrenamiento en un 36 % sobre el valor de entrenamiento.

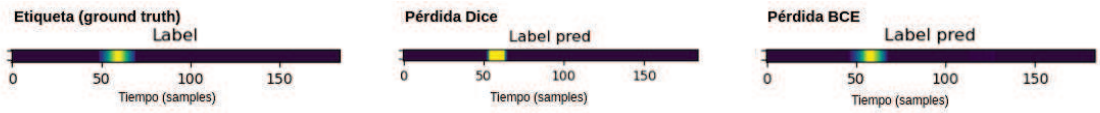


Figura 5.8: *Entrenamiento de modelos con la máscara tipo vector Gaussiano con distintas funciones de pérdida. A la izquierda de la Figura se puede ver la máscara utilizada como etiqueta, en el centro, la máscara predicha con el modelo entrenado con pérdida Dice, y a la derecha, la máscara predicha con el modelo entrenado con pérdida BCE.*

Comparación de desempeño para las distintas máscaras de salida

Tal como se observó en las predicciones de los distintos tipos de modelos, el modelo que mejor reconstruyó las máscaras propuestas fue el de forma vectorial con función

Gaussiana entrenado con pérdida BCE. En la Figura 5.9 se puede observar el desempeño de cada modelo evaluado con respecto a la clasificación del segmento en las clases correspondientes según la presencia de un episodio ictal, lo cual se evaluó con el valor de BACC y el de F1, y con respecto a la estimación del tiempo de inicio ictal, el cual fue evaluado con el valor de MAE. Esta evaluación se realizó a partir de los resultados de postprocesamiento, siguiendo los cálculos especificados en la Sección 5.1.3.

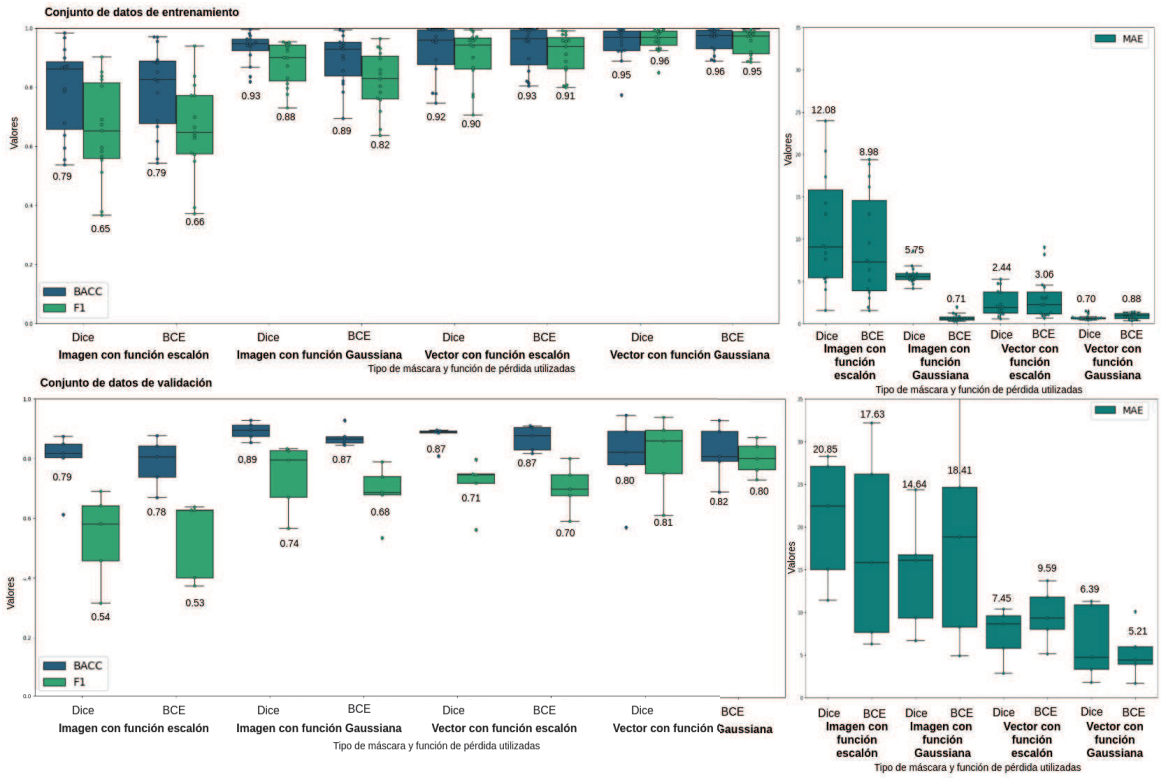


Figura 5.9: *Evaluación comparativa entre modelos generados por distintas máscaras y distintas funciones de pérdida. Cada punto representa el desempeño sobre un paciente. En la primera fila se pueden observar los valores de BACC, F1 y MAE sobre los datos de entrenamiento. En la segunda fila se pueden observar los mismos valores evaluados sobre el conjunto de datos de validación.*

Observando estos valores efectivamente se puede ver que el modelo que mejor desempeño logró fue el de forma vectorial con función Gaussiana entrenado con pérdida BCE. Este modelo logró un desempeño en validación con un valor de BACC del 82 %, F1 de 0.80 y MAE de 5.21 segundos. Esto indica que el modelo aparenta una buena capacidad de generalización y llevó a decidir trabajar con este tipo de máscaras. Todos los pasos de optimización y validación que continuaron luego de estos experimentos se realizaron bajo estas máscaras y función de pérdida.

5.1.4.3. Optimización del valor de umbralizado

En la etapa de optimización, uno de los parámetros claves que debió ser optimizado fue el valor del umbral que se aplicó a los vectores de salida. El mismo buscaba evitar que los valores debajo de cierto valor afectarán el cálculo del máximo de la máscara, a

partir del cual se determinaba el tiempo de inicio ictal. En muchos casos el hecho de aplicar este umbralizado permitió diferenciar de forma más clara las muestras pertenecientes a clases negativas o sin episodio, ya que permitía llevar cualquier valor positivo de las máscaras predichas a cero, de forma que no hubiera ningún valor mayor a cero en todo el vector, y de esa forma el tiempo de inicio predicho sea cero, indicando que no era detectable un patrón ictal en el segmento.

El valor lógico para este umbral debía ser de 0.5, dado que es el valor medio entre el valor mínimo posible en el vector siendo el mismo de 0, y el valor máximo que se esperaba encontrar, es decir 1, y que es el umbral generalmente utilizado en los algoritmos de clasificación para decidir una clase a partir de una determinada probabilidad. Sin embargo, debido a la distribución desbalanceada de los datos y a posibles problemas de calibración en el modelo, era posible que este valor umbral estuviera desplazado. Es por esto que se experimentó con distintos valores umbrales para analizar como este valor afectaba a las métricas de evaluación. En la Figura 5.10 se pueden observar los resultados de BACC, F1 y MAE para cada valor de umbral.

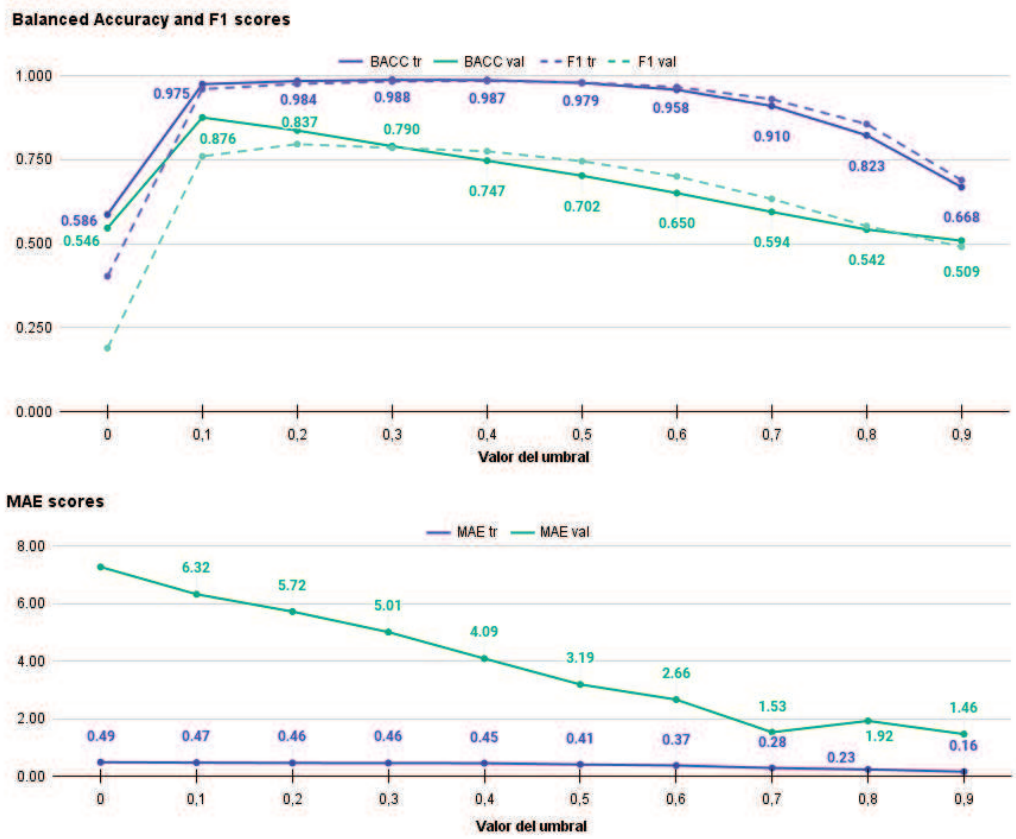


Figura 5.10: *Evaluación del modelo ante distintos umbrales de postprocesamiento de la salida del modelo.*

Al aplicar el postprocesamiento sobre la salida con umbrales altos se observó que los valores de BACC y F1 disminuían en los dos conjuntos de datos evaluados. Lo que sucede en estos casos es que el umbral resulta demasiado alto, incluso para los casos de verdaderos positivos, por lo que se termina clasificando a los mismos como negativos erróneamente. En estos casos la especificidad aumenta hasta llegar a valores cercanos a 1, dado que la mayoría de las muestras se van a clasificar como negativas, y muy

pocas van a superar el umbral y se van a clasificar como muestras con episodio. Con respecto a los valores de error absoluto medio es posible ver que el mismo tiende a disminuir. Esto se da ya que generalmente las muestras que superen estos umbrales altos son las que tienen una probabilidad muy marcada de tener un onset en el sitio indicado, por lo que es más probable que la predicción sea correcta y más precisa. Al haber menor cantidad de muestras clasificadas como positivas y al ser las mismas las que son más fáciles de predecir, el valor del error en el cálculo del onset disminuye notablemente.

Por otro lado, también es posible observar qué sucede en el caso sin umbralizado (es decir con umbral cero). Aquí, gran parte de las muestras que son de clase negativa son clasificadas incorrectamente, ya que en el caso de que algún punto de la máscara tenga un valor mayor a cero, aunque el mismo tenga una intensidad débil, será interpretado como un punto válido para clasificar a la muestra como positiva. Esto genera un aumento de la sensibilidad, pudiendo detectar la gran mayoría de los casos positivos, pero también una disminución de la especificidad a valores muy bajos, ya que el modelo no puede detectar aquellos casos que son de clase negativa. Estos valores confirman que el umbralizado efectivamente es necesario para poder luego llevar a cabo el procesamiento de los datos según la forma propuesta.

Considerando lo expuesto anteriormente, se decidió evaluar el desempeño en valores intermedios. Se decidió utilizar el valor de umbral 0.2, ya que es en este punto que se logra un valor de BACC y F1 balanceados en el conjunto de validación. Se puede ver que yendo a valores umbrales de 0.1 o de 0.3 el F1 disminuye en validación, con un F1 de 0.761 con umbral de 0.1, 0.796 con 0.2 y 0.785 con 0.3, empeorando el rendimiento del postprocesamiento. Otro valor a analizar es el del MAE, el cual empeora en 0.1 y mejora yendo hacia un umbral de 0.3, obteniendo en 0.2 un punto de desempeño medio. Considerando estos valores de MAE, en este caso, se dio prioridad en la elección del umbral al valor de BACC.

5.1.5. Definición final de la arquitectura

La arquitectura completa de la red resultante se puede observar en la Figura 5.11, y la cantidad de parámetros en cada nivel de la red se puede consultar en el Anexo A. En primer lugar, se logró disminuir en gran medida el número de parámetros en la red quitando un nivel de resolución en la arquitectura, según lo evaluado en la Sección 5.1.4.1. Con esta modificación la cantidad de parámetros pasó de 31 millones de parámetros aproximadamente a menos de 8 millones, es decir, pasó a ser un 25 % del tamaño original de la red.

Para las máscaras de salida de la red que funcionaron como etiquetas durante el entrenamiento se utilizaron las máscaras de forma vectorial con función Gaussiana, según lo evaluado en la Sección 5.1.4.2. Esto requirió el agregado de una capa adicional al final de la red, la cual se puede observar dentro de un recuadro en la Figura 5.11. Esta capa adicional surge como un requisito para poder entrenar el modelo con máscaras de forma vectorial, ya que se necesita de una transformación a la salida de la red para que las salidas resulten de tamaño vectorial y puedan ser comparadas con las etiquetas. Esta capa, según lo descripto en la Sección 5.1.2.1, es del tipo convolucional con un

filtro de tamaño $1x120$, lo que permite reducir el eje frecuencial pero mantiene intacto el eje temporal.

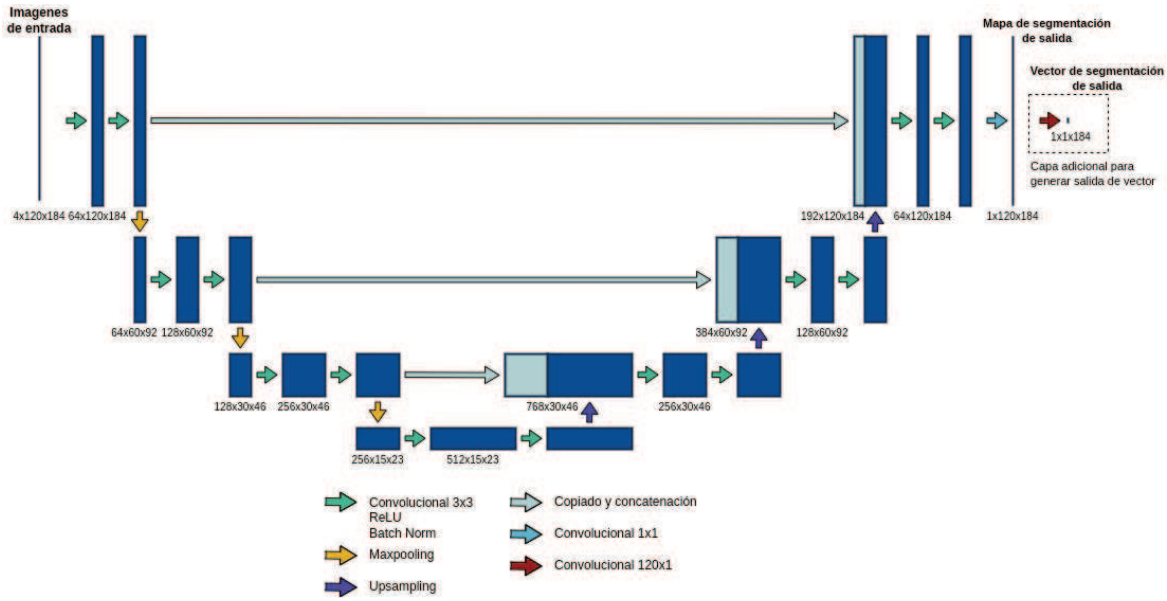


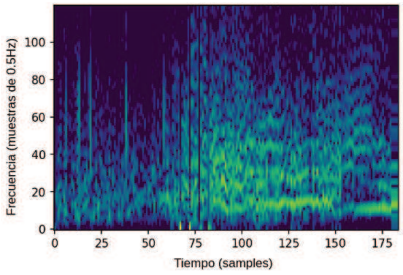
Figura 5.11: *Arquitectura basada en la UNet utilizada en los experimentos. Debajo de cada conjunto de mapas se encuentra el detalle del tamaño del mismo, en el orden (número de canales x Longitud en frecuencia x Longitud temporal). Las flechas indican una operación realizada, representando cada color un tipo de operación distinta. Las flechas verdes indican una convolución con filtro (3x3) y activación ReLU, las amarillas indican una operación de maxpooling, las violetas una operación de upsampling, las celestes claras un copiado y concatenación, las celestes una capa convolucional de filtro* $1x1$ *y la roja indica una capa convolucional de tamaño de filtro* $1x120$.

5.2. Optimización de la red diseñada

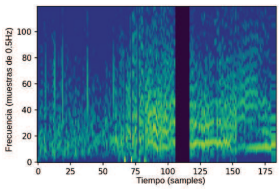
Una vez diseñada la red, junto con las máscaras utilizadas y el método de postprocesamiento se buscó optimizar distintos hiperparámetros y técnicas utilizadas durante el entrenamiento del modelo que permitieran mejorar los resultados. De esta forma se buscó mejorar la capacidad del modelo para reconstruir las máscaras a la salida, y obtener mejores resultados en el postprocesamiento de las mismas.

5.2.1. Aumento de datos de entrenamiento

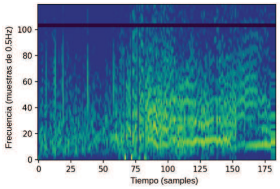
El rendimiento de las redes neuronales es sensible a la cantidad de datos que la misma puede ver durante el entrenamiento. Es por esto que, como se mencionó en la Sección 3.2.5.4, la técnicas de aumento de datos generalmente permiten mejorar el rendimiento de las redes y evitar el sobreajuste. Estas técnicas consisten en aumentar la cantidad de datos de entrenamiento generando nuevas muestras a partir de modificaciones en las existentes.



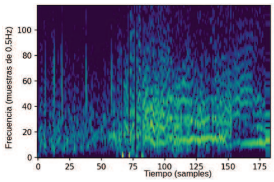
(a) *Imagen original*



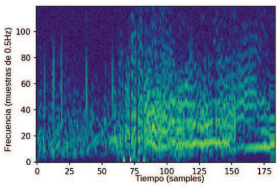
(b) *Enmascaramiento en tiempo*



(c) *Enmascaramiento en frecuencia*



(d) *Escalado de intensidades*



(e) *Suma de ruido gaussiano*

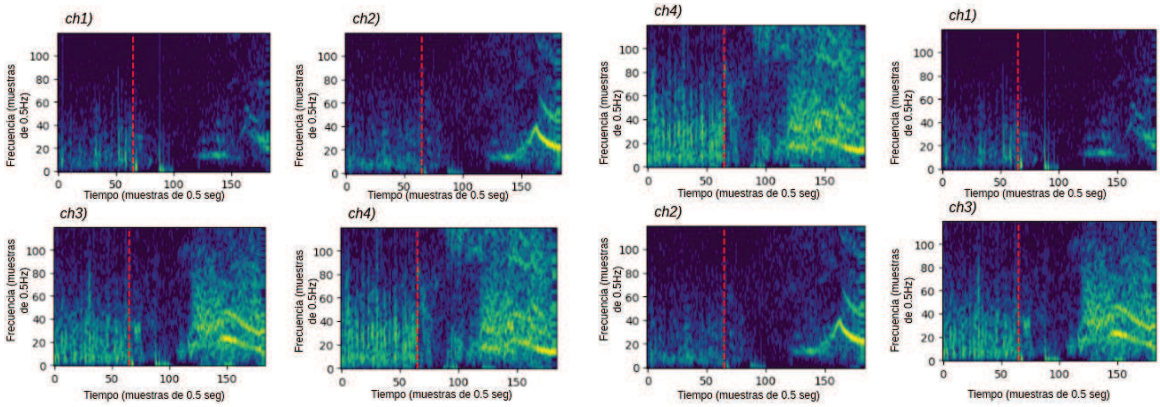
Figura 5.12: *Técnicas de aumento aplicadas a imágenes de entrenamiento.*

Existen diversas transformaciones que se pueden aplicar sobre los datos en el caso del entrenamiento con imágenes. Algunas técnicas clásicas de aumento en imágenes incluyen la rotación en distintos ángulos (pudiendo ser por ejemplo 90°, 180° y 270°), la generación de imágenes en espejo (tanto horizontal como vertical), el escalado, el recorte, la suma de ruido Gaussiano, entre otras. Sin embargo, en este proyecto, aunque se está tratando a los datos de forma similar a como se manipulan las imágenes, es importante destacar que los mismos no son imágenes estándar sino que se trata de espectrogramas, los cuales tienen una forma y estructura particular. Es por esto que las técnicas clásicas de aumento no aplican en general para el problema tratado.

En el desarrollo del proyecto se utilizaron algunas técnicas de aumento de datos compatibles con los espectrogramas. Las técnicas que se analizaron son:

- Enmascaramiento en frecuencia: esta técnica aplica una máscara a una cantidad determinada de *bins* frecuenciales consecutivos, generando un bloqueo de manera uniforme de todo el ancho de banda involucrado. De esta forma al aplicar una máscara en frecuencia hay una banda frecuencial que en el entrenamiento no aparece en el espectrograma evaluado. Clínicamente esta máscara puede bloquear anchos de banda que se activan durante un patrón ictal en un sujeto, pero que podrían no activarse en otro, generando que el modelo se acostumbre a espectrogramas de mayor variedad. El efecto de esta técnica en la imagen original se puede observar en la Figura 5.12c

- Enmascaramiento en tiempo: en este caso se aplica una máscara a un paso de tiempo determinado, generando un bloqueo uniforme en todo el ancho temporal. Clínicamente esto podría significar el momento en que el neuroestimulador deja de medir y comienza a estimular, dado que al ser un único cable por electrodo el dispositivo solo puede realizar una tarea a la vez. Por esta razón se intuyó que esta máscara podría generar problemas en el entrenamiento del modelo, pero se probó igualmente su desempeño. El efecto de esta técnica en la imagen original



(a) *Datos en orden original* (b) *Datos permutados*

Figura 5.13: *Técnicas de permutación del orden de los espectrogramas por muestra*

se puede observar en la Figura 5.12b

- Escalado: esta aumento aplica un escalamiento aleatorio de la intensidad de los espectrogramas. Esto podría permitir generar variaciones en los datos que resulten compatibles con la descripción clínica, dado que la intensidad de los espectros varía según el sujeto y el contexto de la medición. El efecto de esta técnica en la imagen original se puede observar en la Figura 5.12d

- Ruido Gaussiano: esta es una técnica que ha sido utilizada en procesamiento de audio, a partir de la cual se puede hacer que las entradas resulten más suaves y fáciles de aprender. El efecto de esta técnica en la imagen original se puede observar en la Figura 5.12e

Además, para todos los procesos de aumento se aplicó una permutación de los datos, de forma de poder mezclar el orden de las 4 imágenes que se muestran por cada *sample*, evitando que el que la red aprenda un ordenamiento específico de los canales y facilitando la utilización en datos con diferentes configuración de posicionamiento de electrodos. El efecto de este proceso de permutación se puede observar en la Figura 5.13.

5.2.1.1. Evaluación de las técnicas de aumento

El rendimiento de los modelos generados con cada técnica por separado se puede consultar en la Figura 5.14. En principio se puede observar que la aumento de datos durante el entrenamiento del modelo genera mejoras en el rendimiento del mismo con respecto al caso sin aumento. Esto se da en el conjunto de datos de validación pero en el conjunto de entrenamiento no se observan grandes diferencias. De esta manera podemos afirmar que la aumento de datos, incluso sin aplicar transformaciones significativas sobre los datos, solo permutando el orden de las imágenes por *sample*, produce mejoras sobre la capacidad de generalización de la red.

Se encuentra que en el conjunto de entrenamiento no se observan variaciones significativas en los valores de BACC y F1 entre los distintos experimentos. Sin embargo, en

el conjunto de validación sí existen diferencias observables sobre las métricas de evaluación. Principalmente se puede observar que los métodos de escalamiento y frecuencia son los que mejor rendimiento tienen en validación.

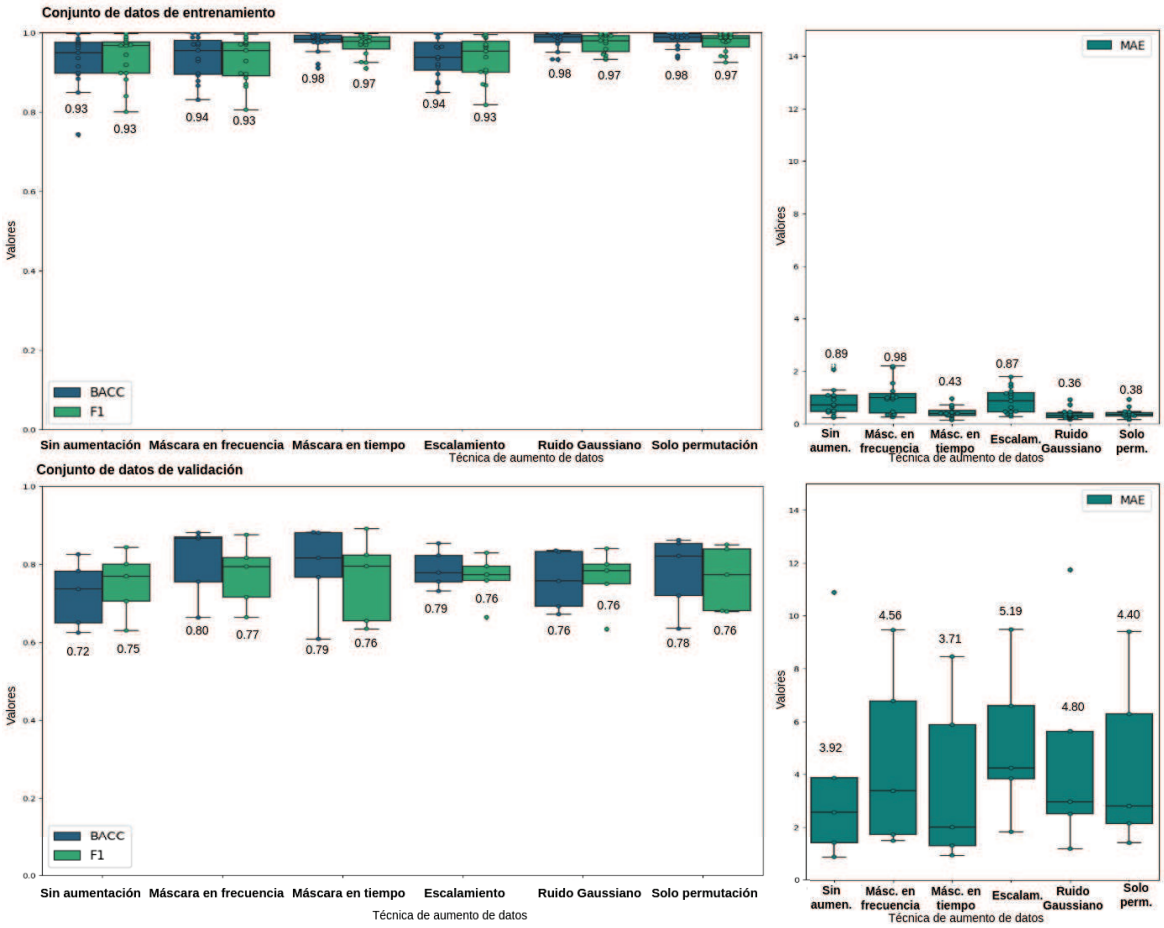


Figura 5.14: *Evaluación del modelo con distintas técnicas de aumento de datos. Cada punto representa el desempeño sobre un paciente. En la primera fila se pueden observar los valores de BACC, F1 y MAE sobre los datos de entrenamiento. En la segunda fila se pueden observar los mismos valores evaluados sobre el conjunto de datos de validación.*

Por otro lado, es posible ver que el método de aplicación de ruido Gaussiano, siendo un método clásico de aumento en imágenes estándar, no produce mejoras sobre el rendimiento del modelo, resultando incluso peor que el caso en el que se aplica únicamente permutación de los datos de cada electrodo. Esto se puede ver por ejemplo en los valores de BACC y F1 alcanzado con esta técnica en validación comparando contra los valores obtenidos en el caso de solo permutación, siendo 0.76 y 0.76 para el ruido gaussiano con permutación, y de 0.78 y 0.76 para la permutación sola. Adicionalmente el valor alcanzado de MAE empeora con el agregado del ruido gaussiano, yendo de 4.40 segundos con solo permutación a 4.80 segundos. Esto indica que no cualquier técnica de aumento de imágenes resulta aplicable para el análisis con espectrogramas, sino que se requieren técnicas de aumento más específicas para este tipo de datos. Finalmente se puede observar que en el conjunto de validación, la técnica con mejores métricas de clasificación es el enmascaramiento en frecuencia, resultando con los mejores valores de BACC, seguido por la técnica de enmascaramiento en tiempo, la cual incluso logra

la obtención de un valor de MAE más pequeño.

A partir de estos resultados se realizaron distintos experimentos combinando las técnicas de aumento, duplicando la cantidad de datos disponibles. A partir de los resultados individuales se evaluó el uso de aumento de datos combinando todas las técnicas presentadas, junto con la permutación de los datos. Los resultados para este modelo se pueden observar en la Tabla 5.1. Se puede observar que el rendimiento de esta combinación es comparable con el de los modelos por separado en BACC y F1. Es decir la combinación logra mantener la capacidad de clasificar las señales en positivas o negativas. A partir de los análisis realizados y considerando la baja cantidad de muestras disponibles para el aprendizaje de los parámetros, se decidió aplicar un aumento en dos instancias, triplicando la cantidad de datos disponibles durante el entrenamiento. Así, sobre una primera instancia se aplicó un enmascaramiento en tiempo, y en un segunda un enmascaramiento en frecuencia, combinando ambas con el escalamiento de las intensidades y la permutación de los canales.

Tabla 5.1

Resultados del modelo con combinación de técnicas de aumento

Métrica	Datos de entrenamiento		Datos de validación	
	Sin aum.	Con aum.	Sin aum.	Con aum.
BACC	0.92 ± 0.07	0.97 ± 0.02	0.72 ± 0.09	0.78 ± 0.09
F1	0.93 ± 0.06	0.97 ± 0.02	0.75 ± 0.08	0.76 ± 0.08
MAE	0.89 ± 0.59	0.48 ± 0.20	3.91 ± 4.05	5.24 ± 2.09

5.2.2. Optimización de hiperparámetros

Cuando se trabaja con modelos paramétricos de aprendizaje automático, se pueden diferenciar dos tipos de parámetros: los parámetros del modelo, que son los diferentes componentes de la red que irán variando su valor durante la fase de entrenamiento y que se aprenden a partir de los datos de entrada, y los parámetros de configuración o hiperparámetros, que son los que permiten configurar el modelo o el proceso de entrenamiento en sí mismo [38]. En este proyecto, el modelo propuesto contó con distintos hiperparámetros que se fueron modificando con el consecuente impacto en su rendimiento. Estos hiperparámetros debieron ser fijados antes del entrenamiento, y de su elección dependió el resultado obtenido.

Los hiperparámetros ideales para cada modelo dependen de los datos que sean analizados. La forma de obtenerlos en este caso comenzó por definir los hiperparámetros de mayor interés para el desarrollo de la red. Luego se buscaron algunos valores dentro de los cuales se podían encontrar estos hiperparámetros, es decir se buscó el espacio en el que se podían encontrar. Por último se trabajó experimentando con diferentes valores y evaluando los resultados. Para todos los experimentos se utilizó el mismo modelo de base en el cual se modificaba únicamente el hiperparámetro a evaluar, de manera de poder identificar su efecto sobre el desempeño final.

Los hiperparametros explorados fueron el *learning rate*, el *weight decay*, la varianza de la Gaussiana, y el *positive weight*. La evaluación y elección realizada sobre cada uno

se presenta a continuación.

5.2.2.1. Valor del *learning rate*

El *learning rate* consiste en el tamaño del paso de los pesos durante el entrenamiento. El valor se seleccionó para permitir que el modelo pudiera entrenar. Este valor fue buscado experimentalmente, con el fin de obtener una curva de entrenamiento que permitiera alcanzar un mínimo. Se encontró que con un valor de 5e-6 este objetivo se lograba.

5.2.2.2. Valor de *weight decay*

El *weight decay* corresponde al tamaño del paso considerado para penalizar los pesos durante el entrenamiento. Es una técnica popular de regularización, que fue presentada en la sección 3.2.5.1.

Para este hiperparámetro se evaluaron distintos valores entre 0.1 y 1e-6. Se puede observar en la Figura 5.15a que a medida que aumenta el valor de *weight decay*, acercando a 0.1, la exactitud del modelo en datos de validación aumenta, al mismo tiempo que aumenta F1. También es posible observar que a medida que aumenta el valor de MAE disminuye. Es por esto que se eligió usar el valor de 0.1 como valor final para el *weight decay*.

5.2.2.3. Varianza de la función Gaussiana

En las máscaras con función Gaussiana el valor de la varianza determina el ancho de la campana. Por lo tanto se exploraron distintos valores para este hiperparámetro. El valor de la varianza determina la cantidad de *samples* que son incluidos dentro de la distribución de probabilidad, por lo que a mayor ancho, mayor es la cantidad de *samples* que obtienen un valor distinto a cero dentro de la máscara utilizada. En principio se evaluó que aumentar el valor de este hiperparámetro podría mejorar la capacidad del modelo para diferenciar aquellos máscaras correspondientes a la clase positiva de las de clase negativa, pero se supuso el riesgo de que un mayor ancho podría disminuir la precisión en el cálculo del tiempo de inicio del patrón ictal. Es por esto que se analizaron distintos anchos de Gaussiana, buscando encontrar uno que balanceara ambas capacidades.

En la Figura 5.15b, se puede ver el desempeño del modelo con los distintos valores de ancho de gaussiana evaluados. Efectivamente, de acuerdo a los supuestos iniciales, se puede observar que a medida que se aumenta el ancho de la gaussiana mejoran los valores de BACC y F1. Este fenómeno se da especialmente en el conjunto de validación, es decir, el aumento del ancho de la gaussiana mejora la generalización del modelo para la tarea de detección. Por otro lado, también es posible ver que el aumento de este ancho no genera diferencias en el desempeño con respecto al MAE al calcular el valor de tiempo de inicio ictal. Incluso, en el conjunto de validación se puede observar que el valor de MAE disminuye al aumentar valor de varianza, incluso al alcanzar mayores

valores de BACC. Es por esto que se seleccionó finalmente un valor de ancho de 10 *samples*, es decir, una varianza de 5.

5.2.2.4. Valor de *positive weight*

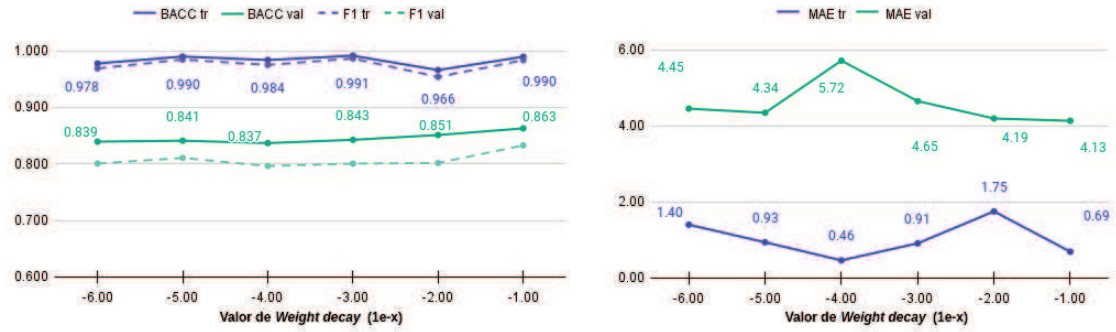
En la sección 3.3.2 se presentó la función de entropía cruzada binaria pesada. El peso que se puede aplicar en esta función de pérdida depende de las características de desbalance de las máscaras utilizadas, por lo que este peso se fue variando para poder determinar la mejor función de pérdida. Es posible usar la función BCE sin aplicar ningún tipo de peso a alguna de las clases, pero también es posible usar la función WBCE, la cual aplica un peso determinado a la clase positiva, generando que los píxeles correspondientes a dicha clase contribuyan en mayor medida a la pérdida. Esto permite que durante el entrenamiento se penalicen los errores asociados a los píxeles de clase positiva, generando una mayor importancia en la correcta clasificación de la misma. En máscaras desbalanceadas como la trabajada en este caso, esto permite obtener una pérdida calculada de manera más equilibrada.

En la Figura 5.15c es posible observar el desempeño del modelo utilizando como función de pérdida la función BCE con distintos valores de pesos para la clase positiva. Es posible observar que la aplicación inicial de *positive weight* mejora el BACC en el conjunto de validación, de un valor de 0.837 sin peso positivo y de 0.886 con la aplicación de un peso positivo de 2. Al mismo tiempo disminuye el valor del MAE, yendo de un error de 5.72seg a 5.20seg. Sin embargo, luego de pasar el valor inicial de 2, a medida que aumenta el *positive weight* disminuye la exactitud y aumenta el MAE tanto en el conjunto de datos de entrenamiento como en validación.

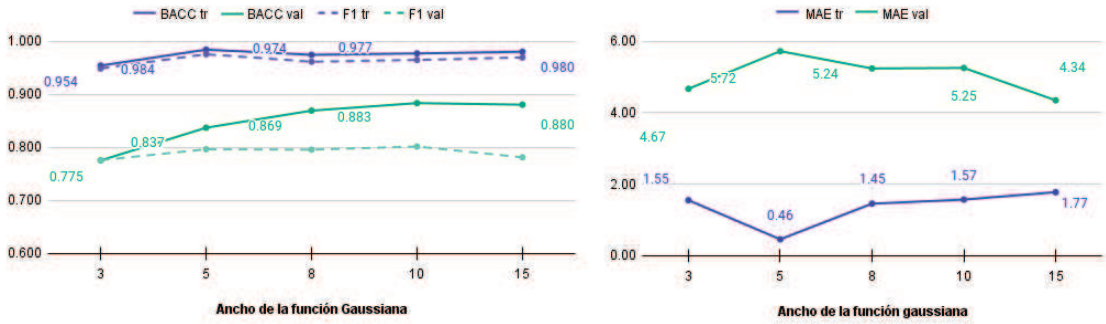
Esto se puede deber a que el algoritmo comienza a centrar la importancia de la clasificación correcta únicamente sobre la clase positiva, despreciando la correcta clasificación de los puntos de valor igual a cero en las máscaras. Esto genera que todos las máscaras predichas comiencen a tener valores positivos, independientemente de la presencia de un episodio ictal. Así, en el postprocesamiento aumenta la sensibilidad, dado que se pueden detectar todos los valores mayores a cero, pero disminuye la especificidad, dado que no se pueden detectar los episodios realmente negativos. Además, el aumento del MAE se puede deber a que los puntos más intensos del *label* predicho comienzan a generarse sobre cualquier ubicación, de forma que el error del cálculo del tiempo de inicio empeora. Considerando lo expuesto, se optó por un valor de *positive weight* de 0.2, dado que es el punto que mejor balance de BACC, F1 y MAE consigue.

5.3. Desempeño del modelo final

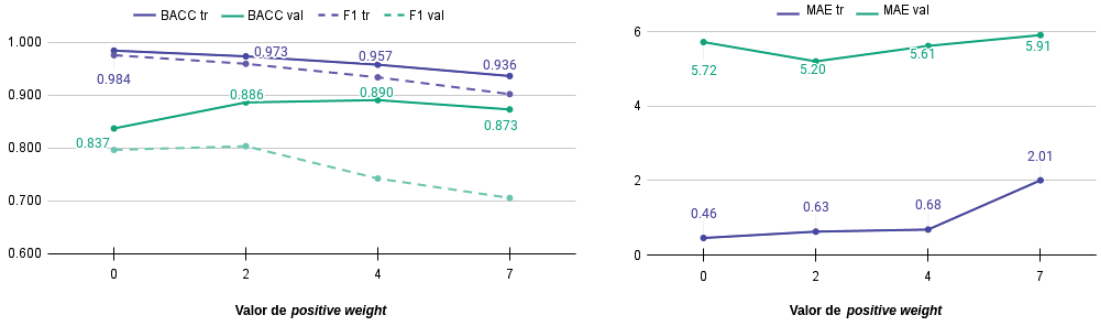
Una vez que se definieron todas las condiciones óptimas para desarrollar el modelo final, el mismo fue entrenado y testeado en los datos reservados para este fin. Las condiciones finales con las que se entrenó este modelo se pueden consultar en la Tabla 5.2. Se entrenó con una máscara de forma vectorial con una función Gaussiana centrada en el tiempo de inicio ictal. La función de pérdida utilizada fue la entropía cruzada binaria pesada, con un peso de 2 para los píxeles positivos. Se utilizaron técnicas de



(a) *Evaluación del modelo con distintos valores de weight decay (escala logarítmica)*



(b) *Evaluación del modelo con gaussianas de distinto ancho*



(c) *Evaluación del modelo con distintos valores de positive weight*

Figura 5.15: *Evaluación del modelo con distintos valores para cada hiperparámetro*

aumento con máscaras en tiempo y frecuencia de los espectrogramas, permutación del orden de los datos de cada electrodo, y escalamiento de las intensidades. Para poder mejorar la capacidad de generalización del modelo se utilizaron dos instancias de aumento, es decir, se triplicó la cantidad de datos disponibles durante el entrenamiento. Así en un conjunto se aplicó escalamiento y máscara de frecuencia, y en otro se aplicó escalamiento y máscara de tiempo, con permutación en ambos.

Por otro lado, se uso un tamaño de *batch* de 128, permitiendo que en cada actualización de pesos se contara con una cantidad amplia y variada de muestras. Esto se logró debido al uso de una GPU con memoria RAM suficiente para procesar esta gran cantidad de datos en simultáneo. Además, se usó un valor de *weight decay* de 0.1 para regularizar el modelo y un *learning rate* de 5e-6. Por último, se usó un valor umbral de 0.2 para el postprocesamiento de las máscaras predichas, permitiendo que se pudieran diferenciar mejor las máscaras correspondientes a casos positivos.

Tabla 5.2

Condiciones experimentales para el entrenamiento del modelo final

Hiperparametro / Condición	Valor
Máscara	Vector con función gaussiana
Tamaño de la gaussiana	n = 10
Función de pérdida	Entropía cruzada binaria pesada, con peso 2
Valor de umbralizado	0.2
Técnicas de aumento	Máscara en frecuencia y máscara en tiempo, con escalamiento y permutación en ambos casos
Batch size	128
Weight decay	1e-1
Learning rate	5e-6

5.3.1. Resultados del entrenamiento y la validación

El modelo entrenó durante 12 épocas, dado que la mejor pérdida de validación resultó ser la correspondiente a esta época, no pudiendo alcanzar en las épocas subsiguientes un valor menor de pérdida (considerando que el algoritmo estaba configurado con un *early stopping* de paciencia de 20 épocas). Las curvas de entrenamiento se pueden observar en la Figura 5.16. En azul se puede ver la pérdida de entrenamiento, la cual comienza desde un valor cercano a 0.25 y disminuye hasta llegar a 0.05 y también es posible observar la curva de pérdida en validación. Se puede observar que la curva de validación se encuentra siempre por debajo de la pérdida de entrenamiento. Esto puede deberse a que los datos de validación tienen una distribución diferente de casos a los datos de entrenamiento. Se entiende entonces que esta menor pérdida en validación puede derivarse a causa de las variaciones en los datos, la posible relación entre clases y las regularizaciones que se aplicaron al modelo en distintas instancias.

Sin embargo, se puede observar que hasta la época 12 la curva de validación tiende a disminuir, mejorando la pérdida en validación, y luego de esta época se comienza a

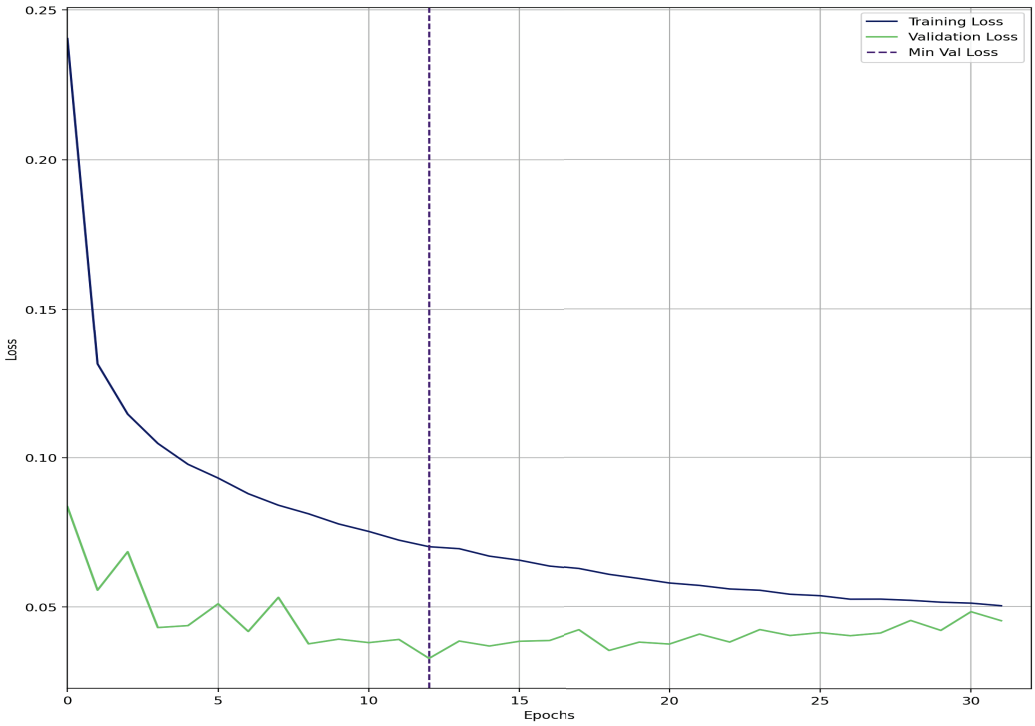


Figura 5.16: *Curvas de pérdida en datos de entrenamiento y de validación durante el entrenamiento del modelo*

notar una tendencia marcada de la curva en aumento. Esto significa que a partir de la época 12 el entrenamiento comienza a sobreajustar, perdiendo el rendimiento en los datos de validación.

Al evaluar el modelo generado con las métricas de postprocesamiento se encontró que el modelo permitía una clasificación de las muestras con un valor de BACC del 95 % en entrenamiento y un 89 % en validación. Con respecto al cálculo del tiempo de inicio ictal, se encontró que en entrenamiento se logró un MAE de 0.70 segundos y en validación de 4.78 segundos. Los valores de desempeño del modelo en el postprocesamiento en ambos conjuntos de datos se puede consultar en la Tabla 5.3.

Tabla 5.3

Desempeño en entrenamiento y validación alcanzado por el modelo final

Métrica	Datos de entrenamiento	Datos de validación
BACC	0.949 ± 0.057	0.891 ± 0.078
Sensibilidad	0.989 ± 0.011	0.852 ± 0.170
Especificidad	0.910 ± 0.118	0.929 ± 0.057
Precision	0.927 ± 0.057	0.783 ± 0.096
F1	0.931 ± 0.063	0.813 ± 0.084
MAE (seg)	0.700 ± 0.329	4.779 ± 3.096

En la Figura 5.17 se pueden observar las curvas ROC construidas para los distintos conjuntos de datos. En estas curvas se puede observar que el área bajo la curva resultó según lo esperado para ambos conjuntos, acercándose ambas a la esquina con sensibilidad 1 y FPR de 0, manteniéndose alejadas de la linea central que hubiera obtenido

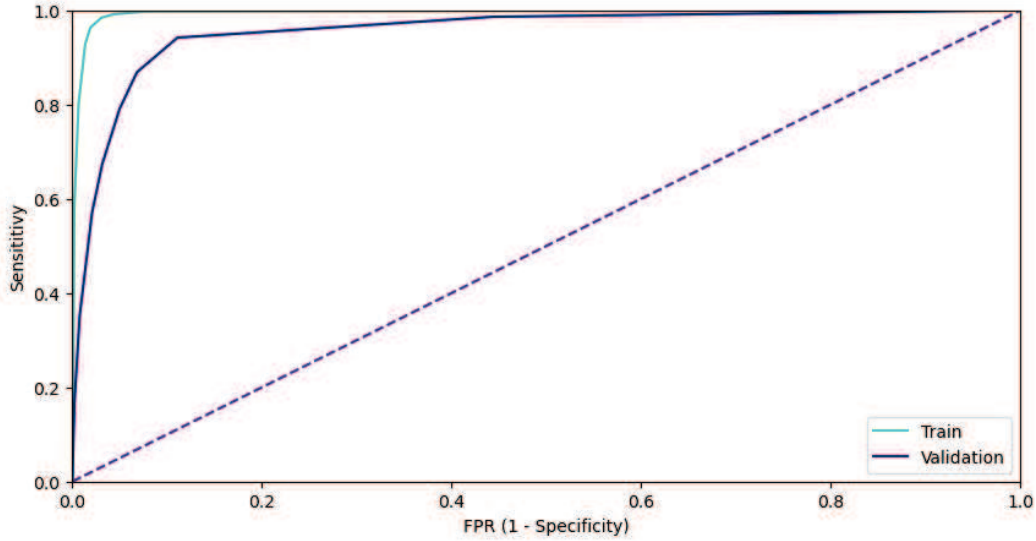


Figura 5.17: *Curvas ROC para los distintos datasets*

un clasificador aleatorio.

Por último, en la Figura 5.18 se pueden observar las matrices de confusión para la clasificación de muestras y los histogramas de errores de cálculo de tiempo de inicio para ambos conjuntos de datos evaluados. Estos histogramas presentan las distribuciones alcanzadas durante el cálculo del tiempo de inicio ictal, representando en cada punto la diferencia o delta entre el tiempo real y el tiempo predicho durante el postprocesamiento. Con respecto a las matrices de confusión, se puede observar que para el conjunto de entrenamiento se obtuvo un valor de TP de 7222 sobre un total de muestras positivas de 7264, y un valor de TN de 15838 sobre un total de 16784 muestras negativas. En el conjunto de validación la cantidad de TP fue de 655 sobre un total de 722 y la cantidad de TN fue de 5680 sobre un total de 6198 muestras negativas.

5.4. Resumen del capítulo

En este capítulo se describieron los experimentos realizados para diseñar el sistema de procesamiento propuesto, incluyendo el diseño de la arquitectura, las máscaras utilizadas como etiquetas para entrenar la misma, y el método de postprocesamiento que permitió obtener los datos de interés a partir de la etiquetas. Luego, se pasó a describir los experimentos que permitieron definir la elección de las técnicas de aumento y los hiperparametros del modelo óptimo, y una vez definidos los mismos se describió el desempeño del modelo resultante. En el siguiente capítulo, se tomará el modelo elegido y se realizará la evaluación sobre los datos de testeo. Además se presentarán los resultados de la validación cruzada llevada a cabo para conocer el desempeño del modelo electo sobre un escenario de simulación clínica, y se evaluará su desempeño sobre los datos de duración original. Por último, se realizará una comparación del modelo UNet con modelos generados mediante técnicas de ML clásico y contra modelos relevados en la bibliografía.

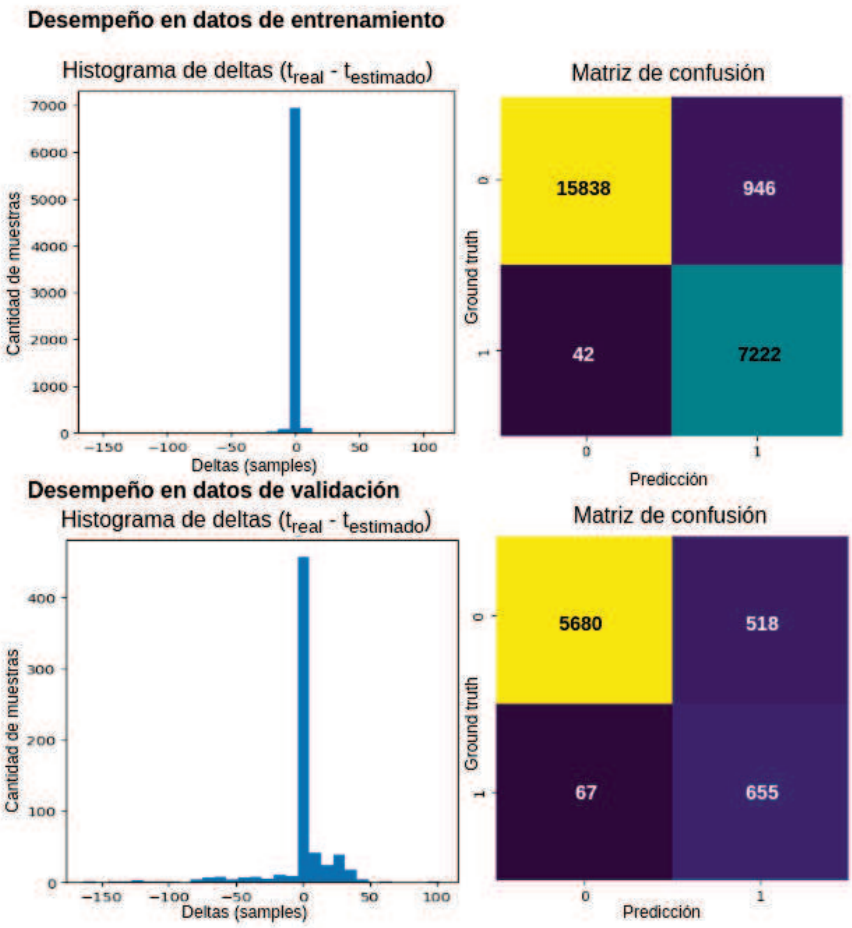


Figura 5.18: *Desempeño del modelo desarrollado. En la figura se incluyen los histogramas de las diferencias (o deltas) entre el tiempo real de las muestras positivas y el tiempo estimado, y las matrices de confusión para la clasificación en cada conjunto de datos.*

Capítulo 6

Validación y análisis comparativo

En este capítulo se describen los resultados alcanzados por el modelo desarrollado en datos desconocidos durante el diseño, y se analiza el significado de los mismos. Además se busca interpretar la significancia clínica de las predicciones realizadas, al evaluar el desempeño de detección de patrones ictales en escenario de validación compatibles con el uso médico. Para esto se realiza una validación cruzada del tipo *Leave One Subject Out*, donde en cada etapa de entrenamiento se separa un paciente y se entrena con todo el resto, simulando así escenarios de aplicación clínicos. En este análisis, además, se evalúa el impacto de utilizar segmentos de iEEG de distinta longitud temporal. De esta forma se analizan las ventajas derivadas del hecho de contar con una arquitectura totalmente convolucional, que resulta adaptable a entradas de distintos tamaños.

Finalmente se realiza una comparación del desempeño del modelo desarrollado contra otros *baselines* que utilizan técnicas clásicas de aprendizaje automático, basadas en el algoritmo XGBoost. También se resume un relevamiento de la bibliografía relacionada y se establece un análisis comparativo del modelo UNet con modelos de la bibliografía que persiguen objetivos similares.

6.1. Evaluación del modelo sobre datos de testeo

En esta primera sección se evalúa el desempeño del modelo en el subconjunto fijo de pacientes de prueba, de forma individual sobre cada sujeto. Este conjunto de prueba es distinto al de validación, por lo que no fue utilizado en la selección de hiperparámetros y construcción del modelo. De esta manera se puede realizar un análisis de desempeño clínico, evaluando en cada sujeto, y analizando la capacidad de adaptación del modelo a distintos patrones en los datos. Es importante notar que los pacientes pertenecen a distintos grupos donde los electrodos del sistema RNS están implantados en diferentes zonas del cerebro. Además, como se puede consultar en el Anexo B, estos pacientes también se diferencian en la cantidad de datos que tienen disponibles, yendo desde 360 a 2147 muestras. La relación de muestras positivas vs. negativas también es muy variable y sujeto dependiente, tomando valores desde el 1.2 % hasta el 36.4 %.

En la Tabla 6.1 se pueden ver los resultados para el conjunto de datos completo y para cada paciente independientemente. Es posible observar que la exactitud balanceada global es del 87.6 %, alcanzando valores muy similares a los del conjunto de validación. A partir de esto se pudo comprobar la capacidad de generalización del modelo, dado que el mismo logró obtener valores de rendimiento altos en datos desconocidos durante el entrenamiento y la validación.

Se puede observar que no se encontró gran variabilidad entre las exactitudes balanceadas calculadas para cada paciente, variando desde un 84.9 % hasta un 91.2 %, aunque para el valor del error MAE calculado, los valores varían desde 2.76 segundos a 8.1 segundos. Por su parte, el valor de F1 también presenta una mayor variabilidad entre sujetos, yendo de un 74.6 % a un 92.5 %.

Tabla 6.1

Desempeño del modelo en pacientes de testeo

Métrica	Test global	RNS1090	RNS7525	RNS8973	RNS1438
BACC	0.876 ± 0.026	0.849	0.912	0.874	0.872
Sensibilidad	0.840 ± 0.099	0.787	0.832	0.982	0.760
Especificidad	0.913 ± 0.104	0.910	0.991	0.767	0.985
Precisión	0.799 ± 0.109	0.799	0.947	0.766	0.685
F1	0.818 ± 0.077	0.820	0.925	0.781	0.746
MAE (seg)	4.939 ± 2.258	4.220	8.096	2.759	4.684

6.2. Validación cruzada *Leave One Subject Out*

Con el fin de evaluar el desempeño del modelo en todos los pacientes que conforman la base de datos, se llevó a cabo un experimento de validación cruzada del tipo *Leave One Subject Out*. En este esquema, se dejó fuera del conjunto de datos de entrenamiento un sujeto a la vez, eligiendo 3 sujetos para validación y entrenando con el resto de los datos. Una vez que se entrenaron todos los modelos, se testeó el desempeño en cada sujeto por separado, y luego se promediaron dichos resultados. El esquema se puede observar en la Figura 6.1. Los sujetos de validación permitieron elegir la cantidad de épocas con la que se entrenó cada modelo, para evitar el sobreajuste. Además, de esta forma se imitó la manera de entrenar el modelo presentado en la Sección 5.3.

6.2.1. Resultados de la validación cruzada

Al entrenar el algoritmo en la configuración de *Leave One Subject Out* se obtuvo para todos los pacientes una exactitud balanceada global de 81.3 ± 10.0 %, con un F1 promedio de 74.6 ± 10.2 % y un error absoluto promedio entre pacientes de 6.2 ± 5.1 segundos. Además se obtuvieron valores globales de precisión de 74.7 ± 10.5 %, sensibilidad de 76.2 ± 19.4 % y especificidad 86.3 ± 11.9 %.

En la Figura 6.2 se pueden ver las distribuciones de las métricas que se obtuvieron para todos los pacientes. En esta figura se puede observar que para el caso de la

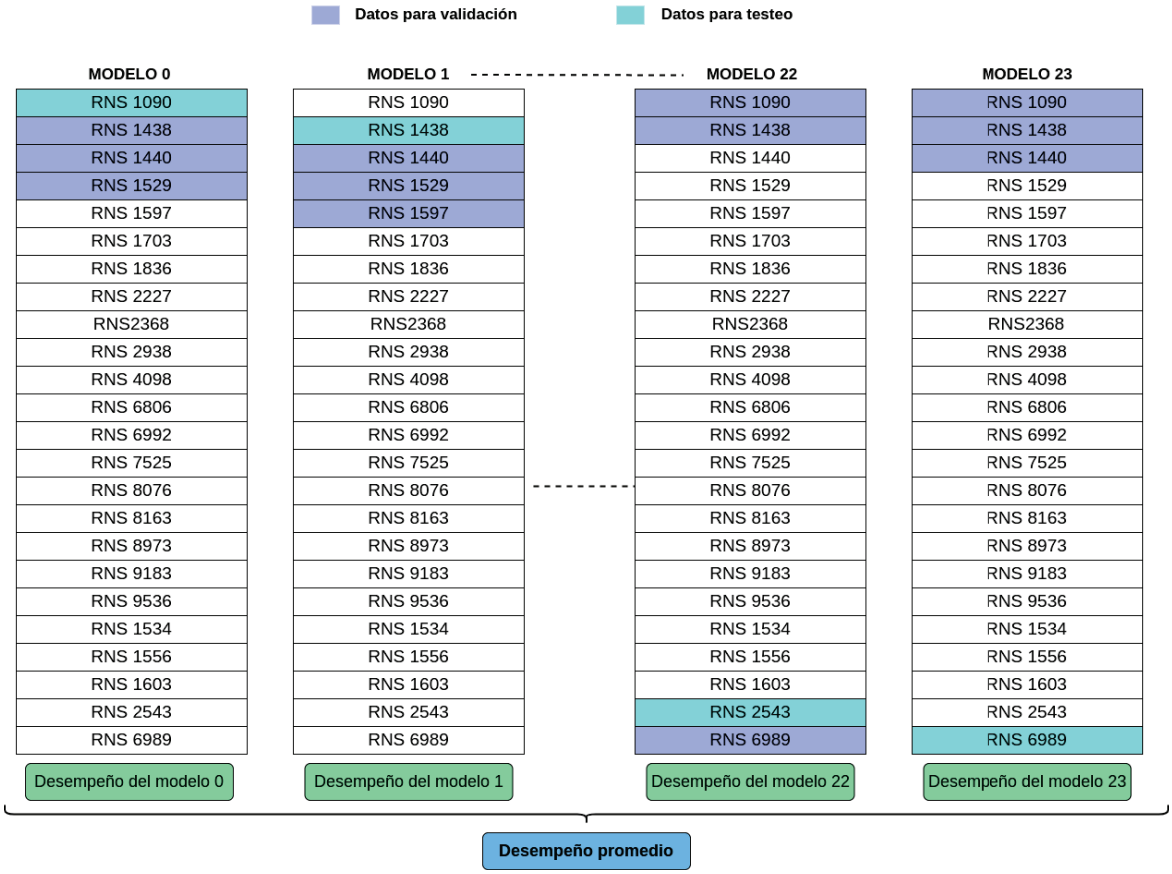


Figura 6.1: *Esquema del validación cruzada en la configuración Leave One Subject Out. Se puede observar que en el esquema propuesto se quita un paciente del grupo de entrenamiento para luego evaluar sobre el, pero además se quitan los 3 pacientes consecutivos para realizar una validación durante el entrenamiento.*

exactitud balanceada la mayoría de los pacientes obtuvieron un valor de BACC en el rango entre 0.8 y 1, y que para el F1 la mayoría obtuvo un valor entre 0.6 y 0.8. En la evaluación del error absoluto medio la mayoría de los pacientes obtuvo un valor de error entre 1 y 3.7 segundos. En total un 91.6 % obtuvo una exactitud mayor a 0.7, un 62.5 % un resultado mayor a 0.8 y un 20.8 % un resultado mayor a 0.9. Al observar los valores de F1 se puede observar que un 79 % obtuvo un valor de F1 mayor a 0.7.

Los resultados individuales por paciente se pueden observar en la Figura 6.3. Para cada paciente se muestra el valor de cada métrica evaluada y además se muestra la distribución de datos positivos y negativos. Por otro lado, junto con la exactitud balanceada y el F1 se indica el valor del F1 *by chance*. Este número indica el valor del F1 que se puede obtener al evaluar las muestras con un clasificador aleatorio. Para la exactitud balanceada el valor *by chance* sería de 0.5, independientemente del paciente. Esto es así dado que la exactitud evaluada es balanceada entre las dos clases posibles, y entonces un clasificador aleatorio tiene una probabilidad de 0.5 de asignar una clase u otra a cada segmento. El valor del F1 *by chance* se puede calcular a partir de la distribución a-prior de casos positivos (el prior π), es decir, la proporción de valores positivos en el conjunto de datos, según la ecuación 6.1. Este valor es el valor de F1 que se obtendría utilizando un clasificador aleatorio para realizar las predicciones, y

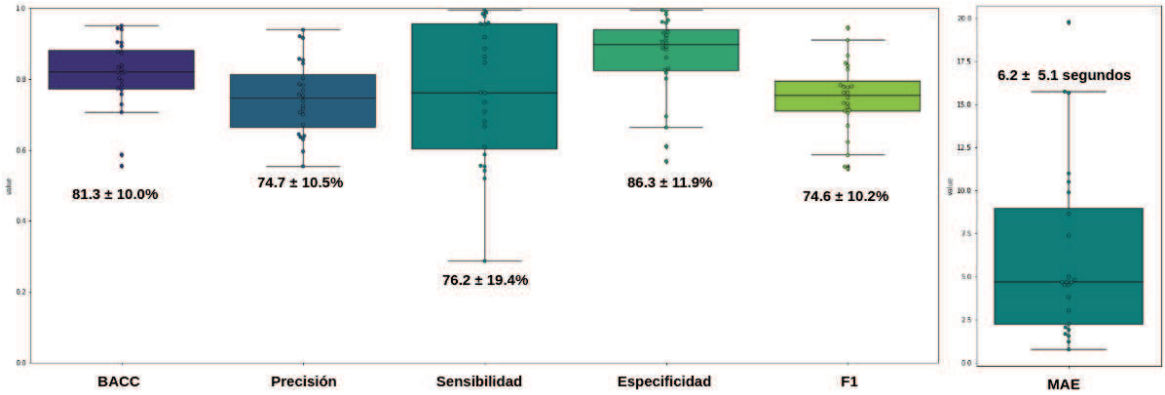


Figura 6.2: *Distribuciones para cada métrica de evaluación para todos los pacientes en la validación cruzada LOSO*

se calcula por paciente debido a que la proporción de segmentos positivos y negativos de cada paciente genera que sea diferente la capacidad de predicción de segmentos positivos.

$$F1_{bychance} = \frac{2\pi}{\pi + 1} \tag{6.1}$$

En 2 pacientes (PIT-RNS6992 y PIT-RNS8163) se obtuvieron valores de BACC menores al 60 %, y en esos mismos 2 el valor del F1 resultante es menor al valor del F1 *by chance*. Al observar en detalle en estos dos pacientes se encuentra que ambos pertenecen al grupo de 4 sujetos que mayor porcentaje de muestras con patrón ictal poseen, con proporciones mayores al 40 % de datos positivos. Estos 4 sujetos también coinciden con los 4 valores de F1 más bajos, siendo todos similares al valor del F1 *by chance*. En estos 4 sujetos se observa que la disminución en la exactitud se da a partir de una baja sensibilidad. Esta baja sensibilidad puede ser explicada a causa de que estos sujetos, que poseen una gran proporción de datos positivos, al ser excluidos dejan afuera del entrenamiento muchas muestras que contienen episodios ictales y, por lo tanto, el modelo resultante tiene mayores dificultades para predecir muestras positivas.

6.3. Evaluación del modelo con datos de duración real

Los datos utilizados para poder entrenar el modelo pasaron por un proceso de pre-procesamiento en el cual fueron recortados y normalizados a un tamaño limitado a 90 segundos, tal como se describió en la Sección 4.2.1. En la Sección 6.1 se evaluó el modelo en los datos de testeo normalizados al igual que los datos de entrenamiento y validación. Esta sección en cambio se enfoca en evaluar el modelo con los datos de duración original. Ejemplos de los espectrogramas generados con las duraciones originales se pueden observar en la Figura 6.4.

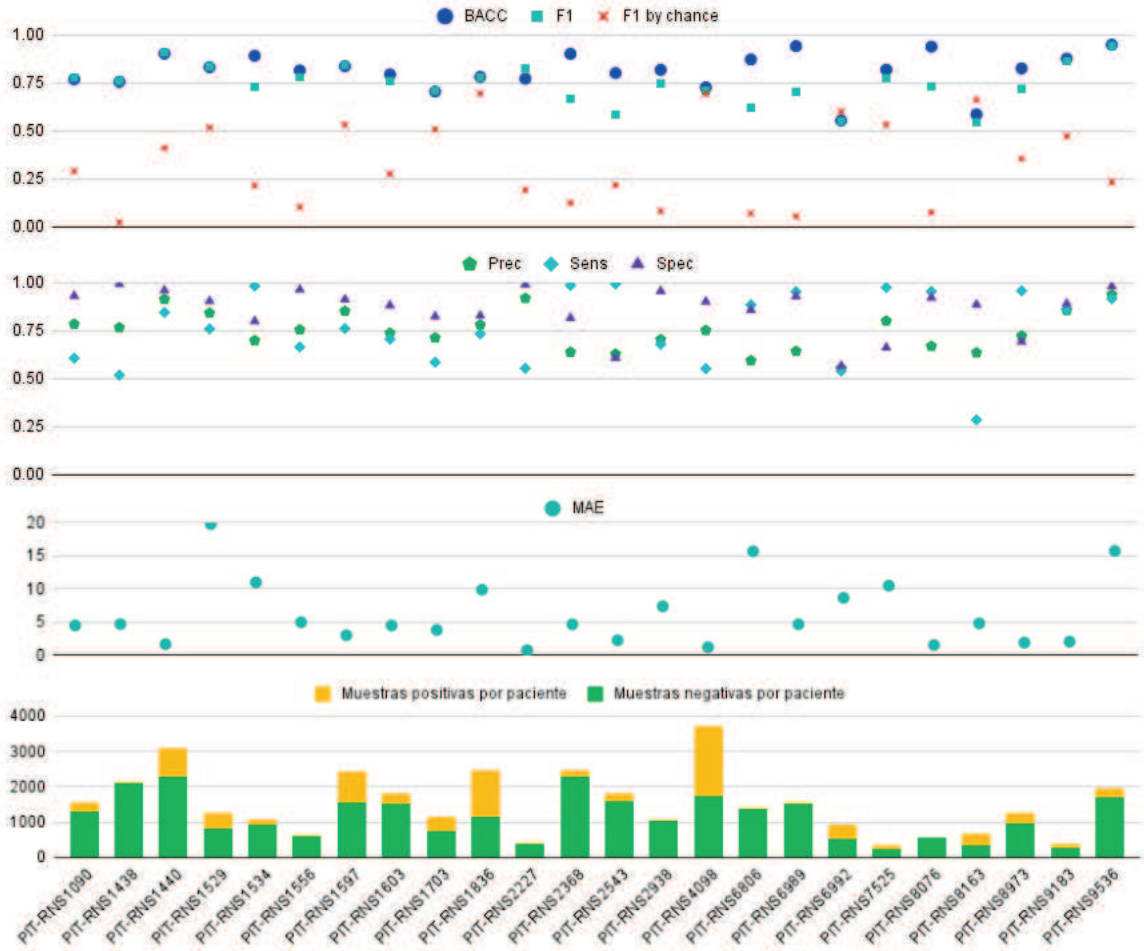
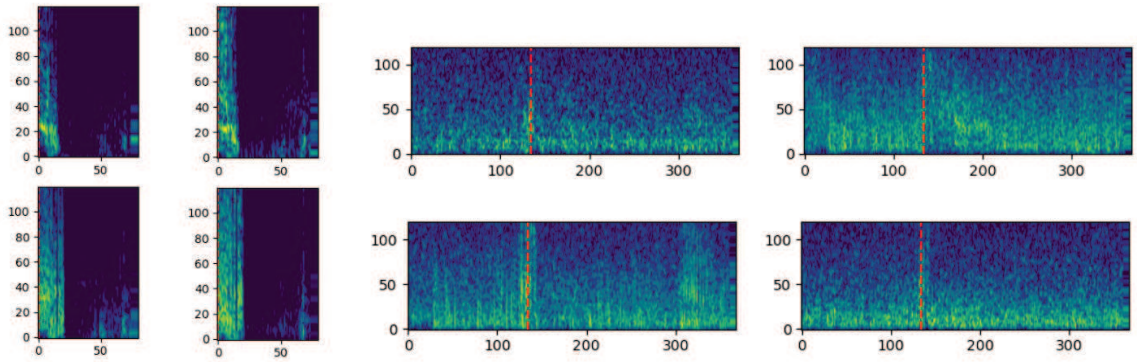


Figura 6.3: *Resultados de la validación cruzada LOO por paciente*

Una de las ventajas de las redes neuronales totalmente convolucionales (en relación a redes en las que se utilizan capas totalmente conectadas) se centra en su flexibilidad con respecto a datos de distintos tamaños. En el caso del modelo desarrollado en este proyecto, al haberse trabajado con una estructura totalmente convolucional, es posible realizar una evaluación utilizando el modelo con espectrogramas de distinta longitud temporal. Para evaluar los datos en este modelo solo se encuentran 2 limitaciones en el tamaño de las muestras. La primera limitación está dada en el eje frecuencial, el cual al convertir el resultado final a un vector debe mantenerse siempre de longitud igual a 120 píxeles. Esto no impide probar el modelo en datos de distinta longitud temporal. La segunda limitación está relacionada con la característica propia de la arquitectura UNet construida, que solo permite procesar imágenes de dimensiones proporcionales a 2^3. Es por esto que para cada muestra de distinta longitud temporal se aplicó la misma técnica de *padding* antes descripta para el caso de los datos estandarizados, agregando el mínimo número de columnas de píxeles posible hasta alcanzar el tamaño que resultará compatible con la red.



(a) *Menor a 90 segundos* (b) *Mayor a 90 segundos*

Figura 6.4: *Ejemplos de señales de distinta duración*

6.3.1. Evaluación del modelo sobre datos de testeo con datos de duración real

Para poder evaluar si el modelo era capaz de procesar señales de longitudes variables, el mismo fue evaluado en los 4 pacientes de testeo considerando sus muestras originales, sin el ajuste a 90 segundos, y se analizó la diferencia con la evaluación en los espectrogramas ajustados. La distribución de longitudes se puede observar en la Figura 6.5. Al contar la cantidad de muestras de longitud mayor a los 90 segundos estándar en cada paciente se encontraron 9, 2, 254 y 4 muestras, para los pacientes de ID RNS1090, RNS7525, RNS8973 y RNS1438 respectivamente, mientras que al contar las muestras de longitud menor a 90 segundos se encontraron 25, 109, 464 y 1025 para cada paciente respectivamente. Se puede observar que hay una mayor cantidad de datos de longitud menor a los 90 segundos estándar que de longitud temporal mayor.

En un primer momento se evaluó el desempeño del modelo en todos los datos de cada paciente, manteniendo sus longitudes temporales originales. Los resultados obtenidos se pueden observar en la Tabla 6.2. En esta evaluación se encontró que los valores de las métricas de evaluación no diferían en gran medida de los obtenidos en los datos de longitud normalizada. En la tabla además se incluyen los valores correspondientes a la media y a la desviación estándar de las distribución de longitud sobre las muestras de cada paciente.

A partir de esto se evaluó si había una diferencia notable en la capacidad predictiva del modelo al ser evaluado sobre datos de longitud estándar, es decir de 90 segundos, frente a muestras de longitud más corta o más larga que los 90 segundos. En este caso no se evaluó el desempeño por paciente, dado que en los 4 pacientes de testeo se contaba con pocos datos e incluso podía no encontrarse una muestra positiva. En la Figura 6.6 se pueden encontrar los resultados obtenidos en la evaluación realizada sobre distintos conjuntos que incluían datos de distintas longitudes. En el primer conjunto se incluyeron todos los datos de testeo disponibles, independientemente de su longitud, mientras que en el segundo conjunto se incluyeron únicamente aquellos datos que superaban los 90 segundos de longitud temporal, y en el tercero se incluyeron aquellos datos menores a los 90 segundos. Por último, en el cuarto conjunto se incluyeron los datos con longitud temporal igual a 90 segundos, es decir, los datos con la misma

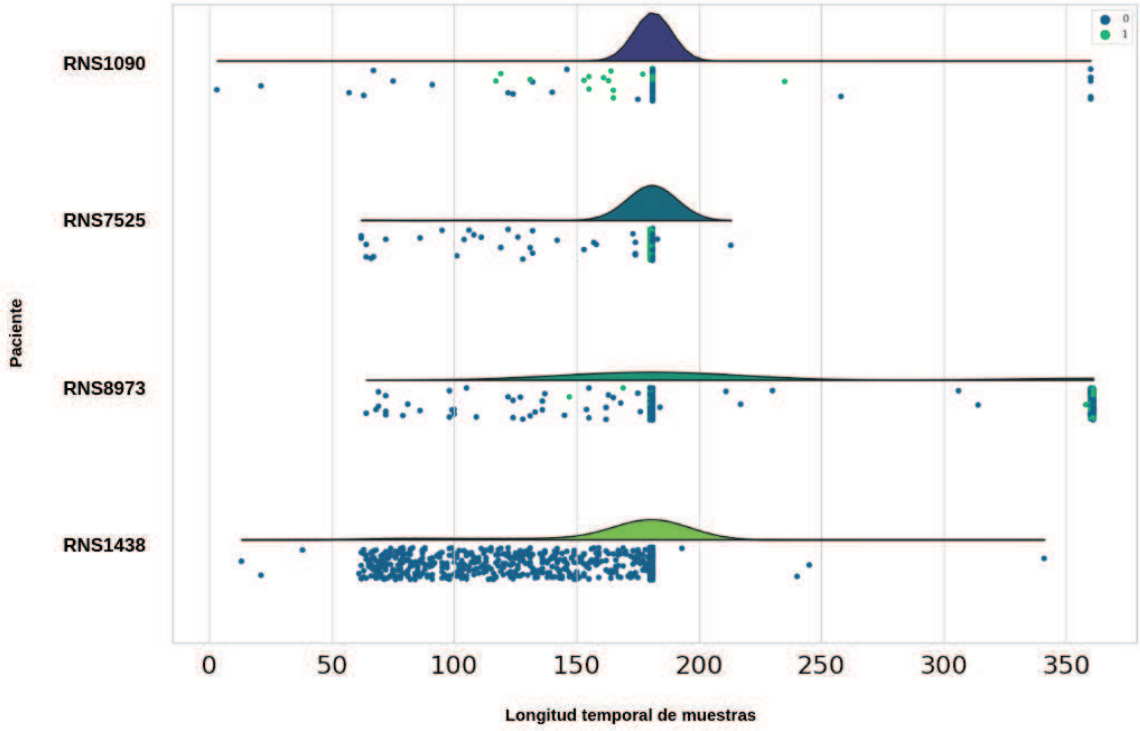


Figura 6.5: *Distribución por paciente de la longitud temporal de las muestras evaluadas*

longitud con la que se entrenó el modelo. Cabe destacar que para esta evaluación se contaba con un total de 5628 datos, de los cuales 269 eran muestras cortas, con un 28.25 % de positividad, 1623 eran muestra largos, con un 17.62 % de positividad, quedando la gran mayoría (con un total de 2726 datos) de longitud estándar de 90 segundos, con un 8.97 % de positividad.

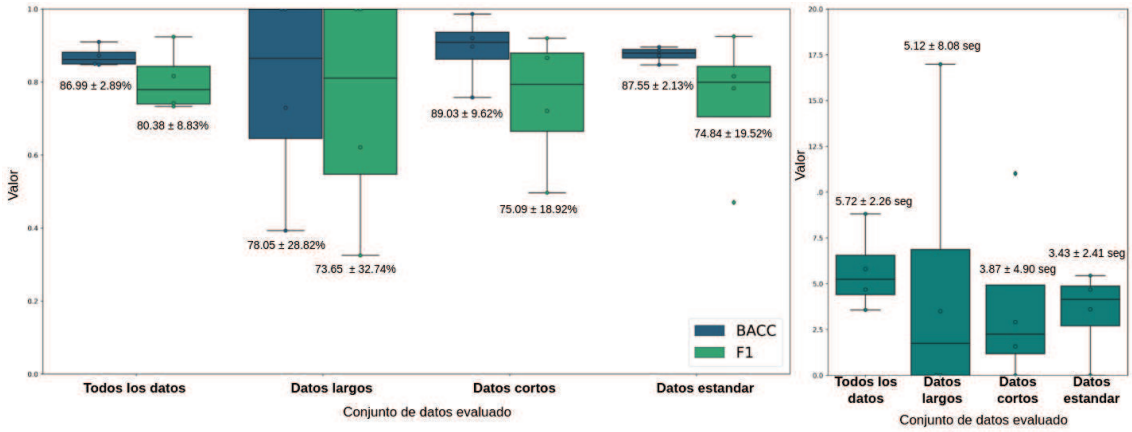


Figura 6.6: *Desempeño del modelo en conjuntos de datos de distinta longitud temporal*

El conjunto de datos que obtuvo un rendimiento más pobre fue el conjunto de datos largos, con duraciones mayores a 90 segundos. Se esperaba que este caso resultara el más difícil de analizar para el modelo dado que sus duraciones extensas podían incluir distintos patrones en una misma imagen, los cuales podían deberse o no a episodios ictales. A su vez se puede encontrar que es el modelo con mayor valor de MAE, lo cual

Tabla 6.2

Desempeño del modelo en pacientes de testeo con datos de duración variable

Métrica	RNS1090		RNS7525		RNS8973		RNS1438	
Long. media	180.94		175.85		214.22		167.67	
Long desv. est.	15.49		20.24		73.48		30.94	
Datos	90seg	Orig.	90seg	Orig.	90seg	Orig.	90seg	Orig.
BACC	0.847	0.849	0.910	0.912	0.850	0.874	0.872	0.872
Sensibilidad	0.788	0.787	0.832	0.832	0.982	0.982	0.760	0.760
Especificidad	0.906	0.910	0.987	0.991	0.740	0.767	0.985	0.985
Precisión	0.794	0.799	0.945	0.947	0.766	0.766	0.675	0.685
F1	0.816	0.820	0.924	0.925	0.742	0.781	0.816	0.746
MAE (seg)	3.57	4.22	8.816	8.096	5.804	2.759	4.684	4.684

era de esperar porque resultaba una tarea más difícil establecer el inicio del patrón ictal de manera precisa en datos de mayor extensión. Sin embargo, también es necesario tener en cuenta que existía un desbalance en la cantidad de datos de cada conjunto evaluado, lo cual pudo generar imparcialidades en los resultados. Por otro lado, en el conjunto de datos cortos el modelo tuvo mejor desempeño que en el conjunto de datos estandarizados, aunque ambos rendimientos resultaron similares. Se puede encontrar que la red resultante no encontró dificultades para clasificar este tipo de datos, logrando identificar correctamente los patrones ictales y su punto de inicio.

6.3.2. Resultados de la validación cruzada con datos de duración real

Para poder evaluar el desempeño del modelo en un contexto clínico se realizó el testeo en la modalidad de validación cruzada LOSO con los modelos generados en la Sección 6.2, pero testeando en los datos de duración real de cada paciente. Para cada paciente se tomaron los datos reales y se calcularon las proporciones correspondientes a datos mayores, menores e iguales a 90 segundos, y luego se registraron las exactitudes balanceadas y los valores de F1 tanto para el caso de evaluación en datos de longitud estándar como en los datos de longitud variable. Los resultados de esta evaluación se pueden consultar en la Figura 6.7.

Se puede observar que en general el desempeño de los modelos para cada paciente resultó similar en los datos de longitud variable y en los de longitud estándar. Se puede ver en el caso del BACC y el F1 no se encuentran diferencias notables entre los valores del testeo en los distintos tipos de datos. Las mayores diferencias se encuentran en el cálculo del error MAE, lo cual era esperado dado que el algoritmo encuentra variaciones en la longitud de tiempo que analiza. Los datos que mayores diferencias presentan con respecto al rendimiento en los datos de duración estándar son aquellos con una proporción alta de muestras de duración mayor a 90 segundos. Esto se corresponde con lo encontrado en la sección anterior (Sección 6.3.1) donde se encontró que el desempeño del modelo en los datos de testeo disminuía al evaluarlo en datos de duración larga.

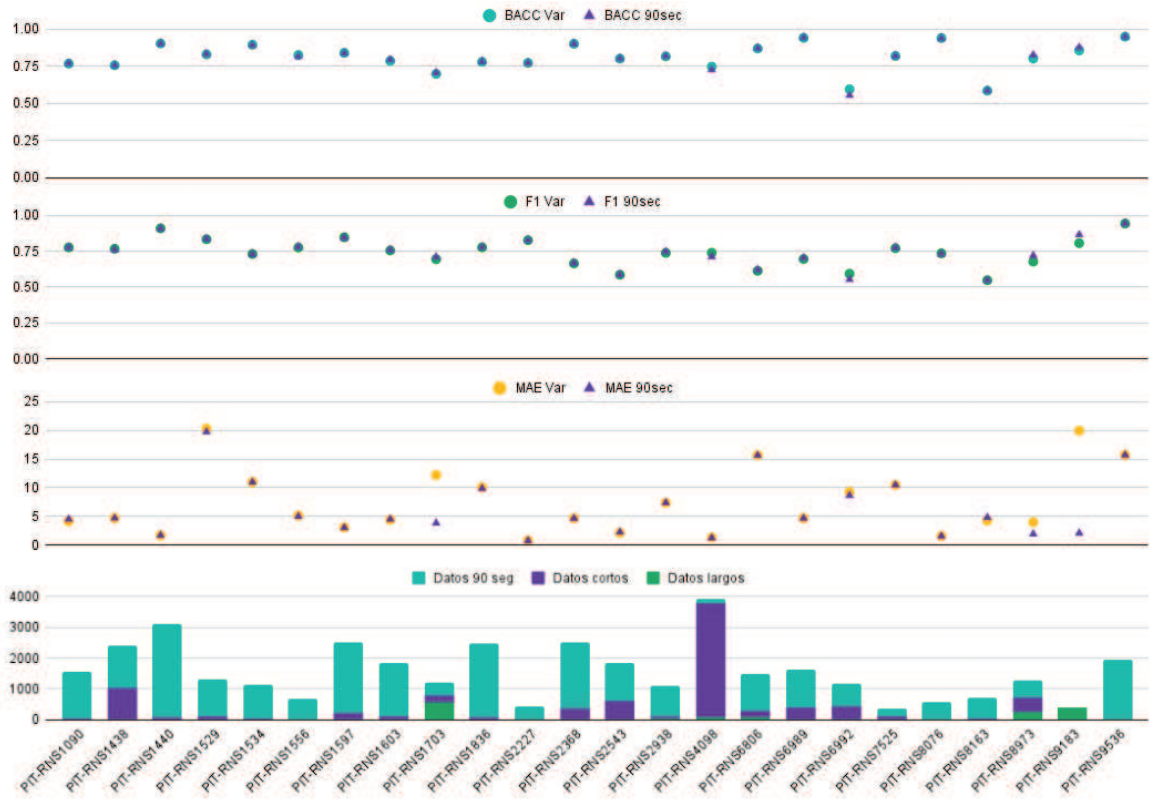


Figura 6.7: *Resultados de la evaluación en escenario LOSO en datos de longitud real*

6.4. Comparación con otras técnicas de aprendizaje automático

Para poder determinar una base de rendimiento contra la cual evaluar los resultados del modelo entrenado a partir de la UNet, se buscó un modelo de base (*baseline*) que fuera capaz de predecir la clase a la que pertenece cada muestra y, además, que pueda estimar el tiempo de inicio ictal en aquellos datos que hayan sido clasificados como positivos. Es por esto que se desarrollaron dos modelos distintos, uno que permitiera clasificar las muestras según la presencia o no de patrones ictales, y otro que permitiera regresionar el tiempo de inicio del patrón ictal en aquellos casos que un evento fuera detectado.

Existen distintos tipos de algoritmos de ML, que se ajustan a los problemas a resolver en cada caso. En especial en este proyecto se trabajó con un algoritmo conocido como XGBoost, en inglés *eXtreme Gradient Boosting* (XGBoost), el cual se utilizó para ambas tareas requeridas. El algoritmo está basado en modelos del tipo árbol, y pertenece a dicha familia junto con otros como el árbol de decisión y el *Random Forest*, utilizando técnicas relacionadas con esta familia como son el *bagging, boosting* y *gradient boosting* [64]. Una descripción más detallada sobre estos modelos puede encontrarse en el Anexo C. Este algoritmo presenta generalmente buen rendimiento en tareas de clasificación y regresión, y fue ésta la razón que motivó su utilización en este proyecto.

6.4.1. Desarrollo y desempeño del modelo de ML

Para el modelo de base se trabajó con una arquitectura en *pipeline*, que primero toma un dato, realiza la extracción de características, luego lo clasifica, y, si el mismo es clasificado dentro de la categoría con episodio, entonces es evaluado con el modelo de regresión y se calcula el tiempo de inicio ictal para el mismo. Al contrario, si la muestra es clasificada dentro de la categoría sin episodio entonces no se calcula el tiempo de *onset*. Este esquema se puede observar en la Figura 6.8. De esta manera se pasó a desarrollar dos modelos que cumplieran con este objetivo:

- un modelo para la clasificación de los datos, es decir, un modelo que permita determinar si en el registro electroencefalográfico existió o no un episodio con características ictales,
- y un segundo modelo para la regresión de los datos, el cual permita determinar el tiempo de inicio del período ictal en aquellos datos que registraran un episodio de características ictales.

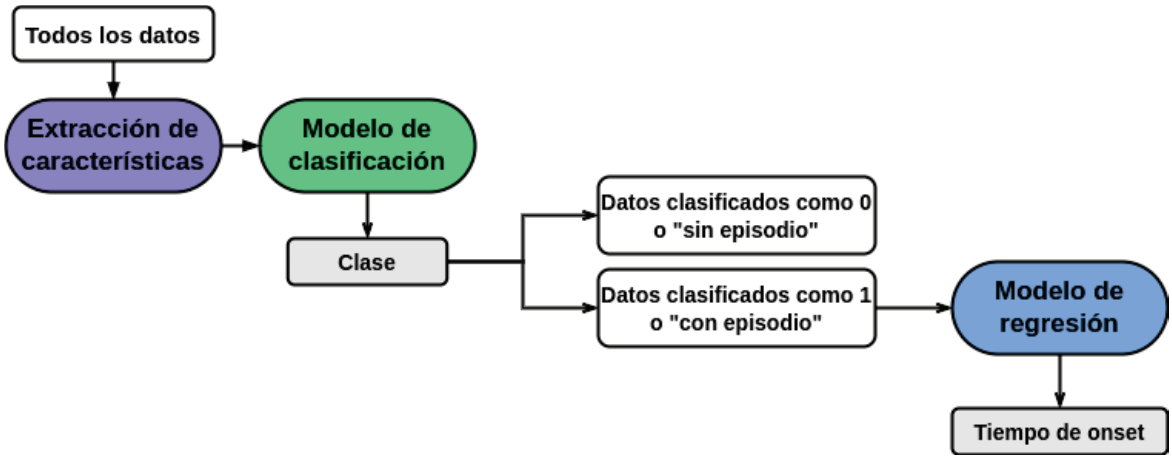


Figura 6.8: *Pipeline de modelos de base*

El primer modelo, de clasificación, se entrenó con un conjunto que incluía todos los datos disponibles, tanto los que presentaban episodio como los que no. El criterio para definir las clases de los datos fue el mismo que se tomó para la clasificación de los espectrogramas en el modelo UNet, el cual consiste en una clase negativa o 0, para aquellos datos que no presentan episodios de características ictales, y una clase positiva o 1, con aquellos datos que presentan algún episodio de características ictales. El segundo modelo, de regresión, se entrenó con un conjunto de datos que incluyó únicamente los datos que estaban identificados como pertenecientes a la clase positiva, es decir solo se tomaron los datos que presentaban algún episodio con un tiempo de inicio determinado. Esto permitió separar de una manera más sistemática los grupos dentro del espacio de parámetros, sin agregar error por regresionar datos sin episodio en la detección del punto de inicio ictal.

Para poder entrenar ambos modelos propuestos no se utilizaron los espectrogramas completos, dado que este tipo de datos resulta demasiado complejo para analizar con algoritmos de este estilo. En lugar de los espectrogramas, se extrajeron características específicas a partir de los datos para describir las señales de iEEG. Las características

a extraer fueron determinados a partir de investigación bibliográfica [65] [66] [67] y calculadas de forma previa a este proyecto. En total se utilizaron 71 características para poder identificar a los datos, y todas correspondían a distintos cálculos, como son: potencia absoluta, potencia relativa y cálculos en el dominio del tiempo, como mínimo, máximo, media, varianza, energía, longitud lineal, kurtosis, ske, entropía.

En ambos casos, para la selección de los hiperparámetros se utilizó el método de optimización bayesiana. Este método permite encontrar el máximo en funciones de costo elevado. Para poder optimizar dichos hiperparámetros se configuró como función objetivo la pérdida de cada modelo entrenado, evaluado en datos de validación. En cada iteración se entrenó el modelo buscado usando datos de entrenamiento, y luego se calculó la pérdida para los datos de validación.

Los hiperparámetros que se buscaron optimizar [68] en ambos casos incluyen:

- Profundidad máxima (*max depth*): este hiperparámetro determina la profundidad máxima de un árbol, la cual, si se incrementa, genera modelos más complejos y tendientes a sobreajustar.

- Gamma: determina la reducción mínima en la pérdida que es requerida para generar una nueva partición en un nodo de un árbol. A mayor gamma, más conservativo resultará el modelo final.

- Tasa de aprendizaje (*learning rate*): determina el tamaño del paso utilizado para la actualización de los pesos.

- Delta: permite establecer un límite para los valores de los pesos de los árboles. Si es igual a cero, entonces no se establecen límites para los pesos.

- *Subsample*: da la fracción de observaciones que van a ser *samples* aleatorios para cada árbol. Un valor de 0.5 significa que el algoritmo va a muestrear aleatoriamente la mitad de los datos antes de comenzar a crecer árboles.

En el caso del modelo de clasificación, la función de pérdida utilizada fue la precisión promedio, la cual es una métrica que indica el área bajo la curva de la curva *precision-recall*. En el caso del modelo de regresión la función de pérdida utilizada para medir al modelo en validación fue el error absoluto medio. Es necesario considerar que para ambos modelos se usó un método de optimización que buscaba minimizar el objetivo. En el caso de la regresión esto era correcto ya que el error absoluto medio es una métrica que se debe minimizar para optimizar el modelo. En el caso de la clasificación, sin embargo, la precisión media debía ser maximizada para obtener el modelo con mejor rendimiento, por lo que se buscó minimizar el valor negativo de la misma.

Por otro lado, en el caso del modelo de clasificación se implementó un método para muestrear los datos, de forma de poder balancear la cantidad de datos pertenecientes a cada clase y aumentar la cantidad de datos disponibles para entrenar el modelo. Se utilizó la técnica de sobre-muestreo de la clase minoritaria para el entrenamiento, logrando así aumentar la cantidad de datos de disponibles y también balancear la cantidad de datos de cada clase.

6.4.2. Resultados del modelo de base

Los resultados del modelo de clasificación en los datos de entrenamiento alcanzaron un 95 % de exactitud balanceada, resultando similar al rendimiento global del 95 % alcanzado con la arquitectura basada en UNet. Sin embargo, al observar el rendimiento sobre los datos de validación, se encontró que era menor que en los datos de entrenamiento, pudiendo clasificar los datos en un 87 %, alejándose del 89 % global alcanzado en el modelo UNet. Lo mismo sucedió al analizar el modelo sobre el conjunto de datos de testeo, alcanzando un 74 % de exactitud balanceada en la clasificación de las muestras contra un promedio de 88 % alcanzado por el modelo UNet en la misma partición. Los valores de desempeño de la clasificación se pueden observar en la Tabla 6.3. Además, en la Figura 6.9 se pueden observar las matrices de confusión para cada conjunto de datos. En los tres casos se puede ver que el modelo de clasificación pareciera sobreajustar a los datos de entrenamiento, alcanzando una generalización más pobre que el modelo UNet.

Tabla 6.3

Rendimiento del modelo de clasificación

Métrica	Datos de entrenamiento	Datos de validación	Datos de testeo
BACC	0.95	0.87	0.74
Sensibilidad	0.94	0.88	0.88
Especificidad	0.96	0.86	0.6
Precisión	0.95	0.92	0.87
F1	0.93	0.78	0.76

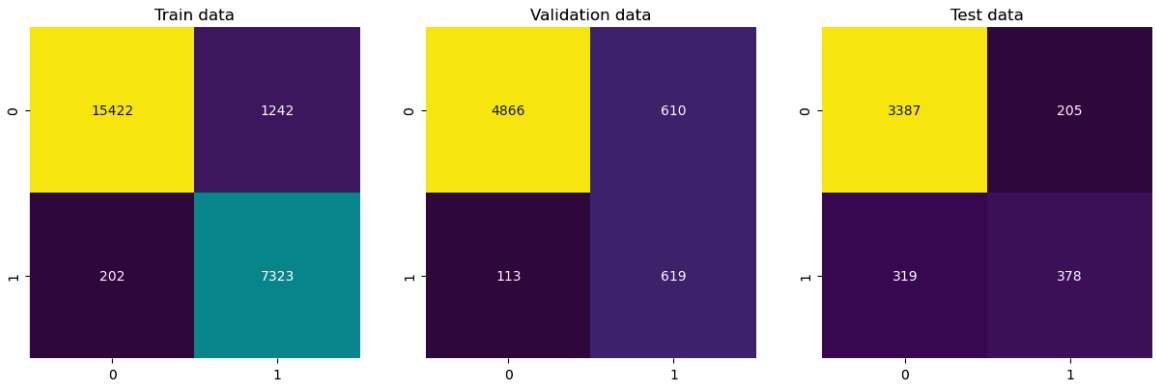


Figura 6.9: *Matrices de confusión para el modelo de clasificación de base*

En la primer matriz es posible observar que el modelo es capaz de clasificar correctamente en gran medida los datos de entrenamiento, aunque no llega a alcanzar el desempeño del modelo desarrollado con la UNet. En ese caso se contaba con un total de 946 datos clasificados como FP y 42 FN, mientras que en este caso se cuenta con 1242 FP y 202 FN. En el caso de los conjuntos de validación y testeo están diferencias se mantienen, teniendo en validación 518 FP y 67 FN contra 610 y 113.

Por otro lado, en el entrenamiento del modelo de regresión se observó que los errores cometidos en todos los datasets resultaron comparables, encontrándose todos

los errores alrededor de 7 y 9 segundos. Para el conjunto de datos de entrenamiento el error de MAE resultó de 7.68 segundos, mientras que en validación fue de 9.16 segundos y en testeo de 8.44 segundos. En la Figura 6.10 se pueden observar las regresiones y los histogramas con la distribución de los errores para cada conjunto de datos. Sin embargo, nuevamente estos valores resultan más pobres que en la evaluación realizada con la arquitectura UNet, en la cual se lograron alcanzar valores más bajos, habiendo resultado de 0.77 segundos en entrenamiento y 5.41 segundos en validación.

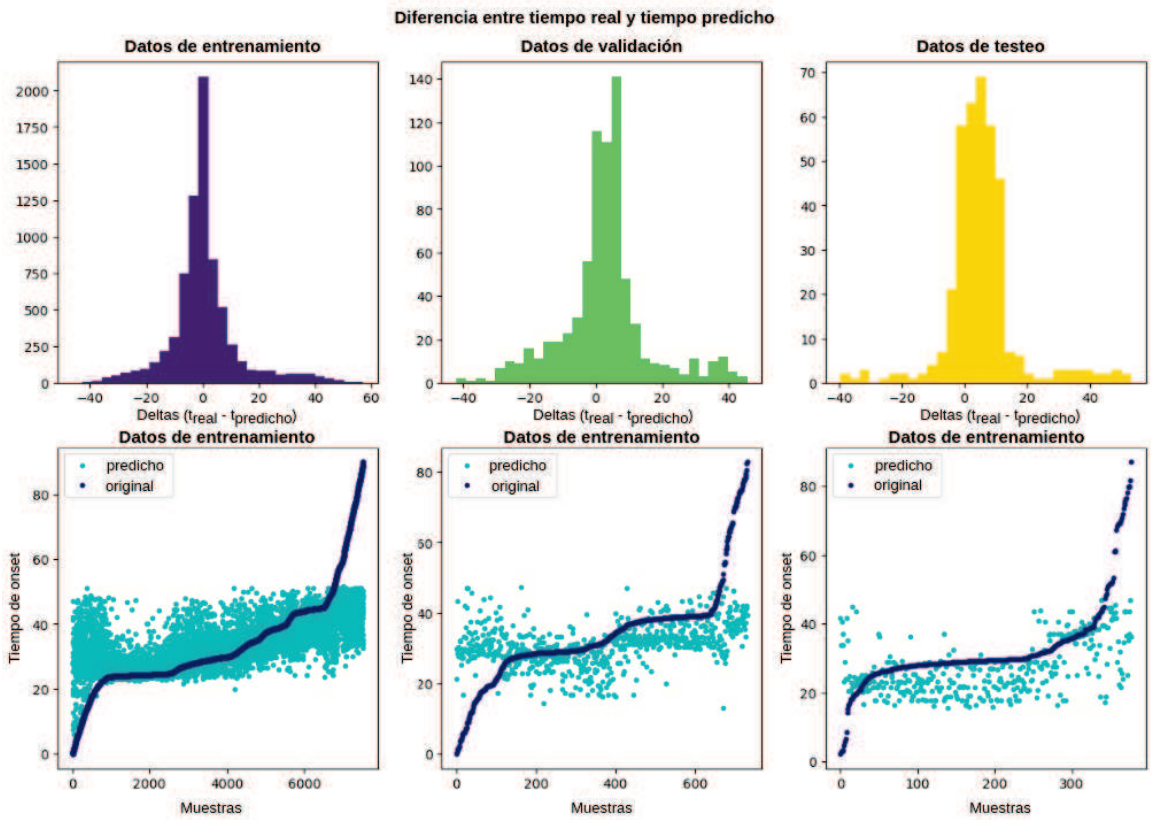


Figura 6.10: *Histogramas y valores de regresión para el modelo de regresión de base*

Por otro lado, además de analizar al conjunto de datos de testeo como un único dato se lo evaluó separando por paciente, de forma de poder realizar la misma evaluación clínica que en la sección 6.1. De esta manera se puede realizar un análisis de desempeño clínico, evaluando en cada sujeto, cuyos resultados se pueden consultar en la tabla 6.4. Se puede ver que los modelos de clasificación y regresión funcionan bien para algunos pacientes pero de manera muy pobre para los restantes. En este caso, los médicos estarían obteniendo muy buen desempeño en los pacientes de testeo 3 y 4, pero muy malos resultados en los pacientes 1 y 2, lo cual complicaría su tarea y generaría desconfianza sobre la validez de los resultados generados por la evaluación con el modelo.

6.4.3. Análisis y comparación

El modelo UNet demuestra un mejor desempeño sobre todos los conjuntos de datos, especialmente sobre los datos de testeo. El desempeño presenta menores variaciones

Tabla 6.4

Desempeño del modelo en pacientes de testeo

Métrica	RNS1090	RNS7525	RNS8973	RNS1438
BACC	0.54	0.57	0.96	1
Sensibilidad	0.78	0.64	0.95	1
Especificidad	0.30	0.50	0.97	1
Precisión	0.74	0.62	0.96	1
F1	0.54	0.57	0.94	0.95
MAE (seg)	13.48	5.12	7.96	9.59

en la evaluación sobre distintos pacientes, y además se logran mejores estimaciones del error de inicio ictal con el modelo basado en espectrogramas. El modelo generado con el algoritmo XGBoost en dos pacientes no logra un valor de especificidad mayor al 50 %, generando así que una gran cantidad de muestras puedan ser identificadas como muestras con episodio de forma incorrecta. Si bien en general la UNet supera al modelo de clasificación basado en XGBoost, esto se da por una muy baja especificidad, siendo que el modelo alternativo tiene mayor sensibilidad y precisión considerando los 4 sujetos totales.

Por otro lado, para llegar a los resultados obtenidos, en el caso del modelo de base es necesario realizar un preprocesamiento mas extenso sobre los datos para poder realizar el entrenamiento, debiendo en primer lugar identificar las características que se incluirán como entradas al algoritmo, y luego calcularlas de forma manual. La identificación de estas características depende del nivel de conocimiento y del criterio utilizado por la persona que realice esta tarea. Además, este algoritmo no permite realizar las dos tareas principales planteadas para este problema de manera conjunta, sino que es necesario hacerlo de forma desacoplada y secuencial, primero realizando la clasificación de los datos, y luego estimando el tiempo de inicio ictal mediante una regresión. Sin embargo, es un algoritmo que se entrena en un tiempo menor al necesario para entrenar la red UNet (siendo el tiempo necesario para entrenar ambos modelos de XGBoost de 20 minutos contra el tiempo requerido para entrenar la UNet, de 3 a 4 horas aproximadamente, según la capacidad del procesador utilizado).

6.5. Comparación con modelos de bibliografía

Además de buscar un punto de comparación con técnicas de aprendizaje automático, se buscó establecer un punto de comparación de resultados para el modelo desarrollado con resultados de distintas investigaciones de la bibliografía que buscaran resolver problemas comparables. Así es como se realizó un relevamiento del estado del arte actual incluyendo distintas investigaciones con objetivos y escenarios similares. En cada uno de los trabajos evaluados se plantearon distintas alternativas para abordar el problema a resolver.

En un primer momento se evaluó un informe de investigación denominado "*Expert-Level Intracranial Electroencephalogram Ictal Pattern Detection by a Deep Learning*

Neural Network"(Constantino Alexander et al., 2021) [11], en el que se utilizó un conjunto de datos similar al estudiado en este proyecto, consistiendo de 5226 patrones de eventos ictales en 18368 muestras de 22 pacientes implantados con el sistema RNS. Resultó interesante poder comparar los resultados con este trabajo, dado que el conjunto de datos provenía del mismo sistema y las diferencias en los resultados tienen menos probabilidades de depender de los datos utilizados. Sin embargo, en este informe los datos se usaron en formato de señales en lugar de espectrogramas. Además, se utilizó una red neuronal convolucional con conexiones residuales y dos salidas, siendo las mismas la probabilidad de que una muestra incluya un patrón ictal, y el tiempo de onset ictal. Para esto se utilizó una función de pérdida que ajusta las dos tareas, al contrario del modelo desarrollado con la arquitectura UNet, donde se usa una función sin términos sumados. Se puede observar que en este caso, al contrario del modelo desarrollado en este proyecto, la red utilizada incluía capas completamente conectadas, de forma que sólo es aplicable en señales de longitud temporal estándar, que hayan pasado por un procesamiento previo.

Además en este caso se plantearon 2 escenarios distintos para el procesamiento de los datos, con un primer escenario que incluía pacientes con patrones convulsivos correspondientes a un periodo de registro largo, y un segundo escenario donde se incluían datos de pacientes en un periodo de post implantación. Se utilizaron distintas técnicas de aumento de datos para evitar el sobreajuste en entrenamiento, pero que resultaban aplicables a señales, al contrario de las aplicadas en este trabajo que son técnicas utilizadas en espectrogramas.

El testeo se realizó con una validación cruzada del tipo *Leave One Subject Out*, pero para mostrar los resultados se incluyeron únicamente los resultados de testeo de 12 pacientes, los cuales tenían un mínimo de 35 muestras con patrones ictales. En el primer caso se logró una AUCPR de 0.84 ± 0.19 y un error en la regresión de 6.3 ± 1 segundos, mientras en el segundo caso se obtuvo un AUCPR de 0.80 ± 0.19 y un error de 6.3 ± 0.9 segundos. Además, en este trabajo se encontró que al analizar los distintos grupos de sujetos separados según lugar de implantación del RNS no se obtuvieron resultados variables.

El siguiente artículo de investigación evaluado fue "*Epileptic seizure detection based on EEG signals and CNN*"(Zhou Mengni et al., 2018) [69], y lo que resulta interesante de este trabajo es que realiza una evaluación comparativa de la clasificación de señales de EEG en el dominio temporal y el frecuencial. En este estudio se evaluó el uso de arquitecturas pequeñas, contando únicamente con un bloque convolucional que es seguido por una capa totalmente conectada que da la salida de la red, lo que permite clasificar la entrada en 3 clases distintas: Ictal, Preictal o Interictal. Las entradas a la red consistieron en muestras conformadas por las señales temporales o frecuenciales con la señal correspondiente a cada canal. Lo que se comparó en este caso fue el desempeño de los modelos desarrollados con señales en el dominio del tiempo y en el dominio frecuencial.

En este caso se utilizaron nuevamente datos de dos fuentes distintas. Por un lado se utilizó una base de datos proveniente del hospital universitario de Freiburg en Alemania, que contiene datos de EEG intracraneal de 21 pacientes con epilepsia focal que se encontraban en estudios invasivos prequirúrgicos. En total se contaba con 87

episodios convulsivos de los 21 pacientes. Las actividades epileptiformes y los tiempos de onset ictales fueron anotados por expertos. La siguiente base de datos en ser usada fue la base CHB-MIT, que contiene datos de 23 pacientes pediátricos de registros de EEG superficial. En la evaluación se analizaron 3 casos de clasificación, con dos casos binarios, en los que se analizó la posibilidad de clasificar en interictal-preictal o entre interictal-ictal, y el caso de clasificación múltiple, buscando entre interictal-ictal y preictal. En el caso de la base Freiburg con el dominio frecuencial se obtuvieron exactitudes de 96.7, 95.4 y 92.3 % y en el dominio temporal fueron de 91.1, 83.8 y 85.1 %. En el caso de la base CHB-MIT se obtuvieron exactitudes de 95.6, 97.5 y 93 % en el dominio del tiempo y de 59.9, 62.3 y 47.9 % en el dominio de la frecuencia. En general, todos los pacientes obtuvieron mejores resultados de exactitud, sensibilidad y especificidad para la clasificación de las señales en el dominio frecuencial.

Otro trabajo evaluado fue "*Automatic seizure detection based on imaged-EEG signals through fully convolutional networks*"(Gomez et al., 2020) [31]. Lo interesante de este trabajo reside en que se utilizaron distintas arquitecturas completamente convolucionales, compuestas de pocas capas, incluyendo capas convolucionales, normalización por batch, activaciones ReLU y capas de pooling, resultando en un modelo liviano, de tan solo 314.000 parámetros. Con los datos se intentó emular la inspección visual que realizan los especialistas médicos cuando leen los registros de EEG, por lo que propusieron una estrategia que genera representaciones en formato de imagen de las señales crudas del EEG, que luego son usadas como entrada para el algoritmo utilizado. Es decir, no se utilizaron los datos en forma de espectrogramas, pero si se aplicó la idea de tomar como datos de entrada imágenes en lugar de señales.

Una diferencia fundamental con este proyecto reside en que se utilizaron dos bases de datos que incluían tanto registros de EEG superficial como intracraneal. Sin embargo, al igual que en este caso, y dada la naturaleza de los datos de EEG tomados de pacientes con patología epiléptica, los conjuntos de datos eran desbalanceados y esto supuso también uno de los mayores desafíos del problema. La primera base de datos, conocida como CHB-MIT, solo contaba con señales de EEG superficial, y los sujetos involucrados consistían en 23 pacientes pediátricos, de los cuales se analizaron 941.6 horas de actividad interictal y solamente 3 horas de actividad ictal. Con esta base se obtuvo una exactitud global de valor medio 92.9 % y una especificidad de 0.93, con una precisión de 0.51, una sensibilidad de 0.53 y un valor de F1 de 0.46, todos calculados con la configuración de *Leave One Patient Out*. La segunda base de datos conocida como EPILEPSIAE, contiene registros de 275 pacientes, incluyendo señales de electroencefalografía intracraneal junto con los datos de EEG superficial. En este caso se analizaron 1302.2 horas de actividad interictal y solamente 2.5 horas de actividad ictal. En este se obtuvo una exactitud de 98 % con una especificidad de 0.983, pero con una precisión de 0.229, una sensibilidad de 0.259 y un valor de F1 de 0.208. Se puede observar que, a pesar de haber obtenido en ambos conjuntos de datos mejores valores de exactitud y especificidad, los valores de evaluación en el LOSO de sensibilidad, F1 y precisión fueron superados con el modelo desarrollado con la UNet.

El último trabajo evaluado se denomina "*A high accuracy electrographic seizure classifier trained using semi-supervised labeling applied to a large spectrogram dataset*"(Barry Wade et al., 2021)[70]. Lo remarcable de este trabajo es que se contó con

Tabla 6.5

Resumen comparativo de trabajos de investigación

Trabajo de investigación	Datos utilizados	Tipo de arquitectura	Métrica y resultados
Constantino A. et al.	iEEG de 22 pac. con RNS	23 capas CNN y 11 bloques residuales	AUCPR 0.80-0.84 MAE 6.3seg
Zhou M. et al.	iEEG de Freiburg de 21 pac. y EEG sup. de CHB-MIT con 23 pac. ped.	Arquitectura con 1 capa CNN y 1 FC	BACC 0.92-0.97 (frec.) BACC 0.48-0.91 (tiempo)
Gomez C. et al.	EEG sup. de CHB-MIT con 23 pac. ped. e iEEG de EPILEPSIAE con 275 pac.	CNN con 9 capas convolucionales y 314 mil parámetros	ACC 0.93-0.98 Sens. 0.26-0.53 Espec. 0.93-0.98 Prec. 0.30-0.51 F1 0.21-0.46
Barry W. et al.	iEEG tomados de 113 pac. con RNS	ResNet50	BACC 0.95 Sens. 0.94 Espec. 0.97 F1 0.94
Proyecto	iEEG tomados de 24 pac. con RNS	UNet modificada	BACC 0.81 Sens. 0.76 Espec. 0.86 Prec. 0.74 F1 0.74 MAE 6.2seg

una base de datos que, al igual que en este proyecto, incluía datos obtenidos con el sistema RNS, pero con señales de 113 pacientes, siendo una base mucho más grande, y por ende variada, a la del presente proyecto. También se utilizaron las señales convertidas en espectrogramas, por lo que es un trabajo que permite validar en un *dataset* extenso el uso de los espectrogramas para la clasificación de las señales de pacientes con patología epiléptica implantados con el sistema RNS. Al contar con esta gran cantidad de datos, el etiquetado no se realizó de forma manual sino que se realizó de modo semi-supervisado, con una técnica de preprocesamiento basada en *clustering*. Para evaluar las distintas redes posibles se utilizaron distintas redes de complejidad variables, siendo la más compleja la arquitectura ResNet50 [71], y se entrenó cada una con distintas combinaciones de los datos, entrenado con conjuntos de 72 pacientes de entrenamiento, 18 de validación y 23 de testeo.

Al contrario de lo que se encontró en las evaluaciones con la arquitectura UNet, donde la red con mejor funcionamiento no resultó ser la de mayor cantidad de parámetros, en esta investigación la arquitectura que obtuvo el mejor desempeño fue la red ResNet50 con un valor de F1 de 94.26 % y exactitud balanceada de 95.72 % con una especificidad del 97.17 % y una sensibilidad del 94.26 %. En este trabajo se encontró que

las redes mas complejas permitían que no se subajuste a los datos de entrenamiento y generaban modelos más robustos en cuanto a su rendimiento en distintas separaciones de los datos. También se encontró que al clasificar señales de menor o mayor longitud temporal que 90 segundos, las exactitudes resultaban menores. Este mismo resultado se encontró en la evaluación del modelo desarrollado en este proyecto, analizado en la Sección 6.3, encontrando que al evaluar sobre señales mayores a 90 segundos la especificidad del modelo empeoraba, no pudiendo clasificar correctamente las muestras con y sin presencia de patrones epilépticos.

Una vez revisados los trabajos de investigación, se pasó a compararlos entre sí y con el modelo desarrollado en este proyecto, de forma de obtener una idea clara del estado del arte y del aporte que realiza este trabajo al mismo. En la Tabla 6.5 se puede observar un resumen comparativo de las investigaciones consultadas, incluyendo también el modelo presentado.

6.6. Resumen del capítulo

Durante este capítulo se presentaron los resultados de desempeño del modelo electo sobre los pacientes separados de la base de datos para testeo, y también sobre la evaluación de validación cruzada realizada para evaluar el modelo sobre todos los pacientes. Luego se evaluó la posibilidad de analizar los datos con su duración original, utilizando los espectrogramas generados sin la estandarización de la longitud temporal. También se desarrollaron comparaciones del modelo con respecto a modelos generados con técnicas de aprendizaje automático clásicas y con modelos de la bibliografía que utilizaron distintos enfoques para resolver el problema planteado en este proyecto. Habiendo presentado estos resultados y comparativas, en el siguiente capítulo se desarrollará una discusión sobre el trabajo realizado, indicando los puntos de mayor relevancia del proyecto.

Capítulo 7

Discusión y conclusiones

Repasando el trabajo realizado y las decisiones tomadas durante este proyecto, en este capítulo se discuten los resultados obtenidos y su relevancia en la ingeniería biomédica y en el área de aplicación. Además, se proponen los trabajos futuros a realizar, y se cierra el capítulo con las conclusiones del proyecto.

7.1. Discusión

En el proyecto se desarrolló un modelo de aprendizaje profundo con el objetivo de automatizar el análisis de señales de electroencefalografía intracraneal del registro diario capturado por el sistema RNS. Con este fin se utilizaron espectrogramas construidos a partir de las señales de iEEG. De esta manera, información tanto del dominio temporal como del frecuencial es conjuntamente extraída, permitiendo así la posibilidad de encontrar el inicio del episodio epiléptico (variable tiempo) dado por un cambio en la energía de la señal (variable frecuencia).

Los espectrogramas pueden pensarse como representaciones bidimensionales, y de manera análoga a lo que son las imágenes, cada píxel representa una información del espacio tiempo-frecuencia. De esta manera, a fin de investigar las ventajas de redes totalmente convolucionales, este proyecto se basó en la arquitectura UNet, la cual fue originalmente creada para segmentación de imágenes médicas. La misma fue modificada y adaptada para poder procesar los datos y generar las salidas deseadas. Los pruebas de evaluación sobre la complejidad de la red en función de su desempeño (Sección 5.1.4.1) mostraron que es posible disminuir la cantidad de parámetros de la red obteniendo un modelo con mejor capacidad de generalización. Por otro lado, al evaluar los diferentes tipos de máscaras que podían ser aplicadas como etiquetas durante el aprendizaje del modelo (Sección 5.1.4.2) se encontró que el mejor desempeño se halló con una salida del tipo vectorial con función de suavizado Gaussiana. A partir de esto se debieron realizar modificaciones adicionales a modo de permitir que la salida de la red tenga un tamaño vectorial en lugar de un tamaño consistente con los datos de entrada, como es su versión original.

El uso de máscaras como etiquetas para el entrenamiento de la red permitió abor-

dar de manera conjunta la clasificación de las muestras y la estimación del tiempo de inicio ictal. Es por esto que para poder interpretar los resultados de las máscaras predichas se añadió una etapa de post-procesamiento de las mismas. Se encontró que estableciendo un umbralizado de los vectores predichos por la red el desempeño del sistema completo de procesamiento mejoraba, permitiendo obtener clasificaciones y estimaciones del tiempo de inicio con mayor exactitud (Figura 5.10). Es importante considerar que esta etapa se podría haber incluido como una capa adicional en la red, implementando una capa totalmente conectada que generara los valores de clase y tiempo de inicio. Sin embargo, realizar esta evaluación de procesamiento por fuera de la red permitió que la arquitectura se mantuviera completamente convolucional, pudiendo así procesar muestras de cualquier longitud temporal, lo cual resulta de gran valor para el problema estudiado. Esta capacidad del sistema permite que no se deban realizar procesamientos previos de recorte y ajuste de tamaño en los segmentos de iEEG.

Un punto considerable sobre los datos evaluados que debió ser tenido en cuenta durante todo el proyecto fue el desbalance de clases que existía en los mismos. De los 24 pacientes incluidos en la base de datos, sólo unos pocos alcanzaban una proporción de valores positivos cercana al 50 %, siendo el promedio general de positividad del 23 % (Figura 4.2). Este desbalance obligó a utilizar métricas de evaluación robustas en el contexto de conjuntos desbalanceados, como lo son la exactitud balanceada y el F1.

Durante el desarrollo, también se experimentó con la aplicación de distintos métodos para evitar el sobreajuste. Se buscó obtener un modelo que resultara confiable en el caso de evaluarlo con datos desconocidos durante el proceso de ajuste, a modo de que pueda ser considerado como una herramienta de soporte para especialistas en neurología en el análisis y etiquetado de nuevos datos disponibles. Entre estas técnicas para evitar el sobreajuste se incluyeron aumento de datos, aplicación de *weight decay* y elección de modelo de entrenamiento mediante *early stopping* (Figura 5.14 y Figura 5.15). Específicamente dentro de las técnicas de aumento se optó por el uso de técnicas específicas para el análisis de espectrogramas.

Una vez encontrado un modelo "óptimo", se pasó a evaluar su desempeño en datos nunca vistos durante el proceso de ajuste y selección de hiperparámetros. Se encontró que en un conjunto de testeo, contemplado por datos de 4 pacientes, la exactitud balanceada fue aproximadamente del 88 % y el F1 del 82 %, con valores de sensibilidad, especificidad y precisión del 84 %, 91 % y 80 % respectivamente. Además se obtuvo un error absoluto medio de 4.9 segundos para la estimación del tiempo de inicio ictal, lo cual es un tiempo aceptable para la aplicación prevista del método (Tabla 6.1). Esto permitió verificar que el modelo no se hubiera sobreajustado a los datos de entrenamiento. Los 4 pacientes de testeo pertenecían a distintos grupos de localización de los electrodos implantados, por lo que se pudo descartar parcialmente la posibilidad de que se dieran cambios en el desempeño en pacientes con diferentes sitios de implantación.

Para evaluar la capacidad real de generalización del modelo elegido se llevó a cabo una validación cruzada del tipo *Leave One Subject Out*. En esta validación se buscó entrenar la red seleccionada múltiples veces, dejando afuera de los datos de entrenamiento un paciente por vez. De esta forma se evaluó la imparcialidad de los resultados

alcanzados debida a los datos utilizados para entrenamiento. En esta evaluación se obtuvo un valor de BACC global de 81.3 ± 10.0 %, con un F1 de 74.6 ± 10.2 % y un MAE de 6.2 ± 5.1 segundos. A estas métricas se les sumó una precisión de 74.7 ± 10.5 %, sensibilidad de 76.2 ± 19.4 % y especificidad 86.3 ± 11.9 %. En esta evaluación se encontró que el rendimiento empeoraba en los casos en que el paciente de testeo contaba con una gran cantidad de datos, y con un porcentaje de positividad mayor al 50 % (Figura 6.3). A partir de los resultados alcanzados se encontró que al entrenar el modelo elegido excluyendo pacientes con una proporción de positividad alta, la red obtiene una menor cantidad de muestras positivas durante el entrenamiento, y entonces luego no es capaz de reconocerlas correctamente en un paciente nuevo. El análisis realizado en este caso permitió concluir que la arquitectura elegida puede resultar sensible a los datos utilizados para entrenar, pero que en la mayoría de los casos es capaz de aprender los patrones de identificación ictales.

Los últimos experimentos realizados con el modelo fueron aquellos relacionados con la evaluación de datos de duración variable. Una de las razones más importantes por la que un modelo se desarrolla de manera completamente convolucional reside en la capacidad de las mismas para analizar datos de diferentes tamaños, al contrario de aquellas redes con capas completamente conectadas. Esta capacidad permitió analizar las muestras de electroencefalografía con su duración original, evitando tener que recortar o aplicar *padding* a los datos de forma previa a la construcción del espectrograma, obligando a que todos tengan la duración estándar de 90 segundos. Se encontró en distintos experimentos realizados que los resultados de las evaluaciones con los datos de duración original no difieren significativamente de aquellos en los que se evaluó con los espectrogramas de duración estándar. La mayoría de las evaluaciones resultaron poco afectadas, con una diferencia máxima del 17 % entre métricas sobre dato de duración real y dato de duración estándar, a excepción del cálculo del MAE (Figura 6.7). Las mayores diferencias en el desempeño se encontraron con los datos de duraciones largas, mayores a los 90 segundos, específicamente sobre el valor del MAE, por lo que debería considerarse que en la evaluación de estos datos el desempeño empeora (Figura 6.6). Esto permitió llegar a la conclusión de que podría ser posible realizar evaluaciones sobre datos de duración variable, específicamente sobre datos de duración menor a los 90 segundos, y que también podría ser posible entrenar este tipo de arquitecturas sin la necesidad de estandarizar el tamaño de los datos. Evitando el agregado de columnas con información falsa durante la preparación de los datos permitiría al modelo aprender y evaluar los patrones en un contexto clínico real.

Por último, se llevaron a cabo experimentos con algoritmos de aprendizaje automático clásico y se investigaron distintos modelos de la bibliografía, ambas tareas con el objetivo de establecer un punto de base para comparar los resultados alcanzados con el modelo UNet aquí desarrollado. Con respecto al algoritmo de aprendizaje automático se encontró que el modelo UNet permite un mejor desempeño en todos los sujetos, y además logra una mejor localización del punto de inicio de los patrones ictales (Sección 6.4.3). Además de las claras ventajas con respecto al desempeño del modelo, es importante resaltar aquí que con las técnicas de DL no es necesario realizar cálculos previos para extraer características de los datos, permitiendo que la red aprenda la mejor representación de características. Eso confiere una ventaja desde el punto de vista de tiempo asociado a la ingeniería de selección de características.

Con respecto a la comparación con la bibliografía, se encontró que los resultados alcanzados en este proyecto son comparables con los del estado del arte actual. En este relevamiento se encontró que es posible alcanzar valores de BACC o ACC mayores al 90 %, los cuales resultan superiores al 81 % alcanzado en este proyecto. Sin embargo se observó también que esta exactitud no siempre se encontraba acompañada de valores de especificidad, sensibilidad o precisión acordes, como es el caso de Gomez C. [31], donde se alcanzó una sensibilidad y precisión menor al 53 %. Además el valor de error en el cálculo del tiempo de inicio ictal alcanzado es comparable con el valor alcanzado en el trabajo de Constantino A. (6.2 segundos vs. 6.3 segundos) (Tabla 6.5). Con este relevamiento se logró observar distintos acercamientos al problema que existen en el estado del arte, enmarcados con diferentes enfoques de DL así como del uso de los datos. En varios de estos trabajos se observa que la evaluación de señales utilizando el dominio frecuencial o el dominio tiempo-frecuencia arroja mayor información que cuando se utiliza únicamente el dominio temporal. Esto permite validar la intuición inicial de que el uso de espectrogramas en este contexto puede arrojar resultados superiores que con el uso de señales en el dominio temporal. Evaluando la cantidad de pacientes utilizados en otros trabajos, se observa que a pesar de contar con una base de datos chica, de tan solo 24 pacientes, se alcanzaron resultados de detección de patrones ictales prometedores para el tipo de arquitectura propuesta.

7.1.1. Trabajos futuros

Con el trabajo realizado se logró alcanzar los objetivos planteados para el desarrollo de este proyecto. Sin embargo, a partir de las limitaciones encontradas durante su desarrollo se presentan distintas propuestas para continuar con la investigación realizada.

En primer lugar, en el proyecto no se realizó ninguna modificación sobre la forma de generar los espectrogramas, sino que se trabajó con los mismos ya generados, tomándolos como datos. A partir de esto, se propone realizar una evaluación del impacto de la limpieza de artefactos sobre los datos de forma previa a la generación de espectrogramas, buscando con esto mejorar la capacidad de la red de reconocer los patrones de interés.

En el alcance del proyecto no se incluyó el análisis en profundidad de la relevancia del lugar de implantación de los electrodos del neuroestimulador en el proceso de clasificación de las muestras. Es por esto que se propone además realizar un análisis en profundidad de la capacidad del sistema de clasificar y estimar el punto de inicio ictal en pacientes que pertenezcan a distintos grupos según su patología. Esta evaluación permitiría dar una idea de la utilidad clínica del sistema en distintos contextos.

En tercer lugar, en este proyecto se realizó una evaluación de los resultados del procesamiento planteados sobre los datos disponibles con su duración original. Sin embargo, esta evaluación se realizó sobre un modelo entrenado con muestras de duración normalizada a 90 segundos. Para un trabajo futuro se propone realizar un entrenamiento del modelo con los datos de duración original, buscando de esta manera que el mismo aprenda sobre una base verídica, y pueda así reconocer los patrones buscado sobre las señales reales. Esto permitiría además que el desbalance entre las muestras

positivas y negativas sea menor, dado que para generar los espectrogramas de 90 segundos de duración se realizó una búsqueda en la que se debieron recortar las muestras de duración larga, generando dos o más a partir de cada una. Por ejemplo, por cada muestra positiva de duración larga se pudieron generar 2, una positiva, con el segmento que contenía el patrón, y una negativa, con el segmento restante.

Por otro lado, a partir del proyecto se plantea la escritura de un trabajo científico. Además, se realizará una presentación de los resultados alcanzados en el laboratorio que permitió el uso de los datos, buscando aportar a su línea de investigación.

7.1.2. Relevancia del trabajo realizado

Esta investigación permitió observar que es posible realizar en simultáneo dos tareas que resultan de interés sobre datos de electroencefalografía de pacientes epilépticos, pudiendo en una misma evaluación obtener la clase de la muestra y el tiempo de inicio ictal. Además, el trabajo realizado plantea las bases para poder aplicar el uso de redes completamente convolucionales sobre señales de electroencefalografía intracraneal. La importancia de esto se basa sobre la posibilidad de realizar este tipo de análisis sobre señales de distinta duración, permitiendo evaluar los datos disponibles sin necesidad de llevar a cabo tareas de recorte previas.

7.2. Conclusiones

En el presente proyecto se logró cumplir con el objetivo propuesto originalmente, que consistía en desarrollar un modelo a partir de una red totalmente convolucional basada en la arquitectura UNet, para automatizar la detección de patrones ictales en señales de electroencefalografía intracraneal tomadas en pacientes con epilepsia con un sistema de neuroestimulación responsiva. Durante el trabajo se investigaron distintas cuestiones relacionadas con la patología epiléptica, evaluando sus bases fisiológicas y su relación con la electroencefalografía. También se analizaron las dificultades que se pueden presentar durante el análisis de este tipo de señales, y se evaluó la capacidad que tienen las técnicas del campo del aprendizaje automático para resolver estas dificultades.

A partir del trabajo realizado se logró obtener un modelo entrenado a partir de la base de datos disponible, utilizando como base una arquitectura de aprendizaje profundo que es generalmente usada para la segmentación de imágenes médicas. Con el modelo entrenado fue posible obtener resultados de eficacia comparables con resultados de la bibliografía, y que resultaron mejores que los alcanzados con técnicas de aprendizaje automático clásico. Lo remarcable de este modelo es su capacidad de evaluar espectrogramas de longitud temporal variable, lo cual permite analizar las señales sin variaciones significativas en la eficacia, debiendo únicamente transformar las mismas al dominio tiempo-frecuencia. Esto puede potencialmente permitir el desarrollo de una herramienta compacta que permita el análisis previo de las señales crudas provenientes del sistema RNS y genere una estimación previa de la presencia y el tiempo

de inicio de un patrón convulsivo de manera confiable, sin la necesidad de realizar grandes transformaciones sobre los datos crudos.

Para poder llevar a cabo el trabajo se debieron superar distintas dificultades, como fueron el aprendizaje de nuevas técnicas de aprendizaje profundo, el trabajo con nuevas librerías y la utilización de métricas específicas. También se debió aprender a trabajar con servidores remotos para acceder a una GPU y se debieron aprender nuevas técnicas para manejar grandes volúmenes de datos, los cuales eran necesarios para desarrollar el proyecto.

Finalmente, el trabajo resultó una posibilidad de incorporar herramientas y conceptos nuevos para la estudiante, los cuales resultaron desafiantes por sus características innovadoras y complejidad técnica. Se considera que estas herramientas resultan potencialmente útiles para la formación en ingeniería biomédica, por su capacidad de ser aplicadas en problemas de complejidad que requieren de análisis de especialistas, como sucede generalmente en el caso de datos médicos. Además el proyecto permitió a la estudiante indagar en distintos métodos de análisis de datos, lo cual, en un contexto de crecimiento de disponibilidad de distintos tipos de datos, resulta esencial para la formación en ingeniería en la actualidad.

Glosario

ACC	*Accuracy*
Adam	*Adaptive Moment Estimation*
BACC	*Balanced Accuracy*
BCE	*Binary Cross Entropy*
CNN	*Convolutional Neural Networks*
CONICET	Consejo Nacional de Investigaciones Científicas y Técnicas
CPU	*Central Processing Unit*
CUDA	*Compute Unified Device Architecture*
DBS	*Deep Brain Stimulation*
DL	*Deep Learning*
EEG	Electroencefalogafía
FDA	*Food and Drug Administration*
FN	*False Negatives*
FP	*False Positives*
FPR	*False Positive Rate*
GD	*Gradient Descent*
GPU	*Graphic Processing Unit*
IA	Inteligencia Artificial
IDE	*Integrated Development Environment*
iEEG	Electroencefalogafía Invasiva
LOSO	*Leave One Subject Out*
MAE	*Mean Squared Error*
ML	*Machine Learning*
PDS	Paroxysmal Depolarizing Shift
ReLU	*Rectified Linear Unit*
RMSProp	*Root Mean Square Propagation*
RNS	*Responsive Neurostimulation System*
ROC	*Receiver Operator Curve*
sinc(i)	Instituto de Investigación en Señales, Sistemas e Inteligencia Computacional
STFT	Short Time Fourier Transform
TanH	*Hyperbolic Tangent*
TN	*True Negatives*
TP	*True Positives*
TPR	*True Positive Rate*

UNL	Universidad Nacional del Litoral
VNS	*Vagus Nerve Stimulation*
WBCE	*Weighted Binary Cross Entropy*
XGBoost	*eXtreme Gradient Boosting*

Bibliografía

[1] Francesco Piccialli y col. «A survey on deep learning in medicine: Why, how and when?» En: *Information Fusion* 66 (2021), págs. 111-137. ISSN: 1566-2535. DOI: https://doi.org/10.1016/j.inffus.2020.09.006. URL: https://www.sciencedirect.com/science/article/pii/S1566253520303651.

[2] Bradley J. Erickson y col. «Machine Learning for Medical Imaging». En: *RadioGraphics* 37.2 (2017). PMID: 28212054, págs. 505-515. DOI: 10.1148/rg.2017160130. URL: https://doi.org/10.1148/rg.2017160130.

[3] Andre Esteva y col. «Deep learning-enabled medical computer vision». En: *Digital Medicine* (2021). DOI: 10.1038/s41746-020-00376-2.

[4] Ryad Zemouri, Noureddine Zerhouni y Daniel Racoceanu. «Deep Learning in the Biomedical Applications: Recent and Future Status». En: *Applied Sciences* 9.8 (2019). DOI: 10.3390/app9081526. URL: https://www.mdpi.com/2076-3417/9/8/1526.

[5] Ezequiel Alvarez, Federico Lamagna y Manuel Szewc. «A Machine Learning alternative to placebo-controlled clinical trials upon new diseases: A primer». En: *Quantitative Biology* (2020). DOI: https://doi.org/10.48550/arXiv.2003.12454.

[6] Victoria Peterson y col. «Transfer Learning Based on Optimal Transport for Motor Imagery Brain-Computer Interfaces». En: *IEEE Transactions on Biomedical Engineering* 69.2 (2022), págs. 807-817. DOI: 10.1109/TBME.2021.3105912.

[7] Gregory D. Cascino, Joseph I. Sirven y William O. Tatum. «Epilepsy». En: *John Wiley and Sons* (2021). DOI: 10.1002/9781119431893.

[8] Yannick Roy y col. «Deep learning-based electroencephalography analysis: a systematic review». En: *Journal of Neural Engineering* 16.5 (2019), pág. 051001. DOI: 10.1088/1741-2552/ab260c. URL: https://doi.org/10.1088/1741-2552/ab260c.

[9] Simanto Saha y Mathias Baumert. «Intra- and Inter-subject Variability in EEG-Based Sensorimotor Brain Computer Interface: A Review.» En: *Frontiers in computational neuroscience* (2019). DOI: 10.3389/fncom.2019.00087.

[10] Chengfan Li y Yuenyu Qi. «A Deep Learning Method Approach for Sleep Stage Classification with EEG Spectrogram». En: *Environmental research and public health* (2022). DOI: 10.3390/ijerph19106322.

[11] Alexander C. Constantino y col. «Expert-Level Intracranial Electroencephalogram Ictal Pattern Detection by a Deep Learning Neural Network». En: *Frontiers in Neurology* 12 (2021). ISSN: 1664-2295. DOI: 10.3389/fneur.2021.603868. URL: https://www.frontiersin.org/article/10.3389/fneur.2021.603868.

[12] Bassem Bouaziz y col. «Epileptic Seizure Detection Using a Convolutional Neural Network». En: (2019), págs. 79-86.

[13] Ian Goodfellow, Yoshua Bengio y Aaron Courville. *Deep Learning*. MIT Press, 2016. URL: http://www.deeplearningbook.org.

[14] Olaf Ronneberger, Philipp Fischer y Thomas Brox. «U-Net: Convolutional Networks for Biomedical Image Segmentation». En: *LNCS* (oct. de 2015). DOI: 10.1007/978-3-319-24574-4_28.

[15] Carlos Acevedo y col. *Informe sobre la epilepsia en Latonoamerica*. 2008. URL: https://www.paho.org/hq/dmdocuments/2008/Informe_sobre_epilepsia.pdf.

[16] Organización Mundial de la Salud. *Epilepsia*. 2022. URL: https://www.who.int/es/news-room/fact-sheets/detail/epilepsy.

[17] Carl Stafstrom y Lionel Carmant. «Seizures and epilepsy: an overview for neuroscientists». En: *Cold Spring Harbor perspectives in medicine* (2015). DOI: 10.1101/cshperspect.a022426. URL: https://pubmed.ncbi.nlm.nih.gov/26033084.

[18] Edward B. Bromfield, José E. Cavazos y Joseph I Sirven. *An Introduction to Epilepsy*. American Epilepsy Society, 2006. URL: https://www.ncbi.nlm.nih.gov/books/NBK2510/.

[19] Hai Chen y Mohamas Koubeissi. «Electroencephalography in Epilepsy Evaluation». En: *Continuum* (2019). DOI: 10.1212/CON.0000000000000705.

[20] W.J. Hendelman, P. Humphreys y C.R. Skinner. *The Integrated Nervous System: A Systematic Diagnostic Case-Based Approach*. 2017. DOI: 10.1201/9781315156767.

[21] Liga Argentina contra la Epilepsia. *Tratamiento para la epilepsia*. URL: https://www.lace.org.ar/tratamiento.html.

[22] «Ley 25.404: Establécense medidas especiales de protección para las personas que padecen epilepsia». En: (). URL: https://www.argentina.gob.ar/normativa/nacional/ley-25404-66578/texto.

[23] Beata Jarosiewicz y Martha Morrell. «The RNS System: brain-responsive neurostimulation for the treatment of epilepsy». En: *Expert Review of Medical Devices* 18.2 (2021), págs. 129-138. DOI: 10.1080/17434440.2019.1683445. URL: https://doi.org/10.1080/17434440.2019.1683445.

[24] Tara L. Skarpaas, Beata Jarosiewicz y Martha J. Morrell. «Brain-responsive neurostimulation for epilepsy (RNS® System)». En: *Epilepsy Research* 153 (2019), págs. 68-70. ISSN: 0920-1211. DOI: https://doi.org/10.1016/j.eplepsyres.2019.02.003. URL: https://www.sciencedirect.com/science/article/pii/S0920121118305758.

[25] Evren Burakgazi Dalkilic. «Neurostimulation Devices Used in Treatment of Epilepsy». En: *Epilepsy Research* (2017). DOI: https://doi.org/10.1007/s11940-017-0442-9.

[26] F. T. Sun y M. J. Morrell. «Closed-loop neurostimulation: the clinical experience». En: *Neurotherapeutics* (jul. de 2014). DOI: 10.1007/s13311-014-0280-3.

[27] Graham Winston y col. «Closed-loop vagal nerve stimulation for intractable epilepsy: A single-center experience». En: *Seizure* (2021). DOI: 10.1016/j.seizure.2021.03.030.

[28] Nathaniel D. Sisterson y Vasileios Kokkinos. «Neuromodulation of Epilepsy Networks». En: *Neurosurg Clin N Am* (2020). DOI: 10.1016/j.nec.2020.03.009.

[29] NeuroPace. *RNS System Physician Manual*. URL: https://www.neuropace.com/wp-content/uploads/2021/02/neuropace-rns-system-manual-320.pdf.

[30] NeuroPace. *RNS System Programming Manual*. URL: https://www.neuropace.com/wp-content/uploads/2021/02/neuropace-rns-system-programming-manual.pdf.

[31] Catalina Gómez y col. «Automatic seizure detection based on imaged-EEG signals through fully convolutional networks». En: *Scientific Reports* (2020). DOI: 10.1038/s41598-020-78784-3. URL: https://doi.org/10.1038/s41598-020-78784-3.

[32] Yi Sun y col. «Electroencephalography: Clinical Applications During the Perioperative Period». En: *Frontiers in Medicine* (2020). DOI: 10.3389/fmed.2020.00251.

[33] Song Cui y col. «Learning EEG synchronization patterns for epileptic seizure prediction using bag-of-wave features». En: *Journal of Ambient Intelligence and Humanized Computing* (2018). DOI: 10.1007/s12652-018-1000-3.

[34] Kaveh Samiee, Péter Kovács y Moncef Gabbouj. «Epileptic Seizure Classification of EEG Time-Series Using Rational Discrete Short-Time Fourier Transform». En: *IEEE Transactions on Biomedical Engineering* 62.2 (2015), págs. 541-552. DOI: 10.1109/TBME.2014.2360101.

[35] Alexandros Tzallas, Markos Tsipouras y Dimitrios Fotiadis. «Epileptic Seizure Detection in EEGs Using Time–Frequency Analysis». En: *IEEE transactions on information technology in biomedicine : a publication of the IEEE Engineering in Medicine and Biology Society* 13 (abr. de 2009), págs. 703-10. DOI: 10.1109/TITB.2009.2017939.

[36] C.M. Guerrero, A.M. Trigueros y J.I. Franco. «Time-frequency EEG analysis in epilepsy: What is more suitable?» En: (2005). DOI: 10.1109/ISSPIT.2005.1577096.

[37] Badreddine Mandhouj, Mohamed Ali Cherni y Mounir Sayadi. «An automated classification of EEG signals based on spectrogram and CNN for epilepsy diagnosis». En: *Analog Integrated Circuits and Signal Processing* (2021). DOI: 10.1007/s10470-021-01805-2.

[38] Adrian Rosebrock. *Deep Learning for Computer Vision with Python*. PyImageSearch, 2017.

[39] Chandradeep Bhatt y col. «The state of the art of deep learning models in medical science and their challenges». En: *Multimedia Systems* (2020).

[40] Carlos Arana. *Redes neuronales recurrentes: análisis de los modelos especializados en datos secuenciales*. 2021.

[41] Agnes Lydia y Sagayaraj Francis. «A Survey of Optimization Techniques for Deep Learning Networks». En: (mayo de 2019), págs. 2454-9150. DOI: 10.35291/2454-9150.2019.0100.

[42] Ruo-Yu Sun. «Optimization for Deep Learning: An Overview». En: *Journal of the Operations Research Society of China* (2020). DOI: 10.1007/s40305-020-00309-6.

[43] Sharma Siddharth, Sharma Simone y Anidhya Athaiya. «Activation functions in neural networks». En: *International Journal of Engineering Applied Sciences and Technology* (2020). ISSN: 2455-2143.

[44] Aston Zhang y col. *Dive into Deep Learning*. 2021. DOI: 10.48550/ARXIV.2106.11342. URL: https://arxiv.org/abs/2106.11342.

[45] Ilya Loshchilov y Frank Hutter. *Fixing Weight Decay Regularization in Adam*. 2018. URL: https://openreview.net/forum?id=rk6qdGgCZ.

[46] Sergey Ioffe y Christian Szegedy. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. 2015. DOI: 10.48550/ARXIV.1502.03167. URL: https://arxiv.org/abs/1502.03167.

[47] Connor Shorten y Taghi M. Khoshgoftaar. «A survey on Image Data Augmentation for Deep Learning». En: *Journal of Big Data* (2019). DOI: 10.1186/s40537-019-0197-0.

[48] Yamashita Rikiya y col. «Convolutional neural networks: an overview and application in radiology». En: *Insights into Imaging* (2018). DOI: 10.1007/s13244-018-0639-9.

[49] *Analyses of Deep Learning*. 2019. URL: https://stats385.github.io/weightlayers.

[50] *Umpsampling operations*. URL: https://pytorch.org/docs/stable/generated/torch.nn.Upsample.html.

[51] Asgari Taghanaki Saeid y col. «Deep semantic segmentation of natural and medical images: a review». En: *Artificial Intelligence Review* (2021). ISSN: 1573-7462. DOI: 10.1007/s10462-020-09854-1.

[52] Minaee Shervin y col. «Image Segmentation Using Deep Learning: A Survey». En: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44.7 (2022), págs. 3523-3542. DOI: 10.1109/TPAMI.2021.3059968.

[53] Nahian Siddique y col. «U-Net and Its Variants for Medical Image Segmentation: A Review of Theory and Applications». En: *IEEE Access* (2021). DOI: 10.1109/access.2021.3086020. URL: https://doi.org/10.11092Faccess.2021.3086020.

[54] Shruti Jadon. «A survey of loss functions for semantic segmentation». En: (2020). DOI: 10.1109/CIBCB48159.2020.9277638.

[55] *Pytorch - BCE with Logits Loss*. URL: https://pytorch.org/docs/stable/generated/torch.nn.BCEWithLogitsLoss.html.

[56] Nvidia. *Familia GeForce RTX 3090*. URL: https://www.nvidia.com/es-la/geforce/graphics-cards/30-series/rtx-3090-3090ti/.

[57] *Anaconda Documentation*. URL: https://docs.anaconda.com/.

[58] *Pytorch*. URL: https://pytorch.org/.

[59] *Pytorch GPU*. URL: https://www.run.ai/guides/gpu-deep-learning/pytorch-gpu.

[60] Marina Sokolova y Guy Lapalme. «A systematic analysis of performance measures for classification tasks». En: *Information Processing and Management* (2009). ISSN: 0306-4573. DOI: https://doi.org/10.1016/j.ipm.2009.03.002. URL: https://www.sciencedirect.com/science/article/pii/S0306457309000259.

[61] Karimolla Hajian-Tilaki. «Receiver Operating Characteristic (ROC) Curve Analysis for Medical Diagnostic Test Evaluation». En: *Caspian journal of internal medicine* (2013).

[62] Stephanie Glen. *ROC Curve Explained in One Picture*. URL: https://www.datasciencecentral.com/roc-curve-explained-in-one-picture/.

[63] Charley Gros, Andreanne Lemay y Julien Cohen-Adad. «SoftSeg: Advantages of soft versus binary training for image segmentation». En: *Medical Image Analysis* 71 (2021), pág. 102038. ISSN: 1361-8415. DOI: https://doi.org/10.1016/j.media.2021.102038. URL: https://www.sciencedirect.com/science/article/pii/S1361841521000840.

[64] Tianqi Chen y Carlos Guestrin. «XGBoost: A Scalable Tree Boosting System». En: (2016). DOI: 10.1145/2939672.2939785.

[65] Z. Zhang y K. K. Parhi. «Low-complexity seizure prediction from iEEG/sEEG using spectral power and ratios of spectral power.» En: *IEEE transactions on biomedical circuits and systems* (2015).

[66] M. K. Siddiqui, M. Z. Islam y M. A. Kabir. «A novel quick seizure detection and localization through brain data mining on ECoG dataset». En: *Neural Computing and Applications* (2019).

[67] J. Craley y col. «Automated inter-patient seizure detection using multichannel Convolutional and Recurrent Neural Networks». En: *Biomedical Signal Processing and Control* (2021).

[68] L. Wei y C. Mooney. «Epileptic Seizure Detection in Clinical EEGs Using an XGboost-based Method». En: (2020). DOI: 10.1109/SPMB50085.2020.9353625.

[69] M Zhou y col. «Epileptic Seizure Detection Based on EEG Signals and CNN». En: *Frontiers in neuroinformatics* (2018). DOI: 10.3389/fninf.2018.00095.

[70] W Barry y col. «A High Accuracy Electrographic Seizure Classifier Trained Using Semi-Supervised Labeling Applied to a Large Spectrogram Dataset». En: *Frontiers in neuroscience* (2021). DOI: 10.3389/fnins.2021.667373.

[71] Kaiming He y col. «Deep Residual Learning for Image Recognition». En: *CoRR* abs/1512.03385 (2015). arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385.

[72] Clifton D. Sutton. *Classification and Regression Trees, Bagging, and Boosting*. Vol. 24. Elsevier, 2005. DOI: https://doi.org/10.1016/S0169-7161(04)24011-1. URL: https://www.sciencedirect.com/science/article/pii/S0169716104240111.

[73] Ander Fernández Jauregui. *Cómo programar un árbol de decisión en Python desde 0*.

[74] John Jair Quiza Montealegre. *Aplicación de modelos de deep learning y machine learning para el pronostico de la serie de tiempo del par de divisas euro-dolar*. 2022.

Apéndice A

Descripción de arquitectura utilizada

En la figura B.1 se describen las capas involucradas en la arquitectura de red utilizada en el proyecto. En la primera columna se indica el número de capa y en la segunda se describe el tipo de capa. Luego, en aquellas capas que involucran algún tipo de filtro se indica el tamaño del mismo, considerando que en los filtros cuadrados se indica un único valor.

En las columnas *number of filters, height* y *width* se indica el tamaño de las salidas que resultan de cada capa. Se indica en primer lugar la cantidad de mapas de segmentación y luego el alto y ancho de cada mapa. En la última capa se indica el número de parámetros entrenables de cada capa, el cual es calculado según el tipo de capa, el tamaño de los datos de entrada y el tamaño de los datos de salida. Los cálculos de cantidad de parametros se realizaron siguiendo la ecuación correspondiente a cada capa, indicadas en la Tabla A.1. El cálculo fue corroborado con las herramientas de la librería torch.

Tabla A.1

Cálculo del número de parámetros entrenables según tipo de capa

Tipo de capa	Expresión
Convolucional	$N = [(mnd) + 1]k$ donde m es el alto del filtro, n el ancho del filtro, d el numero de filtros en entrada y k el numero de filtros en la salida
Maxpool	$N = 0$ ya que se realiza una cuenta genérica
Batch norm	$N = 2k$ donde k es el número de filtros
ReLU	$N = 0$
Upsample	$N = 0$

En total el modelo final contó con 14 capas convoluciones de kernel $3x3$ con las correspondientes 14 capas de normalización por batch y las 14 capas de activación ReLU. Por otro lado se utilizaron 3 capas de *max pooling* y 3 capas de *upsample*.

Hacia el final de la red se utilizó una capa convolucional $1x1$ para generar un solo mapeo y una capa adicional con kernel $120x1$ para generar el vector a partir del mapa final.

Camino	Nro	Capa	Tamaño del filtro	Cant. de filtros	Alto	Ancho	Cant. de parametros
Encoder	**1**	**Entrada**	**3x3**	**4**	**120**	**184**	**0**
	2	Convolutional 3x3	3x3	64	120	184	2368
	3	Batch norm	-	64	120	184	128
	4	ReLU	-	64	120	184	0
	5	Convolutional 3x3	3x3	64	120	184	36928
	6	Batch norm	-	64	120	184	128
	7	ReLU	-	64	120	184	0
	8	Maxpooling	2x2	64	60	92	0
	9	Convolutional 3x3	3x3	128	60	92	73856
	10	Batch norm	-	128	60	92	256
	11	ReLU	-	128	60	92	0
	12	Convolutional 3x3	3x3	128	60	92	147584
	13	Batch norm	-	128	60	92	256
	14	ReLU	-	128	60	92	0
	15	Maxpooling	2x2	128	30	46	0
	16	Convolutional 3x3	3x3	256	30	46	295168
	17	Batch norm	-	256	30	46	512
	18	ReLU	-	256	30	46	0
	19	Convolutional 3x3	3x3	256	30	46	590080
	20	Batch norm	-	256	30	46	512
	21	ReLU	-	256	30	46	0
	22	Maxpooling	2x2	256	15	23	0
Medio	23	Convolutional 3x3	3x3	512	15	23	1180160
	24	Batch norm	-	512	15	23	1024
	25	ReLU	-	512	15	23	0
	26	Convolutional 3x3	3x3	512	15	23	2359808
	27	Batch norm	-	512	15	23	1024
	28	ReLU	-	512	15	23	0
Decoder	29	Upsample	2x2	512	30	46	0
	30	Convolutional 3x3	3x3	256	30	46	1769728
	31	Batch norm	-	256	30	46	512
	32	ReLU	-	256	30	46	0
	33	Convolutional 3x3	3	256	30	46	590080
	34	Batch norm	-	256	30	46	512
	35	ReLU	-	256	30	46	0
	36	Upsample	2x2	256	60	92	0
	37	Convolutional 3x3	3x3	128	60	92	442496
	38	Batch norm	-	128	60	92	256
	39	ReLU	-	128	60	92	0
	40	Convolutional 3x3	3x3	128	60	92	147584
	41	Batch norm	-	128	60	92	256
	42	ReLU	-	128	60	92	0
	43	Upsample	2x2	128	120	184	0
	44	Convolutional 3x3	3x3	64	120	184	110656
	45	Batch norm	-	64	120	184	128
	46	ReLU	-	64	120	184	0
	47	Convolutional 3x3	3x3	64	120	184	36928
	48	Batch norm	-	64	120	184	128
	49	ReLU	-	64	120	184	0
	50	Convolucional 1x1	1x1	1	120	184	65
Out	51	Convolucional 120x1	120x1	1	1	184	121
	52	**Salida**		**1**	**1**	**184**	**0**
Suma de parametros total							**7789242**

Capa	Cantidad
Convolutional 3x3	14
Batch norm	14
ReLU	14
Maxpooling	3
Upsample	3
Convolucional 1x1	1
Convolucional 120x1	1

Figura A.1: *Capas involucradas en la arquitectura UNet*

Apéndice B

Descripción de los datos

Los datos se dividieron en 3 grupos, ubicando todos los datos de un mismo paciente dentro un mismo grupo, de forma de no generar resultados imparciales afectados por la mezcla de datos en los conjuntos.

La Tabla B.1 muestra los pacientes que se seleccionaron para entrenar, indicando la cantidad de muestras y el porcentaje de positividad en los datos de cada sujeto, pero también indicando el grupo clínico al que pertenecían según la clasificación de su patología y la posición en la que se colocaron los electrodos del sistema RNS para su tratamiento. En la Tabla B.2 se pueden observar los mismos datos para los sujetos que se separaron para realizar la validación del modelo.

Tabla B.1

Datos de los sujetos de entrenamiento

ID	Grupo	Cantidad de archivos	% de positividad
RNS1440	Neocortex	3077	25.9
RNS1529	Hippocampus	1255	35.2
RNS1597	Hippocampus	2433	36.3
RNS1703	Neocortex	1159	34.2
RNS1836	Migrational Dysplasia	2477	53.4
RNS2227	Hippocampus	422	10.7
RNS2368	Hippocampus - Neocortex	2460	6.7
RNS2938	Neocortex	1085	4.3
RNS4098	Neocortex	3709	53.2
RNS6806	Neocortex	1427	3.7
RNS6992	Neocortex	950	43.1
RNS8076	Neocortex	577	4.0
RNS8163	Neocortex	689	49.6
RNS9183	Hippocampus	386	31.1
RNS9536	Migrational Dysplasia	1942	13.2

Por último, en la Tabla B.3 se pueden observar los pacientes separados para testear el modelo y evaluar el error de generalización del modelo entrenado. Se puede observar

Tabla B.2

Datos de los sujetos de validación

ID	Grupo	Cantidad de archivos	% de positividad
RNS1534	Hippocampus	1081	12.1
RNS1556	Migrational Dysplasia	657	5.5
RNS1603	Neocortex	1808	16.1
RNS2543	Hippocampus	1805	12.2
RNS6989	Hippocampus - Neocortex	1569	2.9

que todos los pacientes pertenecen a distintos grupos clínicos y que poseen distintas cantidades de muestras y de porcentajes de positividad, de modo de poder evaluar el desempeño del modelo en conjuntos de datos con distintos contextos.

Tabla B.3

Datos de los sujetos de testeo

ID	Grupo	Cantidad de archivos	% de positividad
RNS1090	Migrational Dysplasia	1567	17.1
RNS1438	Hippocampus - Neocortex	2147	1.2
RNS7525	Neocortex	360	36.4
RNS8973	Hippocampus	1251	21.7

Apéndice C

Bases teóricas del algoritmo XGBoost

Un árbol de decisión es un algoritmo no paramétrico de aprendizaje supervisado, el cual puede ser utilizado tanto para tareas de clasificación y regresión. Tiene una estructura de árbol jerárquico, comenzando con un nodo base, a partir del cual comienzan a surgir ramas que alimentan nodos internos, también conocidos como nodos de decisión. En estos nodos se toman decisiones, las cuales llevan a otros nodos internos o finalmente a nodos de hoja, en los que se representan las posibles salidas del modelo generado. Este tipo de estructura permite visualizar fácilmente el proceso de toma de decisión al clasificar cada dato evaluado [72]. En el proceso de entrenamiento de estos modelos el algoritmo realiza una búsqueda exhaustiva para identificar los puntos de decisión óptimos en un árbol. El proceso de división luego es repetido de manera recursiva hasta que todos, o la mayoría de los datos evaluados, hayan sido clasificados en una clase específica.

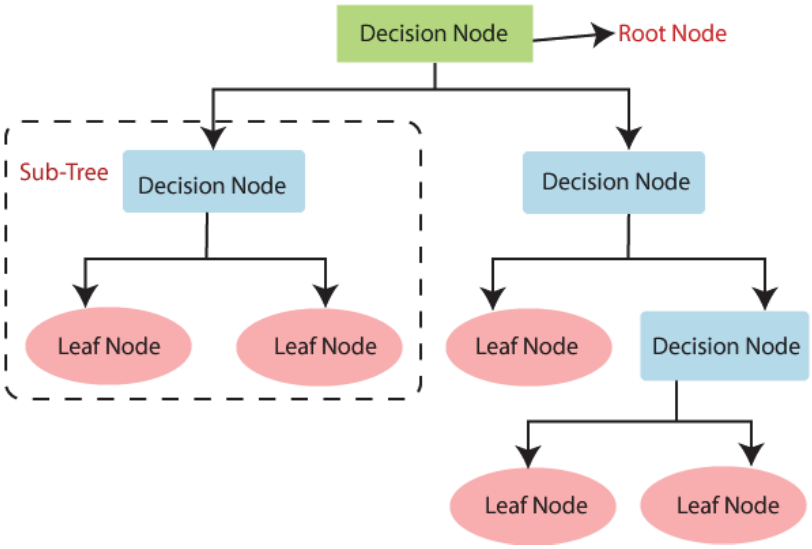


Figura C.1: *Algoritmos con esquema de árboles de decisión. Extraída de [73]*

El *Bagging* y el *Boosting* son dos técnicas generales que permiten mejorar las reglas de predicción, y las mismas pueden ser aplicadas a métodos basados en árboles para mejorar la precisión de los resultados predichos.

El *bagging* es una técnica que puede ser utilizada para reducir la varianza asociada con las predicciones, y así mejorar el proceso predictivo. La forma de procesar los datos consiste en utilizar distintas muestras de forma que cada una de ellas permita entrenar un modelo. De este modo tendremos tantas predicciones como modelos y por tanto tantas predicciones como muestras de entrenamiento. El procedimiento consistente en promediar todas las predicciones anteriores tiene dos ventajas importantes: simplifica la solución y reduce mucho la varianza [72]. El problema reside en que en la práctica no suele disponerse más que de una única muestra de entrenamiento. Lo que se hace entonces es generar cientos o miles de muestras *bootstrap* a partir de la muestra de entrenamiento, y después utilizar cada una de estas muestras como una muestra de entrenamiento. Este método funciona bien con arboles con mucha profundidad, ya que los mismos tienen mucha variabilidad, por lo que es posible entrenar varios y luego calcular la media de todas las predicciones. Así es como se generan los algoritmos de *Random forest*.

Por otro lado tenemos el *boosting*, el cual es un acercamiento que puede ser usado para incrementar la precisión de modelos, pero que utiliza un promedio pesado de los resultados obtenidos a partir de aplicar un método predictor a varios datos. Además en esta técnica las muestras usadas en cada paso no se toman de igual manera, sino que en cada nueva iteración se toman los casos predichos erróneamente en el paso anterior se utilizan con un mayor peso en el durante el siguiente paso. Esta técnica suele funcionar mejor con aprendices débiles, es decir, árboles de baja profundidad [72].

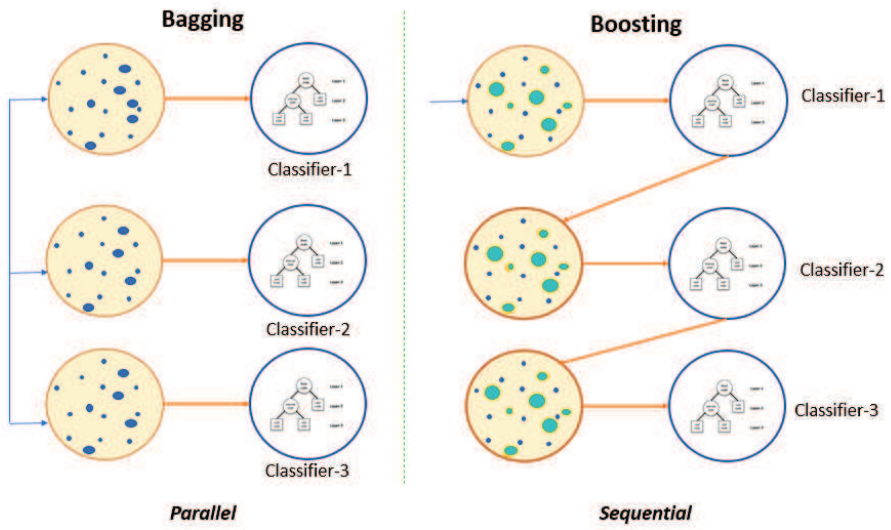


Figura C.2: *Esquema de métodos de bagging y boosting. Extraída de [74]*

A partir del boosting surge el *Gradient boosting*. Este es un método de ensamble que utiliza *boosting*, y es considerado un algoritmo de descenso por gradiente, es decir, un algoritmo con optimización capaz de encontrar soluciones óptimas para una amplia variedad de problemas. La idea general del descenso por gradiente es que se modifican los parámetros de manera iterativa con el objetivo de minimizar una función de costo determinada en cada iteración. En cada iteración el descenso por gradiente mide el gradiente local de la función de costo establecida para un dado conjunto de parámetros y luego modifica los mismos en la dirección del gradiente descendiente.

El algoritmo XGBoost es una extensión del *Gradient boosting* que se ha vuelto muy popular en los últimos años y es hoy en día el método *gold standad* en la aplicación de aprendizaje automático para datos estructurados, por su gran escalabilidad. El mismo fue especialmente diseñado para mejorar la rapidez de entrenamiento y el rendimiento de los modelos resultantes [64] [68].

Esta técnica utiliza métodos de regularización para evitar el sobreajuste de los datos de entrenamiento, lo cual suele ser un problema frecuente en el *gradient boosting*. Además, utiliza métodos de paralelización y organiza los datos en bloques de información para mejorar la capacidad de computo y la velocidad de entrenamiento, permitiendo evaluar conjuntos de datos mas extensos, y así mejorando la capacidad de entrenamiento [64].