



Universidad
Tecnológica
de Pereira

Implementación de un modelo predictivo basado en redes neuronales convolucionales 3D en el paso de deterioro cognitivo leve a Alzheimer sobre imágenes por resonancia magnética

Maria Camila Castaño Martínez

Universidad Tecnológica de Pereira
Facultad de Ingenierías
Pereira, Colombia
2023

Implementación de un modelo predictivo basado en redes neuronales convolucionales 3D en el paso de deterioro cognitivo leve a Alzheimer sobre imágenes por resonancia magnética

Maria Camila Castaño Martínez

Tesis o trabajo de grado presentada(o) como requisito parcial para optar al título de:
Ingeniera física

Director:
Msc. Ing. Jonnatan Arias Garcia

Co-director:
Phd(c). Msc. Ing. Walter Serna Serna

Línea de Investigación:
Visión por computador - Procesamiento de imágenes médicas
Grupo de Investigación en Automática UTP

Universidad Tecnológica de Pereira
Facultad de ingenierías, Ingeniería Física
Pereira, Risaralda, Colombia
2023

A mi mamá y papá, que nunca dudaron. A mis amigos, que siempre me apoyaron. Y a mi pareja Diego, que siempre estuvo ahí para mí. Rodearse de gente maravillosa dió sus frutos. *K Dot.*

Agradecimientos

Quiero agradecer a mis asesores por su constante ayuda y apoyo durante este proceso, la realización de este proyecto fue posible gracias a ellos. A mis amigos, que siempre me apoyaron desde el inicio de este proyecto. En especial a Jhan y a mi pareja, Diego, ya que sin ellos el desarrollo de este modelo no hubiera podido continuar. Gracias por hacer posible esta idea.

Resumen

La enfermedad del *Alzheimer* es un trastorno neurológico que causa la pérdida de autonomía y memoria en las personas que la padecen. Debido al aumento de casos de este padecimiento y la falta de precisión de las herramientas de diagnóstico se da paso al desarrollo de nuevas herramientas capaces de disminuir esta problemática. El objetivo principal de este trabajo investigativo es implementar un modelo de red neuronal convolucional tridimensional con estructura base tipo *AlexNet3D* para obtener la predicción de un posible diagnóstico de la enfermedad *Alzheimer* (AD) a partir del análisis de imágenes por resonancia magnética, utilizando como etapa temprana el síndrome de deterioro cognitivo leve (MCI). Este proyecto brindará la explicación de cada fase planteada, las cuales fueron divididas en selección de las bases de datos, elección de características, procesamiento de los datos, desarrollo del modelo para su entrenamiento y validación, y por último, resultados obtenidos a partir de las pruebas de predicción. Con las cuales pudo obtenerse un porcentaje del 72,222 %, permitiendo catalogar al modelo *K-Net95* como una red estable y eficiente, a pesar de las limitaciones computacionales a las que se vio limitado el proyecto.

Palabras clave: **Alzheimer, Clasificación, Deterioro Cognitivo Leve, Imágenes por Resonancia Magnética, Predicción, Red Neuronal Convolutacional Tridimensional.**

Abstract

Alzheimer's disease is a neurological disorder that causes loss of autonomy and memory in people who suffer from it. Due to the increase in cases of this disease and the lack of accuracy of diagnostic tools, new tools capable of addressing this issue are being developed. The main objective of this research work is to implement a three-dimensional convolutional neural network model with *AlexNet3D* type base structure to predict the possible diagnosis of *Alzheimer's disease* (AD). from the analysis of magnetic resonance images, using as an early stage mild cognitive impairment syndrome (MCI). The construction phases of the project will be explained, divided into database selection, feature selection, data processing, model development for training and validation, and finally, results obtained from prediction tests. With this it was possible to obtain a percentage of 72,222 %, allowing the *K-Net95* model to be cataloged as a stable and efficient network, despite the computational limitations to which the project was limited.

Keywords: **Alzheimer's Disease, Classification, Magnetic Resonance Imaging, Mild Cognitive Impairment, Prediction, Three-Dimensional Convolutional Neural Network.**

Contenido

Agradecimientos	VII
Resumen	IX
Lista de figuras	XI
Lista de tablas	1
1 Introducción	2
1.0.1 Alzheimer	3
1.0.2 Deterioro Cognitivo Leve	4
1.1 Antecedentes y problemática	5
1.1.1 Pregunta de investigación	6
1.2 Objetivos	7
1.2.1 Objetivo General	7
1.2.2 Objetivos Específicos	7
1.3 Justificación	8
1.3.1 Pertinencia	8
1.3.2 Viabilidad	9
1.3.3 Impacto	9
2 Materiales y Métodos	11
2.1 Bases de datos	12
2.1.1 Imágenes de resonancia magnética	13
2.2 Selección de características	14
2.3 Procesamiento de imágenes	17
2.3.1 Free Surfer	24
2.4 Estructura y configuración del modelo	25
2.4.1 Redes Neuronales Convolucionales 3D	30
2.4.2 Funciones de activación	32
2.4.3 Librerías Usadas de Python	33
2.5 Entrenamiento y validación	35
3 Experimentos y resultados	37
3.1 Modificación de parámetros	37

3.2	K-Net95 vs. otros modelos	44
3.3	Predicción	46
4	Conclusiones y recomendaciones	48
4.1	Conclusiones	48
4.2	Reconocimientos:	49
	Bibliografía	50
5	Anexos	56

Lista de Figuras

2-1	Diagrama de flujo.	11
2-2	Cortes axial, sagital y coronal de un cerebro masculino sano sin procesar.	12
2-3	Cortés axial, sagital y coronal de un paciente de 65 años con <i>MCI</i>	14
2-4	Cortés axial, sagital y coronal de un paciente de 77 años con <i>AD</i>	15
2-5	Cortés axial, sagital y coronal de un paciente sano de 76 años.	15
2-6	Primera resonancia de la transición MCI a MCI.	16
2-7	Primera resonancia de la transición Normal a AD.	16
2-8	Primera resonancia de la transición MCI a AD.	16
2-9	Imágenes ponderadas en T1 y T2 respectivamente.	18
2-10	Interfaz de <i>FreeView</i>	19
2-11	Cerebro procesado con <i>Skull stripping</i>	21
2-12	Comparación entre imágenes procesadas y sin procesar por <i>FreeSurfer</i>	22
2-13	Corte transversal con eliminación del 10 % de los bordes.	23
2-14	Comparación entre imagen procesada y sin procesar por parte de la técnica <i>FastSurfer</i>	24
2-15	Estructura original de la red AlexNet (2018) [1].	26
2-16	Estructura de una red <i>AlexNet 3D</i> [2].	27
3-1	Estructura del modelo <i>K-Net95</i>	43
3-2	Resultados obtenidos del modelo <i>K-Net95</i>	47
5-1	Curvas de aprendizaje de la prueba 1.	65
5-2	Curvas de aprendizaje de la prueba 2.	65
5-3	Curvas de aprendizaje de la prueba 3.	66
5-4	Curvas de aprendizaje de la prueba 4.	66
5-5	Curvas de aprendizaje de la prueba 5.	67
5-6	Curvas de aprendizaje de la prueba 6.	67
5-7	Curvas de aprendizaje de la prueba 7.	68
5-8	Curvas de aprendizaje de la prueba 8.	68
5-9	Curvas de aprendizaje de la prueba 9.	69
5-10	Curvas de aprendizaje de la prueba 10.	69

Lista de Tablas

2-1	Resultados del primer entrenamiento de la red base.	29
2-2	Parámetros obtenidos a partir del primer entrenamiento.	30
3-1	Comparación entre las estructuras planteadas.	44

1 Introducción

Conocido como uno de los principales campos investigativos, el sector médico se ha visto rodeado de desafíos por mejorar técnicas no invasivas y de diagnóstico capaces de brindar una mejor calidad de vida para el paciente, siendo la detección temprana de enfermedades terminales uno de ellos. Como consecuencia, el campo médico ha puesto en curso diversas investigaciones que van de la mano con el campo de la computación automática e ingeniería, permitiendo la optimización de los tratamientos y diagnósticos de estas enfermedades [3,4]. El surgimiento de esta necesidad da paso a nuevas técnicas de diagnóstico que puedan unificar el uso de tecnologías emergentes y el método de estudio clínico no invasivo. Es por esto que en los últimos años, ha ido en aumento la utilización de modelos de inteligencia artificial, siendo esta una ciencia de automatización que busca la creación de sistemas inteligentes capaces de obtener información de su entorno, aprender de este y utilizarle para tomar decisiones [5]. Existen diversas aplicaciones de estos modelos, entre estas el reconocimiento de voz, la identificación de estados de ánimo en texto, el reconocimiento de características en imágenes de texto, la segmentación automática de elementos en una imagen, y hasta el análisis de imágenes médicas [6], es destacable que para el caso del análisis de imagen se utiliza una arquitectura perteneciente al subconjunto de Deep Learning conocida como Red Neuronal Convolutacional (CNN) [7].

La CNN Es un tipo de red artificial clasificadora que puede utilizarse como herramienta de apoyo en el sector clínico, haciendo posible la obtención de un posible diagnóstico a partir de la introducción de datos conocidos como Imágenes de Resonancia Magnética (MRI) el cual es un método no invasivo de visualización del cuerpo humano. Con estas imágenes, es posible poner en práctica un nuevo modo de reconocimiento, capaz de presentar resultados sobre una posible detección de diversos tipos de enfermedades sin necesidad de procedimientos complicados. Se ha demostrado que este tipo de red artificial logra extraer de forma automática características de las imágenes con las cuales fue entrenada sin supervisión humana, razón por la cual es el algoritmo más empleado para la creación de modelos de clasificación y predicción [6, 8]. En la actualidad, son bastante utilizadas como alternativa para la diferenciación de distintas características que envuelven las enfermedades neurodegenerativas, logrando alcanzar un nivel de precisión de hasta el 98 % [9] según investigaciones recientes.

En el campo médico, siendo el cerebro un órgano de estudio complejo, es necesaria la implementación de herramientas que sean de ayuda para entender su funcionalidad y estructura, y con esto, determinar cuáles son los factores presentes en las imágenes que contienen la información necesaria para un posible diagnóstico de diferentes neuro-enfermedades, siendo un ejemplo de estas, la enfermedad conocida como *Alzheimer* (AD).

Esta enfermedad se ha visto rodeada de suposiciones en cuanto a su origen se refiere, lo cual se ha convertido en objeto de estudio por diferentes sectores académicos. Varios de los factores asociados con la aparición y desarrollo de esta enfermedad van desde los cambios ambientales que sufre la persona durante años, la genética familiar o mutaciones, e incluso el estilo de vida. Sin embargo, se ha podido lograr un acercamiento a la respuesta de este interrogante a partir del estudio sobre las etapas de mediana y avanzada, ya que las personas pertenecientes a estos grupos son las más propensas a padecer *Alzheimer*. Este acercamiento da paso a un nuevo planteamiento con el cual será posible lograr una posible detección temprana de esta enfermedad. A este nuevo enfoque se le conoce como síndrome de Deterioro Cognitivo Leve (MCI, por sus siglas en inglés), el cual fue considerado como un posible precursor del *Alzheimer* durante este proyecto, ya que ha podido presentarse como una etapa prematura a partir de su desarrollo en diversos estudios [10–12]. Estos mismos presentan esta posibilidad a través del uso de técnicas de aprendizaje profundo capaces de diferenciar esta etapa de la enfermedad neurodegenerativa definitiva (AD).

El presente proyecto tiene como propósito ofrecer una herramienta de apoyo para la identificación de características ligadas al desarrollo de la neuro-enfermedad para lograr una predicción sobre la progresión del *Alzheimer* a partir del deterioro cognitivo para un potencial diagnóstico, utilizando la mencionada red neuronal convolucional entrenada a partir de imágenes de resonancia magnética hasta lograr el reconocimiento de anomalías poco visibles para el ser humano. Se espera que con esta nueva herramienta, el paciente diagnosticado pueda tener diversas opciones para su tratamiento a futuro y desacelerar el progreso de la enfermedad [13].

1.0.1. Alzheimer

Según el Instituto Mexicano del Seguro Social, el *Alzheimer* es una enfermedad que afecta las células cerebrales conocidas como neuronas, provocando su degeneración de forma progresiva hasta que las mismas mueren, ocasionando la atrofia cerebral (disminución de la masa cerebral). Es la forma más común de demencia entre las personas pertenecientes a la mediana y avanzada edad, causa problemas en la memoria, pensamiento y comportamiento hasta el punto de afectar gravemente la capacidad que tiene una persona de llevar a cabo actividades

ordinarias [14]. Este padecimiento se caracteriza por el deterioro cognitivo y conductual de inicio temprano que sufre el paciente, y curso progresivo de aparición en la edad adulta. Es bien sabido que esta patología neuronal puede comenzar incluso hasta una década antes de que se inicie la sintomatología clínica.

Actualmente, es considerada como la causa más frecuente de demencia neurodegenerativa. Además, existen diversos estudios que presentan variantes tanto biológicas y genéticas como ambientales para explicar la aparición de la enfermedad; sin embargo, aún no es claro y hoy en día un buen punto de apoyo podría ir ligado no directamente evitar la enfermedad, sino que mientras esto se logra, dar un apoyo a aquellos que ya la padecen o que son propensos a padecerla como por ejemplo aquellos con deterioro cognitivo Leve.

1.0.2. Deterioro Cognitivo Leve

El Deterioro Cognitivo Leve (MCI) se conoce como un síndrome ocasionado por procesos patológicos del sistema nervioso central junto con los cambios relacionados con la vejez. Las personas que padecen del deterioro se encuentran ubicados en un estado intermedio entre el envejecimiento normal y la demencia, incluyendo la enfermedad de Alzheimer; poseen un declive cognitivo mayor que el esperado para su edad, pero pueden llevar a cabo actividades cotidianas sin problema [15]. Durante el inicio de este desorden neurocognitivo, se manifiestan problemas de memoria, lenguaje, atención, toma de decisiones y cambios de comportamiento; y que, según estudios, puede ser considerada como una etapa temprana de Alzheimer, ya que aproximadamente el 10-15 % de las personas con MCI desarrollan dicha enfermedad cada año, además siendo esto un riesgo que aumenta con la edad [16].

1.1. Antecedentes y problemática

El *Alzheimer* es conocido como un trastorno neurológico que afecta a más de 30 millones de personas alrededor del mundo y que, según pronósticos, se presentará un aumento de esta cifra que podrá llegar a 46,8 millones de pacientes para 2030 [17]. Este padecimiento comienza a desarrollarse en personas de 30 a 65 años [18], las cuales pueden mostrar desde síntomas leves, como la dificultad para realizar tareas en entornos sociales y tener problemas de organización; hasta síntomas graves, como necesitar asistencia para el cuidado personal, dando lugar a la pérdida de independencia del paciente, y experimentar cambios en las capacidades físicas como el habla.

La problemática se origina por la falta de una detección temprana y precisa con la cual pueda encontrarse un tratamiento preventivo, estableciéndose como una de las principales causas por las cuales esta enfermedad se sitúa como la séptima causa de muerte en el mundo [19]. Generalmente, este diagnóstico se obtiene cuando el paciente ya se encuentra en una de las etapas más avanzadas del trastorno, en donde no es posible la recuperación ni la reformación neuronal debido al desarrollo del *Alzheimer* en años anteriores.

Para poder ampliar en la problemática presentada, es necesario tomar en cuenta las técnicas de diagnóstico tradicionales y realizar un análisis de la evolución que estas han tenido durante los últimos años. Realizando este estudio, será posible tener una vista previa sobre su desarrollo, efectividad y como han podido dar paso a nuevas técnicas innovadoras basadas en métodos convencionales.

Partiendo desde las primeras técnicas presentadas en los años 80 para tener un acercamiento a la aparición del *Alzheimer*, las cuales se basaban en evaluaciones cognitivas que identificaban cambios en la memoria y lenguaje, no otorgaban un diagnóstico concreto que pudieran indicar la presencia de la enfermedad [20]. En términos de técnicas de radiología, en la década de 1980 ya se utilizaban técnicas de imagen como la tomografía computarizada (CT) y la resonancia magnética (MRI) para examinar el estado estructural del cerebro, sin embargo, realizar una examinación de estas imágenes es complicado, ya que las imágenes obtenidas no eran lo suficientemente definidas para realizar una diferenciación entre elementos normales y anormales, además teniendo en cuenta las propias variaciones del cerebro de paciente a paciente.

En la época de los 2000 se introducen nuevas técnicas de neuroimagen, las cuales toman un peso aún más importante que los exámenes cognitivos. Uno de los métodos desarrollados que se siguen utilizando en la actualidad es la resonancia magnética funcional (fMRI), la cual permite un análisis más profundo del comportamiento, evolución y cambios en la ac-

tividad cerebral relacionados con el *Alzheimer* en tiempo real. Durante estos años, se abre paso a una nueva razón del surgimiento de esta enfermedad, conocidos como biomarcadores cerebroespinales. Estos marcadores son conocidos actualmente como el péptido-amiloide y proteína tau, pueden ser detectados mediante pruebas realizadas al líquido cefalorraquídeo obtenido a través de una punción lumbar. El análisis realizado demuestra una disminución del líquido ocasionado por la sobreproducción de estas proteínas [21]; sin embargo, es un método con costos muy por encima de los de una MRI normal y no es necesariamente el que mejor podría caracterizar el neuro-deterioro.

Teniendo en cuenta lo anteriormente explicado, es posible ver problemas significativos en cada método mostrado, resaltando que se presentan dos dificultades principales, las técnicas en cuanto a términos prácticos, no son muy viables, ya que los exámenes necesarios para un análisis completo son invasivos y dolorosos. Por otro lado, se tiene poca precisión de un diagnóstico por la parte de imágenes radiológicas y la calidad que presentan para una evaluación de patrones característicos de la enfermedad. Debido a esto, se han implementado técnicas que faciliten el entendimiento de pruebas cognitivas, análisis de antecedentes médicos, evaluación del estado mental, físico y neurológico. Estas soluciones son presentadas por los científicos durante la época comprendida entre 2012 y 2022, años en los que se ha producido un avance significativo en la detección del *Alzheimer*, mediante el uso de la tecnología, la cual pasa a ser una herramienta fundamental para el análisis de esta enfermedad. Se han desarrollado diversas aplicaciones de técnicas de aprendizaje automático y, evaluación de datos como imágenes estructurales y aumento de biomarcadores [22, 23] que permitan disminuir la problemática que engloba la precisión en la detección temprana y el diagnóstico de esta enfermedad, llegando así a nuestra pregunta de investigación.

1.1.1. Pregunta de investigación

¿Cómo se puede implementar eficazmente un modelo de aprendizaje profundo basado en redes neuronales convolucionales que permita el análisis de enfermedades neurodegenerativas y predicción de la progresión del Deterioro Cognitivo Leve a *Alzheimer*, a partir del aprendizaje sobre las variaciones de formas cerebrales no rígidas vistas en imágenes MRI?

1.2. Objetivos

1.2.1. Objetivo General

Desarrollar un modelo de red neuronal convolucional 3D para el análisis de imágenes de resonancias magnéticas cerebrales que permita cuantificar las variaciones cerebrales frente a la progresión de enfermedades neurodegenerativas.

1.2.2. Objetivos Específicos

1. Diseñar una estrategia de preprocesamiento que permita resaltar la información relevante en una imagen MRI, enfocada a los posibles factores causantes de las neuro-enfermedades de Alzheimer y MCI para la minimización de impurezas e información poco útil de las imágenes médicas.
2. Acondicionar las imágenes de la base de datos ADNI de personas pertenecientes al grupo pertenecientes al rango de edad entre 50 a 80 años, en la etapa de preprocesamiento para su purificación.
3. Implementar una red neuronal convolucional 3D en Python apoyada en AlexNet que permita el procesamiento de imágenes MRI a partir del entrenamiento basado en aprendizaje supervisado, para el análisis de características presentes en enfermedades neurodegenerativas.
4. Evaluar el desempeño red neuronal convolucional 3D para la predicción de una potencial progresión del MCI a Alzheimer, como herramienta para un posible diagnóstico.

1.3. Justificación

1.3.1. Pertinencia

El deterioro cognitivo leve (MCI) se encuentra en un estado intermedio entre el envejecimiento normal y la demencia avanzada, como el *Alzheimer*, dándolo a entender como un indicador de riesgo para predecir la transición hacia esta enfermedad neurodegenerativa u otra forma de demencia [24]. No todas las personas con MCI desarrollarán *Alzheimer*, ya que algunas pueden estabilizarse en su estado actual o incluso revertir sus síntomas. Sin embargo, se estima que aproximadamente el 10-15 % de las personas con MCI desarrollarán *Alzheimer* por año [25].

Los médicos realizan evaluaciones periódicas para monitorear cualquier cambio en el estado cognitivo para ayudar a la identificación de los factores de riesgo para una posible conversión [25]. El diagnóstico y la intervención tempranos son fundamentales para explorar opciones de tratamiento y estrategias de estilo de vida que puedan ayudar a retardar la progresión de la enfermedad. Además, es importante destacar que existen subcategorías de MCI, como el deterioro cognitivo medio temprano y tardío (EMCI, LMCI), donde el EMCI tiene un mayor avance y, por lo tanto, una mayor probabilidad de demencia [26].

El diagnóstico del *Alzheimer* se basa típicamente en el análisis de imágenes médicas que examinan diversas estructuras y actividades cerebrales en comparación con patrones normales de desarrollo [27]. En el campo de la visión por computadora, se han desarrollado diversas metodologías automáticas que se enfocan en esta comparación y buscan proporcionar un diagnóstico temprano o evaluar la probabilidad de transición de MCI a AD [24].

Es notorio que existen diferentes enfoques automáticos para la clasificación del *Alzheimer*. Algunos de ellos analizan el ámbito de las variantes y marcadores genéticos asociados a la enfermedad [28], mientras que otros se basan en técnicas de visión por computadora, como el aprendizaje de patrones característicos en imágenes de resonancia magnética [29, 30]. Sin embargo, algunos de estos métodos de aprendizaje de patrones suelen basarse en un simple corte plasmado en una imagen 2D, lo que puede pasar por alto información estructural relevante del cerebro que no puede ser visible. Por lo tanto, es necesario desarrollar una metodología que reconozca patrones en las estructuras cerebrales y permita cuantificar rasgos comunes en la estructura [31], y posteriormente, identificar aquellas diferencias que puedan estar relacionadas con el *Alzheimer*.

1.3.2. Viabilidad

El apoyo a los adultos mayores, especialmente a aquellos en situación de discapacidad, es un tema recurrente en las políticas estatales y sociales. Es fundamental brindarles un apoyo social integral para mejorar su calidad de vida [32].

La Política Nacional de Vejez y Envejecimiento (2022-2031), presentada por el Ministerio de Salud y Protección Social en 2022, establece parámetros respaldados por los derechos humanos de los adultos mayores. También aborda las determinantes del envejecimiento y la vejez, y destaca la necesidad de proyectos que promuevan un envejecimiento saludable [33]. Además, es importante mencionar que, al tratar con personas con posibles limitaciones mentales, se deben tener en cuenta las políticas relacionadas con las personas discapacitadas, como la ley estatutaria frente a personas con discapacidad de 2013, que establece la necesidad de garantizar los derechos de todas las personas con discapacidad [34]. En este sentido, clínicas como Comfamiliar ofrecen programas de atención a la población discapacitada, que incluyen actividades orientadas a brindar apoyo y fomentar el desarrollo neuro-cognitivo para que estos pacientes puedan llevar una vida plena en la sociedad.

Adicionalmente, grupos de investigación, como los de la Universidad Tecnológica de Pereira (UTP), han desarrollado productos en el campo de las imágenes médicas y sistemas de aprendizaje automático. Estos proyectos de investigación se enfocan en el análisis de sistemas predictivos que garantizan la viabilidad en el desarrollo de nuevas tecnologías basadas en el estado del arte, con respecto a modelos de aprendizaje y reconocimiento cerebral. Estos avances contribuyen al desarrollo del conocimiento en el área [7].

En cuanto a la metodología a utilizar, es importante destacar que existen diversas metodologías validadas en el aprendizaje automático de características cerebrales. Estas metodologías suelen utilizar sistemas de convolución para extraer patrones en imágenes volumétricas. Mediante la integración de capas densas y clasificadores de patrones, se puede establecer una base metodológica para el reconocimiento de patrones cerebrales y la detección de coincidencias en nuevos datos o imágenes en relación con los patrones aprendidos [2, 9]. Esto demuestra que la metodología planteada y utilizada en este proyecto se encuentra en línea con los avances más recientes y ofrece resultados prometedores.

1.3.3. Impacto

Se pretende aportar en el ámbito investigativo y de salud, desarrollando modelos que permitan un análisis volumétrico, de estructura y desarrollo cerebral, los cuales a su vez permitan

la creación de herramientas predictivas basadas en aprendizajes automático para médicos e investigadores, con las cuales puedan tener un apoyo en el diagnóstico temprano y por consiguiente brindar planes terapéuticos para tratamientos adecuados.

También se pretende impactar la población adulta con el objetivo de que estos modelos permitan dar una mejoría en la decisión de tratamientos, enfocándose de forma más directa en la predicción de problemas neuronales y enfermedades neurodegenerativas.

Ahora, desde el punto de vista investigativo en la ingeniería, se pretende implementar una metodología basada en modelos de redes neuronales profundos que cuantifiquen información relevante en las volumetrías cerebrales y dar una visión de coincidencia frente a la enfermedad neurodegenerativa de Alzheimer.

2 Materiales y Métodos

Durante este capítulo, se explicará el desarrollo del proyecto y los elementos utilizados para la construcción de este, partiendo desde la obtención de los datos hasta el entrenamiento del modelo. Se muestra en la siguiente figura 2-1 un diagrama de flujo que de forma explicativa expone el paso a paso realizado durante la investigación.

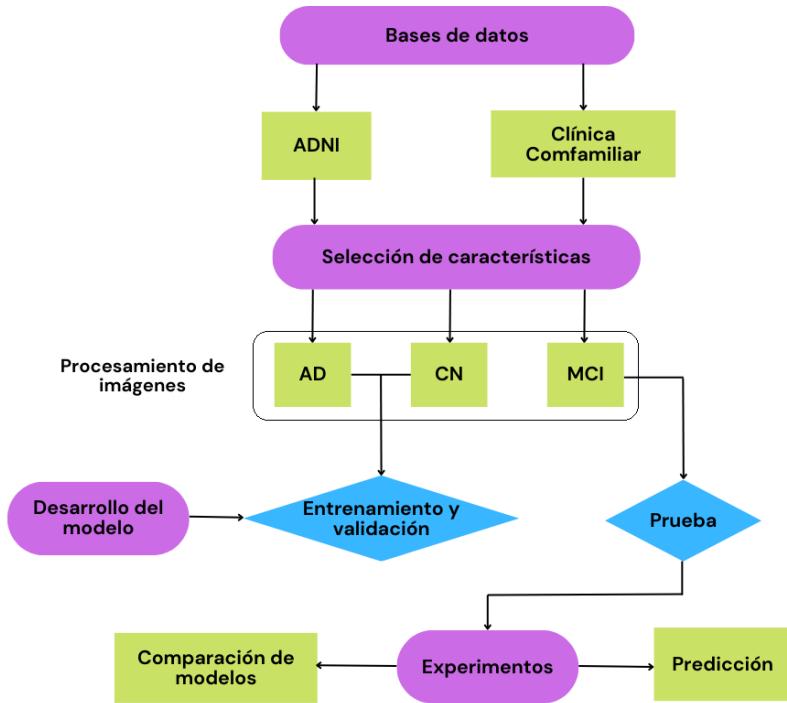


Figura 2-1: Diagrama de flujo.

Notando así que inicialmente se utilizarán datos de la base ADNI [35] para entrenar la red, mientras que la clínica Comfamiliar proporcionará las imágenes para hacer predicciones y *testing*. Esta sección se basará en el procesamiento de imágenes de resonancia magnética, la generación de un modelo capaz de reconocer patrones normales y anormales encontrados en los datos, y posteriormente a través de comparaciones frente a resonancias de MCI, logre percibir variaciones y dar alertas tempranas de la enfermedad. Cabe resaltar que los experimentos planteados irán desde la creación de un modelo base propio, la comparación con modelos del estado del arte y finalmente la predicción de Alzheimer.

2.1. Bases de datos

Para este proyecto se utilizaron imágenes por resonancia magnética (MRI) obtenidas a partir de dos repositorios diferentes, estas serán presentadas en el transcurso de esta sección. Los datos fueron divididos en 3 grupos de estudio, conocidos como datos de entrenamiento, validación y prueba. Ambas bases de datos contienen la información clínica y las neurimágenes de pacientes sanos (Cognitivo Leve), afectados por el *Alzheimer* y el deterioro cognitivo leve (MCI).

Profundizando un poco más acerca del MCI, se planteó esta fase como una posible etapa de transición entre el envejecimiento normal y el padecimiento de una enfermedad como lo es el *Alzheimer*. En diversos estudios [36, 37] se ha demostrado la similitud de síntomas entre estos dos estados, como la dificultad para recordar información reciente y los problemas de memoria a largo plazo. Teniendo en cuenta esta explicación, se utilizará el MCI y sus variantes en el grupo de prueba como una posible fase inicial del *Alzheimer* para este estudio. Se encontrará su aplicación en la sección de *test* del modelo para la predicción.

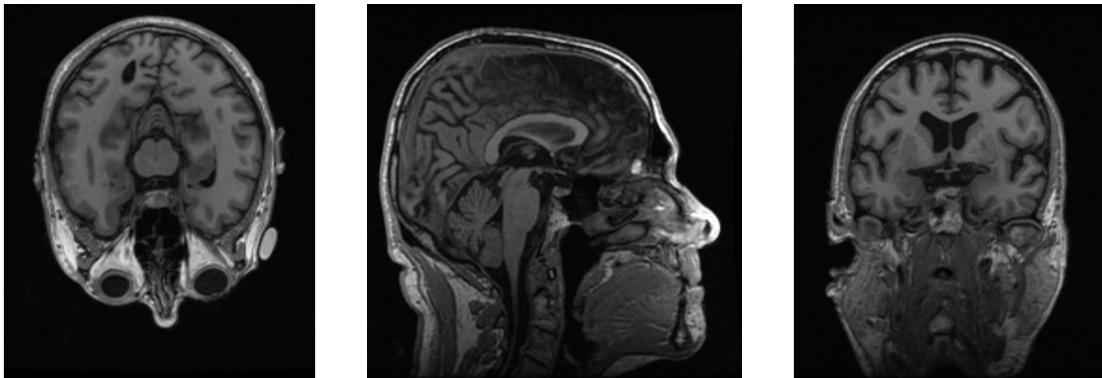


Figura 2-2: Cortes axial, sagital y coronal de un cerebro masculino sano sin procesar.

Las fases que conformarán los dos primeros grupos son *Alzheimer* (AD) y Cognitivo Leve (CN). Estos datos fueron tomados del repositorio libre ofrecido por *Alzheimer's Disease Neuroimaging Initiative* (ADNI) [35]. El repositorio ADNI contiene una gran cantidad de imágenes de resonancia magnética de pacientes entre los 40 a 90 años, también, presentan información relevante recolectada en exámenes médicos. por lo general, las MRI dadas, contienen datos sin procesar (*Original*) que muestran elementos anatómicos que no son necesarios para el entrenamiento de la red (cráneo, nariz, ojos, entre otros). Como ejemplo, en la figura 2-2, puede observarse la imagen por resonancia magnética sin procesar de un paciente masculino sano de 85 años.

El número de paquetes de imágenes que serán utilizados durante el entrenamiento y validación del modelo se especificarán durante esa misma sección, ya que estos datos van a variar dependiendo de las modificaciones que pueda necesitar la red. Se habla de paquetes de imágenes y no de pacientes, ya que estos se sometieron a diversas resonancias magnéticas a lo largo del tiempo, actualizando el estado de la etapa en la que se encontraban anteriormente. Por esta razón se contarán únicamente estos paquetes como datos definitivos de estos dos grupos.

Pasando ahora al grupo de prueba, los paquetes de imágenes que lo componen fueron extraídos de la base de datos suministrada por la Clínica Comfamiliar. Como el trastorno de MCI se utilizará durante la fase de *test*, se optó por una selección de pacientes que cumplieran con esta característica. Es por esto que las imágenes escogidas para el grupo de *test*, contienen los 3 grupos de estudio planteados. Estos datos se especificarán en la siguiente sección

Los datos proporcionados por ambos repositorios son imágenes con información principalmente estructural y texturas del cerebro, a través de píxeles volumétricos o Voxels, que a su vez pueden ser vistos como agrupaciones de imágenes en dos dimensiones para los tres ejes cerebrales (axial, coaxial y sagital). Cabe resaltar que dichas imágenes se encuentran en un formato conocido como *Neuroimaging Informatics Technology Initiative* (NIfTI), el cual presenta paquetes de imágenes en 2D y 3D almacenados en su estructura flexible. Para este proyecto se optó por trabajar con estos arreglos en tercera dimensión, ya que permitieron obtener una mejor perspectiva de las diferentes características estructurales del cerebro y las variaciones a las que están sometidas.

2.1.1. Imágenes de resonancia magnética

La resonancia magnética es conocida por ser una técnica no invasiva que utiliza campos magnéticos y ondas de radio, con el fin de obtener imágenes detalladas del interior del cuerpo humano. Estas imágenes proporcionan información sobre el estado estructural y la función de los tejidos blandos como el cerebro. La implementación de esta técnica para la obtención de imágenes se basa en la creación de un campo magnético homogéneo y estable aplicado en el área de interés. La información capturada se procesa mediante algoritmos informáticos que crean la imagen conocida, permitiendo así, la examinación tanto estructural como funcional de la anatomía corporal [38]. El procedimiento puede generar imágenes volumétricas en 2D y 3D de diferentes órganos vitales de alta calidad y resolución sin la necesidad de utilizar radiación [39].

2.2. Selección de características

Para esta sección, se realizó una selección minuciosa de las características que deben tener las imágenes de entrenamiento y validación, teniendo en cuenta que estos datos fueron obtenidos en una base de datos extensa (ADNI).

La plataforma de la base de datos (ADNI) se encuentra incluida dentro un sistema denominado Archivo de Imágenes y Datos (IDA) [40] proporcionado por el Laboratorio de Neuroimagen (LONI) de la Universidad del Sur de California (USC). Con este recurso es posible tener acceso directo a imágenes cerebrales para su visualización y descarga.

En cuanto a las imágenes de los grupos de entrenamiento y validación, se escogieron las fases *ADNI-1* y *ADNI-2*, ya que son las que contienen mayor cantidad de datos de los grupos elegidos y establecen una estandarización de calidad. Para escoger la formación de estos arreglos 3D, se decidió elegir un límite de 500 imágenes por paquete de cada paciente, esto con el fin de optimizar la capacidad computacional y simplificar el análisis. Se estableció un rango de edad entre los 50 hasta los 80 años para lograr una mejor comparación entre un cerebro sano y enfermo de los grupos de mediana-avanzada edad. Como último filtro, se seleccionó una ponderación de *T1*, la cual maneja un mejor contraste de la estructura cerebral.

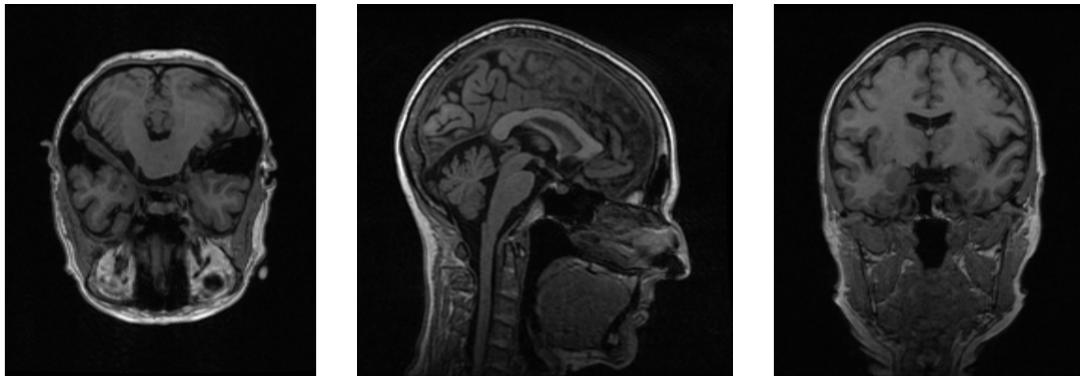


Figura 2-3: Cortés axial, sagital y coronal de un paciente de 65 años con *MCI*.

Las figuras **2-3**, **2-4** y **2-5** presentan las imágenes resultantes, al aplicar los filtros anteriormente mencionados, se tomarán 3 pacientes diferentes, cada uno perteneciente a los grupos de estudio seleccionados entre edades mencionadas y sin importar el género.

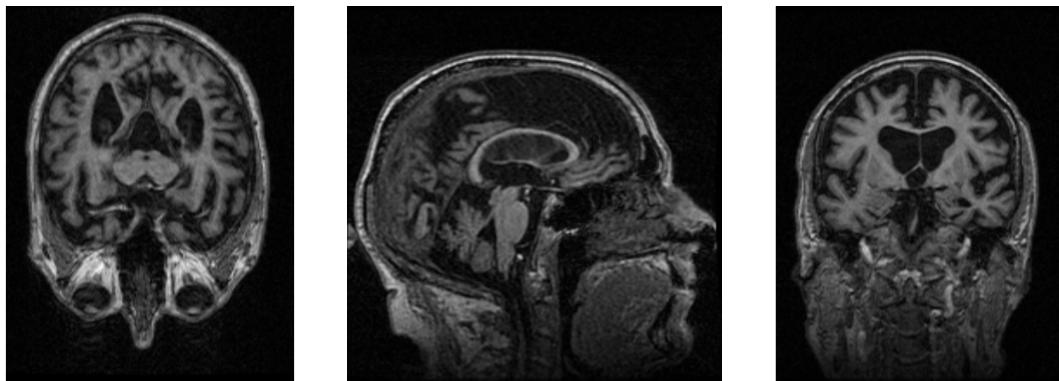


Figura 2-4: Cortés axial, sagital y coronal de un paciente de 77 años con *AD*.

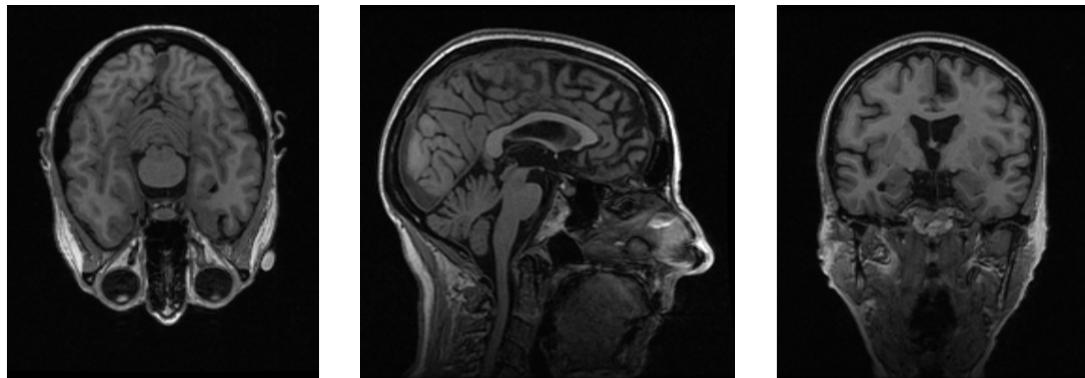


Figura 2-5: Cortés axial, sagital y coronal de un paciente sano de 76 años.

Del repositorio de la clínica Comfamiliar se escogieron 3 pacientes que tenían una edad de 79 años durante su primera resonancia magnética. Estás resonancias se obtuvieron en un período de tiempo comprendido entre 2007 a 2012, durante el cual los pacientes se sometieron a múltiples capturas de imágenes con un intervalo aproximado de un año entre cada una de ellas para poder monitorear cualquier tipo de cambios. Estas imágenes fueron tomadas con una ponderación de *T1*.

Cada paciente representa una transición entre fases seleccionadas, lo cual implica que a medida que se realizaron las tomas de imágenes, se observaron cambios significativos que alteraron su pronóstico. Uno de los pacientes que se diagnosticó como sano, desarrolló *Alzheimer*. Otro paciente pasó de tener deterioro cognitivo leve (MCI) a padecer *Alzheimer*. Y el último paciente no presentó cambios y se mantuvo en la etapa de MCI.

Estas imágenes fueron claves para la comprobación del modelo durante la toma de pruebas finales.

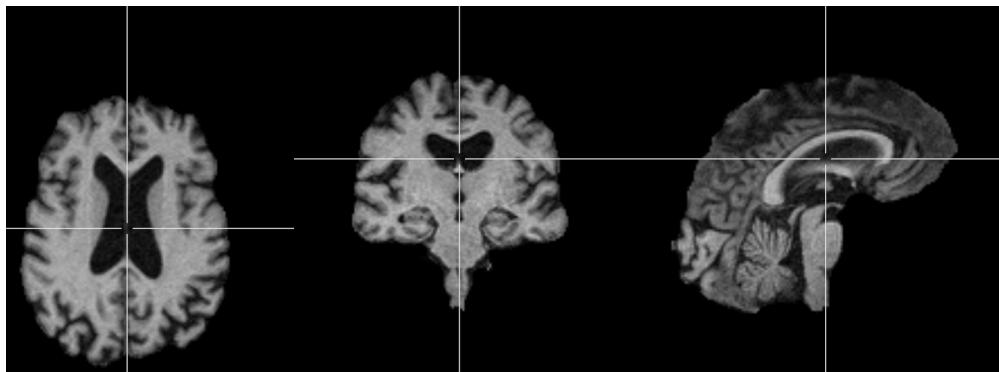


Figura 2-6: Primera resonancia de la transición MCI a MCI.

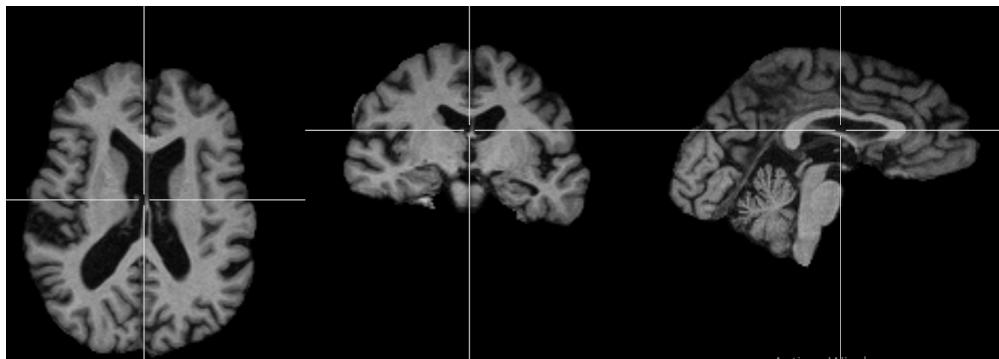


Figura 2-7: Primera resonancia de la transición Normal a AD.

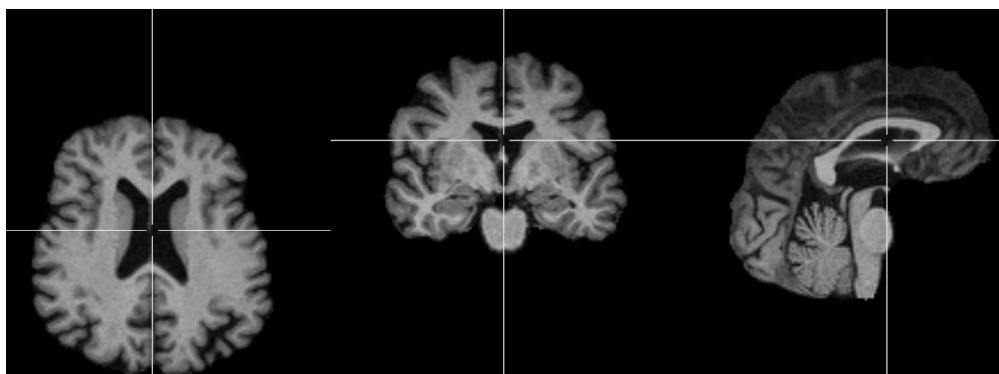


Figura 2-8: Primera resonancia de la transición MCI a AD.

En las figuras **2-6**, **2-7** y **2-8**, pueden observarse los 3 cortes principales de las primeras imágenes tomadas de los 3 pacientes escogidos, a las cuales se les aplicó el procesamiento explicado en la sección 2.3.

2.3. Procesamiento de imágenes

Los objetivos del procesamiento de las imágenes son mejorar su calidad, eliminar elementos innecesarios (cráneo y tejidos blandos) y destacar detalles del cerebro que la red neuronal podrá identificar y aprender con mayor facilidad durante su entrenamiento. Para esta etapa se utilizó *FreeSurfer* [41], un software utilizado para procesar imágenes de resonancia magnética del cerebro y generar modelos tridimensionales de su anatomía.

Este software es gratuito y de código abierto, además, Uno de sus principales usos es la visualización y segmentación del cerebro en estructuras anatómicas como la corteza cerebral, hipocampo y cerebelo. Utiliza algoritmos de procesamiento que incluyen la transformada de Fourier rápida (*FFT*). Contiene funciones de procesamiento que permiten la manipulación de los datos *MRI* como el *Skull stripping*, la cual permite el recorte del cráneo y duramadre del elemento principal que es el cerebro. Para este proyecto, se dispuso de este software para la limpieza de elementos innecesarios, como el cráneo y tejidos blandos, de las imágenes de resonancia magnética.

Como se explicó en la sección 2.1, los datos utilizados durante este proyecto se dividieron en para entrenamiento, prueba y validación. Las imágenes pertenecientes a los grupos de entrenamiento y prueba se sometieron a la técnica ofrecida por *FreeSurfer* para su procesamiento, donde son refinadas en cuanto a contraste y eliminación de elementos sin importancia. Algunas de las imágenes de validación se sometieron a un procedimiento más sencillo conocido como *FastFreesurfer*, una línea de neuroimagen rápida y precisa basada en el aprendizaje profundo [42], implementada para una obtención más acelerada de datos procesados. Cabe aclarar que no se profundizará principalmente en el procesamiento de las imágenes de entrenamiento, ya que se le dio prioridad a la etapa de aprendizaje de la red.

Para el procesamiento de las imágenes, se utilizó el paquete de herramientas de *skull stripping* denominado *SkullStripFix_tktools* como etapa principal para obtener los archivos *brain-mask.mgz*, conocidos como superficie binaria que representa la máscara del cerebro.

En relación con lo mencionado en la sección 2.2, cabe destacar la selección de las imágenes MRI ponderadas en T1 (*T1-weighted*), ya que presentan un mejor contraste entre tejidos (materia gris y materia blanca) permitiendo así diferenciarlos de forma clara y facilitando la detección de anomalías para la red neuronal. Dicho esto, en la figura 2-9 puede apreciarse una comparación entre imágenes que demuestra la razón por la que se escogió la ponderación T1.

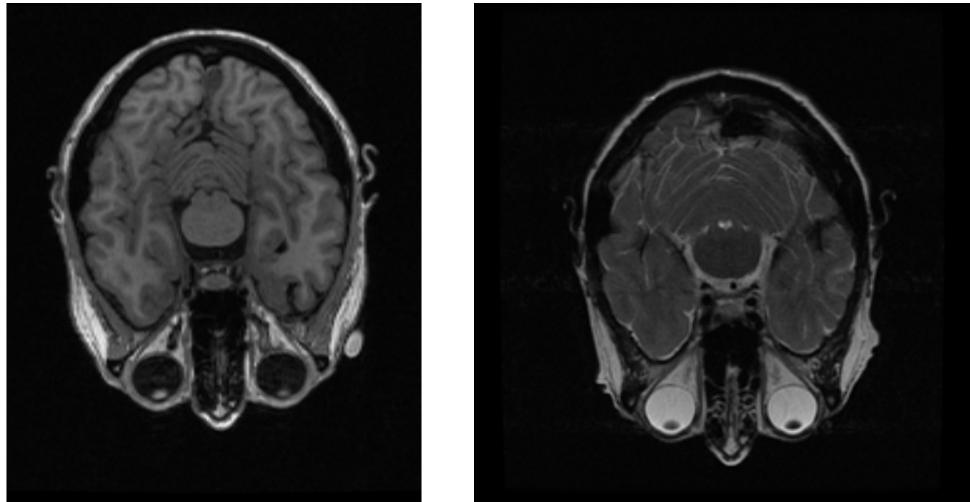


Figura 2-9: Imágenes ponderadas en T1 y T2 respectivamente.

Para la utilización del software, fue necesario aplicarlo sobre el sistema operativo *MacOS* o *Linux* y utilizar la ventana de comandos para correr todas las instrucciones necesarias proporcionadas por *FreeSurfer* [41]. El primer comando permitirá la instalación del tutorial del set de datos de *FreeSurfer* el cual facilitó el procesamiento. Es necesaria la creación de una carpeta llamada **tutorial_data** para correr los siguientes comandos

```
>curl https://surfer.nmr.mgh.harvard.edu/pub/data/tutorial_data.tar.gz -o
>tutorial_data.tar.gz
>tar -xzvf tutorial_data.tar.gz
>rm tutorial_data.tar.gz
```

Para poder ejecutarlo, fue necesario definir una variable de entorno llamada **TUTORIAL_DATA** la cual será guardada en la misma dirección de la carpeta que contiene el set de datos. Se define lo anterior en el siguiente comando:

```
>export SUBJECTS_DIR=\ \TUTORIAL_DATA/buckner_data/tutorial_subjs
>cd \SUBJECTS_DIR
```

Ahora comenzando el procesamiento, se utilizó la herramienta de visualización predeterminada de *FreeSurfer* denominada *FreeView*. Donde la interfaz de *FreeView* se abre con el siguiente comando

```
>freeview -v \
```

Fue posible obtener un primer procesamiento básico de la imagen denominado *Skull stripping*, encargado de eliminar elementos como el cráneo y tejidos blandos. Esto permitió la creación de una superficie *brainmask.mgz* que incluye la marcación de la materia gris y la materia blanca en color rojo y azul. Se procedió a abrir un primer paquete de imágenes que tiene un formato .nii en la interfaz utilizando los siguientes comandos.

```
>good_output/mri/T1.mgz \
>good_output/mri/brainmask.mgz \
>good_output/mri/wm.mgz:visible=0 \
>-f good_output/surf/lh.white:edgecolor=blue \
>good_output/surf/rh.white:edgecolor=blue \
>good_output/surf/rh.pial:edgecolor=red \
>good_output/surf/lh.pial:edgecolor=red \
>good_output/surf/rh.inflated:visible=0 \
>good_output/surf/lh.inflated:visible=0 \
>-viewport cor -layout 1
```

Al ejecutar los comandos anteriores, en la interfaz *FreeView* se obtuvo lo siguiente

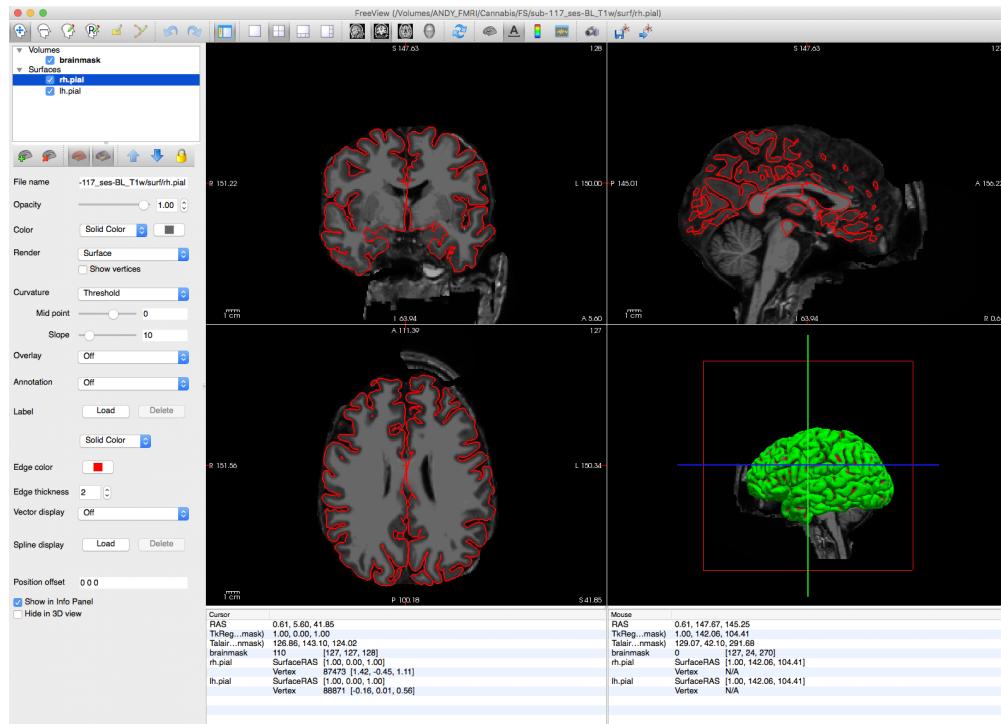


Figura 2-10: Interfaz de *FreeView*.

En la figura **2-10** se presenta la superficie *brainmask.mgz* anteriormente mencionada, la cual está dividida en 1 sección central y 3 secciones secundarias que se encuentran al lado derecho del visualizador. La sección central presenta la imagen del corte coronal, mientras que en el lado derecho se presentan los cortes sagital y transversal junto con la estructura 3D del paquete de datos introducido. Cabe aclarar que es posible cambiar el orden de las imágenes.

Los paquetes de imágenes procesados y no procesados son denominados volúmenes por el *FreeView*. Luego del procesamiento de estos paquetes, el visualizador realizó una división de dos grupos que contienen la división de los tejidos del cerebro y las primeras capas de imágenes modificadas.

La interfaz crea dos grupos denominadas *Volumes* y *Surfaces* respectivamente. El grupo *Volumes* utilizó un formato de capas, las cuales son *T1* que son las imágenes sin procesar, *brainmask* que son las imágenes con el procesamiento básico y *wm* la cual presenta la denominación de la materia blanca única únicamente. En este caso, solo se utilizó la capa *brainmask*. El segundo grupo *Surfaces* presentó la marcación de tejidos anteriormente mencionada, dividida en 2 capas principales llamadas *rh.pial* y *lh.pial*. El *pial* resalta la materia gris para el hemisferio derecho e izquierdo. Como puede observarse, es posible activar y desactivar estas capas para realizar un análisis de los límites de estos tejidos.

Para la siguiente parte del *Skull stripping*, las imágenes se procesaron por medio del proceso denominado *Watershed threshold*, el cual permitió encontrar los límites entre cráneo y cerebro. Este método es utilizado principalmente para evitar que alguna parte del cerebro o cerebelo sea cortada del *brainmask* y para evitar una eliminación incorrecta del cráneo que pueda dejar restos.

En este caso, como los primeros resultados del procesamiento básico de *Skull stripping* arrojaron imágenes que contenían restos de cráneo y de duramadre, se utilizaron los siguientes comandos para corregir este error.

```
>recon-all -skullstrip -wsthresh 5 -clean-bm -no-wsgcaatlas -subjID  
>-clean-bm - <subject>
```

El comando anterior creó un nuevo archivo de *brainmask* que tendrá una mayor cantidad de cráneo eliminado. Cabe resaltar, que para obtener un *Skull stripping* más agresivo, el valor introducido debe ser menor a 25. El espacio de *<subject>* está para colocar el nombre del archivo de cada paciente.

Puede observarse en la figura 2-11 que aún existen trozos de duramadre que no pudieron ser extraídos en el procesamiento anterior. Sin embargo, no se consideran de gran relevancia, por lo que estas no se sometieron a una metodología extra.

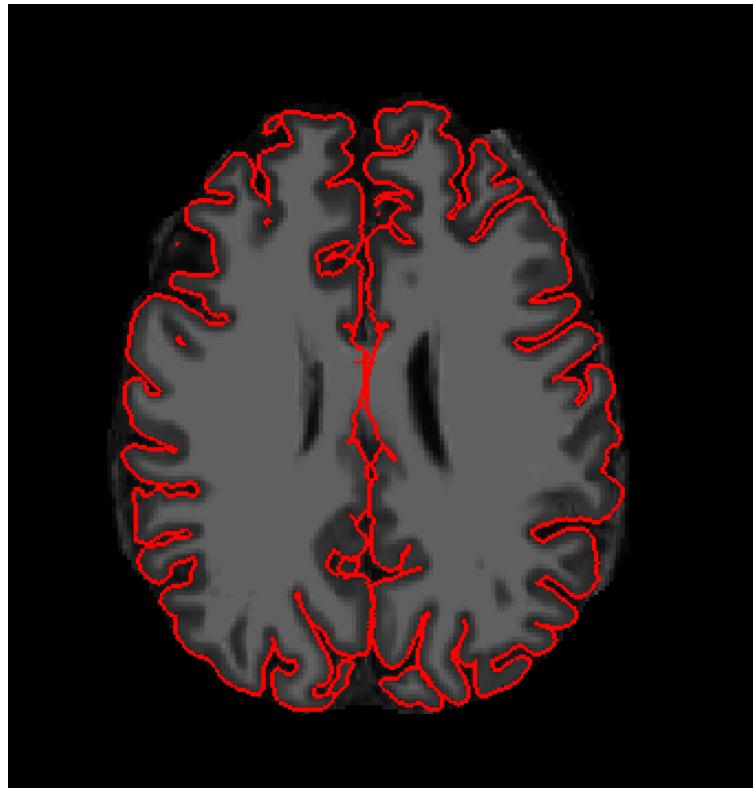
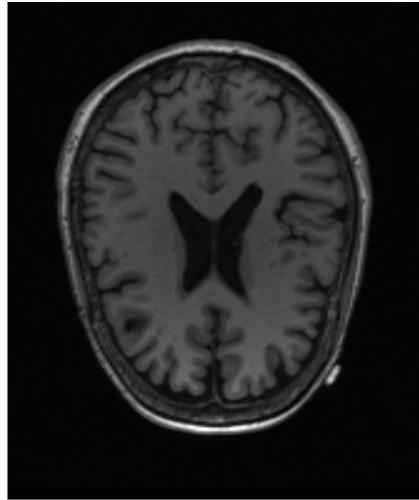


Figura 2-11: Cerebro procesado con *Skull stripping*.

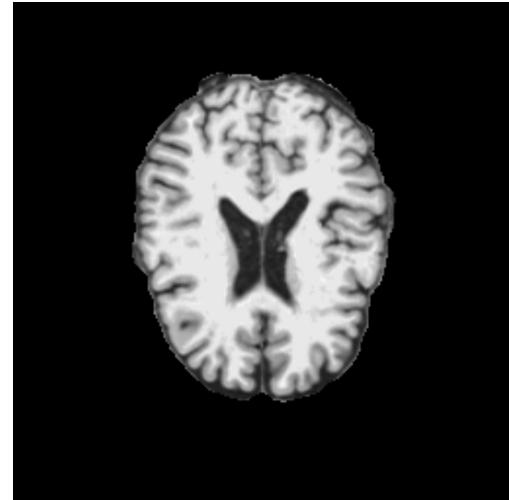
Por último, se procedió a la mejora del contraste de las imágenes para una mejor diferenciación entre los tejidos (materia blanca y materia gris). Para este paso, se utilizó la barra de herramientas encontrada en el panel de control, la cual contiene opciones para cambiar la opacidad, el contraste y el mapa de color de las imágenes. Solo se hicieron cambios en contraste y brillo, dejando las imágenes en su escala de grises original.

Luego de realizar los pasos anteriores, fue posible obtener paquetes de imágenes con características visibles para la red neuronal. A continuación un ejemplo que presenta la diferencia entre una imagen procesada y una imagen sin procesar.

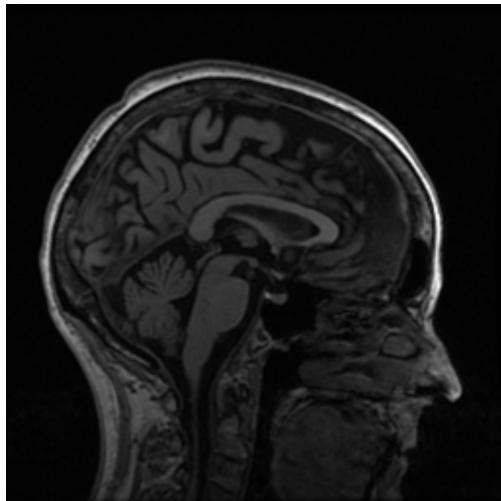
Es posible observar en la figura 2-12, que la imagen post-procesada muestra el cerebro recortado en el centro de un fondo negro. Se decidió realizar el recorte del 10 % de este fondo como parte de un procesamiento extra, ya que puede afectar el resultado de aprendizaje y ocasionar ruido. Para esto fue necesario quitar las filas y columnas pertenecientes al fondo de la imagen para su modificación.



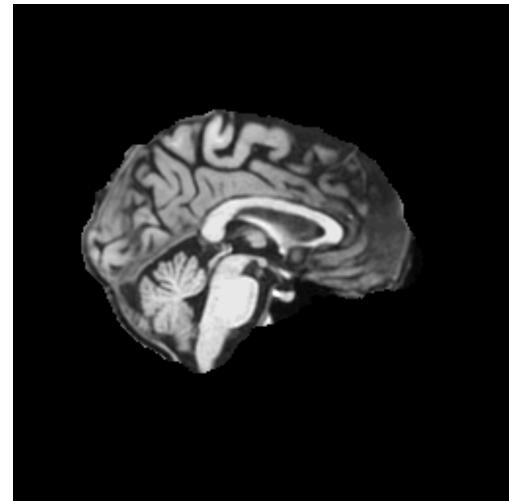
(a) Corte coronal sin procesar.



(b) Corte coronal procesado.



(c) Corte sagital sin procesar.



(d) Corte sagital procesado.

Figura 2-12: Comparación entre imágenes procesadas y sin procesar por *FreeSurfer*.

También se eliminaron las imágenes vacías, estas no contienen ningún elemento útil. Para lograr este paso extra, se realizó el siguiente código para la eliminación de los elementos cero.

```
truemp = np.delete(truemp, 0)
datanp = datanp.reshape(truemp.shape[0]+1,256,256,256,1)
datanp = np.delete(datanp,0, axis=0)
print(truemp.shape)
print(datanp.shape)
```

Obteniendo el resultado en la figura **2-13**

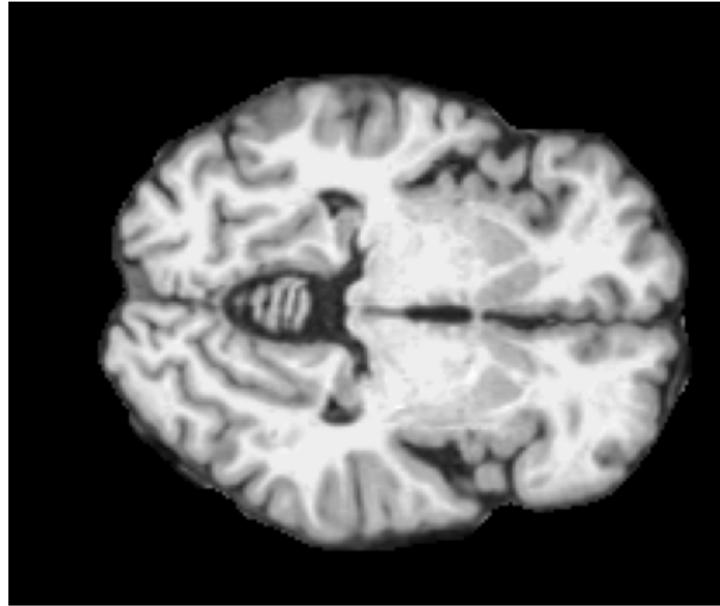
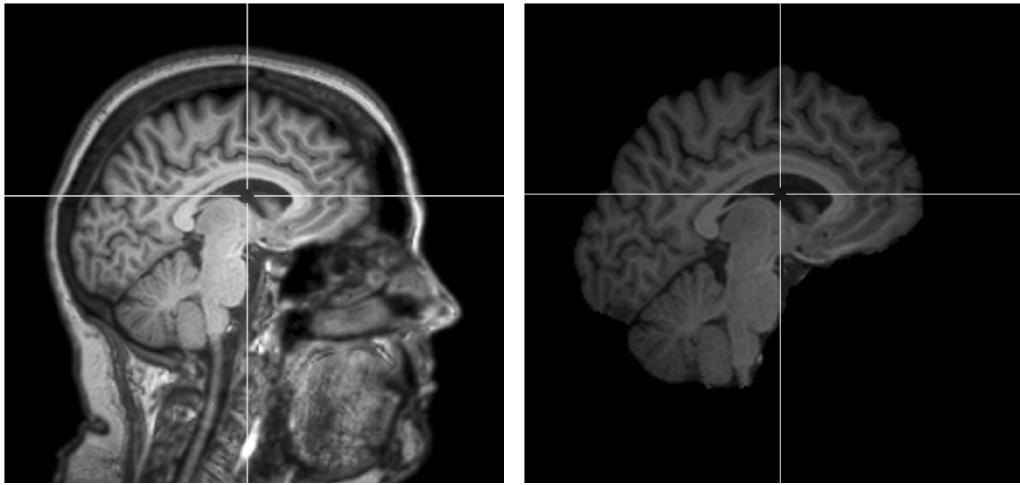


Figura 2-13: Corte transversal con eliminación del 10 % de los bordes.

La implementación del *FastSurfer* para el procesamiento de las imágenes de prueba proporciona una alternativa rápida, basada en la técnica del software *FreeSurfer*, que permite obtener la superficie requerida.

Este procesamiento se basa en la segmentación del cerebro para crear un *brainmask*. Primero, se crea una malla de triángulos de superficie inicial para cada hemisferio cerebral basada en la segmentación de materia blanca, esta debe ser suavizada para la formación de una superficie concreta. Luego, se coloca una segunda superficie expandida en el límite externo de la materia gris, logrando una estimación de espesor para cada punto de la corteza cerebral. Teniendo estos dos elementos, es posible dar paso a la construcción de la superficie *brainmask*, la cual contiene la imagen volumétrica del cerebro procesada. Se presenta, en la figura **2-14**, la comparación entre la imagen antes y después del procesamiento.

Finalizando esta sección de procesamiento, se aclara que se optó por utilizar un método sencillo para los datos de prueba para optimizar el avance del proyecto. Se dio prioridad a los datos de entrenamiento y validación con un procesamiento más exhaustivo, ya que estos hicieron parte directa del desarrollo de la red, por lo que deben ser de calidad y ser lo más completos posible. Es necesario tener en cuenta que cuando se aplique el modelo en un futuro con datos externos, estos no siempre se encontrarán procesados en su totalidad.



(a) Corte coronal sin procesar.

(b) Corte coronal procesado.

Figura 2-14: Comparación entre imagen procesada y sin procesar por parte de la técnica *FastSurfer*

2.3.1. Free Surfer

Es un software utilizado para el procesamiento de imágenes cerebrales. Proporciona herramientas avanzadas para la segmentación y reconstrucción automática del cerebro a partir de los datos que se deseen implementar, siendo el caso del proyecto, imágenes de resonancia magnética. *Free Surfer* utiliza algoritmos avanzados de procesamiento que permite generar modelos cerebrales con los cuales se realizarán análisis volumétricos y morfológicos, y con estos, monitorear la existencia de anomalías y el cambio estructural del cerebro [43].

Además de aplicar la segmentación y construcción del cerebro para exámenes específicos, la técnica que maneja el software permite eliminar elementos que no son útiles para el análisis cerebral, como lo es el cráneo y otros tejidos blandos que se presente en la imagen. Durante la investigación, el método aplicado de *Free Surfer* fue de gran utilidad, permitiendo obtener una imagen limpia del cerebro en tercera dimensión sin afectar su calidad ni dimensiones.

2.4. Estructura y configuración del modelo

Con el objetivo de encontrar soluciones a problemas complejos, como el reconocimiento e identificación de enfermedades, los científicos han desarrollado modelos computacionales con la capacidad de imitar el funcionamiento del cerebro humano. A estos modelos se les conoce como redes neuronales, los cuales son entrenados para tener la capacidad de aprender, clasificar y reconocer patrones a partir de una serie de datos específicos con el propósito de solucionar tareas complejas. Dentro de la amplia variedad de redes neuronales existentes, las redes neuronales convolucionales (CNN) se destacan por su capacidad para procesar imágenes. Estas arquitecturas están diseñadas para analizar y comprender el contenido visual, y han demostrado ser altamente efectivas en la clasificación de objetos, detección de características y segmentación de imágenes.

Es necesario definir la estructura de la red neuronal convolucional que será entrenada con las imágenes previamente procesadas. Para esto, se optó por un modelo de aprendizaje con estructura *AlexNet*. Esta reconocida estructura diseñada por Alex Krizhevsky, es utilizada para cumplir tareas de clasificación y reconocimiento de imágenes a gran escala [1]. Se diferencia de las CNN convencionales ya que es una red mucho más amplia y profunda compuesta por 5 tipos de capas principales organizadas:

1. Capa inicial: Recibe los datos de entrada (imágenes) a la red. Debe encontrarse acondicionada para las dimensiones de dichos datos.
2. Capas convolucionales: Aplica filtros receptivos para extraer características relevantes de las imágenes de entrada en diferentes niveles de abstracción.
3. Capas de agrupación: También conocida como *max pooling*, reduce las dimensiones de las imágenes manteniendo las características relevantes identificadas por los filtros de las capas convolucionales.
4. Capas totalmente conectadas: Capas semifinales que se encuentran conectadas aprenden los resultados más abstractos ofrecidos por las características extraídas. Contienen *dropout* y funciones de activación, las cuales impactan el rendimiento y la capacidad de diferentes partes de la red en su procesamiento interno.
5. Capa de salida: Siendo también una capa conectada, posee una función conocida como *softmax*, esta se utiliza para clasificar las imágenes de entrada dando una probabilidad de la cual se obtiene la identificación del dato de entrada.

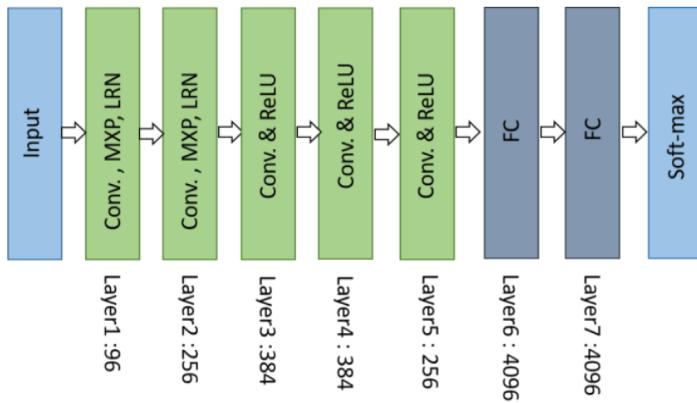


Figura 2-15: Estructura original de la red AlexNet (2018) [1].

La elección de la estructura tipo *AlexNet* se debe al desempeño que presenta en comparación con otros modelos de aprendizaje profundo. Se destaca por ser de las principales arquitecturas de redes convolucionales capaz de mejorar el rendimiento del aprendizaje y entrenamiento del modelo. Cabe aclarar que uno de los objetivos de esta investigación es poder implementar y poner a prueba la capacidad de clasificar imágenes que tendrá el modelo a partir de su entrenamiento. A partir de la clasificación de estas imágenes, la red brindará la probabilidad que tiene cada paciente de sufrir un posible desarrollo de *Alzheimer*. Esto quiere decir que la capacidad de predicción de la red vendrá de la extracción y aprendizaje de características, y como a partir de este proceso, la red podrá diferenciar un paciente posiblemente enfermo de uno sano.

Es necesario aclarar que estos datos de entrada se presentan en escala de grises, haciendo compleja la detección de características sutiles a simple vista. La red tipo *AlexNet* es capaz de identificar anomalías y patrones claves presentes en este tipo de imágenes gracias a los filtros denominados *kernels*, matriz de pesos que se desliza por la imagen de entrada para extraer características locales a partir de la realización de múltiples convoluciones, que se encuentran presentes en las capas convolucionales.

Por el lado de las funciones de activación, estas permiten la no linealidad en múltiples dimensiones, facilitando la detección de características y patrones complejos [44]. Debido a la densidad de los datos de entrada, es necesario un parámetro que pueda evitar el sobreentrenamiento del modelo, aportando regularización y aumento de la capacidad de la red. Siendo la técnica de *dropout* ideal para el cumplimiento de estas tareas.

Después de conocer la arquitectura en la cual se basará el proyecto, es necesario aclarar que esta fue acondicionada para ser una red tridimensional conocida como *AlexNet 3D* capaz de recibir paquetes de 500 imágenes que forman arreglos en tercera dimensión como datos de

entrada. Este tipo de datos tridimensionales contienen una gran cantidad de características que permitirán el reconocimiento y clasificación de las diferentes fases durante la prueba del modelo. Para entender mejor este tipo de modelo, se presentará su estructura base.

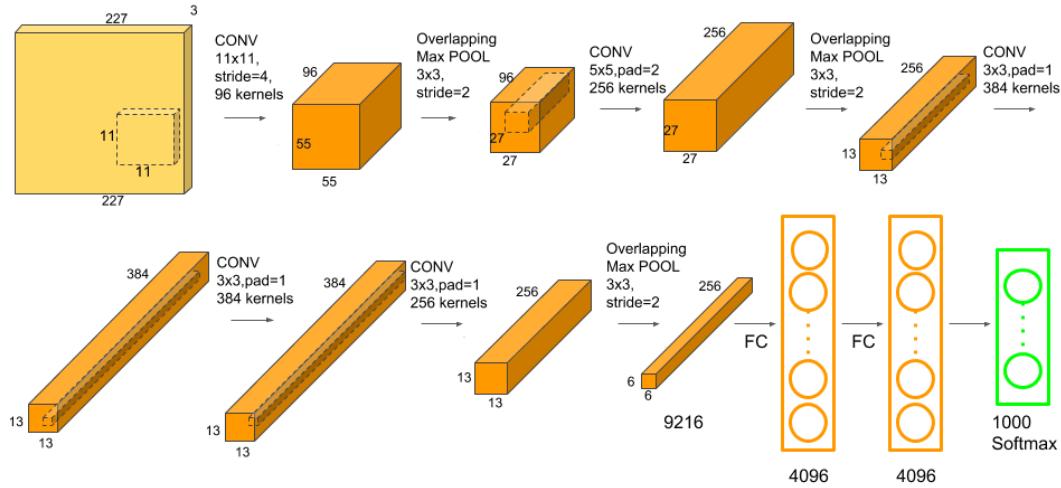


Figura 2-16: Estructura de una red *AlexNet 3D* [2].

La figura **2-16** presenta un ejemplo de la estructura preestablecida de una red *AlexNet 3D*. Cabe aclarar que los valores presentados en cada capa de la imagen no son los mismos que están siendo utilizados en este proyecto.

Para realizar el acondicionamiento de la red *AlexNet*, es necesario realizar cambios en su estructura que puedan admitir datos de tipo 3D. Estos cambios se dividieron en el ajuste de dimensiones de entrada, modificación de capas convolucionales y capas de agrupación 2D a 3D, el ajuste de los tamaños de los filtros presentes en dichas capas y demás cambios en la arquitectura principal.

Para el ajuste de dimensiones en la capa de entrada, fue necesario realizar un acondicionamiento que permitiera el ingreso de datos en tercera dimensión. Para las capas de convolución, se realizó el reemplazo de estas a partir de la utilización de la librería *Keras*, que brindará las nuevas capas convolucionales 3D. Esta librería también entrega las capas tridimensionales de agrupación, las cuales permitieron la reducción de medidas de los datos de entrada preservando las características más importantes. Dentro de estas capas, los filtros y *kernels* también fueron acondicionados a un formato 3D que les permita realizar la convolución a lo largo de las tres dimensiones. A continuación se muestra el algoritmo de la primera estructura de red utilizada para este proyecto.

```

model = Sequential()
model.add(Conv3D(7, kernel_size=(3, 3, 3),
                activation='relu',
                kernel_initializer='he_uniform',
                input_shape=(X_train.shape[1],
                            X_train.shape[2],
                            X_train.shape[3],
                            X_train.shape[4])))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.7))
model.add(Conv3D(4, kernel_size=(3, 3, 3), activation='relu',
                kernel_initializer='he_uniform'))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.5))

model.add(Flatten())
model.add(Dense(30, activation='sigmoid', kernel_initializer='he_uniform'))
model.add(Dense(30, activation='relu', kernel_initializer='he_uniform'))
model.add(Dense(2, activation='softmax'))

```

Algorithm 1: Primera estructura del modelo.

En el algoritmo presentado puede observarse la inclusión de las capas mencionadas anteriormente (*Conv3D*, *MaxPooling3D*). También se agregaron las funciones *Dense()* y *Flatten()* a las capas finales con el fin de aplanar los datos de salida de la capa anterior a un formato unidimensional para su procesamiento en las siguientes capas conectadas, las cuales contienen las funciones de activación *softmax*, *ReLU* y *sigmoid* que podrán mejorar el rendimiento de aprendizaje de la red.

Es necesario resaltar que este primer modelo fue utilizado para una prueba piloto con 10 paquetes de imágenes de dimensiones $5 \times 256 \times 256 \times 256$. Para probar esta estructura inicial, los datos fueron divididos en 2 directorios, los cuales contienen 5 paquetes de imágenes de resonancia magnética de los 2 grupos definidos en la primera sección, *Alzheimer* (AD) y cognitivo normal (CN). Luego de introducir los datos al modelo, se obtuvieron la forma de los datos de salida en cada capa de la red, los parámetros totales, entrenables y no entrenables presentados en la siguiente tabla

Modelo: sequential_1		
Layer (type)	Output Shape	Parameter (#)
conv3d_2 (Conv3D)	(None, 173, 206, 173, 7)	196
max_pooling3d_2 (MaxPooling 3D)	(None, 86, 103, 86, 7)	0
batch_normalization_2 (BatchNormalization)	(None, 86, 103, 86, 7)	28
dropout_2 (Dropout)	(None, 86, 103, 86, 7)	0
conv3d_3 (Conv3D)	(None, 84, 101, 84, 4)	760
max_pooling3d_2 (MaxPooling 3D)	(None, 42, 50, 42, 4)	0
batch_normalization_2 (BatchNormalization)	(None, 42, 50, 42, 4)	16
dropout_3 (Dropout)	(None, 42, 50, 42, 4)	0
flatten_1 (Flatten)	(None, 352800)	0
dense_3 (Dense)	(None, 30)	10584030
dense_4 (Dense)	(None, 30)	930
dense_5 (Dense)	(None, 2)	62

Tabla 2-1: Resultados del primer entrenamiento de la red base.

La tabla **2-1** muestra tres secciones y presenta la información sobre la estructura organizada de la red. Primero, las capas por las que atraviesan las imágenes, la forma de salida de los datos introducidos y el número de parámetros entrenables.

Parameters		
Total params	Trainable params	Non-trainable params
10,586,022	10,586,000	22

Tabla 2-2: Parámetros obtenidos a partir del primer entrenamiento.

La tabla **2-2** presenta el número de parámetros obtenidos por toda la red, es decir, la suma de los parámetros de cada capa. También, muestra el número de parámetros entrenables, los cuales pueden ser ajustados durante el entrenamiento [45]. Por último, los parámetros no entrenables, que son aquellos que no necesitan ajustarse durante el proceso de entrenamiento de la red neuronal [45].

Al realizar esta prueba piloto del modelo, fue posible obtener un acercamiento sobre las modificaciones que podrá necesitar la red para mejorar su desempeño en las tareas de clasificación y predicción, a medida que se agreguen más datos de prueba y entrenamiento. Estos cambios se presentarán durante la siguiente sección, la cual se basará en el entrenamiento del modelo.

2.4.1. Redes Neuronales Convolucionales 3D

Las redes neuronales convolucionales son ampliamente utilizadas para el procesamiento y análisis de datos tridimensionales, como lo son los volúmenes de imágenes. Son conocidas en el aprendizaje de redes profunda, utilizando como principal herramienta la operación de convolución. Suelen también trabajar correctamente con dimensiones temporales, lo que les permite capturar elementos espaciales y cronológicos de forma conjunta. La red se compone de múltiples capas, las cuales realizan las operaciones de convolución, agrupación y transformación no lineal que extraen características relevantes de los datos 3D de entrada [46].

Las operaciones de convolución 3D se expresan con base en la ecuación general de una red neuronal convencional de la siguiente forma:

$$z = W * X + b \quad (2-1)$$

$$Y = f(z) \quad (2-2)$$

En donde W es la matriz de pesos que representa las conexiones entre las neuronas de la capa anterior y la actual. X es el vector de entrada a la capa actual. b es el vector que se agrega a la suma ponderada $W * X$. Y por último, la función de activación aplicada al valor resultante para obtener la salida.

Con base a la expresión anterior, es posible obtener la operación de una convolución 3D presentada de la siguiente manera:

$$Conv3D(X, W, b) = \sigma(\sum(X \otimes W) + b) \quad (2-3)$$

La ecuación de convolución 2-3 que realizan las primeras capas del modelo, utiliza las variables X , W , \otimes y b , siendo estas:

- X : Volumen de entrada a la capa convolucional.
- W : Filtro de convolución el cual se desliza sobre el volumen de entrada y realiza operaciones de suma y multiplicación para calcular los valores de entrada.
- \otimes : Denota la operación de convolución, en donde se multiplica cada elemento del filtro con los elementos del volumen de entrada.
- b (*bias*): Se añade a cada canal de salida después de aplicar la convolución para ayudar a introducir desplazamiento y flexibilidad en la salida.
- σ : Representa la función de activación que desea aplicarse a la capa.

Por la parte de estructura, la cual fue explicada en esta sección, se encuentran cambios en las capas convolucionales y en las capas de agrupación debido a esta dimensión extra. También se presenta este cambio en los filtros conocidos como *kernels* pertenecientes a estos dos tipos de capas.

- Capas convolucionales 3D: Realizan operaciones de convolución en los datos utilizando filtros 3D que extraen características relevantes.
- Capas de agrupación 3D: Reducen las dimensiones de los datos mediante la selección de valores máximos y promedios.

En cuanto a las aplicaciones de este tipo de red neuronal, son altamente utilizadas para el análisis de imágenes médicas, demostrando ser eficaces y tener mejor rendimiento que una red neuronal convencional [47]. Para este proyecto, la implementación de esta estructura 3D permite aprovechar las características más relevantes de los datos 3D para el cumplimiento de tareas de clasificación y análisis de objetivos específicos.

2.4.2. Funciones de activación

Determinan la salida generada por esta neurona dados ciertos datos de entrada, y así, le permite decidir si debe activarse o no al transmitir una señal a los siguientes nodos occasionando una reconstrucción o predicción [48]. Para el modelo planteado en este proyecto, estas funciones permiten que la red aprenda relaciones y comportamientos no lineales en los datos. Añadir estas funciones en las capas ocultas, otorga control sobre el proceso de entrenamiento. Las funciones utilizadas fueron las siguientes:

Función Rectified Linear Unit (ReLU)

$$f(x) = \max(0, x) \quad (2-4)$$

La función *ReLU* transforma los valores introducidos, anula los valores negativos, deja sin modificar los positivos y cualquier valor inferior a cero se establece como cero [44]. Dado el caso de que la función sea igual a 0 junto con su derivada, se da la muerte de neuronas otorgando regularización del *dropout*. En la ecuación, la x representa la entrada a la función. Esta función toma el valor máximo entre 0 y la entrada x .

Leaky ReLu

$$f(x) = \max(ax, x) \quad (2-5)$$

Se le conoce como una variante de la función ReLu anteriormente explicada. Introduce una pendiente pequeña para los valores de entrada negativos en lugar de anularlos completamente, lo que permite que el gradiente fluya, aun así, cuando los valores sean negativos.

Función Sigmoide

$$f(x) = \frac{1}{1 + e^{-x}} \quad (2-6)$$

Conocida como función logística, presenta una salida interpretada como una probabilidad y mapea cualquier valor a un rango entre 0 y 1 [44]. Para la optimización del modelo, implementa la no linealidad en la red y soluciona problemas de clasificación binaria.

Función Softmax

$$f(x_i) = \frac{e^{x_i}}{\sum_{j=0...k} e^{x_j}}, \quad i = 0, 1, 2, \dots, k \quad (2-7)$$

Utilizada para la solución de problemas de clasificación multiclase, esta función toma un vector de entrada y produce una distribución de probabilidades en donde la suma de todos estos resultados es igual a 1 [44]. En la ecuación, x_i representa la entrada i -ésima neurona en la capa de salida, y n es el número total de clases a las que clasificarán los datos. Asigna una probabilidad a cada clase presentada por el modelo, lo que la hace útil para la posible predicción de las fases cerebrales que se están estudiando.

2.4.3. Librerías Usadas de Python

En esta sección se presentan las librerías más importantes que fueron utilizadas durante el desarrollo y construcción del modelo de red neuronal. Cabe aclarar que se tomó en cuenta la explicación de los siguientes paquetes debido a su nivel de importancia.

Keras

Keras [49] es una librería de código abierto de alto nivel escrita en Python, desarrollada para la creación de modelos de aprendizaje profundo y redes neuronales centrados en la experimentación rápida que simplifica el proceso de construcción, entrenamiento y evaluación. Permite realizar la creación de capas para las redes a partir de sus módulos configurables, los cuales pueden combinarse para crear nuevos modelos.

Numpy

La librería NumPy [50](abreviación de "*numerical Python*") es propia del lenguaje Python utilizada principalmente para la computación científica. Es un paquete de procesamiento de matrices que contiene una larga lista de funciones matemáticas, siendo álgebra lineal una de las más utilizadas. Resulta eficiente a la hora de acortar la escritura de código al requerir pocos ciclos, ya que las operaciones se manejan a través de matrices y vectores. Utilizada en el proyecto para la conversión de paquetes de imágenes 3D a arreglos de matrices de 3 dimensiones que la red puede leer con facilidad.

Matplotlib

Esta es una librería de visualización, la cual permite la generación de gráficos 2D-3D personalizados a partir de los datos guardados en listas o vectores realizados por la red [51]. En este caso, luego del entrenamiento de la red se toman datos de prueba y validación, los cuales son expresados a través de gráficas de 2 dimensiones. Estas gráficas presentan el comportamiento de estas etapas a partir de líneas de tendencia diferenciadas por diferentes colores. Matplotlib se integra con librerías como Numpy y Pandas, lo que facilita la visualización de datos a partir de arreglos y *dataframes*.

Pandas

La librería Pandas [52] es utilizada para la manipulación y análisis de estructura de datos en diversos formatos. Define estructuras de datos basadas en arreglos matriciales apoyados en dos librerías de Python conocidas como Matplotlib, que permite la visualización. y Numpy la cual se encarga de las operaciones matemáticas. Para el proyecto, se utilizó la estructura de datos bidimensional *DataFrame*, en este caso presenta el progreso de entrenamiento del modelo a través de la disminución de la función de pérdida y aumento de exactitud divididos en datos de entrenamiento y datos de validación. Permite visualizar si el modelo está sufriendo de *Overfitting*.

Nibabel

Conocida como una librería de Python, Nibabel [53] es utilizada para la lectura y manipulación de archivos de imagen de formato *NIfTI* (*Neuroimaging Informatics Technology Initiative*), este tipo de archivos son paquetes de imágenes médicas 3D utilizadas en neurociencias. Permite acceder a los datos y propiedades de la imagen. Para este proyecto, fue de gran ayuda para la visualización de las imágenes y su conversión a arreglos Numpy para su lectura.

TensorFlow

La librería TensorFlow [54] permite realizar cálculos de aprendizaje automático diseñada para la realización de tareas como visión artificial y aprendizaje profundo. Se basa en grafos computacionales que describen las relaciones entre las operaciones matemáticas que se realizan en el modelo. Fue de utilidad en el proyecto por su capacidad para entrenar redes neuronales con grandes conjuntos de datos, además, proporcionó herramientas y recursos para acelerar el entrenamiento de este modelo.

2.5. Entrenamiento y validación

En esta sección, se explicará el proceso de entrenamiento y aprendizaje del modelo tipo *AlexNet*. Es necesario saber que esta etapa fue fundamental para que la red neuronal obtenga la capacidad de realizar las tareas que se le indiquen, como la clasificación y predicción, que en este caso son fundamentales para cumplir con los objetivos de este proyecto. El objetivo del entrenamiento de la red neuronal se basó en la implementación de diferentes pruebas en donde van a realizarse cambios significativos que podrán mejorar la precisión del modelo, como por ejemplo, la ampliación del conjunto de datos, el aumento o disminución de la tasa de aprendizaje, el ajuste de los parámetros de la red mediante la modificación del número de neuronas de las capas convolucionales y la aplicación o eliminación de las funciones de activación.

Para determinar si estos cambios fueron correctos y mejoraron la calidad de la red, se analizarán los porcentajes obtenidos de exactitud (*accuracy*), pérdida (*loss*) y las curvas de aprendizaje que representan los cambios de estos valores durante las épocas aplicadas durante el entrenamiento. En el caso de las redes neuronales, la exactitud indica la precisión alcanzada por el modelo durante el aprendizaje de los datos para predecir la clase correcta de la muestra de datos brindada. Su curva de aprendizaje muestra el cambio de la precisión a medida que aumenta el número de épocas del entrenamiento. Por otro lado, la pérdida presenta la incongruencia entre las predicciones presentadas por la red y los valores verdaderos expresados como etiquetas. Su curva de aprendizaje muestra la disminución de la función de pérdida a medida que el modelo está siendo entrenado. A partir de estos valores, fue posible determinar nuevas modificaciones de los parámetros de la red para obtener las respuestas deseadas.

En cuanto a los datos de aprendizaje, fue necesario aplicar una división extra en los grupos de imágenes anteriormente mencionados, ya que con estos será posible cumplir con el entrenamiento requerido para la red. Para los siguientes ensayos del modelo, se realizó un *split* de los datos adquiridos, donde el 30 % se destinará a la validación y el 70 % se utilizará para el entrenamiento. Cabe resaltar que en la sección 2.1, los grupos de estudio seleccionados para estas etapas fueron *Alzheimer* (AD) y paciente sano (CN). Estos dos grupos se escogieron con el fin de conformar el conjunto de datos de entrenamiento predeterminado. El objetivo de esta selección es lograr que la red adquiera la capacidad de aprender patrones específicos de un cerebro sano y de un cerebro enfermo, logrando así una diferenciación de ambos.

Además, durante estas pruebas, se aplicará un aumento de datos con el fin de enriquecer el conjunto de entrenamiento, permitiendo que la red aprenda una amplia gama de características relevantes, y al mismo tiempo, evite el *overfitting*. Es necesario aclarar que este proyecto se vió limitado en este aspecto debido a la limitación de la capacidad computacional.

Volviendo a las modificaciones estructurales del modelo, es necesario mantener la arquitectura base de una red tipo *AlexNet*. Teniendo en cuenta esta necesidad, fue necesario determinar el tipo de cambios a los que se someterá la red sin afectar su base inicial. Los valores de porcentaje entregados por la validación de la red serán clave para determinar estos ajustes. Durante las pruebas a las que se enfrentó el modelo, se propuso la adición o eliminación de regularizadores de *kernel* dependiendo del desempeño que presente. También, se tuvo en cuenta el aumento de neuronas y capas neuronales en la red para aumentar su complejidad y así, extraer más características reconocibles para el sistema neuronal. Para mejorar la generalización y evitar la dependencia de neuronas, se optó por una regularización *dropout* que se mantuvo en constante cambio. Por último, las funciones de activación del algoritmo base anteriormente presentado serán cambiadas según las necesidades de la red.

Se realizaron aproximadamente **10 pruebas del modelo**, en las cuales se aumentó el conjunto de entrenamiento y se modificaron los parámetros explicados anteriormente. Para presentar los resultados y cambios de cada *test*, se dispondrá de la siguiente organización:

- Número de imágenes
- Modificaciones del modelo con respecto a la prueba anterior
- Valores de precisión, pérdida y curvas de aprendizaje
- Explicación de las modificaciones con respecto a la prueba anterior

Durante el proceso de entrenamiento del modelo, se tomaron decisiones específicas sobre los parámetros que afectan este proceso, los cuales no presentarán ningún cambio a lo largo de las pruebas. Estos parámetros incluyen la tasa de aprendizaje, el tamaño del lote, las épocas de entrenamiento y el conjunto de datos de validación. El tamaño del lote se estableció en 10 muestras por iteración de entrenamiento. Se decidió mantener este tamaño constante en todo el proceso. El número de épocas de entrenamiento se fijó en 100. Esta elección garantiza un número adecuado de iteraciones para que el modelo aprenda y se ajuste a los datos. Para el conjunto de datos de validación, se reservó el 30 % del total de datos disponibles. Esta división permitirá evaluar el rendimiento del modelo en un conjunto independiente de datos y verificar su generalización. En cuanto a la tasa de aprendizaje, se optó por mantenerla en 0,0001. Esta elección se basa en la búsqueda de una convergencia suave y estable durante el entrenamiento, evitando oscilaciones o divergencias del modelo.

El siguiente capítulo presentará cada una de las pruebas realizadas durante el proceso de entrenamiento y los resultados obtenidos.

3 Experimentos y resultados

En este capítulo, se presentarán y analizarán los resultados obtenidos a partir de tres experimentos clave que fueron planteados con el propósito de demostrar la efectividad del modelo construido. Se examinará detalladamente cada uno de los experimentos realizados, describiendo el propósito de cada uno a partir de los procedimientos llevados a cabo y los resultados obtenidos. Cada experimento fue diseñado con el objetivo de dar a conocer la aplicación de nuevas herramientas de apoyo para la obtención de un posible diagnóstico de enfermedades neurodegenerativas.

3.1. Modificación de parámetros

Este experimento se basó en realizar modificaciones a la red base que se planteó en el capítulo anterior a lo largo de diferentes pruebas, con el fin de acondicionarla a las necesidades del proyecto. La división de estas pruebas fue planteada en la sección de Entrenamiento y Validación.

Las curvas de aprendizaje nombradas en la sección anterior representan la exactitud y pérdida del modelo, las cuales serán divididas en dos figuras. Estas figuras presentan dos curvas denominadas *train* y *val*, en las cuales se monitoreó el comportamiento del modelo durante el entrenamiento y la evaluación su rendimiento. Cabe resaltar que las curvas y las estructuras de los modelos se puede ver en los anexos A y B respectivamente.

PRUEBA 1

Para la primera prueba de la red, se mantuvo la arquitectura del modelo base presentado en la sección 2.4. En este caso, se emplearon 10 paquetes de imágenes por cada grupo seleccionado, teniendo un total de 20 datos en el conjunto de entrenamiento.

La forma cuadrada que presentan estas curvas en la figura 5-1, se da debido a la poca cantidad de datos iniciales que se incluyeron en esta prueba. Se obtuvo como resultado un valor de exactitud del 100 % a partir de la época 10, interpretándose como un no-aprendizaje profundo de más características significativas por parte de la red, dando paso al *overfitting* tanto para el entrenamiento como para la validación. Por otro lado, la curva

que representa la pérdida (initialmente descendiente) deja de presentar una variación en su comportamiento, mostrando una línea suave. Cabe resaltar que esta prueba se realizó para tener un primer vistazo de los resultados de entrenamiento. También, es necesario saber se estará incrementando de forma gradual los datos del conjunto, de esta forma se podrá mejorar la estabilidad y capacidad del aprendizaje.

PRUEBA 2

En esta segunda prueba, no se realizaron cambios con respecto a la arquitectura anterior. Sin embargo, si hubo un incremento de 5 paquetes de imágenes por cada grupo seleccionado, dando como resultado un total de 30 datos en el conjunto de entrenamiento. Con este aumento, se espera disminuir el *overfitting*.

La figura 5-2 muestra que ambas curvas presentaron cambios a partir del aumento de datos. Las curvas de aprendizaje de la precisión presenta más variación de valores, sin embargo, comienza a presentar un comportamiento constante y lineal a mitad de las épocas de entrenamiento y validación, presentando un sobreajuste de la red. Se presentó un aumento en la pérdida durante la validación del entrenamiento, lo que significa que la mayoría de predicciones dadas por la red fueron erróneas. También se notó un avance en cuanto al valor de la exactitud, ya que este resultado alcanzó un valor del 100 % en la época 36, demostrando así una disminución de sobreajuste.

PRUEBA 3

En la tercera prueba se mantuvieron los mismos parámetros de la red anterior, ya que se decidió dar prioridad al aumento de datos en el conjunto de entrenamiento. Para esto, se duplicaron los datos con respecto a la prueba 2 para un total de 60 imágenes del conjunto de entrenamiento. A continuación se muestran las curvas de aprendizaje obtenidas.

Las curvas presentadas en la figura 5-3 muestran cambios significativos en comparación con los resultados obtenidos en la prueba anterior. Las curvas de precisión muestran una mayor variación en los valores debido al nuevo aprendizaje de características al que el modelo fue sometido, dando como resultado un porcentaje de precisión del 83,3333 %. En cuanto a la curva de pérdida, se demuestra una disminución similar a las pruebas anteriores, dando a entender un incremento en la mejoría de las predicciones.

PRUEBA 4

Para esta nueva prueba, se realizaron modificaciones en la estructura de la red y se aumentaron los datos del conjunto de entrenamiento. El algoritmo 5 presenta la estructura.

Al modelo se le agregó una nueva capa de convolución 3D que contiene una capa de *MaxPooling*, también tiene dos neuronas y un valor de *dropout* de 0,7. En la segunda capa de convolución, se aumentó el valor de *dropout* a 0,7. Por último, se aumentó a un total de 8 neuronas en la primera capa de convolución. Con esto se tiene una disminución de neuronas de forma descendente. En cuanto a los datos, se realizó un aumento de 20 imágenes por cada grupo seleccionado, obteniendo un total de 100 datos en el conjunto de entrenamiento.

En cuanto al comportamiento de las curvas de aprendizaje, la figura 5-4 muestra una tendencia variante en la precisión hasta la época 40, luego, se presenta sobreajuste en la parte de entrenamiento. La gráfica de pérdida muestra un descenso en la curva de entrenamiento, mientras que la curva de validación todavía presenta predicciones erróneas. Para esta prueba, se obtuvo una precisión del 53,333 %. Con este resultado, puede verse un progreso estructural del modelo al hacerse más complejo para el aprendizaje de nuevos patrones.

PRUEBA 5

Se agregó al modelo anterior un regularizador de *kernel* tipo *L2* con el objetivo de experimentar con los parámetros de la red. Este regularizador permitirá reducir el *overfitting* y controlar la complejidad del modelo. Estos cambios pueden verse en el algoritmo 2. Al conjunto de entrenamiento no se añadieron más datos con la intención de poner a prueba estos nuevos parámetros y como trabajan con la cantidad de imágenes establecidas en el entrenamiento anterior.

La aplicación de estos nuevos parámetros puede verse reflejada en el comportamiento de las curvas de aprendizaje presentes en la figura 5-5. Las curvas de precisión del modelo dejaron de manifestar una tendencia constante, presentando más variación durante las épocas de entrenamiento y validación del modelo, lo cual puede concluirse como una disminución significativa de *overfitting*. El modelo presentó un porcentaje de precisión del 56,6666 %. Por otro lado, se obtuvo un progreso en la pérdida del modelo. Ambas curvas muestran un comportamiento descendente, curvo y suave, lo que significa una disminución de errores predictivos durante las épocas de entrenamiento.

PRUEBA 6

Para la prueba 6, se realizaron modificaciones tanto en el conjunto de datos como en la estructura de la red. Los cambios realizados se encuentran en el algoritmo 3

Se realizó un incremento de 10 imágenes por cada grupo, teniendo un total 120 datos en el conjunto de entrenamiento. A la arquitectura de la red, se le agregó una capa convolucional

que contiene 16 neuronas junto con otra capa de *MaxPooling*. Cada capa de la red contiene *dropout*, para las dos primeras capas de la red, se ajustó un valor de 0,7, mientras que las dos últimas capas tienen un valor de 0,5. La intención de estos cambios es evitar el aumento de *overfitting* y experimentar la complejidad del modelo.

Los resultados de estos cambios pueden verse reflejados en la figura 5-6. Las curvas de precisión presentan una tendencia ascendente durante el entrenamiento, sin embargo, el comportamiento de los valores de validación se mantiene estático durante la fase inicial de las épocas, dando a entender un sobreajuste. Se obtuvo un 50% de precisión en este *test*. La gráfica de pérdida del modelo sigue teniendo un comportamiento descendente, aunque se presenten variaciones de los datos tanto para validación como para entrenamiento. Se concluye una disminución en los errores de las predicciones.

PRUEBA 7

Se realizó un cambio en la estructura del modelo y un aumento de los datos que permitieron aprovechar los recursos computacionales. La estructura de la red puede verse en el algoritmo 4.

En esta prueba fue necesario disminuir la cantidad de datos de 120 a 100 paquetes de imágenes en el conjunto de entrenamiento debido a la limitación en la capacidad computacional y el aumento de complejidad del modelo. Se aumentó el número de neuronas en las dos primeras capas densas, proporcionando un mayor aprendizaje de características cada vez más complejas. La última capa densa se mantiene sin cambios. Por último, se agregó un regularizador de *kernel* tipo *L2* en la primera capa para evitar el sobreajuste.

Se tuvo un retroceso en la etapa de validación, en este caso, la variación en los valores de precisión se ve frenada antes de la época 40, mientras que la curva de entrenamiento presenta una tendencia ascendente y variante de los valores en cada época. Estos comportamientos se presentan cuando el modelo está experimentando *overfitting*, siendo la disminución de los datos en el conjunto de entrenamiento una razón principal. El porcentaje de precisión se mantuvo en 53,33333 %. En cuanto a las curvas de pérdida del modelo, el entrenamiento y la validación presentan una tendencia descendente y poco variante. La figura 5-7 presenta las curvas anteriormente explicadas.

PRUEBA 8

En esta prueba el modelo no se sometió a cambios significativos. Su estructura en el algoritmo 5.

En cuanto al conjunto de entrenamiento, pudo realizarse un incremento de 15 imágenes con respecto a la prueba anterior, dando como resultado un total de 130 datos. Se duplicaron el número de neuronas en cada capa de convolución con el objetivo de aumentar la capacidad de reconocimiento de características y mejorar la flexibilidad de la red para el ajuste de sus parámetros.

El comportamiento del modelo en cuanto a precisión y pérdida mostró pocos cambios. La curva de precisión del entrenamiento no presentó un cambio significativo en comparación con la prueba anterior, esta se mantiene ascendente. Mientras que la curva de validación presentó un aumento en la variación de los valores en cada época, demostrando una disminución de *overfitting*. Se obtuvo un porcentaje de 53,8461 % de precisión. La pérdida del modelo no presentó cambio significativo en su comportamiento.

Para las siguientes pruebas, se realizaron dos rondas de entrenamiento del modelo. En la primera, se entrenó la red utilizando un conjunto significativo de imágenes por cada grupo de estudio, y posteriormente se guardó el modelo completo. Luego, en la segunda fase, se sometió el modelo a un nuevo conjunto de imágenes, aprovechando el pre-entrenamiento obtenido en la fase anterior. Esta división se llevó a cabo para incrementar gradualmente la cantidad de imágenes utilizadas en el entrenamiento, dado que la capacidad computacional es limitada.

PRUEBA 9

Durante esta prueba, la red y el conjunto de entrenamiento sufrió nuevos cambios. El nuevo modelo puede verse en el algoritmo 6.

El modelo mencionado fue entrenado con un total de 80 paquetes de imágenes para el pre-entrenamiento. Durante las modificaciones de la red, se implementó una nueva función conocida como *LeakyReLU* en las primeras capas densas, la cual introducirá una pequeña pendiente y con esto logrará que las neuronas puedan seguir actualizarse y ser activas en el aprendizaje. Este cambio fue necesario debido al tamaño del conjunto de datos aplicado a una arquitectura compleja. Con esta modificación se busca evitar un aumento en el sobreajuste o una disminución en el porcentaje de precisión. También se incluyó cierto valor de *dropout* para cada capa densa y se aumentaron el número de neuronas en cada una de estas.

Para el segundo entrenamiento al que fue sometido el modelo guardado, se planteó un total de 90 imágenes para el nuevo conjunto. Los resultados obtenidos durante esta segunda fase pueden verse reflejados en la figura 5-9. En la gráfica de precisión del modelo, la curva de entrenamiento se mantiene variante y ascendente como en las pruebas anteriores, por otro lado, la curva de pérdida presenta un comportamiento variante, demostrando una reducción de *overfitting* por parte del modelo. Para esta prueba, el porcentaje de precisión fue del 50,7058 %. Las curvas de aprendizaje en la gráfica de pérdida del modelo muestran un comportamiento similar a las últimas pruebas realizadas, manteniendo su forma constante.

PRUEBA 10

Para las últimas modificaciones del modelo, se tomaron en cuenta las tendencias presentadas por las curvas de precisión y pérdida junto con los valores porcentuales de su desempeño. Con estos resultados pudo concluirse la falta de capacidad computacional para aplicar un aumento considerable del conjutno de datos para el entrenamiento del modelo. Debido a esto, se realizaron modificaciones en los parámetros pero no en la estructura del modelo.

En esta prueba fue posible corregir varios errores en los parámetros de la red. También se dio prioridad al aumento de imágenes del conjunto de entrenamiento. La red modificada puede verse en el algoritmo 7.

En el pre-entrenamiento, se incluyeron 50 imágenes para el grupo de CN y 55 imágenes para el grupo de AD, dando como resultado un total de 105 datos del conjunto. Los cambios realizados en el algoritmo fueron con base a las limitaciones presentadas por la capacidad computacional. Por esta razón, se optó por la disminución de los valores de *dropout* en cada capa presente en la red e incluir regularizadores de *kernel* en las dos primeras capas. Como se presentó una limitación en el aumento de los datos, los valores anteriores en el parámetro *dropout* pueden ocasionar que haya pérdida de información, reducción de aprendizaje y una menor estabilidad durante el entrenamiento. También se redujo el número de neuronas de la primera capa densa a la mitad para poder controlar el sobreajuste. Resumiendo lo anterior, estas modificaciones fueron necesarias para que el modelo pudiera trabajar mejor con los datos entregados.

Para el segundo entrenamiento, se utilizaron un total de 80 paquetes de imágenes para el conjunto de datos final. Se realizó el proceso de aprendizaje con las condiciones anteriormente presentadas para dar lugar a los siguientes resultados. Es posible ver que las gráficas presentadas en la figura 7 muestran una variación mayor en comparación con las pruebas tomadas anteriormente, dando a entender un aumento de datos, permitiendo el aprendizaje de cada vez más patrones y características. Las curvas de precisión del modelo presentan constante cambio durante las 100 épocas de entrenamiento. La curva *train* presenta

un comportamiento ascendente y variante, demostrando que durante su entrenamiento pudo desarrollar un aprendizaje significativo de características presentes en las imágenes, mientras que la curva *val* presenta poca variación en su comportamiento pero un gran cambio en los valores de cada época. La pérdida del modelo mantuvo la misma tendencia presentada en las pruebas anteriores, junto con un comportamiento variable en los valores obtenidos en cada época, tanto para el entrenamiento como para la validación del modelo.

Tras la realización de las pruebas anteriores, el modelo obtenido en el entrenamiento final logró presentar un porcentaje de precisión del 72,222 %. Este resultado demuestra la estabilidad de la red y la efectividad de las diversas pruebas a lo largo de la etapa anterior, que permitieron realizar un seguimiento del comportamiento del modelo y así, determinar las modificaciones adecuadas para alcanzar un desempeño satisfactorio en la fase de validación. La aplicación de los nuevos parámetros radica en la capacidad del modelo para adaptarse y generalizar los patrones encontrados en el conjunto de datos, lo que a su vez permite obtener predicciones más precisas y consistentes.

Se optó por nombrar el modelo desarrollado *K-Net95*, el cual fue basado en la canción *N95* del rapero Kendrick Lamar. En la figura 3-1 se presenta el prototipo de la red obtenida.

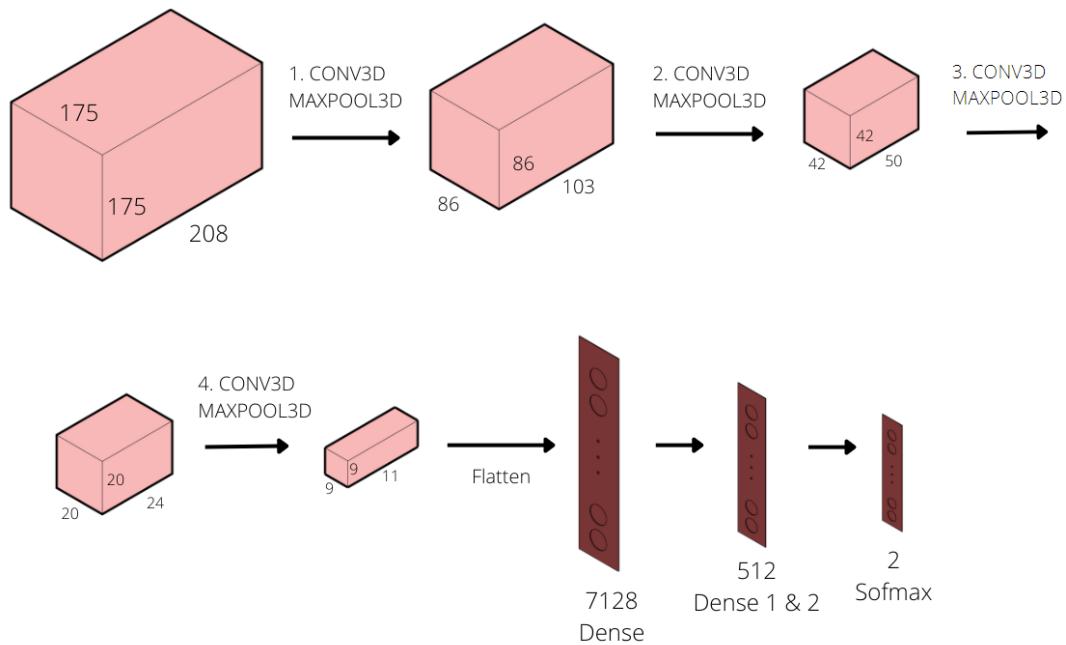


Figura 3-1: Estructura del modelo *K-Net95*.

3.2. K-Net95 vs. otros modelos

Con el fin de demostrar el rendimiento superior del modelo *K-Net95* en comparación con otras arquitecturas neuronales reconocidas, se llevó a cabo una comparación entre el modelo desarrollado y dos arquitecturas destacadas: *ResNet3D* [55] y *UNet3D* [56]. Los algoritmos correspondientes a estas redes estarán disponibles en los anexos del presente trabajo.

La elección del modelo de red tipo *AlexNet3D* se basó en una serie de criterios necesarios para llevar a cabo la investigación, como por ejemplo, rendimiento, cantidad de parámetros y estabilidad. Esta red ha demostrado la capacidad de detectar, en el caso de su aplicación a imágenes médicas, características malignas de diversos tejidos para su debida clasificación, superando los resultados de cualquier otro modelo de aprendizaje profundo [57]. Por otro lado, en comparación con otros métodos detección profunda, el desempeño y eficiencia de la red *AlexNet3D* ha demostrado ser superior en términos de tiempo de entrenamiento y validación en términos de rendimiento [58].

Para poder realizar una comparación entre estos tres modelos, se entrenaron las redes *UNet3D* y *ResNet3D* con una cantidad similar de datos a los introducidos en la red *K-Net95*. No fue posible implementar el mismo grupo de entrenamiento debido a la limitación en la capacidad computacional.

Este experimento se basó en los resultados obtenidos durante las fases de entrenamiento y validación del modelo, enfocando el porcentaje final de precisión de cada modelo. Se implementaron los mismos parámetros de tamaño de lote, tasa de aprendizaje y número de épocas para ambas redes, obteniendo los siguientes resultados.

Modelo	<i>Accuracy</i>
K-Net95	72,2222 %
UNet3D	50 %
ResNet3D	53,333 %

Tabla 3-1: Comparación entre las estructuras planteadas.

Durante esta prueba, se monitorearon los valores obtenidos en las épocas de entrenamiento y validación para analizar la presencia de *overfitting* o *underfitting*, el consumo computacional, parámetros establecidos en su arquitectura base y complejidad estructural.

La red *ResNet3D* presenta una arquitectura profunda que se considera pesada en términos

de rendimiento computacional debido a los elementos que la componen. Posee múltiples capas residuales, implicando diversas conexiones. Durante este experimento, la red no presentó un óptimo desempeño debido a la falta de *dropout* en su estructura base, viéndose expresada durante la fase de validación. Por otro lado, también debe tomarse en cuenta la complejidad estructural que tiene este tipo de modelo en comparación con arquitecturas más simple, lo que aumentó el tiempo de entrenamiento haciéndola poco práctica. Debido a estas razones, el porcentaje de precisión (53,333 %) que presentó la red fue bajo en comparación con el obtenido durante las pruebas del modelo *K-Net95*.

En cuanto a la red *UNet3D*, es una estructura compleja debido a la cantidad de capas y parámetros que tiene. La arquitectura de este modelo posee una combinación de capas convolucionales, de muestreo y de convolución transpuesta implicando un mayor número de operaciones, lo que implica un mayor consumo de recursos computacionales. También puede presentar problemas en la fase de entrenamiento al tener limitaciones en la captura de características a gran escala, con las cuales se auemnta la posibilidad de perder detalles finos durante el proceso de entrenamiento. En comparación con los otros experimentos, la red *UNet3D* consumió muchos más recursos computacionales. Es por esta razón que fue necesario disminuir el conjunto de datos definitivo y así, poder comenzar su entrenamiento. Debido a estas razones, la red presentó un porcentaje de precisión (50 %) mucho más bajo en comparación a los primeros modelos.

Con las pruebas realizadas durante este experimento, pudo concluirse que el modelo *K-Net95* se destacó por su desempeño y eficiencia en comparación las redes *UNet3D* y *resNet3D* en varios aspectos clave. En términos de desempeño de clasificación, el modelo desarrollado demostró la capacidad para extraer características relevantes a partir del conjunto de datos tridimensionales. Además, permitió el aprovechamiento de la capacidad computacional utilizada, logrando un rendimiento superior con menos parámetros a diferencia de las otras redes. En cuestión de entrenamiento, el modelo *K-Net95* pudo disminuir considerablemente el sobre-ajuste de los datos teniendo en cuenta su reducción necesaria para este proyecto, mientras que las otras redes presentaron un comportamiento constante en los valores obtenidos durante las épocas de aprendizaje. La red tipo *AlexNet3D* se definió como la opción más óptima y eficiente entre las redes *UNet3D* y *resNet3D*.

3.3. Predicción

Para demostrar la capacidad de clasificación adquirida por la red durante las fases de entrenamiento, se realizará el experimento de predicción. Es necesario aclarar que el modelo *K-Net95* se definió como una herramienta de clasificación que predice la probabilidad que tendrán los pacientes de estudio de padecer *Alzheimer*. Esto quiere decir que su objetivo es predecir la clase a la que pertenece cada imagen de entrada debido al conjunto de características que esta posee a través de la categorización.

Como se explicó en la sección 2.2, el grupo de datos de prueba está conformado por los paquetes de imágenes proporcionados por el repositorio de la clínica Comfamiliar, los cuales fueron clasificados en las fases de transición establecidas. Estas divisiones se han realizado con la intención de permitir que la red pueda llevar a cabo la predicción y clasificación de manera precisa, y así, evaluar su capacidad y desempeño.

Para la implementación del modelo, se utilizó un total 10 paquetes de imágenes como datos de entrada para la sección de predicción. El objetivo es predecir la probabilidad que presenta cada paciente de sufrir *Alzheimer* mediante un porcentaje. Durante el experimento, se seleccionaron 5 paquetes de imágenes para ser analizados por la red y así, obtener los resultados. Estos resultados incluirán el corte horizontal del cerebro, el porcentaje de probabilidad y la fase a la que pertenece cada paciente. A continuación se muestran los resultados entregados por el modelo en la siguiente figura.

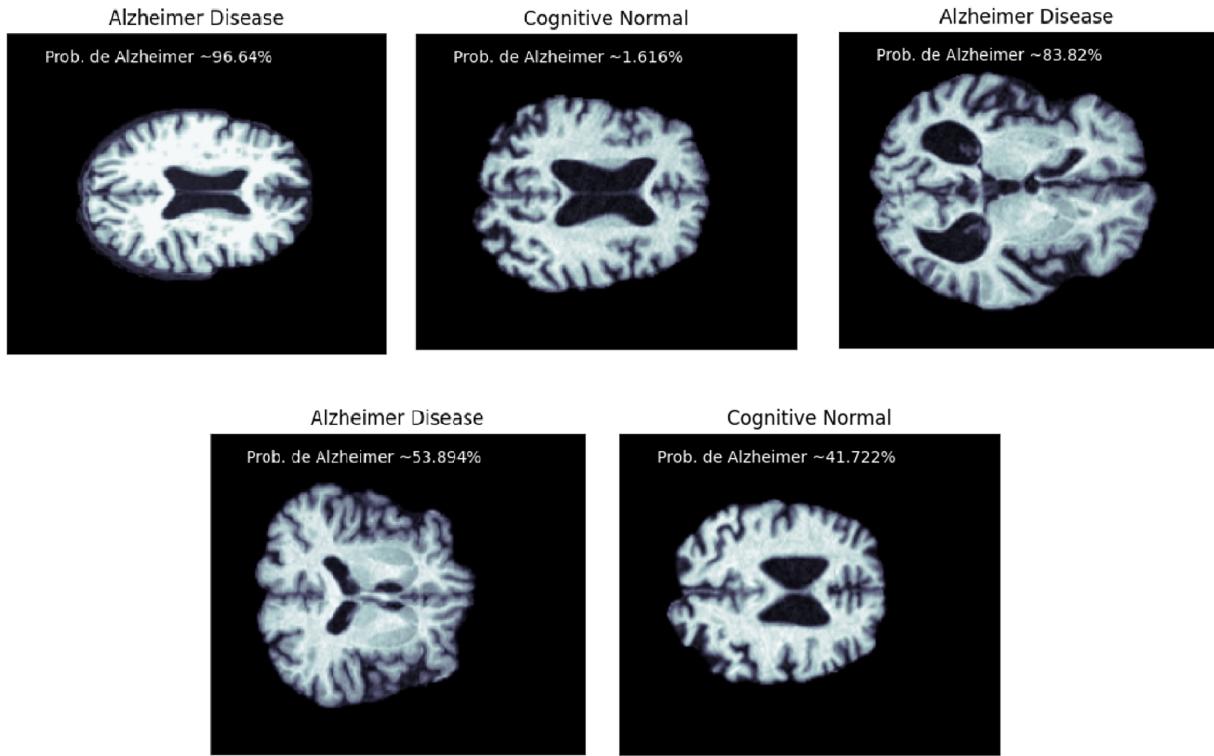


Figura 3-2: Resultados obtenidos del modelo *K-Net95*.

La figura 3-2 presenta 5 cortes con sus respectivos porcentajes de probabilidad. Esto permitirá una lectura sencilla de las predicciones hechas por el modelo, con el objetivo de ser una herramienta útil capaz de presentar un posible diagnóstico.

En comparación con las fases de entrenamiento y validación, la etapa de prueba no necesitó de una capacidad computacional robusta. Una vez que el modelo ha sido guardado, solo es necesario utilizar los paquetes de imágenes de cada paciente para su análisis y así, determinar el porcentaje de probabilidad que tienen de padecer *Alzheimer*. Además, clasificó a cada paciente en las fases principales establecidas en la sección de la base de datos, es decir, *Alzheimer Disease* (AD) y *Cognitive Normal* (CN).

4 Conclusiones y recomendaciones

4.1. Conclusiones

Durante el desarrollo del proyecto, se encontraron ciertas limitaciones en términos de capacidad computacional, los cuales afectaron la velocidad de procesamiento del modelo, dando paso a una reducción considerable del conjunto de datos total. Debido a este impedimento, fue necesario disminuir la complejidad del modelo en cuanto a los parámetros que lo componen. Sin embargo, estas modificaciones permitieron la obtención de un modelo estable capaz de entregar una precisión del 72,222% capaz de entregar una predicción acertada de los pacientes pertenecientes al grupo de *test*. Aunque se logró obtener tal porcentaje, se reconoce que aún puede hacerse avances. Es recomendable seguir con el entrenamiento del modelo utilizando una muestra más amplia del conjunto de datos total para mejorar aún más su rendimiento y precisión.

Se pudo demostrar que el modelo *K-Net95* basado en la estructura base de la red tipo *AlexNet3D* superó en rendimiento a las arquitecturas planteadas en investigaciones previas [56, 57]. Esto se evidencia en el porcentaje de precisión alcanzado por cada uno durante las etapas de entrenamiento y validación, los cuales respaldan la eficacia y superioridad del modelo desarrollado.

A lo largo del desarrollo de las fases de entrenamiento y validación, se resalta el efecto del *overfitting* y como afecta el rendimiento del modelo al permitir un ajuste excesivo a los datos de entrenamiento. Para contrarrestar este problema, se implementaron técnicas de regularización como la inclusión de *dropout* para mejorar la generalización del modelo y el aprendizaje de características específicas.

La elección de imágenes volumétricas 3D se realizó por diversas razones. Estas imágenes proporcionan una mejor contextualización de las regiones de interés al presentar características de los 3 cortes principales (transversal, sagital y coronal), permitiendo una percepción completa del estado cerebral. También permiten monitorear cambios sutiles en la estructura del cerebro a lo largo del tiempo, lo cual permite un acercamiento a una posible detección temprana. En el desarrollo del proyecto, permitieron un análisis longitudinal más detallado y preciso.

Por último, lo que se pretendió llevar a cabo en esta investigación fue el desarrollo de una herramienta de apoyo médico para un posible diagnóstico y seguimiento de pacientes en riesgo, lo que podría llevar a un tratamiento más temprano y efectivo.

4.2. Reconocimientos:

Agradecemos al Ministerio de Ciencias de Colombia por apoyar el proyecto 111089784907 CTO 897-2021, código de registro: 84907

Bibliografía

- [1] M. Z. Alom and T. M. e. a. Taha, “The history began from alexnet: A comprehensive survey on deep learning approaches,” *arXiv*, 2018. [Online]. Available: <http://arxiv.org/abs/1803.01164>
- [2] P. Varshney, “Alexnet architecture: A complete guide,” jul 2020. [Online]. Available: [blurredmachine/alexnet-architecture-a-complete-guide](https://blurredmachine.com/alexnet-architecture-a-complete-guide)
- [3] E. J. Topol, “High-performance medicine: the convergence of human and artificial intelligence,” *Nature medicine*, vol. 25, no. 1, p. 44–56, 2019. [Online]. Available: <https://www.nature.com/articles/s41591-018-0300-7>
- [4] M. M. Ahsan, S. A. Luna, and Z. Siddique, “Machine-learning-based disease diagnosis: A comprehensive review,” *Healthcare (Basel, Switzerland)*, vol. 10, no. 3, p. 541, 2022. [Online]. Available: <https://pubmed.ncbi.nlm.nih.gov/35327018/>
- [5] J. H. Fetzer and J. H. Fetzer, *What is Artificial Intelligence?* Springer, 1990.
- [6] Q. Zhou, X. Li, L. He, Y. Yang, G. Cheng, Y. Tong, L. Ma, and D. Tao, “trans,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 45, no. 6, p. 7853–7869, 2023. [Online]. Available: <https://arxiv.org/pdf/2201.05047.pdf>
- [7] J. Martínez Llamas, “Reconocimiento de imágenes mediante redes neuronales convolucionales,” Ph.D. dissertation, Universidad Politecnica de Madrid, Madrid, 2018.
- [8] M. Tan and Q. V. Le, “Efficientnet: Rethinking model scaling for convolutional neural networks,” *arXiv*, 2020.
- [9] M. Odusami, R. Maskeliūnas, R. Damaševičius, and T. Krilavičius, “Analysis of features of alzheimer’s disease: Detection of early stage from functional brain changes in magnetic resonance images using a finetuned resnet18 network,” *Diagnostics (Basel, Switzerland)*, vol. 11, no. 6, p. 1071, 2021. [Online]. Available: <https://www.mdpi.com/2075-4418/11/6/1071>
- [10] M. Velazquez and Y. Lee, “Multimodal ensemble model for alzheimer’s disease conversion prediction from early mild cognitive impairment subjects,” *Computers in biology and medicine*, vol. 151, no. Pt A, p. 106201, 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S001048252200909X>

- [11] C. Platero, M. C. Tobar, and A. D. N. Initiative, “Predicting alzheimer’s conversion in mild cognitive impairment patients using longitudinal neuroimaging and clinical markers,” *Brain imaging and behavior*, vol. 15, no. 4, p. 1728–1738, 2021. [Online]. Available: <https://pubmed.ncbi.nlm.nih.gov/33169305/>
- [12] S. Ghafoori and A. Shalbaf, “Predicting conversion from mci to ad by integration of rs-fmri and clinical information using 3d-convolutional neural network,” *International journal of computer assisted radiology and surgery*, vol. 17, no. 7, p. 1245–1255, 2022. [Online]. Available: <http://dx.doi.org/10.1007/s11548-022-02620-4>
- [13] E. D. G. Sinapsis, “7 ventajas del diagnóstico temprano en la enfermedad de alzheimer.” [Online]. Available: <https://www.gsinapsis.com/7-ventajas-del-diagnostico-temprano-en-la-enfermedad-de-alzheimer/>
- [14] M. d. S. M. IMSS, “Enfermedad de alzheimer: Tratamiento, recomendaciones y cuidado.” [Online]. Available: <http://www.imss.gob.mx/salud-en-linea/enfermedad-alzheimer>.
- [15] K. Ritchie, “Mild cognitive impairment: an epidemiological perspective,” *Dialogues in Clinical Neuroscience*, vol. 6, pp. 401–408, 04 2022.
- [16] Alzheimer’s Association, “What is dementia? related conditions: Mild cognitive impairment,” https://www.alz.org/alzheimers-dementia/what-is-dementia/related_conditions/mild-cognitive-impairment, Accessed 2023.
- [17] World Health Organization (WHO), “World failing to address dementia challenge,” <https://www.who.int/es/news/item/02-09-2021-world-failing-to-address-dementia-challenge>, 2021, accessed on June 6, 2023.
- [18] National Institute on Aging (NIA), “What causes alzheimer’s disease?” <https://www.nia.nih.gov/health/what-causes-alzheimers-disease>, Accessed 2023.
- [19] G. D. Rabinovici, “Late-onset alzheimer disease,” *Continuum: Lifelong Learning in Neurology*, vol. 25, no. 1, p. 14, 2019.
- [20] E. C. E. Mark W. Bondi and D. P. Salmon, “Alzheimer’s disease: Past, present, and future,” *ncbi*, 2020. [Online]. Available: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5830188>
- [21] A. M. Fagan, M. A. Mintun, R. H. Mach, S.-Y. Lee, and e. a. Dence, “Inverse relation between in vivo amyloid imaging load and cerebrospinal fluid abeta42 in humans,” *Annals of neurology*, vol. 59, no. 3, p. 512–519, 2006. [Online]. Available: <https://pubmed.ncbi.nlm.nih.gov/16372280/>

- [22] T. K. K. Ho, M. Kim, Y. Jeon, B. C. Kim, J. G. Kim, K. H. Lee, J.-I. Song, and J. Gwak, “Deep learning-based multilevel classification of alzheimer’s disease using non-invasive functional near-infrared spectroscopy,” *Frontiers in aging neuroscience*, vol. 14, p. 810125, 2022. [Online]. Available: <http://dx.doi.org/10.3389/fnagi.2022.810125>
- [23] S. L. Amir Ebrahimi, “Convolutional neural networks for alzheimer’s disease detection on mri images,” *ncbi*, 2020. [Online]. Available: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8083897>
- [24] e. a. Iacono, Resnick, “Mild cognitive impairment and asymptomatic alzheimer disease subjects,” *Journal of Neuropathology & Experimental Neurology*, vol. 73(4), p. 295–304, 2014.
- [25] e. a. Serrano. Cecilia Mariela. Dillon. Leis, “Deterioro cognitivo leve: riesgo de demencia según subtipos,” *Actas Españolas de Psiquiatría; Actas Españolas de Psiquiatría*, vol. 41, 2014.
- [26] M. Ganguli and e. a. Jia, “Mild cognitive impairment that does not progress to dementia: A population-based study,” *Journal of the American Geriatrics Society*, vol. 67, no. 2, p. 232–238, 2018.
- [27] P. Vemuri and C. R. Jack, “Role of structural mri in alzheimer’s disease,” *Alzheimer’s research & therapy*, vol. 2, no. 4, pp. 1–10, 2010.
- [28] P. Johnson, L. Vandewater, W. Wilson, P. Maruff, G. Savage, P. Graham, L. S. Macaulay, K. A. Ellis, C. Szoke, R. N. Martins *et al.*, “Genetic algorithm with logistic regression for prediction of progression to alzheimer’s disease,” *BMC bioinformatics*, vol. 15, pp. 1–14, 2014.
- [29] V. S. Rallabandi, K. Tulpule, M. Gattu, A. D. N. Initiative *et al.*, “Automatic classification of cognitively normal, mild cognitive impairment and alzheimer’s disease using structural mri analysis,” *Informatics in Medicine Unlocked*, vol. 18, p. 100305, 2020.
- [30] B. F. Marghalani and M. Arif, “Automatic classification of brain tumor and alzheimer’s disease in mri,” *Procedia Computer Science*, vol. 163, pp. 78–84, 2019.
- [31] G. Dimauro and L. Simone, “Novel biased normalized cuts approach for the automatic segmentation of the conjunctiva,” *Electronics*, vol. 9, no. 6, p. 997, 2020.
- [32] C. A. Robledo Marín, C. P. Duque Sierra, J. A. Hernández Calle, M. A. Ruiz Vélez, and R. B. Zapata Monsalve, “Envejecimiento, calidad de vida y políticas públicas en torno al envejecimiento y la vejez,” *CES Derecho*, vol. 13, no. 2, p. 132–160, 2022.
- [33] m. d. s. y. p. s. MinSalud, “Politica pública nacion de envejecimiento y vejez, dec. 681 de 2022,” *Minsalud*, 2022.

- [34] Minsalud, “Ley estatutaria 1618 de 2013,” in *Disposiciones para garantizar preno ejercicio de los derechos de la persona con discapacidad*, A. gaviria uribe, Ed., 2017. [Online]. Available: <https://www.minsalud.gov.co/sites/rid/Lists/BibliotecaDigital/RIDE/DE/PS/documento-balance-1618-2013-240517.pdf>
- [35] M. W. Weiner, D. P. Veitch, P. S. Aisen, L. A. Beckett, N. J. Cairns, R. C. Green, D. Harvey, C. R. Jack Jr, W. Jagust, E. Liu *et al.*, “The alzheimer’s disease neuroimaging initiative: A review of papers published since its inception,” *Alzheimer’s & Dementia*, vol. 9, no. 5, pp. e111–e194, 2013.
- [36] Y. Chen, X. Qian, Y. Zhang, W. Su, Y. Huang, X. Wang, X. Chen, E. Zhao, L. Han, and Y. Ma, “Prediction models for conversion from mild cognitive impairment to alzheimer’s disease: A systematic review and meta-analysis,” *Frontiers in aging neuroscience*, vol. 14, p. 840386, 2022. [Online]. Available: <http://dx.doi.org/10.3389/fnagi.2022.840386>
- [37] J. Chen, G. Chen, H. Shu, G. Chen, B. D. Ward, Z. Wang, D. Liu, P. G. Antuono, S.-J. Li, Z. Zhang, and A. D. N. Initiative, “Predicting progression from mild cognitive impairment to alzheimer’s disease on an individual subject basis by applying the care index across different independent cohorts,” *Aging*, vol. 11, no. 8, p. 2185–2201, 2019. [Online]. Available: <https://www.aging-us.com/article/101883/text>
- [38] N. I. of Biomedical Imaging and B. (NIBIB), “Imagen por resonancia magnética (irm).” [Online]. Available: <https://www.nibib.nih.gov/espanol/temas-cientificos/imagen-por-resonancia-magn%C3%A9tica-irm>
- [39] R. H. Hashemi, W. G. Bradley, and C. J. Lisanti, *MRI: the basics: The Basics*. Lippincott Williams & Wilkins, 2012.
- [40] T. A. W. Crawford K. L, Neu S. C., “Ida - the image and data archive at the laboratory of neuro imaging. neuroimage.” [Online]. Available: <https://ida.loni.usc.edu/>
- [41] B. Fischl, “Freesurfer,” *NeuroImage*, vol. 62, no. 2, pp. 774–781, 2012.
- [42] L. Henschel, S. Conjeti, S. Estrada, K. Diers, B. Fischl, and M. Reuter, “Fastsurfer - a fast and accurate deep learning based neuroimaging pipeline,” *NeuroImage*, vol. 219, no. 117012, p. 117012, 2020. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1053811920304985>
- [43] M.-S. J. D. Gallegos-Duarte M, “Free surfer.” [Online]. Available: https://www.researchgate.net/profile/Martin-Gallegos-Duarte/publication/282542006_Alteraciones_en_la_via_Ventral_relacionadas_con_el_estrabismo/links/5611ce8608aec422d117174b/Alteraciones-en-la-via-Ventral-relacionadas-con-el-estrabismo.pdf

- [44] A. I. Bootcamp, “Redes neuronales - bootcamp ai,” nov 2019. [Online]. Available: <https://bootcampai.medium.com/redes-neuronales-13349dd1a5bb>
- [45] P. Probst, A.-L. Boulesteix, and B. Bischl, “Tunability: Importance of hyperparameters of machine learning algorithms,” *The Journal of Machine Learning Research*, vol. 20, no. 1, pp. 1934–1965, 2019.
- [46] S. Ji, M. Yang, and K. Yu, “3d convolutional neural networks for human action recognition,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 1, p. 221–231, 2013. [Online]. Available: https://www.dbs.ifl.lmu.de/~yu_k/icml2010_3dcnn.pdf
- [47] S. Rani, D. Ghai, S. Kumar, M. P. Kantipudi, A. H. Alharbi, and M. A. Ullah, “Efficient 3d alexnet architecture for object recognition using syntactic patterns from medical images,” *Computational intelligence and neuroscience*, vol. 2022, p. 7882924, 2022. [Online]. Available: <https://www.hindawi.com/journals/cin/2022/7882924/>
- [48] S. Sharma, “Activation functions in neural networks,” sep 2017. [Online]. Available: <https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6>
- [49] F. Chollet *et al.*, “Keras,” <https://keras.io>, 2015.
- [50] I. Idris, *NumPy: Beginner’s guide - third edition*, 3rd ed. Birmingham, England: Packt Publishing, 2015. [Online]. Available: <https://books.google.at/books?id=m2T9CQAAQBAJ>
- [51] J. D. Hunter, “Matplotlib: A 2d graphics environment,” *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007.
- [52] T. pandas development team, “pandas-dev/pandas: Pandas,” Feb. 2020. [Online]. Available: <https://doi.org/10.5281/zenodo.3509134>
- [53] M. Brett, C. J. Markiewicz, M. Hanke, M.-A. Côté, and B. . e. a. Cipollini, “nipy/nibabel: 3.2.1,” Nov. 2020. [Online]. Available: <https://doi.org/10.5281/zenodo.4295521>
- [54] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, “TensorFlow: Large-scale machine learning on heterogeneous systems,” 2015, software available from tensorflow.org. [Online]. Available: <https://www.tensorflow.org/>

- [55] H. Kataoka, T. Wakamiya, K. Hara, and Y. Satoh, “Would mega-scale datasets further enhance spatiotemporal 3d cnns?” 2020. [Online]. Available: <http://arxiv.org/abs/2004.04968>
- [56] F. Milletari, N. Navab, and S.-A. Ahmadi, “V-net: Fully convolutional neural networks for volumetric medical image segmentation,” 2016. [Online]. Available: <http://arxiv.org/abs/1606.04797>
- [57] N. Brancati, G. De Pietro, M. Frucci, and D. Riccio, “A deep learning approach for breast invasive ductal carcinoma detection and lymphoma multi-classification in histological images,” *IEEE access: practical innovations, open solutions*, vol. 7, p. 44709–44720, 2019. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/8678759/references#null-all-ref>
- [58] J. Chen, Z. Wan, J. Zhang, W. Li, Y. Chen, Y. Li, and Y. Duan, “Medical image segmentation and reconstruction of prostate tumor based on 3d alexnet,” *Computer methods and programs in biomedicine*, vol. 200, no. 105878, p. 105878, 2021. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0169260720317119>

5 Anexos

Anexo A: Algoritmos modificados durante la fase de entrenamiento

```
model = Sequential()
model.add(Conv3D(8, kernel_size=(3, 3, 3),
                activation='relu',
                kernel_initializer='he_uniform',
                input_shape=(X_train.shape[1],
                            X_train.shape[2],
                            X_train.shape[3],
                            X_train.shape[4])))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.7))
model.add(Conv3D(4, kernel_size=(3, 3, 3), activation='relu',
                kernel_initializer='he_uniform'))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.7))
model.add(Conv3D(2, kernel_size=(3, 3, 3), activation='relu',
                kernel_initializer='he_uniform'))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.7))
model.add(Flatten())
model.add(Dense(30, activation='sigmoid',
                kernel_initializer='he_uniform'))
model.add(Dense(30, activation='relu',
                kernel_initializer='he_uniform'))
model.add(Dense(2, activation='softmax'))
```

Algorithm 2: Modelo modificado para la prueba 4.

```
model = Sequential()
model.add(Conv3D(8, kernel_size=(3, 3, 3),
                activation='relu',
                kernel_initializer='he_uniform',
                input_shape=(X_train.shape[1],
                            X_train.shape[2],
                            X_train.shape[3],
                            X_train.shape[4])))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.7))
model.add(Conv3D(4, kernel_size=(3, 3, 3), activation='relu',
                kernel_initializer='he_uniform',
                kernel_regularizer=regularizers.L2(0.01)))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.7))
model.add(Conv3D(2, kernel_size=(3, 3, 3), activation='relu',
                kernel_initializer='he_uniform',
                kernel_regularizer=regularizers.L2(0.01)))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.7))
model.add(Flatten())
model.add(Dense(30, activation='sigmoid',
                kernel_initializer='he_uniform'))
model.add(Dense(30, activation='relu',
                kernel_initializer='he_uniform'))
model.add(Dense(2, activation='softmax'))
```

Algorithm 3: Modelo modificado para la prueba 5.

```
model = Sequential()
model.add(Conv3D(16, kernel_size=(3, 3, 3),
                activation='relu',
                kernel_initializer='he_uniform',
                input_shape=(X_train.shape[1],
                            X_train.shape[2],
                            X_train.shape[3],
                            X_train.shape[4])))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.7))
model.add(Conv3D(8, kernel_size=(3, 3, 3), activation='relu',
                kernel_initializer='he_uniform',
                kernel_regularizer=regularizers.L2(0.01)))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.7))
model.add(Conv3D(4, kernel_size=(3, 3, 3), activation='relu',
                kernel_initializer='he_uniform',
                kernel_regularizer=regularizers.L2(0.01)))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.5))
model.add(Conv3D(2, kernel_size=(3, 3, 3), activation='relu',
                kernel_initializer='he_uniform',
                kernel_regularizer=regularizers.L2(0.01)))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.5))
model.add(Flatten())
model.add(Dense(30, activation='sigmoid',
                kernel_initializer='he_uniform'))
model.add(Dense(30, activation='relu',
                kernel_initializer='he_uniform'))
model.add(Dense(2, activation='softmax'))
```

Algorithm 4: Modelo modificado para la prueba 6.

```
model = Sequential()
model.add(Conv3D(16, kernel_size=(3, 3, 3),
                activation='relu',
                kernel_initializer='he_uniform',
                input_shape=(X_train.shape[1],
                            X_train.shape[2],
                            X_train.shape[3],
                            X_train.shape[4])))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.7))
model.add(Conv3D(8, kernel_size=(3, 3, 3), activation='relu',
                kernel_initializer='he_uniform'))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.7))
model.add(Conv3D(4, kernel_size=(3, 3, 3), activation='relu',
                kernel_initializer='he_uniform'))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.5))
model.add(Conv3D(2, kernel_size=(3, 3, 3), activation='relu',
                kernel_initializer='he_uniform'))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.5))
model.add(Flatten())
model.add(Dense(100, activation='sigmoid',
                kernel_initializer='he_uniform',
                kernel_regularizer=regularizers.L2(0.01)))
model.add(Dense(50, activation='relu',
                kernel_initializer='he_uniform'))
model.add(Dense(2, activation='softmax'))
```

Algorithm 5: Modelo modificado para la prueba 7.

```
model = Sequential()
model.add(Conv3D(32, kernel_size=(3, 3, 3),
                activation='relu',
                kernel_initializer='he_uniform',
                input_shape=(X_train.shape[1],
                            X_train.shape[2],
                            X_train.shape[3],
                            X_train.shape[4])))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.7))
model.add(Conv3D(16, kernel_size=(3, 3, 3), activation='relu',
                kernel_initializer='he_uniform'))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.7))
model.add(Conv3D(8, kernel_size=(3, 3, 3), activation='relu',
                kernel_initializer='he_uniform'))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.5))
model.add(Conv3D(4, kernel_size=(3, 3, 3), activation='relu',
                kernel_initializer='he_uniform'))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.5))
model.add(Flatten())
model.add(Dense(100, activation='sigmoid',
                kernel_initializer='he_uniform',
                kernel_regularizer=regularizers.L2(0.01)))
model.add(Dense(50, activation='relu',
                kernel_initializer='he_uniform'))
model.add(Dense(2, activation='softmax'))
```

Algorithm 6: Modelo modificado para la prueba 8.

```

model = Sequential()
model.add(Conv3D(32, kernel_size=(3, 3, 3),
                activation='relu',
                kernel_initializer='he_uniform',
                input_shape=(X_train.shape[1], X_train.shape[2],
                            X_train.shape[3], X_train.shape[4])))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.5))
model.add(Conv3D(8, kernel_size=(3, 3, 3), activation='relu',
                kernel_initializer='he_uniform'))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.5))
model.add(Conv3D(4, kernel_size=(3, 3, 3), activation='relu',
                kernel_initializer='he_uniform'))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.5))
model.add(Conv3D(2, kernel_size=(3, 3, 3), activation='relu',
                kernel_initializer='he_uniform'))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.4))
model.add(Flatten())
model.add(Dense(1024, activation=LeakyReLU(alpha=0.01),
                kernel_initializer='he_uniform',
                kernel_regularizer=regularizers.L2(0.01)))
model.add(Dropout(0.6))
model.add(Dense(512, activation=LeakyReLU(alpha=0.01),
                kernel_initializer='he_uniform',
                kernel_regularizer=regularizers.L2(0.01)))
model.add(Dropout(0.7))
model.add(Dense(128, activation=LeakyReLU(alpha=0.01),
                kernel_initializer='he_uniform',
                kernel_regularizer=regularizers.L2(0.01)))
model.add(Dropout(0.6))
model.add(Dense(2, activation='softmax'))

```

Algorithm 7: Modelo modificado para la prueba 9.

```

model = Sequential()
model.add(Conv3D(32, kernel_size=(3, 3, 3),
                activation='relu',
                kernel_initializer='he_uniform',
                kernel_regularizer=regularizers.L2(0.001),
                input_shape=(X_train.shape[1], X_train.shape[2],
                            X_train.shape[3], X_train.shape[4])))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.4))
model.add(Conv3D(16, kernel_size=(3, 3, 3), activation='relu',
                kernel_initializer='he_uniform',
                kernel_regularizer=regularizers.L2(0.001)))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.4))
model.add(Conv3D(8, kernel_size=(3, 3, 3), activation='relu',
                kernel_initializer='he_uniform'))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.3))
model.add(Conv3D(8, kernel_size=(3, 3, 3), activation='relu',
                kernel_initializer='he_uniform'))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(BatchNormalization(center=True, scale=True))
model.add(Dropout(0.2))
model.add(Flatten())
model.add(Dense(512, activation=LeakyReLU(alpha=0.01),
                kernel_initializer='he_uniform',
                kernel_regularizer=regularizers.L2(0.001)))
model.add(Dropout(0.1))
model.add(Dense(128, activation=LeakyReLU(alpha=0.01),
                kernel_initializer='he_uniform',
                kernel_regularizer=regularizers.L2(0.001)))
model.add(Dropout(0.1))
model.add(Dense(2, activation='softmax'))

```

Algorithm 8: Modelo modificado para la prueba 10.

```

model = Sequential()
model.add(Conv3D(8, kernel_size=(3, 3, 3), strides=(1, 1, 1),
                activation='relu', padding='same',
                input_shape=(X_train.shape[1],
                            X_train.shape[2],
                            X_train.shape[3],
                            X_train.shape[4])))
model.add(BatchNormalization(center=True, scale=True))
model.add(Conv3D(8, kernel_size=(3, 3, 3), strides=(2, 2, 2),
                activation='relu', padding='same'))
model.add(BatchNormalization())
model.add(Conv3D(8, kernel_size=(3, 3, 3), strides=(1, 1, 1),
                activation='relu', padding='same'))
model.add(BatchNormalization())
model.add(Conv3D(16, kernel_size=(3, 3, 3), strides=(2, 2, 2),
                activation='relu', padding='same'))
model.add(BatchNormalization())
model.add(Conv3D(16, kernel_size=(3, 3, 3), strides=(1, 1, 1),
                activation='relu', padding='same'))
model.add(BatchNormalization())
model.add(Conv3D(32, kernel_size=(3, 3, 3), strides=(2, 2, 2),
                activation='relu', padding='same'))
model.add(BatchNormalization())
model.add(Conv3D(32, kernel_size=(3, 3, 3), strides=(1, 1, 1),
                activation='relu', padding='same'))
model.add(BatchNormalization())
model.add(Conv3D(64, kernel_size=(3, 3, 3), strides=(2, 2, 2),
                activation='relu', padding='same'))
model.add(BatchNormalization())
model.add(Conv3D(64, kernel_size=(3, 3, 3), strides=(1, 1, 1),
                activation='relu', padding='same'))#512 x 64
model.add(BatchNormalization())
model.add(GlobalAveragePooling3D())
model.add(Dense(2, activation='softmax'))

```

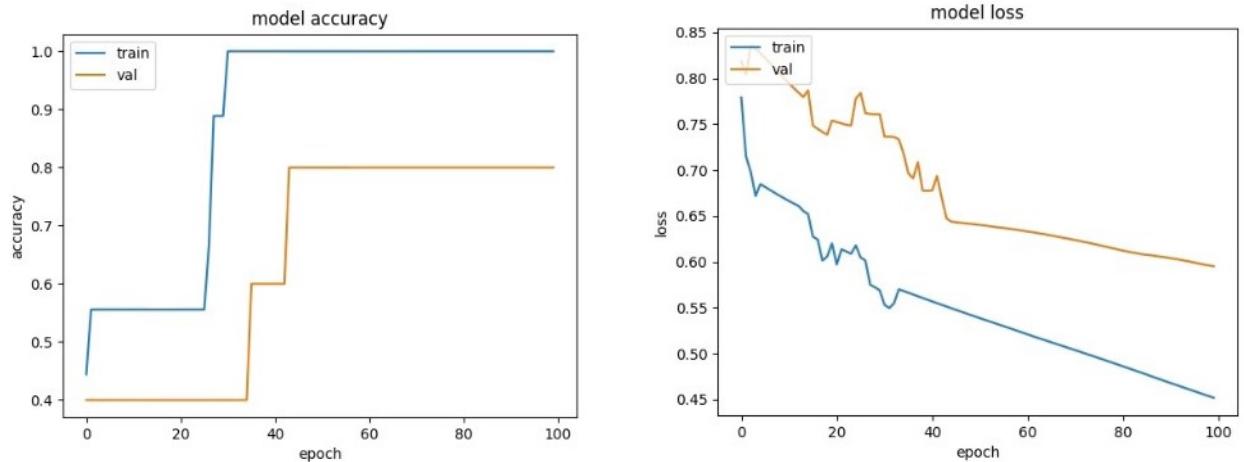
Algorithm 9: Modelo de red neuronal tipo ResNet 3D.

```
model = Sequential()
model.add(Conv3D(8, kernel_size=(3, 3, 3),
                activation='relu', padding='same',
                input_shape=(X_train.shape[1], X_train.shape[2],
                            X_train.shape[3], X_train.shape[4])))
model.add(Conv3D(8, kernel_size=(3, 3, 3),
                activation='relu', padding='same'))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(Conv3D(16, kernel_size=(3, 3, 3),
                activation='relu', padding='same'))
model.add(Conv3D(16, kernel_size=(3, 3, 3),
                activation='relu', padding='same'))
model.add(Dropout(0.5))
model.add(MaxPooling3D(pool_size=(2, 2, 2)))
model.add(Conv3D(32, kernel_size=(3, 3, 3),
                activation='relu', padding='same'))
model.add(Conv3D(32, kernel_size=(3, 3, 3),
                activation='relu', padding='same'))
model.add(Dropout(0.5))
model.add(UpSampling3D(size=(2, 2, 2)))
model.add(Conv3D(16, kernel_size=(3, 3, 3),
                activation='relu', padding='same'))
model.add(Conv3D(16, kernel_size=(3, 3, 3),
                activation='relu', padding='same'))
model.add(UpSampling3D(size=(2, 2, 2)))
model.add(Conv3D(8, kernel_size=(3, 3, 3),
                activation='relu', padding='same'))
model.add(Conv3D(8, kernel_size=(3, 3, 3),
                activation='relu', padding='same'))
model.add(Conv3D(2, kernel_size=(1, 1, 1),
                activation='softmax'))

model.add(GlobalAveragePooling3D())
model.add(Dense(2, activation='softmax'))
```

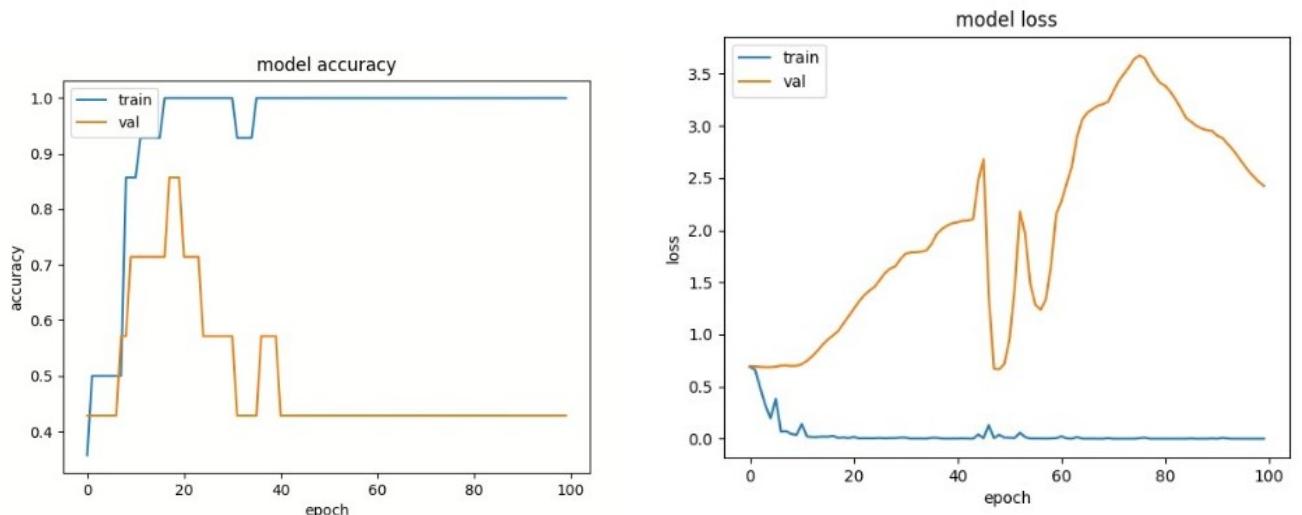
Algorithm 10: Modelo de red neuronal tipo UNet 3D.

Anexo B: Curvas de aprendizaje (Precisión y Pérdida)



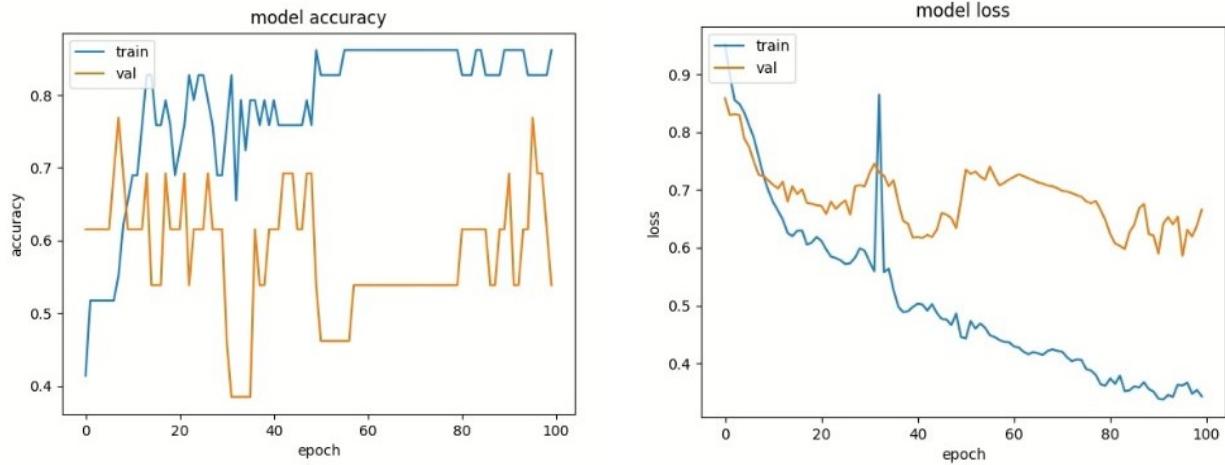
- (a) Precisión en etapa de entrenamiento y validación en la prueba 1.
(b) Pérdida en etapa de entrenamiento y validación en la prueba 1.

Figura 5-1: Curvas de aprendizaje de la prueba 1.



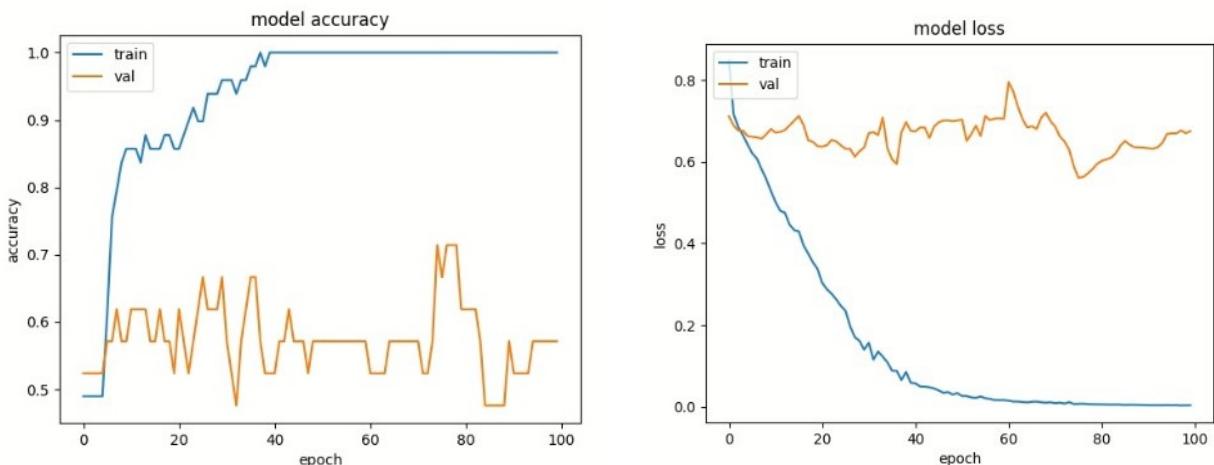
- (a) Precisión en etapa de entrenamiento y validación en la prueba 2.
(b) Pérdida en etapa de entrenamiento y validación en la prueba 2.

Figura 5-2: Curvas de aprendizaje de la prueba 2.



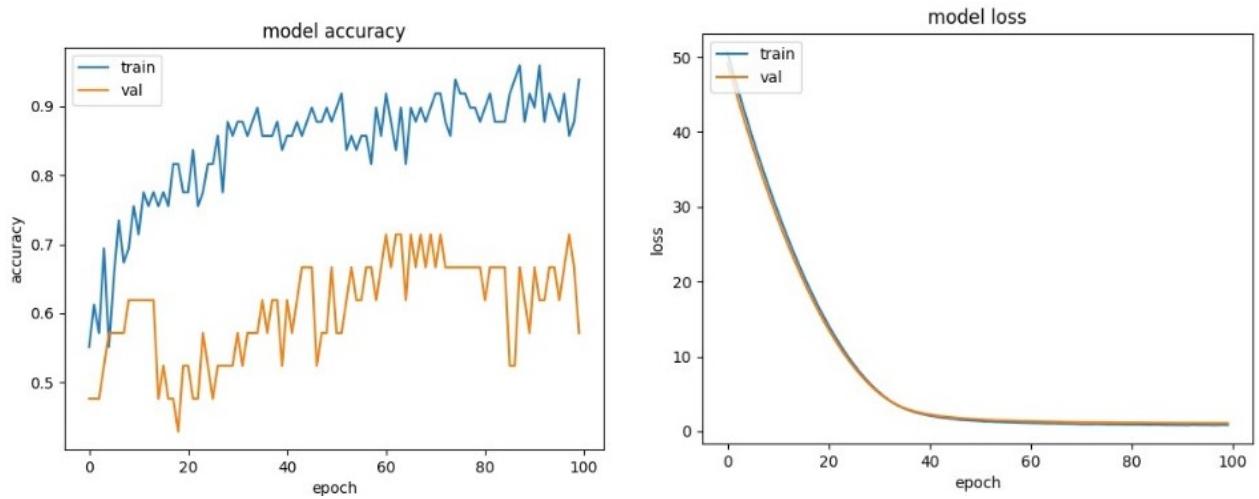
- (a) Precisión en etapa de entrenamiento y validación en la prueba 3.
(b) Pérdida en etapa de entrenamiento y validación en la prueba 3.

Figura 5-3: Curvas de aprendizaje de la prueba 3.



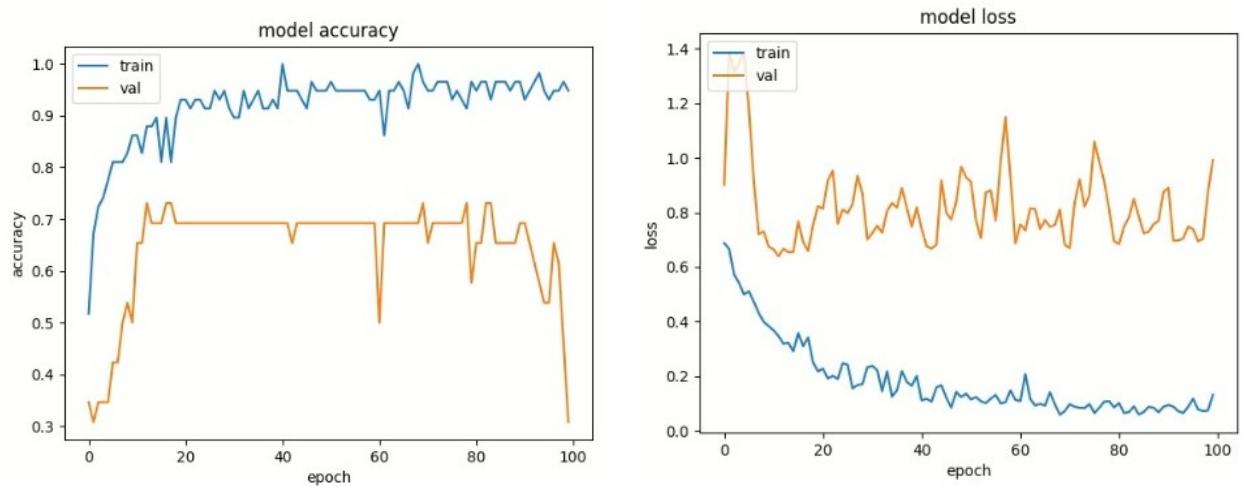
- (a) Precisión en etapa de entrenamiento y validación en la prueba 4.
(b) Pérdida en etapa de entrenamiento y validación en la prueba 4.

Figura 5-4: Curvas de aprendizaje de la prueba 4.



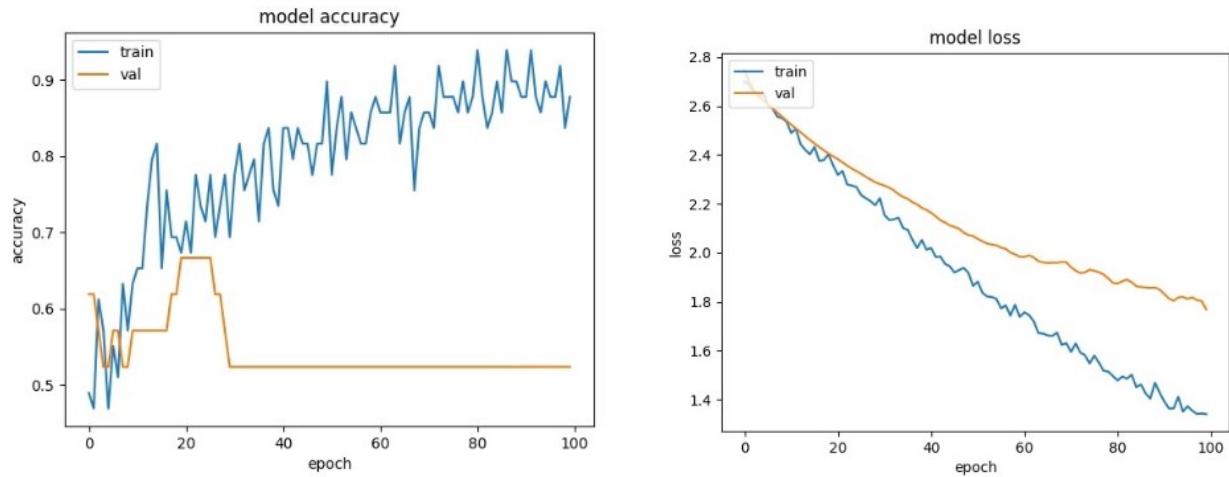
- (a) Precisión en etapa de entrenamiento y validación en la prueba 5.
(b) Pérdida en etapa de entrenamiento y validación en la prueba 5.

Figura 5-5: Curvas de aprendizaje de la prueba 5.



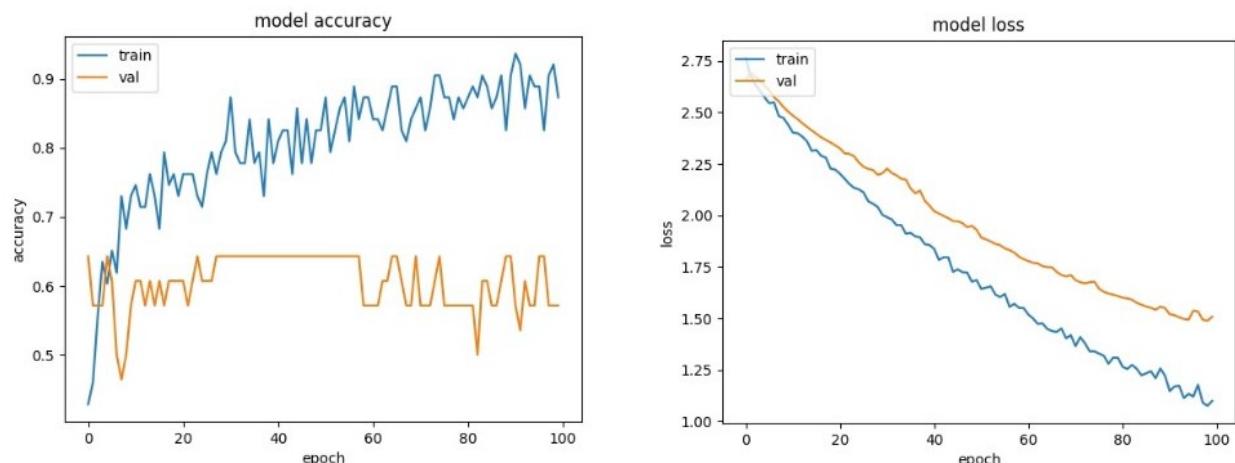
- (a) Precisión en etapa de entrenamiento y validación en la prueba 6.
(b) Pérdida en etapa de entrenamiento y validación en la prueba 6.

Figura 5-6: Curvas de aprendizaje de la prueba 6.



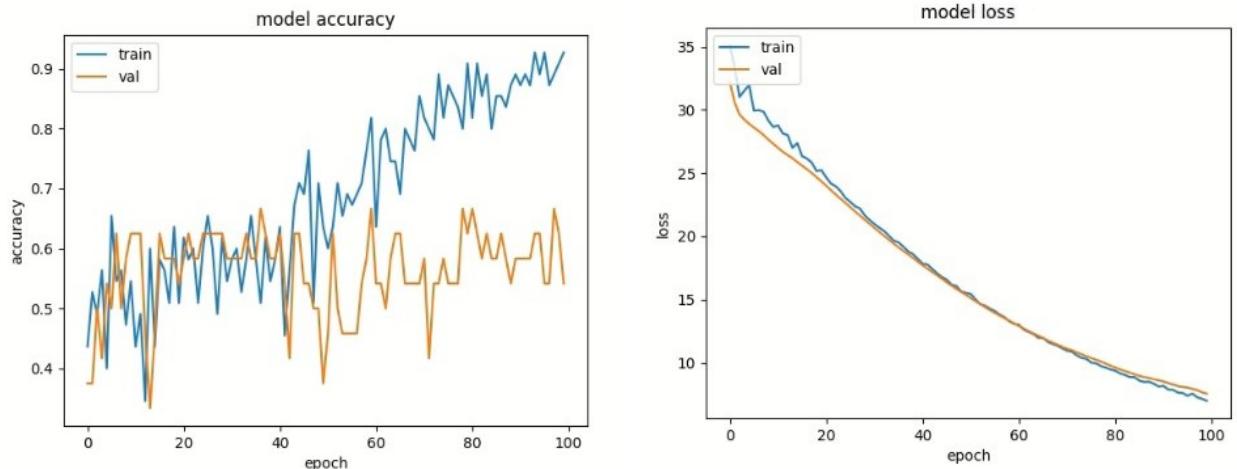
- (a) Precisión en etapa de entrenamiento y validación en la prueba 7.
(b) Pérdida en etapa de entrenamiento y validación en la prueba 7.

Figura 5-7: Curvas de aprendizaje de la prueba 7.



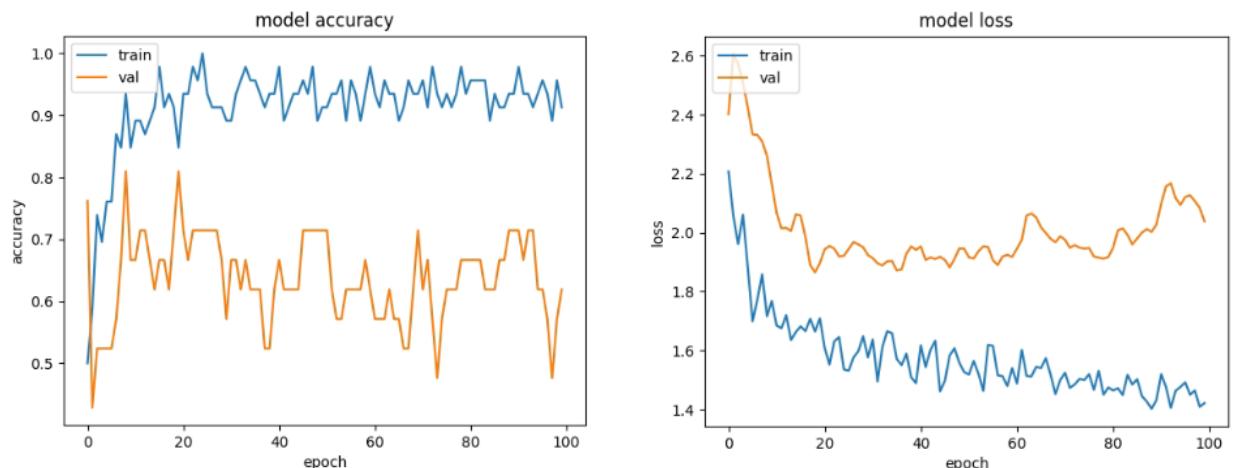
- (a) Precisión en etapa de entrenamiento y validación en la prueba 8.
(b) Pérdida en etapa de entrenamiento y validación en la prueba 8.

Figura 5-8: Curvas de aprendizaje de la prueba 8.



(a) Precisión en etapa de entrenamiento y validación en la prueba 9.
(b) Pérdida en etapa de entrenamiento y validación en la prueba 9.

Figura 5-9: Curvas de aprendizaje de la prueba 9.



(a) Precisión en etapa de entrenamiento y validación en la prueba 10.
(b) Pérdida en etapa de entrenamiento y validación en la prueba 10.

Figura 5-10: Curvas de aprendizaje de la prueba 10.