UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf

Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Análisis de imágenes médicas para dar soporte al diagnóstico de Alzheimer

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

Autor: Rafael Vicente Sánchez Romero

Tutores: Jon Ander Gómez Adrián, María de la Iglesia Vayá

Curso 2018-2019

Resum

L'anàlisi d'imatge mèdica és, cada vegada més, una eina que s'està convertint en un suport per als metges a l'hora d'emetre un diagnòstic.

En aquest treball, abordem el tema de l'anàlisi d'imatge mèdica per proposar un classificador que siga capaç de distingir entre els pacients que pateixen d'Alzheimer i els que no. Per a la construcció del classificador es proposarà l'ús de tècniques d'intel·ligència artificial (IA), en concret de les xarxes neuronals convolucionals (CNN).

A més, també es revisaran diferents tècniques de curació de les dades d'entrada que són necessàries per a la correcta classificació de les mostres.

Paraules clau: Inteligència Artificial, Xarxes Neuronals, Alzheimer, Imatges Mèdiques, Diagnòstic mèdic.

Resumen

El análisis de imagen médica es, cada vez más, una herramienta que se está convirtiendo en un apoyo para los médicos a la hora de emitir un diagnóstico.

En este trabajo, abordamos el tema del análisis de imagen médica para proponer un clasificador que sea capaz de distinguir entre los pacientes que padecen Alzheimer y los que no. Para la construcción del clasificador se propondrá el uso de técnicas de inteligencia artificial (IA), en concreto de las redes neuronales convoulcionales (CNN).

Además, también se revisarán diferentes técnicas de curación de los datos de entrada que son necesarias para la correcta clasificación de las muestras.

Palabras clave: Inteligencia Artificial, Redes Neuronales, Alzheimer, Imágenes Médicas, Diagnóstico Médico.

Abstract

The medical images analysis is a tool which is increasingly becoming a support for doctors when it comes the time to issue a diagnosis.

In this work, we address the medical images analysis issue in order to propose a classifier that is able to distinguish between patients suffering from Alzheimer's and those who do not. For the classifier development, the use of artificial intelligence (AI) techniques will be proposed, specifically we propose the use of convolutional neural networks (CNN).

In addition, different data curation techniques will be reviewed because they are needed for the correct classificaction of the samples.

Key words: Artificial Intelligence, Neural Networks, Alzheimer's Disesase, Medical Imaging, Medical Diagnosis.

Índice general

Índice de figuras

Índice de tablas

CAPÍTULO 1
Introducción

La enfermedad de Alzheimer (AD) es la causa más común de demencia. Esta enfermedad mental genera un gran deterioro cognitivo que se caracteriza por la pérdida de memoria, sufrir alteraciones en el lenguaje o la dificultad del paciente para hacer planes o resolver problemas. Además de todos estos síntomas, los pacientes también pueden desarrollar cambios de comportamiento y de personalidad, llegando a perder su identidad en estadíos avanzados de la enfermedad.

En la actualidad, no hay causas conocidas de la enfermedad y, como resultado de ello, no existen aún tratamientos que puedan prevenirla, pararla o curarla completamente. Las investigaciones, como el Estudio Alfa de la Fundación Pasqual Maragall, se están centrando en conocer cuál es el desarrollo natural de la enfermedad. Para ello, entre otras, se hace uso de pruebas de imagen médica y justo en estas pruebas se centrará el estudio del presente trabajo. [1]

Según la Organización Mundial de la Salud (OMS) [2], la demencia afecta mundialmente a unos 50 millones de personas creciendo este número en unos 10 millones de casos nuevos cada año. La propia OMS calcula que para el año 2030 el número de casos de demencia alcance los 82 millones. Más concretamente, de todos los casos de demencia existentes, entre un 60 % y un 70 % de ellos son Alzheimer.

Este proyecto ha sido desarrollado en colaboración con dos compañeros del grado: Álvaro López Chilet y Silvia Nadal Almela. Una primera parte del proyecto ha sido realizada en conjunto; ésta ha consistido principalmente en el análisis de los datos de los que disponíamos, investigación del estado del arte, preprocesado de los datos y unos primeros experimentos de clasificación. Posteriormente, cada uno se ha centrado en un tipo de experimentación concreto según el tipo de datos utilizados: Álvaro se ha centrado en experimentar con imágenes en tres dimensiones aplicando diferentes preprocesados (BET y FLIRT), Silvia se centró en la experimentación con los datos de volumetría obtenidos gracias a FreeSurfer y yo me he centrado en la experimentación de las imágenes en 2D aplicando diferentes preprocesados (BET y FLIRT). Además, dentro de la Unidad Mixta de Imagen Biomédica FISABIO-CIPF, ha colaborado también José Manuel Saborit.

1.1 Planteamiento del problema

El problema que se pretende abordar es la agilización, mediante la utilización de Inteligencia Artificial (IA), del proceso de diagnóstico médico de la enfermedad de Alzheimer. En la actualidad, a parte de las pruebas clínicas convencionales que por lo general son económicamente costosas y alugunas de ellas invasivas para el paciente, está tomando fuerza el campo de los sistemas de diagnóstico médico guiados por computador. Es en

este último campo donde este trabajo encuentra su motivación. Así pues, en este proyecto se proponen varios modelos de Deep Learning (DL) que, haciendo uso de resonancias magnéticas de cerebro[1], sean capaz de discriminar aquellos pacientes que padecen de Alzheimer de los que no e incluso detectar si el paciente se encuentra en un estado intermedio como puede ser el deterioro cognitivo medio o leve.

1.2 Objetivos

En este trabajo final de grado se plantean tres objetivos:

- Estudiar aproximaciones y soluciones recientemente publicadas por varios investigadores y reproducir algunas de ellas para contrastar resultados.
- Analizar y probar distintas técnicas de preprocesado de datos para decidir, en base a los resultados obtenidos, cuál o cuales son las más apropiadas para el problema que se aborda en este trabajo.
- Diseñar varias topologías de redes neuronales profundas y evaluar los resultados obtenidos utilizando el conjunto de datos ADNI[2] para poder compararlos con los resultados publicados por otros autores sobre el mismo conjunto de datos.

1.3 Estructura del trabajo

La estructura de este trabajo se divide en cinco capítulos. En el Capítulo 2 se realiza una revisión del estado del arte. A continuación, en el Capítulo 3, se describe el problema que se pretende abordar con profundidad y se explican las soluciones que se proponen. En el Capítulo 4, revisamos las herramientas software y hardware que se han utilizado a lo largo de todo el proyecto. Seguidamente, en el Capítulo 5, se comentan los experimentos realizados junto con los resultados obtenidos de los mismos. Por último, en el Capítulo 6, se plantean las conclusiones extraídas a lo largo del proyecto y las posibles líneas futuras de trabajo. Al final se incluyen dos anexos: en el Apéndice A se amplia la explicación de diferentes capas, funciones de activación y técnicas de regularización que se han utilizado durante el proyecto. En el Apéndice B se incluye información acerca de los *datasets* utilizados.

[1] La resonancia magnética (MRI por sus siglas en inglés), es una técnica médica que utiliza campos magnéticos y ondas de radio para crear una imagen 3D detallada del cerebro.

[2] Alzheimer's Disease Neuroimaging Initiative. Este *dataset* se introduce en el Capítulo 5 del presente trabajo.

CAPÍTULO 2
Estado del arte

El análisis de imagen médica cerebral para caracterizar la evolución del Alzheimer, así como para entrenar clasificadores que reconozcan la enfermedad a partir de dichas imágenes vivió un gran impulso a partir del año 2003. Fue en este año cuando se puso en marcha la *Alzheimer's Disease Neuroimaging Initiative* (ADNI) como una colaboración público-privada liderada por el investigador principal Michael W. Weiner. Esta iniciativa puso al alcance de cualquier investigadora o investigador un amplio *dataset* de imágenes MRI, PET y algunos datos clínicos de pacientes, lo que incentivó a la comunidad científica a trabajar en este problema. [14]

En este capítulo nos centraremos en tres aspectos fundamentales de las soluciones que conforman el estado del arte. En primer lugar analizaremos los datos que se utilizan, así como las técnicas de preprocesado y "curación" a las que se someten los conjuntos de datos antes de pasar por las redes neuronales. Por otro lado, analizaremos las topologías y los modelos que mejores resultados han tenido haciendo un breve comentario de sus puntos fuertes. Para finalizar, realizaremos una breve crítica sobre la producción científica en esta materia que podría mejorar en algunos aspectos.

2.1 Datos y preprocesado

En la gran mayoría de los casos, los datos se extraen del *dataset* de la iniciativa ADNI. Aunque en el Capítulo 5 se describe en detalle el *dataset*, cabe destacar que las imágenes incluidas en ADNI pueden ser 2D o 3D. En las soluciones que se han consultado para realizar este trabajo se trabaja con ambos formatos, obteniendo resultados ligeramente mejores en aquellas soluciones que trabajan con imágenes en tres dimensiones, como se demostrará más adelante.

Por otro lado, entre las técnicas de preprocesado es habitual realizar siempre una normalización de intensidades con el objetivo de llevarlas todas a un nuevo espacio donde puedan ser comparadas. Las imágenes obtenidas por resonancia magnética, normalmente se encuentran en formato NIFTI[1]. Los píxeles de las imágenes, pues, tienen valores que representan la intensidad de campo electromagnético que se aplicó para obtener la imagen, en el punto del espacio que representan. A parte de la normalización de intensidad, la mayoría de experimentos pasan las imágenes por una o varias herramientas de la librería FSL[2]. Las herramientas más utilizadas son BET[3] y FLIRT[4]. La primera se encarga de realizar la extracción del parénquima cerebral. De esta manera nos quedamos única-

[1] Neuroimaging Informatics Technology Initiative.
[2] La librería FSL se comenta con más detalle en el apartado 4.3 de este trabajo.
[3] La herramienta BET se comenta con más detalle en el apartado 4.3.1 de este trabajo.
[4] La herramienta FLIRT se comenta con más detalle en el apartado 4.3.2 de este trabajo.

mente con la zona de interés para nuestro problema. La segunda herramienta se encarga de realizar una transformación de las imágenes para llevarlas a un mismo espacio. El espacio de uno de los mapas de referencia que se pueden seleccionar. Esto, como veremos más adelante, consigue que la estructura cerebral característica de cada paciente no sea el aspecto relevante que aprende la red, sino que ésta se centra en sacar las características propias de la enfermedad. Típicamente, FLIRT suele utilizarse junto con BET ya que los mapas cerebrales de transformación no son más que imágenes cerebrales a las que se les ha extraído el parénquima.

Además de todo esto, algunas soluciones únicamente utilizan las imágenes para extraer datos de volumetría cerebral a partir de ellas. Son los propios datos volumétricos, extraídos gracias a la herramienta FreeSurfer[5] los que se introducen a las redes neuronales para analizarse.

2.2 Modelos y topologías

Principalmente los modelos de redes neuronales que se utilizan en los experimentos del estado del arte se pueden clasificar en dos grupos. Por un lado, aquellos modelos que se construyen desde cero, específicamente diseñados para un problema de clasificación. Por otro, aquellas soluciones que utilizan modelos de redes neuronales que ya han sido entrenadas para otros propósitos y que, por tanto, tienen ya unos pesos entrenados. De esta manera, mediante la técnica de *transfer learning* se modifica ligeramente el modelo para adaptarlo al nuevo problema a solucionar.

Como ejemplo del primer caso, podemos citar el contenido del artículo de Wang y Shen [10]. Estos autores proponen un modelo que se diseña específicamente para resolver el problema que nos atañe. Para ello, proponen el diseño de un bloque que va concatenando tensores un número determinado de veces y que se intercala con convoluciones. El resultado de esta primera parte se introduce en una red *fully-connected* que es la encargada de la clasificación en su última capa con función de activación *soft-max*. Se puede ver el modelo que se propone en dicho artículo en la Figura 2.1, extraída del mismo artículo

En el trabajo de Jain y Aggarwal [12] vemos un ejemplo del segundo caso que se propone. En esta ocasión los autores deciden partir de la conocida red VGG16 [13] que en Keras viene con pesos entrenados para el *dataset* de Imagenet y mediante *transfer learning* solucionar el nuevo problema. Para ello, se retira la parte encargada de la clasificación de las muestras (*fully-connected*) de la VGG16 que, originalmente, está preparada para clasificar las imágenes en 1000 clases diferentes y se le adhiere una nueva para clasificar las muestras en dos clases (AD vs. CN).

Además, estos dos trabajos en los que se ha puesto énfasis, también nos permiten ver cómo se puede trabajar tanto con las imágenes en tres dimensiones [10] como en dos dimensiones [12]. Y los resultados que se obtienen para ambos casos son bastante parecidos. Se pueden consultar en la Tabla 2.1

Tabla 2.1: Comparación entre las tasas de acierto de [10] y [12] tanto para los experimentos 3-*way* como para AD vs. CN.

Experimento/Trabajo	*Accuracy* [10]	*Accuracy* [12]
3-way	**97.52 %**	95.73 %
AD vs. CN	98.83 %	**99.14 %**

[5] La herramienta FreeSurfer se comenta con más detalle en el apartado 4.4 de este trabajo.

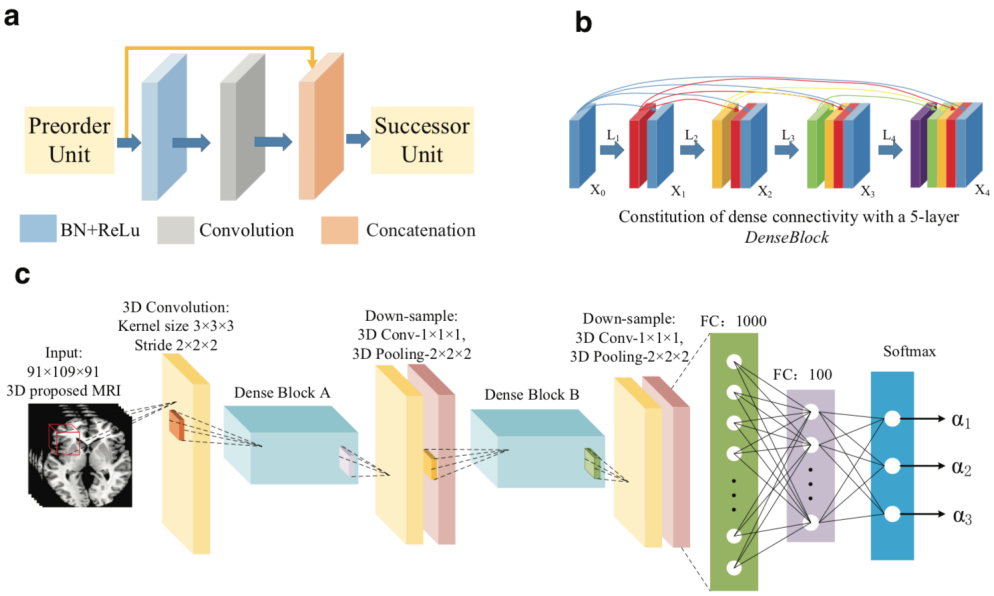


Figura 2.1: Esquema del modelo que se propone en [10]. En (a) podemos ver la construcción de un elemento del tipo *DenseBlock*. En (b) vemos como se construye un bloque del tipo *DenseBlock*. Por último en (c) vemos como se construye todo el modelo que se propone. Imagen extraída de [10].

2.3 El problema del particionado del conjunto de datos

Para poder realizar el entrenamiento de una red neuronal, es necesario dividir el conjunto de los datos que disponemos en tres subconjuntos. Estos tres subconjuntos tienen muestras diferentes y cada uno de ellos tiene un uso diferente durante el entrenamiento y la evaluación de una red neuronal.

Cabe destacar que, en la gran mayoría de las producciones científicas no se detalla cómo se hace la partición de los datos en los tres conjuntos diferentes que se utilizan en las etapas de entrenamiento y evaluación de los modelos. Para entrenar una red neuronal, separamos los datos que disponemos en tres subconjuntos diferentes: *training* (entrenamiento), *validation* (validación) y test. El primero de ellos se utiliza para entrenar el modelo y suele tener un 60 % de los datos. El segundo, se utiliza para calcular la pérdida o *loss* y la tasa de acierto tras cada *epoch* con muestras que no se han utilizado durante el entrenamiento y que por tanto, son totalmente nuevas para la red. Este subconjunto suele contener un 20 % de los datos. Por último, el subconjunto de test se utiliza para comprobar el comportamiento de la red una vez el entrenamiento ha finalizado. El subconjunto de test suele contener un 20 % de los datos.

Una vez sabemos esto, en nuestro caso, si un mismo paciente aparece en más de uno de estos conjuntos, estamos distorsionando la tasa de acierto. Esto se debe a que las estructuras cerebrales de una misma persona no varían demasiado entre diferentes sesiones de resonancia magnética. Por tanto, al aparecer un mismo paciente, por ejemplo en el conjunto de *training* y en el de validación, es altamente probable que la red acierte la clasificación al haber visto un cerebro casi idéntico durante la fase de entrenamiento.

El problema que encontramos en las publicaciones consultadas es que no es habitual encontrar una explicación de cómo se ha llevado a cabo la partición [12] o, si se encuentra, normalmente suele ser una vaga descripción que lleva a confusión [10]. Con lo que los resultados del estado del arte hay que analizarlos minuciosamente y con cautela. Di-

ferentes comparaciones se presentan en el Capítulo 5 donde se puede ver que, variando la forma de particionar los datos, los resultados varían significativamente.

CAPÍTULO 3
Descripción del problema y de la solución propuesta

Para la detección de la enfermedad de Alzheimer los médicos necesitan realizar un gran número de pruebas clínicas que les ayuden a diagnosticar a un paciente como sano o enfermo. Estas pruebas son variadas y van desde entrevistas con familiares y amigos del paciente hasta exámenes neuropsicológicos, pasando por pruebas de imagen cerebral obtenida por resonancia magnética. Justo en este último tipo de pruebaf es donde se encuentra el fundamento y base para la realización de este trabajo.

En los experimentos que hemos llevado a cabo durante la realización del trabajo hemos utilizado imágenes cerebrales. Éstas han sido obtenidas por resonancia magnética y se extrajeron de un conjunto de datos (*dataset*) de acceso público que introduciremos en el Capítulo 5. Las imágenes incluidas en este conjunto de datos se han obtenido a lo largo de varios años a partir de pruebas de pacientes de la enfermedad de Alzheimer y algunos sujetos control sanos. Todo ello gracias a diferentes hospitales colaboradores que han cedido las imágenes correctamente anonimizadas.

El hecho de que las imágenes hayan sido extraídas de hospitales colaboradores nos proporciona la certeza de que éstas están correctamente etiquetadas por sus médicos. Las imágenes están etiquetadas en cinco clases diferentes: *Alzheimer's Disease* (AD), *Cognitive Normal* (CN), *Early Mild Cognitive Impairment* (EMCI), *Mild Cognitive Impairment* (MCI) y *Late Mild Cognitive Impairment* (LMCI). De todas estas etiquetas solo utilizaremos AD y CN para intentar construir un clasificador binario. La decisión de utilizar únicamente estas dos etiquetas viene fundamentada por las investigaciones realizadas en el Centro de Investigación Príncipe Felipe (CIPF). Diferenciar entre tres estadíos diferentes de un deterioro cognitivo a partir de biomarcadores en imagen es, a día de hoy, una tarea aún compleja. Es por ello que las imágenes etiquetadas como MCI, LMCI y EMCI no han sido consideradas para los experimentos que se presentan en este trabajo.

El conjunto de datos cuenta con imágenes en tres dimensiones así como imágenes en dos dimensiones. Aunque las imágenes obtenidas por resonancia magnética son, por naturaleza, imágenes en 2D, en el campo del análisis de imágenes médicas consideramos que una sesión de resonancia magnética es tridimensional cuando ésta se compone de un número mínimo de cortes (150 típicamente) que nos permita predecir el contenido entre dos cortes consecutivos sin problemas. De esta manera, podemos reconstruir todo el volumen a partir de cortes bidimensionales.

El objetivo de este problema es conseguir clasificar las muestras mediante un clasificador binario en las dos clases que hemos mencionado. Además, la clasificación que se busca debería cumplir una restricción, ser una clasificación con la menor entropía posible. Si esto se consiguiera, los resultados del clasificador podrían ser un *input* más a conside-

rar por los facultativos a la hora de emitir un diagnóstico. Incluso se podría ir algo más lejos. Si se consigue clasificar una muestra con una entropía baja, el doctor que utilizara esta información podría estar seguro del diagnóstico y no hacer más pruebas ahorrando, de esta manera, tiempo y dinero al sistema público de salud. Si por el contrario, la clasificación de una muestra no tiene la suficiente certeza, el médico debería seguir realizando otro tipo de pruebas para terminar de emitir el diagnóstico.

Es necesario, para comprender el problema con exactitud, que definamos el concepto de entropía. La entropía no es más que una magnitud que mide el "desorden" que presentan nuestros datos. En nuestro problema estamos intentando clasificar una muestra en una de las dos etiquetas disponibles AD o CN. El clasificador binario que resuelve nuestro problema nos dará dos probabilidades: una que indicará cuán seguro está de que la muestra pertenezca a la etiqueta AD y otra que hará lo propio para la etiqueta CN. Así pues, la entropía (desorden) será baja cuanto más extremos sean los valores y, más alta cuanto más se parezcan.

Para la construcción del mencionado clasificador binario en este trabajo se propone la utilización de modelos de Deep Learning (DL) como las redes neuronales convolucionales (CNN). En concreto se han propuesto dos clasificadores binarios con los que se han realizado experimentos que se presentan más adelante. Introduciremos la topología y las características más relevantes de los clasificadores en el Capítulo 5.

CAPÍTULO 4

Descripción de las herramientas hardware y software utilizadas

Durante el proyecto se han utilizado multitud de herramientas software y hardware con el objetivo de agilizar los procesos de entrenamiento de los modelos, o bien para realizar una ”curación de los datos” más sofisticada que nos permitiera realizar diferentes experimentos con los mismos o diferentes modelos con el fin de comparar resultados. A continuación se detallan todas ellas.

4.1 Tensorflow + Keras

Tensorflow es una biblioteca de código abierto utilizada para aprendizaje automático. Esta biblioteca fue desarrollada por el equipo Google Brain de Google y se lanzó por primera vez en 2015 como sucesora de Distbelief –biblioteca para el mismo uso, pero de código cerrado, que Google utilizaba internamente para crear sus propios modelos de redes neuronales–. [5]

Tensorflow actualmente proporciona APIs estables para C así como para Python, pero también para otros lenguajes de programación como C++, JavaScript o Swift. En este trabajo hemos escogido Python ya que esto nos permite utilizar Keras.

Keras es una librería escrita en Python, open-source y capaz de proporcionar una abstracción a alto nivel de la complejidad de Tensorflow. Keras también se puede ejecutar sobre otras bibliotecas como por ejemplo, Theano. Keras se diseñó para permitir una rápida experimentación con redes neuronales profundas puesto que es modular y extensible. Por tanto, la decisión de usar Keras viene dada por la facilidad de probar experimentos con diferentes modelos de redes neuronales profundas optimizando el tiempo de trabajo [6]. En el apéndice Apéndice A se amplía la información sobre Keras introduciendo las explicaciones de las diferentes capas, funciones de activación y técnicas de regularización que se han utilizado en la experimentación.

4.2 GPUs

Podemos decir que el auge de la inteligencia artificial (IA) y el desarrollo y expansión de las GPUs [1] van unidos de la mano. Esto se debe a dos factores: el primero, que la IA necesita que los cálculos en los que se basa se realicen lo más rápido posible y el segundo

[1] Graphic Processor Units.

que estos cálculos comparten muchas similitudes con las operaciones que soportan las GPUs y para los que fueron diseñadas, esto es, cálculos matriciales.

Con todo esto, en los últimos años las GPUs se han estado utilizando como aceleradores para aplicaciones de IA. En nuestro caso hemos utilizado varias máquinas que cuentan con varias GPUs instaladas en ellas que han acelerado extraordinariamente nuestros experimentos. La utilización de estos aceleradores ha sido posible gracias al acceso proporcionado a las máquinas del CEIB[2] y del PRHLT[3]. En concreto, agradecemos a NVIDIA el apoyo proporcionado con la donación de la tarjeta gráfica NVIDIA Quadro P5000 que ha sido utilizada durante este trabajo.[4] Las GPUs utilizadas junto con sus especificaciones se encuentran en la Tabla 4.1.

Tabla 4.1: GPUs utilizadas durante la experimentación para acelerar las ejecuciones.

GPU	CUDA Cores	Memoria GPU (GB)
NVIDIA Quadro P5000	2560	16
NVIDIA GEFORCE GTX 1080	2560	8
NVIDIA GEFORCE RTX 2080	2944	8

4.3 FMRIB Software Library

FSL[5] es una librería software que contiene herramientas de análisis de imágenes cerebrales obtenidas por resonancia magnética tanto funcionales, estructurales como de difusión. Está desarrollada y mantenida por el FMRIB Analysis Group de la Universidad de Oxford.[6]

Como se ha explicado antes, en este trabajo se ha trabajado con imágenes de resonancia magnética potenciadas en T1[7]. Es por ello que, entre todas las funcionalidades disponibles en FSL, únicamente se han utilizado la *Brain Extraction Tool* (BET) y FLIRT. Estas funcionalidades se explican con detalle en los siguientes subapartados.

4.3.1. BET

BET[8] es la herramienta dentro de la librería FSL[8] que se encarga de eliminar la sustancia no cerebral[9] de las imágenes.

La herramienta en cuestión cuenta con multitud de parámetros ajustables que eliminan, con mayor o menor agresividad, la materia no cerebral. De entre todos los parámetros de que dispone BET pondremos especial énfasis en: *fractional*, *remove_eyes* y *bias*.

[2] Centro de Excelencia e Innovación Tecnológica de Bioimagen de la Consellería de Sanitat.

[3] Pattern Recognition y Human Language Technology Research Center.

[4] We gratefully acknowledge the support of NVIDIA Corporation with the donation of the NVIDIA Quadro P5000 GPU used for this research.

[5] FMRIB Software Library.

[6] https://www.win.ox.ac.uk/research/analysis-research

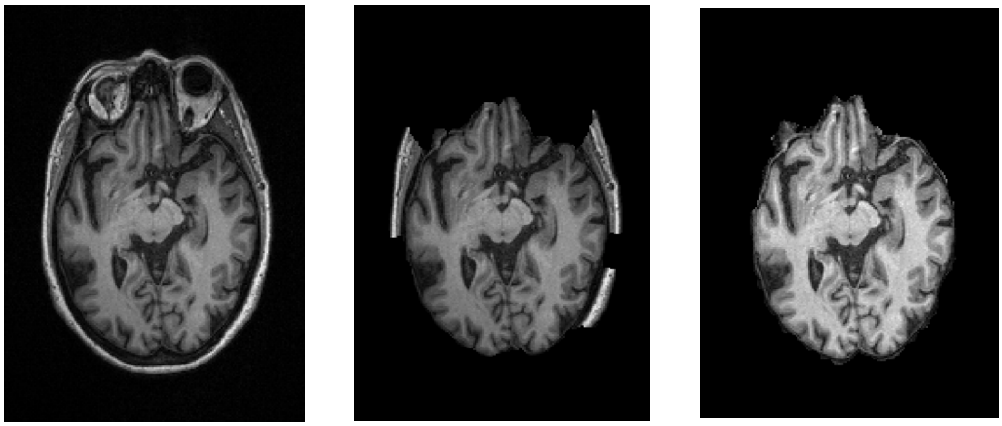
[7] Las imágenes por resonancia magnética pueden estar potenciadas en T1 o en T2. Las potenciadas en T1 reflejan mejor las estructuras, en este caso, cerebrales. Las potenciadas en T2, también llamadas funcionales, representan mejor las zonas 'blandas' del cerebro.

[8] Brain Extraction Tool.

[9] La sustancia no cerebral es el nombre que reciben todas aquellas estructuras que aparecen en las imágenes médicas cerebrales (en nuestro caso imágenes por resonancia magnética) que no pertenecen al parénquima, esto es, el cerebro en si. Así pues, consideramos sustancia no cerebral los huesos del cráneo o las cavidades y globos oculares.

El parámetro *fractional* espera un numero decimal en el rango [0,1]. Ajusta la precisión del recortado. Cuanto mayor es el valor, menor es la precisión del recortado. En segundo lugar, el parámetro *remove_eyes* es un booleano. Si activamos la opción, el procesado de la imagen es más lento, pero a cambio elimina las cavidades y los nervios oculares, que suponen ruido para la red neuronal. Por último, el parámetro *bias* espera también un booleano. Se recomienda activarlo para así eliminar la inhomogeneidad del campo magnético que se genera en los cortes más altos y en los más bajos, así como eliminar partes de cuello que pueden aparecer en algunas sesiones y que no ayudan al entrenamiento de la red neuronal.

Con todo ello, y tras probar diferentes combinaciones de estos parámetros, nos quedamos con la que mejor nos ha funcionado. Esta combinación se usa en [7] y establece los siguientes valores: *fractional* = 0.1, *remove_eyes* = True, *bias* = True. Los resultados del preprocesado de las imágenes con BET se pueden ver en la Figura 4.1.



(a) (b) (c)

Figura 4.1: Dos procesados del mismo corte con diferentes valores en los parámetros de BET. En (a) la imagen original. En (b), el procesado usando únicamente *fractional* = 0.3. En (c) se utilizó el procesado propuesto en [7] *fractional* = 0.1, *remove_eyes* = True, *bias* = True.

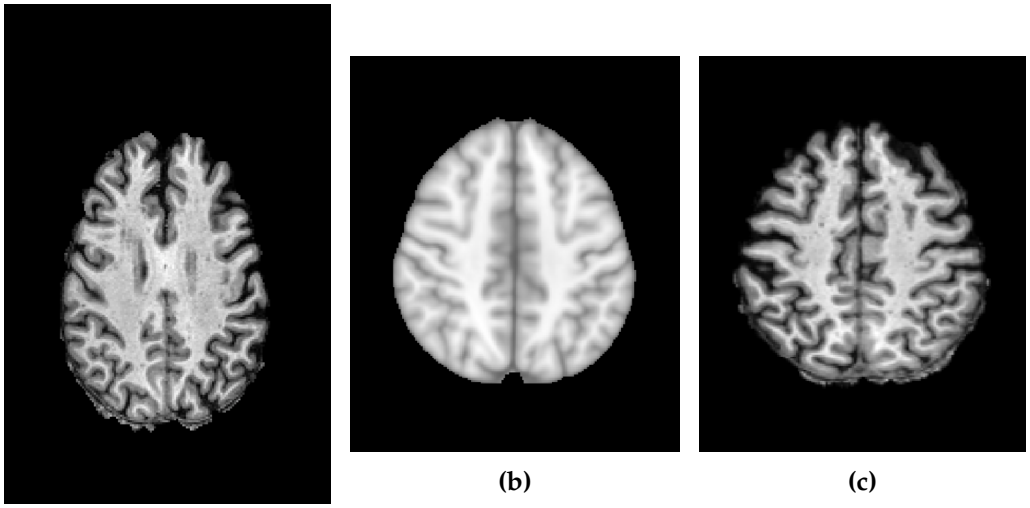
4.3.2. FLIRT

FLIRT es la herramienta dentro de la librería FSL[8] que se encarga de transformar las imágenes por resonancia mágnetica en 3D de un cerebro al espacio de un cerebro de referencia seleccionado de entre todos los que FLIRT tiene disponibles. Esto es necesario para reducir la variabilidad de los datos de entrada. Aunque los cerebros son diferentes internamente, todas las imágenes, tras el procesado con FLIRT, tienen el centro en el mismo lugar y el parénquima tiene la misma forma. De esta manera podemos reducir, como hemos dicho, la variabilidad de los datos de entrada y con ello, el ruido que esto genera. Se puede ver un ejemplo de procesado con FLIRT en la Figura 4.2.

4.4 FreeSurfer

FreeSurfer es una librería software que se utiliza para el análisis de resonancias magnéticas de cerebro. Está desarrollado y mantenido por el Martinos Center for Biomedical Imaging del Massachusetts General Hospital. Esta librería es ampliamente utilizada por la comunidad científica para el segmentado de las imágenes de cerebro que se puede hacer siguiendo diferentes atlas[10]. Además de todo ello, FreeSurfer es capaz de obtener

[10]Un atlas cerebral es un cerebro ideal con diferentes zonas segmentadas.



(a) (b) (c)

Figura 4.2: En esta figura podemos ver en (a) el corte original, en (b) el cerebro de referencia dentro del estándar FLIRT (en este caso MNI152) y en (c) el resultado de la transformación de la imagen original al espacio del cerebro de referencia.

datos de volumetría cerebral, esto es, las medidas de volumen de cada una de las áreas del cerebro segmentadas según el atlas utilizado.

En este trabajo se ha utilizado FreeSurfer con el atlas Aparc2009 y con el atlas que segmenta las Brodmann Areas –un atlas especializado en áreas afectadas por el Alzheimer–. El objetivo ha sido obtener los datos de volumetría de estos dos atlas para poderlos incluir junto con las imágenes en la red neuronal. De esta manera se ha observado si han servido o no de ayuda para la tarea de clasificación. Aunque en este trabajo no abordamos el uso de esta herramienta, recordamos que como parte de la experimentación conjunta con otros dos compañeros, los resultados obtenidos con el uso de esta herramienta se exponen en el trabajo de mi compañera Silvia Nadal Almela.

4.5 La clase *DataGenerator*

Uno de los problemas que se encontró al principio de la etapa de experimentación fue que las imágenes necesarias para el entrenamiento y validación de un modelo y el propio modelo no cabían en la memoria RAM de ninguna de las tarjetas gráficas de las que disponíamos. Así pues, nos surgió la necesidad de introducir *on the fly* los *batches*[11], es decir, cargar cada uno de los *batches* durante el entrenamiento del modelo de manera que se perdiera en el proceso el menor tiempo posible. Así pues, encontramos en [15] la solución. Utilizamos la clase *DataGenerator* adaptándola a nuestras necesidades. Finalmente, se consiguió que la carga de los batches no hiciera perder casi tiempo, ya que el propio *DataGenerator* se encarga de hacer las tareas necesarias para tal fin, minimizando el tiempo de espera de los datos.

[11] Un *batch* es un conjunto de muestras que se toman como una unidad. Después de cada *batch* es cuando se actualizan los pesos.

CAPÍTULO 5
Experimentación y resultados

5.1 Descripción del *dataset* utilizado: ADNI

El objetivo principal del *Alzheimer's Disease Neuroimaging Initiative* (ADNI) ha sido probar si las imágenes tomadas por resonancia magnética, la tomografía por emisión de positrones (PET) y algunos datos de etiquetado clínico se podían combinar para medir la progresión del deterioro cognitivo leve o de la enfermedad de Alzheimer.

Para entender correctamente cómo se ha realizado el preprocesado de las imágenes, es necesario que primero definamos dos conceptos: *subject* y *session*. El término *subject* (o sujeto) hace referencia a un paciente único, esto es, una persona independientemente del número de pruebas de resonancia magnética que le hayan sido realizadas para la iniciativa. Por otro lado, tenemos el término *session* (o sesión). Un sujeto puede tener dentro del *dataset* varias pruebas realizadas. A cada una de las pruebas las denominamos sesiones.

La importancia de la definición previa de estos dos conceptos radica en la separación de las imágenes en los subconjuntos de entrenamiento (tr), validación (dev) y test (te) de los que ya hemos hablado anteriormente. Diferentes sesiones de un mismo sujeto no deben aparecer en más de un subconjunto, de otra manera la partición estaría mal hecha puesto que estaríamos falseando los resultados. Esto, como se ha explicado en el Capítulo 2, es algo que con altas probabilidades y dados los experimentos que presentaremos más adelante, sucede en la gran mayoría de las soluciones que conforman, a día de hoy, el estado del arte de esta materia.

Con todo ello, para los experimentos de este trabajo se han utilizado imágenes tanto en 3D como en 2D. Realmente, en el campo de la tomografía por resonancia magnética solo se obtienen imágenes en 2D corte a corte. En cualquier caso, si tenemos un número de cortes suficiente, típicamente más de 150 cortes, se puede considerar que la sesión es una sesión 3D ya que se pueden utilizar herramientas para predecir los vóxeles que faltan sin problema.

Las imágenes obtenidas han sido sometidas a diferentes técnicas de preprocesado obteniendo así diferentes variaciones del *dataset*. Se han obtenido 3 variaciones en 3D: a) el original sin ninguna transformación, b) el procesado con la herramienta BET para la extracción de parénquima y c) el procesado con BET + FLIRT para extracción de parénquima y normalización de las imágenes. De estas 3 variaciones, considerando cada corte como una imagen independiente, se han extraído los *datasets* en 2D.

Recordemos que las imágenes están clasificadas en 5 etiquetas diferentes: *Alzheimer's Disease* (AD), *Cognitive Normal* (CN), *Early Mild Cognitive Impairment* (EMCI), *Mild Cognitive Impairment* (MCI), *Late Mild Cognitive Impairment* (LMCI), de las cuales únicamente

Tabla 5.1: Recuento de sesiones por tamaño de imagen 2D.

Tamaño	Sesiones
176x240	1078
160x192	682
170x256	590
150x256	1
160x240	190
208x240	46
176x256	8
211x256	24
184x256	19
170x288	3
230x230	6
176x248	3
230x240	5
140x256	2
512x174	1
160x256	1

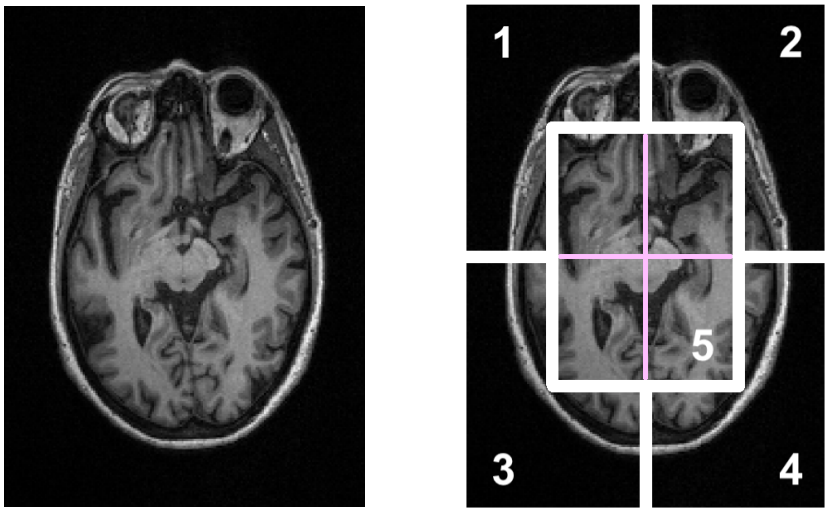
se han utilizado para experimentos de clasificación AD y CN. Las tablas con el conteo de imágenes por cada variación del *dataset* se pueden consultar en el Apéndice B.

5.2 Experimentos iniciales

Una vez descargados los datos, se decide iniciar la experimentación trabajando únicamente con imágenes en 2 dimensiones. Para ello, se considera cada uno de los cortes de las imágenes como una muestra.

Para poderlas utilizar como entrada a una red neuronal, las imágenes tienen que pasar, como mínimo, un pequeño preprocesado que las transforme de tal forma que todas tengan el mismo tamaño. Este proceso siempre suele introducir distorsiones en los datos de entrada, con lo que para minimizar este efecto, se hizo un estudio previo de las imágenes 2D que teníamos. En este estudio, primero se descubrieron los distintos tipos de tamaño que existían en el *dataset* y después se contó el número de imágenes que existían de cada tamaño. Los resultados del estudio se pueden consultar en la Tabla 5.1.

El tamaño con más imágenes resultó ser el de 176x240, con lo que se decidió que las imágenes de ese tamaño no se modificarían y el resto se llevarían a este nuevo tamaño. Ante esta situación, se plantean tres casos: 1) que la imagen que tengamos que transformar sea más pequeña que el tamaño objetivo en todas sus dimensiones, 2) que sea más pequeña también en ambas dimensiones, o bien 3) que una de las dimensiones sea más pequeña que la misma de referencia y la otra más grande. En el caso de ser más pequeña en ambas dimensiones, se decide ampliar el tamaño mediante la técnica del *zero-padding*, esto es, colocar a la imagen un marco con píxeles con valor 0 hasta conseguir el tamaño deseado. Por otro lado, si la imagen resulta ser más grande en ambas dimensiones, se decide realizar 5 cortes de tamaño 176x240 como se muestra en la Figura 5.1. Se decide realizar los cortes de esta manera para que no quede ningún píxel sin considerar ya que todos los píxeles aparecerán en, al menos, uno de los cortes. En el último caso lo que se hace es aplicar *zero-padding* a la dimensión que es más pequeña y después obtener tres cortes en la dimensión que es más grande. Si la dimensión mayor es la vertical, los cortes serían: arriba, centro y abajo. En caso contrario, los cortes serían: izquierda, centro y



(a) (b)

Figura 5.1: En esta figura podemos ver en (a) la imagen original, más grande que el tamaño de moda (176x240), y en (b) los 5 recortes que se obtienen del tamaño objetivo.

derecha. Finalmente, se normalizaron las imágenes con respecto al píxel con mayor valor de la misma. Más tarde veremos que este tipo de normalización no es del todo correcta.

Con este preprocesado sencillo y sin aplicar ninguna técnica de preprocesado más, procedimos a entrenar un *autoencoder* para comprobar cómo se comportaban los datos. Un *autoencoder* es un tipo de red neuronal que trata de aprender una representación de dimensionalidad reducida (*encoding*) de los datos de entrada. Junto con esto, también trata de reconstruir los datos iniciales a partir del *encoding*. En resumen, un *autoencoder* se compone de dos partes la primera, el *encoder* y la segunda, la que se encarga de reconstruir, el *decoder*. La topología del *autoencoder* propuesto se puede ver en la Figura 5.2. Así pues, se decidió que esto era un buen experimento inicial porque, si se obtenían buenos resultados, podíamos deshechar la parte del *decoder* y quedarnos con la reducción de dimensionalidad hecha por el *encoder* para poder usarla como entrada a una parte *fully-connected*, encargada de la tarea de clasificación en si.

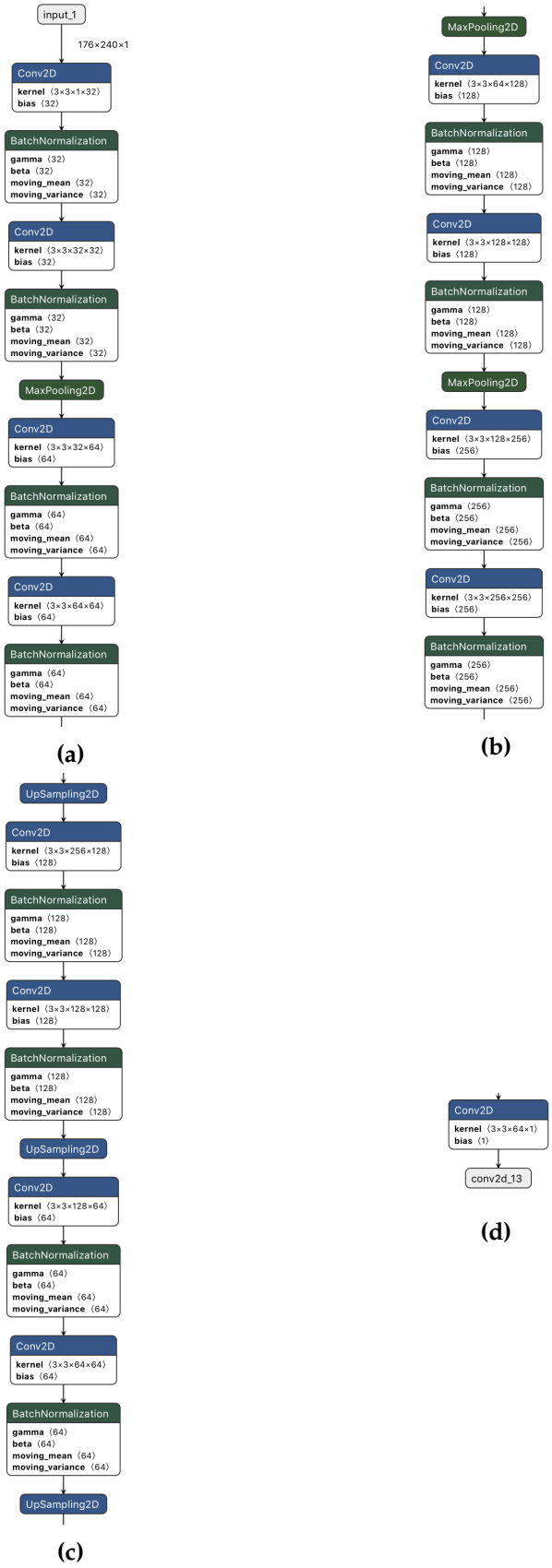


Figura 5.2: Topología del *autoencoder* propuesto para el primer experimento realizado con el *dataset* ADNI 2D.

Finalmente, comentaremos la partición de los datos en los subconjuntos de entrenamiento, validación y test. Para realizar la partición, primero se barajaron las imágenes aleatoriamente sin imponer ningún tipo de restricción. Después, siguiendo los porcentajes habituales en este tipo de casos, se formó el conjunto de entrenamiento (tr) con un 60 % de las imágenes, el de validación (dev) con un 20 % y el de test (te) con el 20 % restante.

Los resultados obtenidos fueron buenos. Se llegó a obtener un *loss* en validación cercano a 1×10^{-4}, por tanto, se decide continuar con un segundo experimento en el que se elimina la parte de la reconstrucción de la imagen (*decoder*) y la salida del *encoder*, ya entrenado, se pasa como entrada a una segunda parte *fully-connected* cuya última capa tiene 5 neuronas, una por cada clase, y activación *softmax*. Este experimento da resultados sorprendentes, una tasa de acierto durante la etapa de validación de aproximadamente 98 %. Este resultado, sin embargo, se pone en duda ya que con un modelo extremadamente sencillo se consiguen resultados que superan los resultados disponibles en el estado del arte.

Por otro lado, se decide utilizar la técnica de *transfer-learning* en el marco de la creación de un poster que se expuso en el III Congreso Nacional de Jóvenes Investigadores en Biomedicina. En él, se propuso el modelo que se puede ver de manera esquemática en la Figura 5.3. Aunque los datos obtenidos no fueron buenos –por debajo del 40 % en la tasa de acierto– se valora positivamente el experimento porque el mal resultado obtenido dio como fruto una revisión de las técnicas de preprocesado utilizadas.

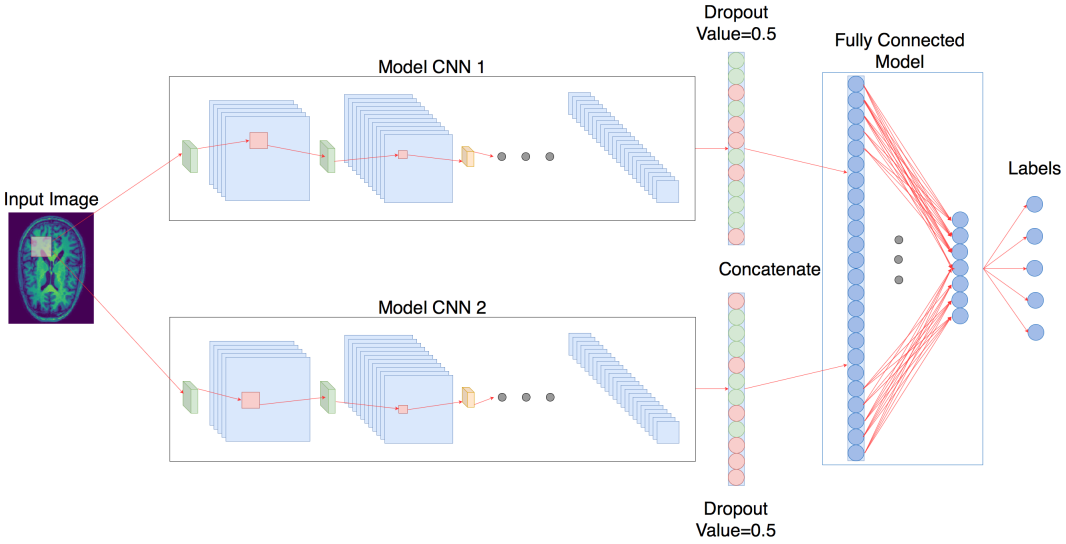


Figura 5.3: Esquema del modelo que se propuso para el póster expuesto en el III Congreso Nacional de Jóvenes Investigadores en Biomedicina. Los pesos de ambos modelos CNN fueron entrenados anteriormente como *encoders* de un *autoencoder* como el de la Figura 5.2. Así pues, al entrenar la red completa, los pesos de los modelos CNN no eran aleatorios, sino que se habían transferido desde otro modelo.

5.2.1. Detalles, resultados y valoración del experimento

En este primer grupo de experimentos, salta a la vista que los resultados obtenidos no son, en absoluto, significativos. Sin embargo, esto nos hace ahondar aún más en el problema que estamos resolviendo. En este punto de la experimentación, se piensa que la región de interés del problema es únicamente el parénquima. Esto provoca que en las imágenes, las zonas en las que aparece hueso craneal así como cavidades y globos oculares deberían ser eliminadas ya que únicamente introducen ruido en la red neuronal. Por tanto, tal y como sucede en la gran mayoría de soluciones que se proponen en el

estado del arte, se procederá a aplicar la extracción del parénquima en las imágenes del *dataset*. Este nuevo *dataset* resultante de aplicar el preprocesado con BET será uno de los que utilicemos en los siguientes experimentos.

Por otro lado, debido a los resultados obtenidos en el clasificador presentado anteriormente (recordemos que la tasa de acierto en validación llegaba al 98 %), se revisa la forma de hacer las particiones de los datos en los tres subconjuntos de entrenamiento, validación y test. Efectivamente, el resultado no es verosímil. La manera de partir los datos hasta ahora se basaba en barajar todos los cortes 2D para después repartirlos en los tres conjuntos siguiendo las proporciones 60 %-20 %-20 % en entrenamiento, validación y test, respectivamente. Esto no tiene ningún fundamento puesto que la estructura cerebral de un paciente (sujeto) no varía demasiado entre cortes adyacentes. Así pues, si en el conjunto de entrenamiento aparece un corte, por ejemplo el corte n de la sesión X de un paciente y en el conjunto de validación aparece otro corte, por ejemplo el corte $n+1$ de la misma sesión X del mismo paciente, la red acertará su clasificación casi con toda seguridad. Y ese acierto no se deberá a que internamente la red haya podido reconocer un biomarcador de la enfermedad en la imagen, sino porque ha sido capaz de reconocer que una imagen muy similar ha pasado previamente durante el entrenamiento y le ha asignado la misma etiqueta.

Así pues, se propone una nueva manera de configurar las particiones de los datos en los mencionados subconjuntos. A partir de ahora, todos los cortes 2D de todas las sesiones de un mismo paciente solamente podrán aparecer en uno de los tres subconjuntos. Con esto se consigue que aunque la red memorice los pacientes que conforman el conjunto de entrenamiento, si se produce un acierto en la etapa de validación sea porque realmente se basa en algo más que la similitud extrema entre dos cortes, esto es lo que se conoce como generalización. En la Figura 5.4 se puede ver, esquemáticamente, las dos maneras de partir los datos que hemos mencionado en esta sección. Cabe destacar que la partición se realiza únicamente una vez al principio y se guarda la distribución de las imágenes en cada subconjunto. Esto nos permite después replicar los experimentos si fuera necesario con, exactamente, la misma distribución de imágenes.

Además, mencionamos que esta manera que proponemos de partir los datos no es casi utilizada en los experimentos que conforman el estado del arte. Como se demostrará en algunos experimentos a continuación, la diferencia de los resultados varía altamente si cambiamos la manera en que hacemos la partición.

Asimismo, también se propone cambiar la manera en la que se normalizan las imágenes. Hasta ahora se han venido normalizando o mediante la división de los valores de los píxeles entre el valor del píxel con mayor intensidad de la imagen (Ecuación 5.1). El problema que presenta esta normalización es que se ve afectada por los máximos absolutos en intensidad. Recordamos que las imágenes por resonancia magnética que se utilizan en este trabajo, se toman mediante la variación de campos electromagnéticos creados por las máquinas que toman las imágenes. Por tanto, los píxeles tienen un valor que representa la intensidad del campo magnético en el punto del espacio que representan. Como las máquinas siempre introducen algún tipo de ruido, puede ser que el valor del píxel con más intensidad sea lo suficientemente grande como para conseguir que, tras la normalización, el resto de píxeles acaben cercanos al 0, esto es, atenuados. Como se puede observar en la Figura 5.5, se da el caso de algunas imágenes que cuentan con este problema que acabamos de describir. Por tanto, se propone una nueva normalización que resuelve el problema que había surgido.

La nueva normalización se expresa en las ecuaciones 5.2 y 5.3. Dadas las gráficas en la Figura 5.5, vemos que hay un conjunto pequeño de imágenes que tienen el máximo valor de intensidad demasiado alto. Si quitamos este conjunto de imágenes que, de alguna ma-

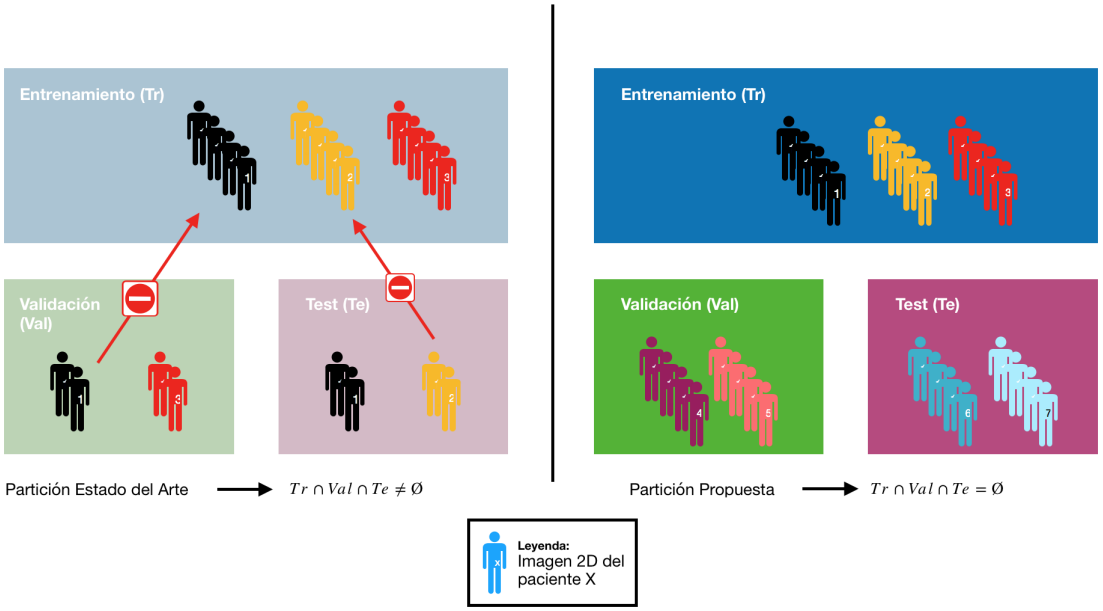


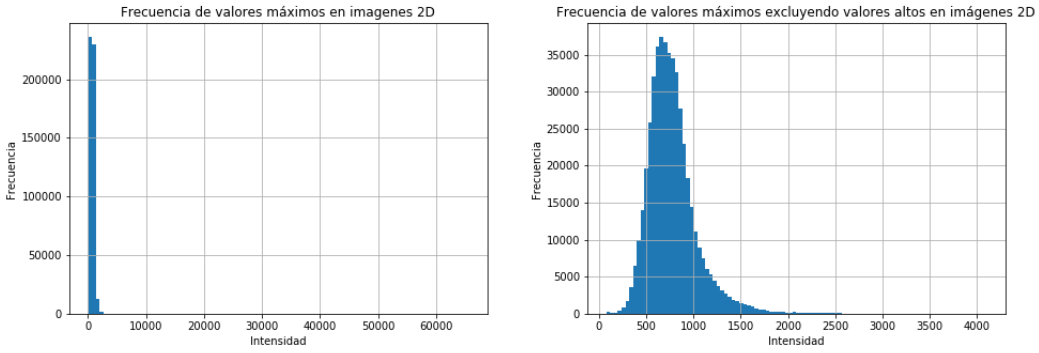
Figura 5.4: En esta figura se pueden ver esquemáticamente los dos modos de partir los datos que se presentan en este trabajo. A la izquierda, la manera que usan la mayoría de experimentos del estado del arte. Imágenes 2D de un mismo paciente aparecen en más de un subconjunto. A la derecha, la que nosotros consideramos correcta. Todas las imágenes de un mismo paciente están, juntas, en el mismo subconjunto.

nera, nos distorsionan el estudio de intensidades máximas, vemos que la distribución es algo más normal. Es por ello que se decide, arbitrariamente y a partir de los nuevos valores de intensidad, establecer el máximo valor de intensidad en un valor de 2000. Por este motivo en la Ecuación 5.2, estamos dividiendo entre 2000. Sin embargo, debido a las imágenes con máximos de intensidad poco usuales, tras la división puede que el algún valor normalizado sea mayor que 1. Así pues en la la Ecuación 5.2, cortamos estos valores a 1 utilizando la operación mín. Pero, en este punto, debemos contemplar un problema. Si el máximo de intensidad en una imagen era bastante más pequeño que el máximo que habíamos elegido, pongamos por ejemplo 300, entonces todos los valores de los píxeles de esa imagen han quedado cercanos al cero en el rango [0,1]. Como consecuencia en la Ecuación 5.3, volvemos a normalizar los valores de la imagen sumando al máximo valor un pequeño valor, en nuestro caso 1×10^{-3}. Con esto conseguimos que los valores se amplifiquen y vuelvan a quedar, ahora realmente, en el rango [0,1]

$$pixel_i = \frac{pixel_i}{max_{0 \leq j \leq NxM} pixel_j} \forall 0 \leq j \leq NxM \tag{5.1}$$

$$pixel_i = min(1, pixel_i/2000) \forall 0 \leq i \leq NxM \tag{5.2}$$

$$pixel_i = \frac{pixel_i}{1 \times 10^{-3} + max_{0 \leq j \leq NxM} pixel_j} \forall 0 \leq i \leq NxM \tag{5.3}$$



(a) **(b)**

Figura 5.5: En esta figura podemos ver como las intensidades máximas de las imágenes varían en el conjunto de datos. En (a) se muestran las intensidades máximas de todas las imágenes 2D frente a la frecuencia del campo magnético. En (b) si eliminamos tan solo 423 imágenes con máximos de intensidad que están fuera de lo normal (>6000), vemos que la distribución mejora. Ahora los máximos están en torno a 2000, valor que utilizamos para normalizar en la Ecuación 5.2.

Finalmente, destacamos que estos experimentos también hicieron aflorar el problema de trabajar con las cinco etiquetas con las que cuenta el *dataset*. De momento, no es posible diferenciar, mediante biomarcadores en imagen, el grado preciso de deterioro cognitivo de un paciente. Con lo cual, introducir las imágenes con etiquetas EMCI[1] y LMCI[2], significa introducir ruido en la red puesto que son bastante parecidas a las muestras etiquetadas como MCI[3]. Basados en esto, se decide realizar los experimentos siguientes con únicamente dos etiquetas (AD vs. CN).

5.3 Introducción a los experimentos posteriores

En esta sección sentaremos las bases necesarias para poder comprender los experimentos que se presentarán en las secciones siguientes. En primer lugar hablaremos de los dos clasificadores que se han utilizado para realizar los experimentos. Por cada experimento, se han utilizado ambos clasificadores para contrastar resultados. Por otra parte, hablaremos de cómo hemos partido los datos en los tres subconjuntos necesarios. Para cada experimento utilizaremos la partición que se propone como correcta en este trabajo, y la que se utiliza con frecuencia en el estado del arte. Además miraremos si en un mismo experimento, la aplicación de la técnica *data augmentation* tiene algún tipo de efecto significativo.

Destacamos que estos experimentos se han llevado a cabo habiendo pasado el ADNI *dataset* en 2D por la herramienta BET. Con lo cual, los tamaños variaron con respecto a las imágenes en el *dataset* original. En esta ocasión, tras realizar un nuevo estudio, se decidió llevar las imágenes a un nuevo tamaño objetivo 256x256 píxeles. Al ser este el tamaño más grande existente, únicamente se tuvo que emplear la técnica del *zero-padding* en el resto de las imágenes (obviamente, más pequeñas) para llevarlas al nuevo tamaño.

[1] Early Mild Cognitive Impairment.
[2] Late Mild Cognitive Impairment.
[3] Mild Cognitive Impairment.

5.3.1. Topología de los clasificadores

Se han desarrollado dos clasificadores que llamaremos, de ahora en adelante, clasificador A y clasificador B. Ambos son clasificadores que comienzan con una primera parte de convoluciones que hacen uso de filtros grandes para, cada vez, ir haciéndose más pequeños en tamaño. Después de ello, los datos entran en una segunda parte *fully-connected* que se encarga de la clasificación en una de las dos clases: AD, o bien CN.

El clasificador A, cuya topología se puede ver en la Figura 5.6, comienza introduciendo ruido gaussiano a la imagen de entrada. Una vez se ha distorsionado la imagen, se procede a la entrada de la misma en la parte convolucional. En esta primera parte se extraen filtros primero de 11x11, después de 7x7 y se termina con 4 capas convolucionales que extraen filtros de tamaño 5x5. Después, a las características extraidas de esta primera etapa se les aplica la operación *flatten* que consigue expresar esas caracterísiticas en un vector unidimensional que se usará como entrada en la parte *fully-connected*. En la segunda parte, la entrada pasa por tres bloques idénticos compuestos de *BatchNormalization*, *GaussianNoise* y la propia capa oculta compuesta por 1024 neuronas. Tras esto, la capa de salida compuesta por dos neuronas y activación *softmax* es la encargada de realizar la clasificación final de la muestra.

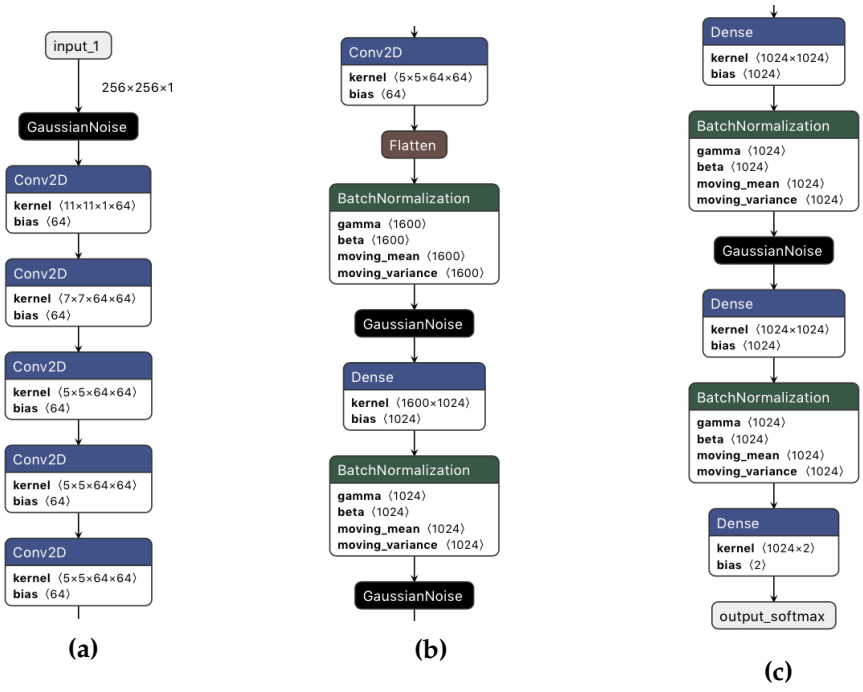


Figura 5.6: Topología del clasificador A con el que trabajaremos en los experimentos de este trabajo.

En cuanto al clasificador B, cuya topología se puede ver en la Figura 5.7, podríamos decir que es casi igual que el clasificador A. La única diferencia la vemos en la parte convoclucional. Mientras el clasificador A comienza con filtros de tamaño 11x11, el B comienza utilizando filtros de 13x13. Después, como en el clasificador A, utiliza filtros de 7x7 seguido de cuatro capas convolucionales que utilizan filtros de 5x5. La parte *fully-connected* es idéntica a la utilizada en el clasificador A.

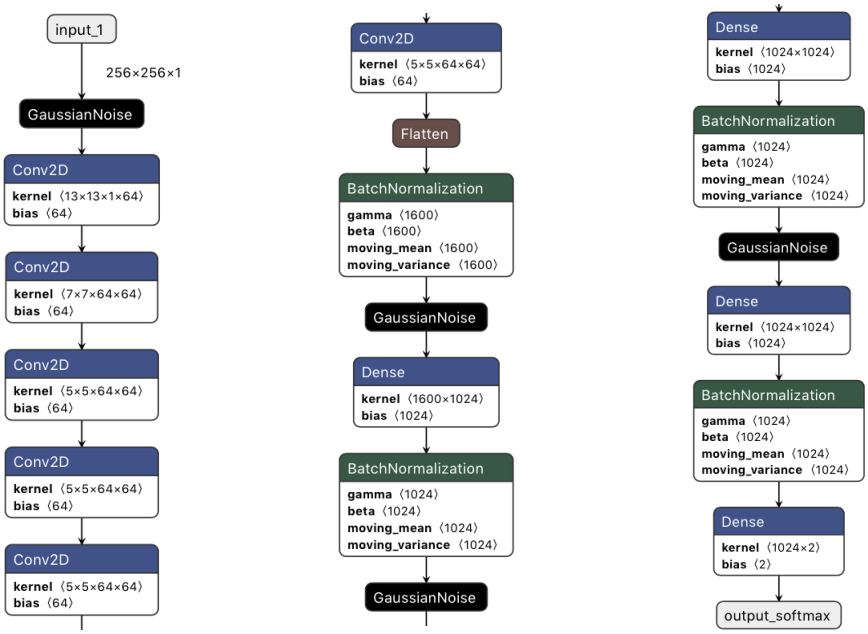


Figura 5.7: Topología del clasificador B con el que trabajaremos en los experimentos de este trabajo.

En el último experimento que se expondrá en este trabajo las imágenes han cambiado de tamaño. Esto se produce porque, para el experimento, se utilizan las imágenes preprocesadas con FLIRT. Así pues debemos modificar ligeramente los modelos que hemos presentado para que sean capaces de aceptar estas nuevas imágenes. De esta manera en las figurar 5.8 y 5.9 se presentan los clasificadores C y D que son variaciones de A y B, respectivamente.

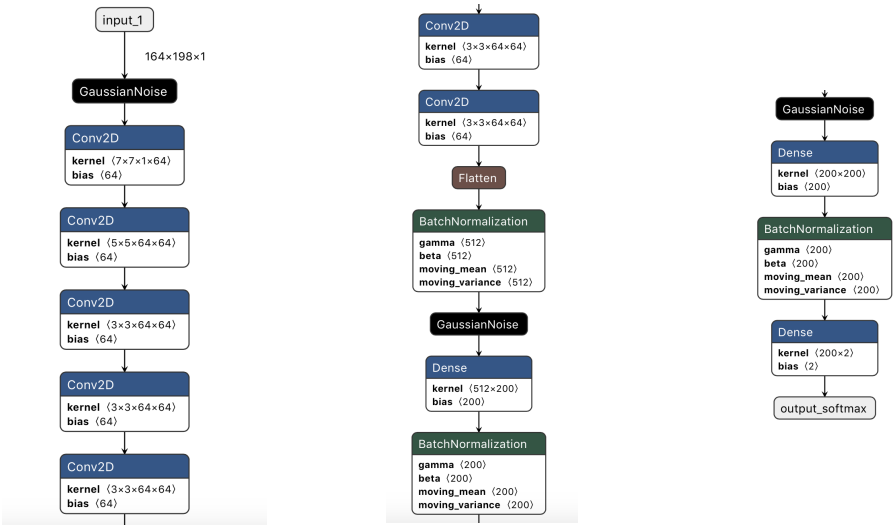


Figura 5.8: Topología del clasificador C con el que trabajaremos en los experimentos de este trabajo.

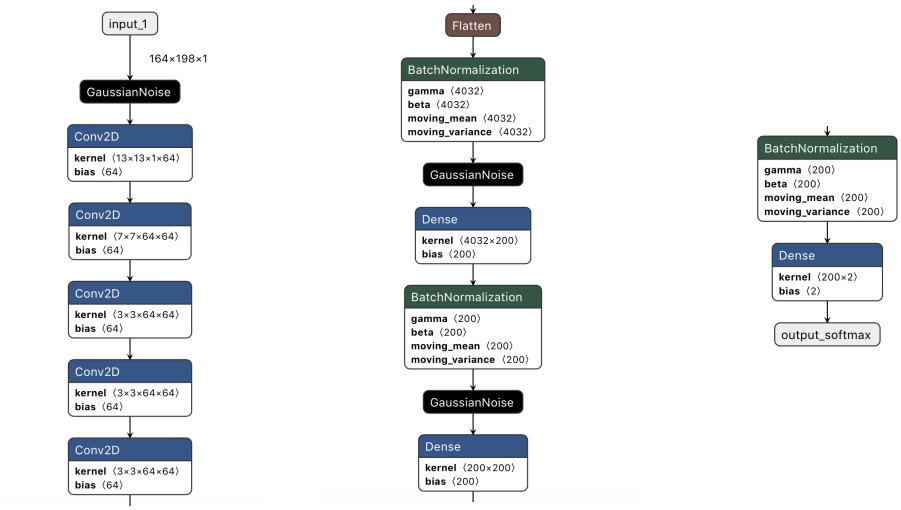


Figura 5.9: Topología del clasificador D con el que trabajaremos en los experimentos de este trabajo.

5.3.2. Variaciones del *dataset*

En los experimentos que presentaremos más adelante utilizaremos tanto nuestro *dataset* base (ADNI 2D preprocesado con BET), así como algunas variaciones que presentaremos en esta sección y que tienen que ver, bien con la manera de partir los datos, con la manera de tratarlos, o bien con la aplicación o no de técnicas de *data augmentation*.

En cuanto a los procedimientos diseñados para particionar los datos, utilizaremos las dos maneras que ya se han presentado en este trabajo y que se pueden ver gráficamente en la Figura 5.4. Por otra parte, trataremos los datos de tres maneras. La primera de ellas será tratando cada corte como una imagen 2D independiente, como se ha venido haciendo hasta ahora que, en adelante, llamaremos "ADNI 2D BET". La segunda, tratará de poner sobre cada corte un poco más de "contexto", esto es, tratará realmente con imagenes 3D formadas por el corte en cuestión, el corte anterior y el posterior en un volumen que resultará de 256x256x3 tal y como se propone en [3]. Esta variación la denominaremos, en adelante, "ADNI 2D 3 cortes". La tercera variación, consistirá en preprocesar las imágenes 2D del conjunto de datos original con BET y FLIRT. A esta variación del *dataset* nos referiremos, en adelante, como "ADNI 2D FLIRT".

Por último, cuando hagamos referencia, en la explicación de los experimentos, a que se ha utilizado *data augmentation* se ha de saber que las técnicas que se han utilizado para este fin son: a) *zoom in/out*, b) *shifting* y c) rotación. Cualquiera de ellas ha podido hacerse o no, puesto que se selecciona aleatoriamente para cada imagen qué técnicas deben aplicarse. Cabe destacar que el *zoom in* (aumento) no incrementa el tamaño de la imagen, esto es, todo contenido que quede fuera del tamaño se recorta. Por otro lado, el *zoom out* (decremento) tampoco modifica el tamaño de la imagen. Se rellena el área restante hasta llegar al tamaño objetivo mediante la técnica de *zero-padding*. El *shifting* consiste en mover el contenido dentro del marco de la imagen, es decir, mover el centro de la imagen por todo el marco. Finalmente, la rotación se aplica en el rango [-30º, +30º]. Todas estas técnicas se pueden ver gráficamente en las figuras 5.10, 5.11 y 5.12

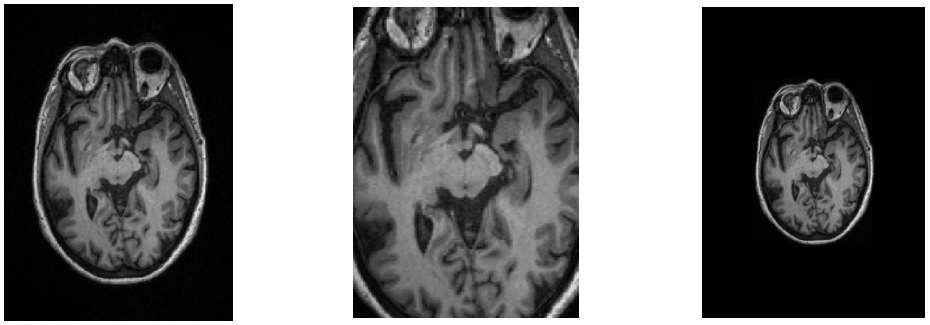


Figura 5.10: En esta figura observamos los resultados de aplicar *zoom*. En (a) vemos la imagen sobre la que aplicamos *zoom in* en (b) y *zoom out* en (c).

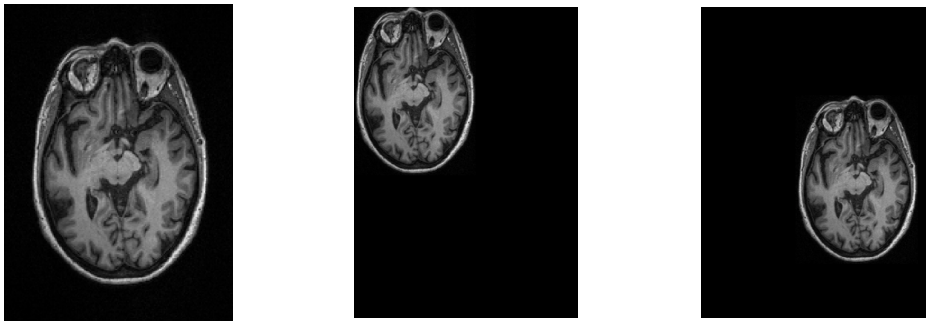


Figura 5.11: En esta figura observamos los resultados de aplicar *shift*. En (a) vemos la imagen sobre la que aplicamos *shift* en (b) y (c).

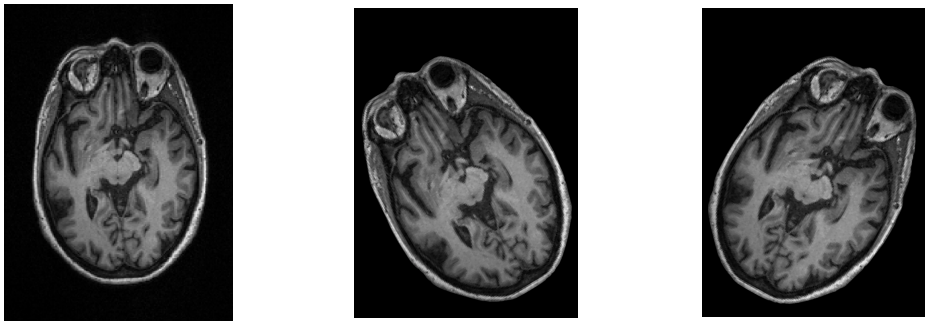
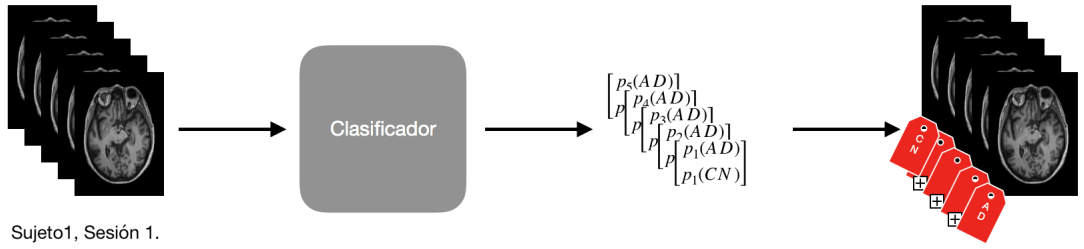


Figura 5.12: En esta figura observamos los resultados de aplicar rotaciones. En (a) vemos la imagen sobre la que aplicamos las rotaciones. En (b) se ha aplicado una rotación de 30º y en (c) de -30º.

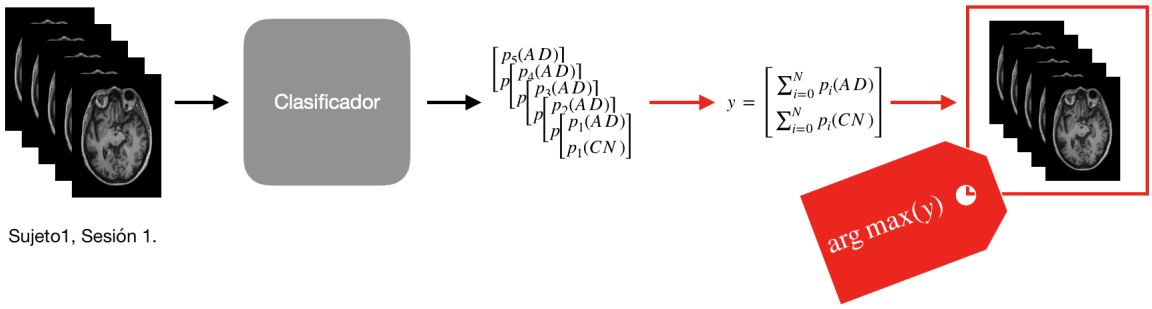
5.3.3. Clasificación por votación

En los experimentos que se presentarán a continuación, se recogen resultados de la tasa de acierto en la etapa de test utilizando dos modos diferentes de llevar a cabo la clasificación. Por un lado, utilizamos una clasificación estándar. Cada una de las imágenes del subconjunto de test pasan por el clasificador entrenado y se clasifica en la clase que éste determine. Por otro lado, proponemos una clasificación que proporciona algo más de contexto. Esto se consigue realizando una clasificación por votación de los cortes 2D que conforman una sesión de resonancia magnética de un paciente. Con ello, estamos logrando clasificar todo el cerebro y no solo una parte de él. De esta manera, conseguimos un doble objetivo: a) atenuar el efecto negativo que puedan tener aquellos cortes 2D que no tengan biomarcadores de la enfermedad y que por tanto, al estar mal clasificados, reducen la tasa de acierto y por otro, b) emitir un veredicto más fiable al tener en cuenta toda

la información analizada de un paciente. En la Figura 5.13 se puede ver una explicación gráfica de los dos modos de clasificación presentados.



(a)



(b)

Figura 5.13: En esta figura podemos observar en (a) el modo de clasificación estándar. En él, cada imagen pasa por el clasificador y se le asigna la etiqueta con mayor probabilidad en el vector de probabilidades que obtenemos en la salida del clasificador. En (b) podemos ver que todas las imágenes pasan por el clasificador y obtienen su vector de probabilidades. Despúes se obtiene un vector suma y, finalmente, a todas las imágenes de la misma sesión se les asigna la misma etiqueta, a saber, la que tenga mayor suma de probabilidades en el vector suma.

5.3.4. *Precision, Recall* y *F1-score*

En los experimentos que se presentarán en la próxima sección se presentarán junto con los datos obtenidos de tasa de acierto, otras tres métricas: *Precision*, *Recall* y *F1-score*. Para entender, más adelante, las conclusiones que se exponen de cada experimento, explicaremos en este apartado.

La precisión es la medida que nos proporciona cuánto acierta nuestra red neuronal. Es decir, tomando como referencia la clase AD, cuántas muestras de las que se han clasificado como AD son realmente de esa clase. Es una buena medida cuando los falsos positivos no tienen un impacto alto en nuestro problema. En nuestro caso, podemos tomar la precisión como una medida importante ya que buscamos cierta precisión en nuestro clasificador para que no clasifique demasiados sujetos sanos como si estuvieran enfermos. De esta manera ahorraremos segundos diagnósiticos para comprobar que los detectados como enfermos, lo sean realmente La fórmula con la que se calcula esta medida (adaptada a nuestro problema) se puede consultar en la Ecuación 5.4

$$Precision = \frac{VerdaderasAD}{VerdaderasAD + FalsasAD} = \frac{VerdaderasAD}{TotalPredichasAD} \tag{5.4}$$

Por otro lado, debemos prestar atención al *Recall*. Esta medida nos proporciona información, en nuestro caso, sobre cuántos pacientes que tienen Alzheimer, hemos clasificado como sanos. Por tanto necesitamos valores de *Recall* altos. Si no fuera el caso, nuestro clasificador no podría ser utilizado para ayudar al diagnóstico. El *recall* se calcula, en nuestro problema, como se detalla en la Ecuación 5.5

$$Recall = \frac{VerdaderasAD}{VerdaderasAD + FalsasCN} = \frac{VerdaderasAD}{TotalAD} \tag{5.5}$$

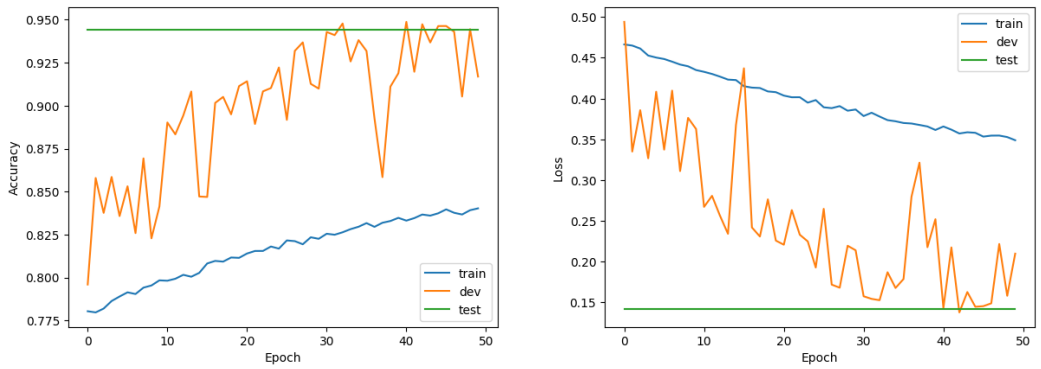
Por último, tenemos la medida del *F1-score*, que también nos es de gran utilidad en nuestro problema. Nos es de utilidad puesto que es un indicador del balance que existe entre Precisión y *Recall*. Como se puede deducir, en nuestro problema, buscamos un clasificador que tenga un balance entre precisión y *recall*, es decir, que no se deje pacientes enfermos por detectar, y que no clasifique pacientes que son sanos como si fueran enfermos. Así pues, un valor alto de *F1-score* es lo que buscaremos en los experimentos realizados. La fórmula con la que calculamos el *F1-score* se puede consultar en la Ecuación 5.6

$$F1\_score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{5.6}$$

5.4 Experimento 1

En este experimento vamos a comparar el comportamiento de los clasificadores A y B utilizando el *dataset* base (ADNI 2D BET). Además también veremos cómo se comporta este experimento cuando hacemos la partición que se propone en este trabajo frente a la que se utiliza en el estado del arte. Por otro lado, veremos si aplicar *Data Augmentation* a la entrada, tiene efectos significativos en los resultados. Por último, compararemos las tasas de acierto (*accuracy*) obtenidas utilizando los dos métodos de clasificación que se han comentado anteriormente: estándar y votación.

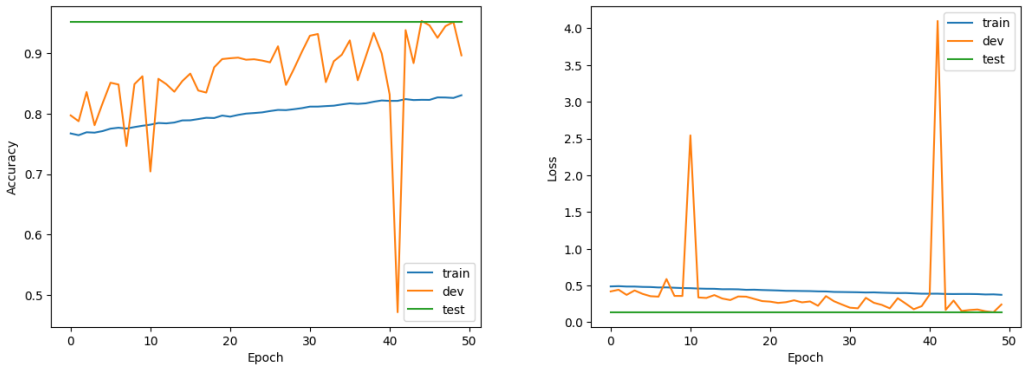
Para el clasificador A, utilizando data augmentation y la partición que se puede encontrar en el estado del arte, obtenemos los resultados de *accuracy* y *loss* que se pueden observar en la Figura 5.14. En este caso se ha utilizado la clasificación estándar.



(a) **(b)**

Figura 5.14: En (a) podemos ver las gráficas de *accuracy* en entrenamiento, validación y test. En (b) podemos ver el *loss* también en las tres etapas. Los resultados se corresponden al Experimento 1 utilizando el clasificador A y la partición utilizada por el estado del arte. En esta ocasión se ha utilizado data augmentation y clasificación estándar.

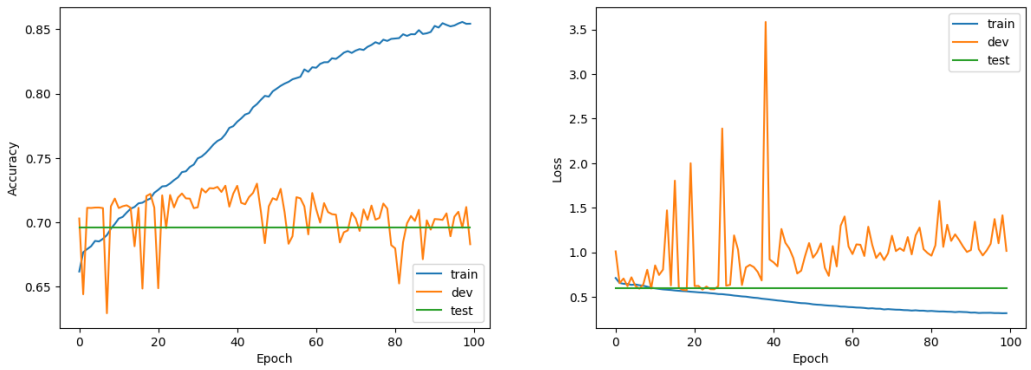
Para el clasificador B, bajo las mismas condiciones, obtenemos los resultados de *accuracy* y *loss* que se pueden observar en la Figura 5.15.



(a) **(b)**

Figura 5.15: En (a) podemos ver las gráficas de *accuracy* en entrenamiento, validación y test. En (b) podemos ver el *loss* también en las tres etapas. Los resultados se corresponden al Experimento 1 utilizando el clasificador B y la partición utilizada por el estado del arte. En esta ocasión se ha utilizado data augmentation y clasificación estándar.

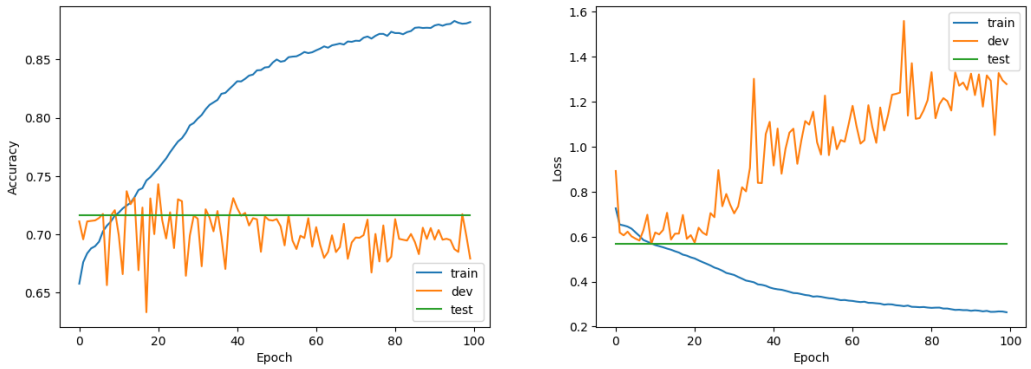
Vistos estos resultados, vamos a compararlos con los obtenidos cuando cambiamos la manera de particionar los datos. Ahora el particionado se realizada del modo que proponemos en este trabajo. De nuevo, utiilizando el clasificador A, data augmentation y la clasificación estándar, obtenemos los resultados presentados en la Figura 5.16



(a) **(b)**

Figura 5.16: En (a) podemos ver las gráficas de *accuracy* en entrenamiento, validación y test. En (b) podemos ver el *loss* también en las tres etapas. Los resultados se corresponden al Experimento 1 utilizando el clasificador A y la partición propuesta en este trabajo. En esta ocasión se ha utilizado data augmentation y clasificación estándar.

En este caso, para el clasificador B, bajo las mismas condiciones, obtenemos los resultados mostrados en la Figura 5.17.



(a) **(b)**

Figura 5.17: En (a) podemos ver las gráficas de *accuracy* en entrenamiento, validación y test. En (b) podemos ver el *loss* también en las tres etapas. Los resultados se corresponden al Experimento 1 utilizando el clasificador B y la partición propuesta en este trabajo. En esta ocasión se ha utilizado data augmentation y clasificación estándar.

A modo de resumen presentamos los datos obtenidos para la *accuracy* en la etapa de test en la Tabla 5.2. En esta tabla se presentan también los resultados en la etapa de test de los mismos experimentos, pero sin aplicar *data augmentation* (DA) a los datos. Por otro lado, se recogen, en la Tabla 5.3, los datos para la *accuracy* en la etapa de test, pero aplicando la clasificación por votación.

Tabla 5.2: Resultados de *accuracy* en test para el experimento 1.

Partición	Clasificador	DA	Accuracy	Precision	Recall	F1-score
Estado Arte	A	Sí	91,24 %	0,909401	0,974397	0,940778
Propuesta	A	Sí	69,62 %	0,746769	0,868969	0,803248
Estado Arte	A	No	96,54 %	0,971477	0,980326	0,975881
Propuesta	A	No	67,00 %	0,736475	0,837223	0,783624
Estado Arte	B	Sí	91,91 %	0,921755	0,968953	0,944765
Propuesta	B	Sí	71,65 %	0,719173	0,989004	0,832777
Estado Arte	B	No	96,73 %	0,971905	0,982644	0,977245
Propuesta	B	No	69,33 %	0,745452	0,865951	0,801196

Tabla 5.3: Resultados de *accuracy* en test para el experimento 1 con clasificación por votación.

Partición	Clasificador	DA	Accuracy	Precision	Recall	F1-score
Estado Arte	A	Sí	98,11 %	0,974358	1	0,987012
Propuesta	A	Sí	72,16 %	0,738461	0,946398	0,829971
Estado Arte	A	No	100 %	1	1	1
Propuesta	A	No	69,33 %	0,714285	0,953947	0,816901
Estado Arte	B	Sí	99,52 %	0,993464	1	0,996721
Propuesta	B	Sí	71,89 %	0,716981	1	0,835164
Estado Arte	B	No	100 %	1	1	1
Propuesta	B	No	72,16 %	0,729064	0,973684	0,833802

5.4.1. Detalles, resultados y valoración del experimento

Como ya se anticipó en anteriores capítulos, la diferencia en cuanto a resultados entre el uso de una partición y la otra es, como se puede apreciar en las tablas 5.2 y 5.3, significativa. Como se ha podido ver en las gráficas adjuntas, en ninguna de las cuatro, la tasa

de acierto en entrenamiento supera el 90 %. Sin embargo, podemos ver que en validación, los resultados cambian entre particiones. En las figuras 5.14 y 5.15 podemos ver que la tasa de acierto en validación es siempre más alta que en entrenamiento. Esto se debe a que imágenes del mismo paciente aparecen en los dos subconjuntos, con lo cual, al reconocer el paciente, asigna la etiqueta que tiene ese paciente y que se ha aprendido durante el entrenamiento. Sin embargo, en las figuras 5.16 y 5.17 vemos que la tasa de acierto en validación está, casi siempre, por debajo del acierto en entrenamiento. A pesar de ello, creemos que aunque la tasa sea más baja, el reconocimiento de la enfermedad mejora con la partición propuesta que con la utilizada en los experimentos del estado del arte. Asimismo, en la etapa de test observamos lo que habíamos avanzado desde el principio del trabajo, dada una misma topología, se alcanzan resultados peores si se hace la partición como se propone en este trabajo. En cuanto al uso o no de técnicas de *data augmentation* vemos que los resultados no son esclarecedores. La variación de los resultados entre una opción y la otra no es suficiente como para poder determinar si se mejora o se empeora la tasa de acierto.

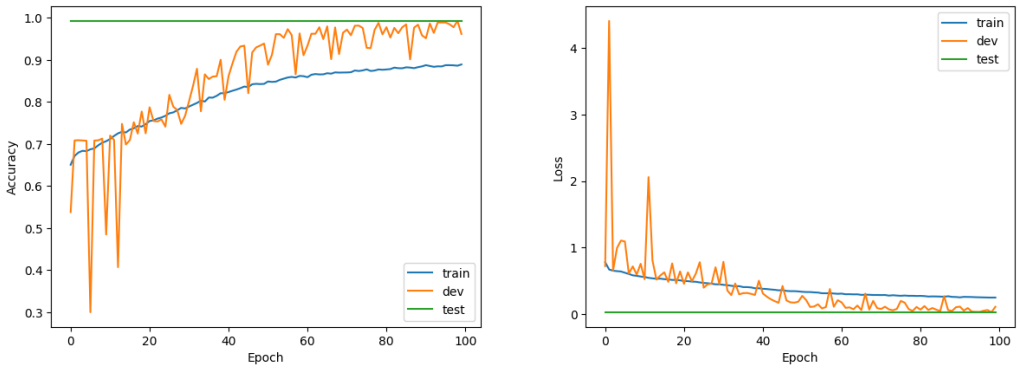
Si comparamos los resultados que presentamos en la Tabla 5.2 y los que presentamos en la 5.3, vemos que en esta última los resultados son, por lo general, varios puntos porcentuales mejores que en la primera en términos de tasa de acierto. Pero no solo son mejores cuantitativamente (*accuracy*) ,sino también cualitativamente. Podemos decir que la precisión en ambos casos se mantiene alrededor del mismo valor sin variar demasiado. Además, no es un valor que podamos considerar bajo, aunque sí que tiene cierto margen de mejora. De la misma manera, vemos que los valores de *recall* y, por tanto, de *F1-score* son mayores con la clasificación mediante votación. En este sentido, podemos decir que los resultados con la clasificación por votación son mejores que los resultados obtenidos al utilizar la clasificación estándar. En general, con la clasificación por votación se suelen etiquetar como enfermos bastantes sujetos que realmente están sanos, pero sí que podemos asegurar que casi ningún paciente enfermo es clasificado erróneamente por el clasificador.

Por lo tanto, el experimento es mejorable en términos de precisión. Se necesita encontrar una forma de reducir el número de pacientes sanos que se clasifican como enfermos.

5.5 Experimento 2

En este experimento vamos a comparar el comportamiento de los clasificadores A y B, pero esta vez utilizando el *dataset* "ADNI 2D 3 cortes" que hemos introducido en la Subsección 5.3.2. De la misma forma que hemos hecho con el Experimento 1, compararemos las variaciones que existen entre ambas maneras de realizar la partición de los datos y veremos si el uso de *data augmentation* produce resultados relevantes o no. Asimismo, veremos cómo afecta a los resultados el uso de la clasificación estándar frente a la clasificación por votación.

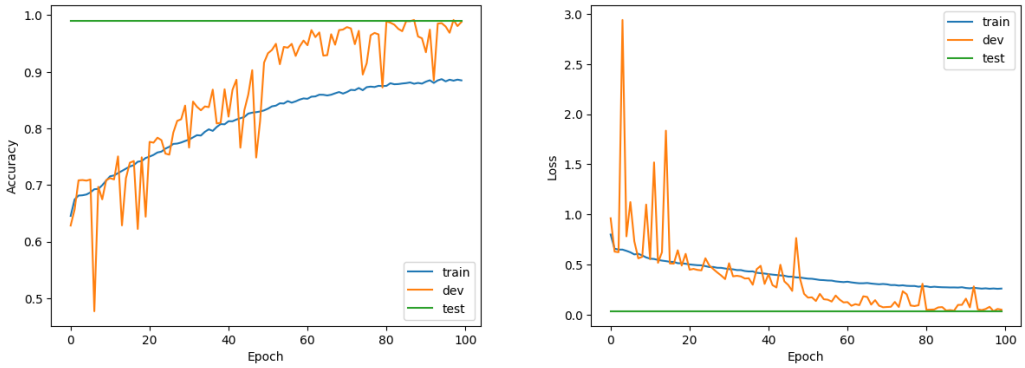
Comenzamos por el comportamiento de los dos clasificadores para el caso en el que se utiliza la partición de los datos presente en los experimentos que conforman el estado del arte y se aplica el método de clasificación estándar. Los resultados de *accuracy* y *loss* para el clasificador A son los que se presentan en la Figura 5.18.



(a) **(b)**

Figura 5.18: En (a) podemos ver las gráficas de *accuracy* en entrenamiento, validación y test. En (b) podemos ver el *loss* también en las tres etapas. Los resultados se corresponden al Experimento 2 utilizando el clasificador A y la partición propuesta en el estado del arte. En esta ocasión se ha utilizado data augmentation y clasificación estándar.

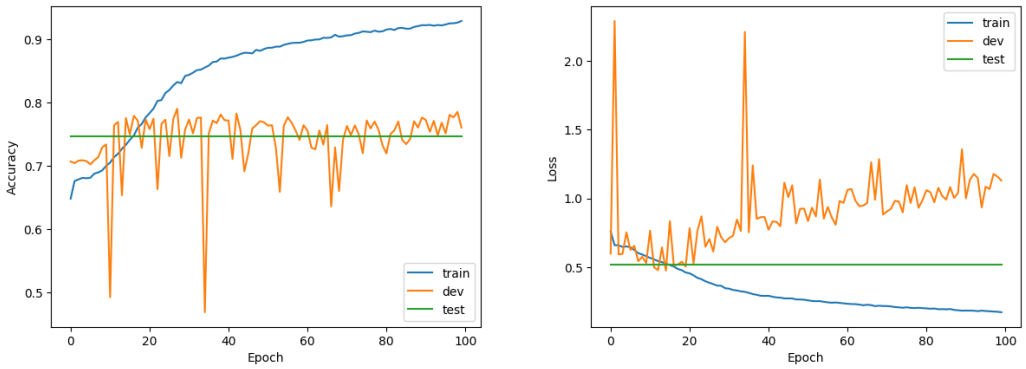
Por otro lado, los resultados del clasificador B con la partición presentada en los artículos consultados y clasificación estándar se presentan en la Figura 5.19



(a) **(b)**

Figura 5.19: En (a) podemos ver las gráficas de *accuracy* en entrenamiento, validación y test. En (b) podemos ver el *loss* también en las tres etapas. Los resultados se corresponden al Experimento 2 utilizando el clasificador B y la partición propuesta en el estado del arte. En esta ocasión se ha utilizado data augmentation y clasificación estándar.

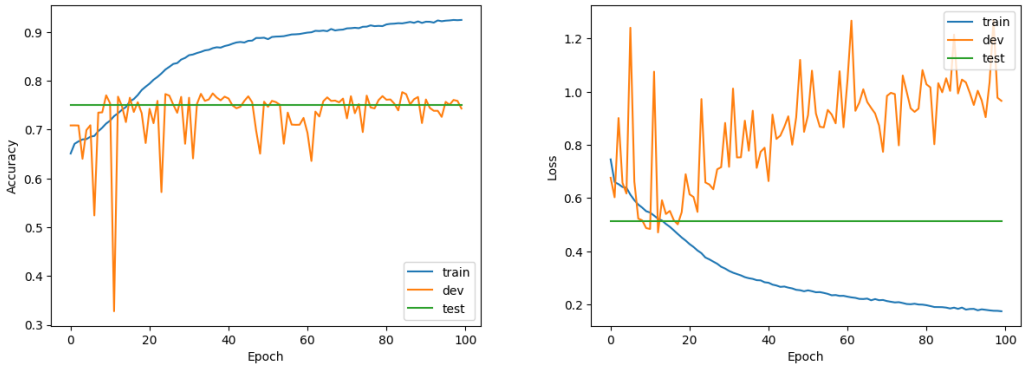
De forma análoga al Experimento 1, ahora mostraremos el comportamiento de ambos clasificadores, entrenados con el mismo *dataset* (ADNI 2D 3 cortes), pero utilizando, en este caso, la partición propuesta en este trabajo. Para ambos clasificadores se utilizad la clasificación estándar. Los resultados del clasificador A se presentan en la Figura 5.20.



(a) **(b)**

Figura 5.20: En (a) podemos ver las gráficas de *accuracy* en entrenamiento, validación y test. En (b) podemos ver el *loss* también en las tres etapas. Los resultados se corresponden al Experimento 2 utilizando el clasificador A y la partición propuesta en este trabajo. En esta ocasión se ha utilizado data augmentation y clasificación estándar.

Los resultados obtenidos con el clasificador B, se presentan en la Figura 5.21.



(a) **(b)**

Figura 5.21: En (a) podemos ver las gráficas de *accuracy* en entrenamiento, validación y test. En (b) podemos ver el *loss* también en las tres etapas. Los resultados se corresponden al Experimento 2 utilizando el clasificador B y la partición propuesta en este trabajo. En esta ocasión se ha utilizado data augmentation y clasificación estándar.

A modo de resumen recogemos los resultados obtenidos para la *accuracy* en test en las tablas 5.4 y 5.5. En ambas tablas se comparan los resultados obtenidos cuando se aplica o no *data augmentation* (DA). El la primera tabla el método de clasificación usado es el estándar. En la segunda se utiliza la clasificación por votación.

Tabla 5.4: Resultados de *accuracy* en test para el experimento 2.

Partición	Clasificador	DA	Accuracy	Precision	Recall	F1-score
Estado Arte	A	Sí	96,49 %	0,970512	0,980397	0,975430
Propuesta	A	Sí	74,62 %	0,796682	0,862976	0,828505
Estado Arte	A	No	97,96 %	0,982161	0,989334	0,985734
Propuesta	A	No	71,18 %	0,716334	0,983953	0,829082
Estado Arte	B	Sí	96,29 %	0,964231	0,984337	0,974180
Propuesta	B	Sí	75,20 %	0,764256	0,941385	0,843623
Estado Arte	B	No	97,87 %	0,979210	0,991159	0,985148
Propuesta	B	No	68,58 %	0,780657	0,775631	0,778136

Tabla 5.5: Resultados de *accuracy* en test para el experimento 2 con clasificación por votación.

Partición	Clasificador	DA	Accuracy	Precision	Recall	F1-score
Estado Arte	A	Sí	100 %	1	1	1
Propuesta	A	Sí	78,40 %	0,795772	0,93421	0,860606
Estado Arte	A	No	100 %	1	1	1
Propuesta	A	No	71,36 %	0,713615	1	0,832876
Estado Arte	B	Sí	100 %	1	1	1
Propuesta	B	Sí	79,81 %	0,779487	1	0,876080
Estado Arte	B	No	100 %	1	1	1
Propuesta	B	No	72,30 %	0,781818	0,848684	0,813880

5.5.1. Detalles, resultados y valoración del experimento

En este experimento se puede comprobar que el comportamiento al cambiar el modo de particionar los datos, afecta a los resultados de manera similar a como afectaba en el Experimento 1. Por tanto, se demuestra de nuevo la hipótesis que venimos planteando desde el inicio del trabajo. Por otro lado, si que podemos ver que todos los resultados son ligeramente mejores que los presentados en el Experimento 1. Así pues, podríamos decir que la técnica en la cual se añade algo de contexto a un corte 2D, mejora los resultados.

Si comparamos los resultados obtenidos al cambiar el método de clasificación, observamos que todos tienen una tasa de acierto superior en varios puntos porcentuales cuando se utiliza la clasificación por votación. Además, como norma general, los resultados en la tasa de acierto son mayores que los obtenidos en el Experimento 1. Como sucedía en el anterior experimento, la calidad del los clasificadores también es mayor. En el experimento 1 extraíamos como conclusión que era necesario aumentar la precisión del clasificador y como se puede comprobar, en este experimento se ha conseguido. Mientras en el experimento 1 conseguíamos que de cada 10 casos de Alzheimer detectados 3 fueran falsos positivos, ahora hemos reducido esa tasa a tan solo 2 falsos positivos por cada 10 casos de Alzheimer detectado. Además, el *recall* es mayor, del 100 % en la mayoría de variantes presentadas. Como consecuencia del aumento de precisión y *recall*, el *F1-score* también ha incrementado. Esto significa que los clasificadores obtenidos en este experimento son, en general, mejores que los obtenidos en el experimento 1 y por tanto serían de mayor ayuda para los facultativos.

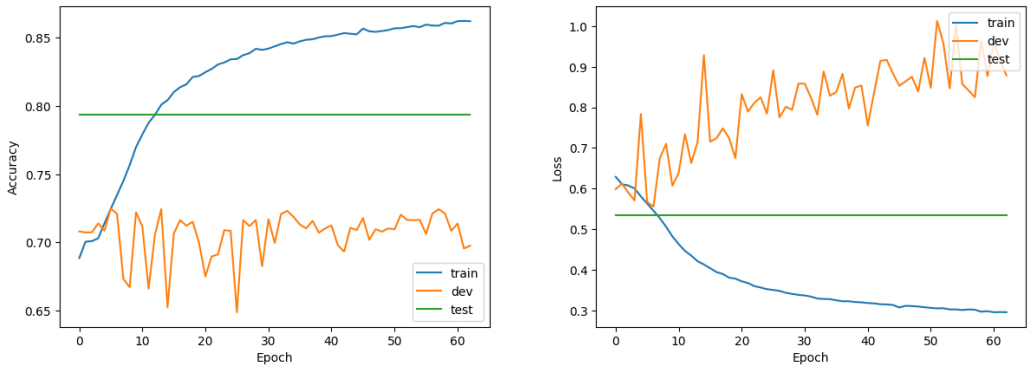
Sin embargo, se podría plantear la hipótesis de que, si trabajamos con las imágenes en 3D, los resultados serán mejores ya que cada corte tiene el contexto completo del cerebro al que pertenece. Comprobar esta hipótesis, escapa los límites de este trabajo. Sin embargo, como se ha comentado en la introducción, las investigaciones en la materia no se han hecho de forma individual, sino en conjunto con otros dos compañeros. Es en el trabajo de Álvaro López Chilet dónde podemos encontrar experimentos sobre la materia utilizando las imágenes de ADNI en tres dimensiones.

5.6 Experimento 3

En este experimento vamos a comparar el comportamiento de los clasificadores C y D utilizando el *dataset* preprocesado con la herramienta FLIRT (ADNI 2D FLIRT). Recordemos que los clasificadores C y D son pequeñas variaciones de los clasificadores A y B que se han adaptado al nuevo tamaño de los datos. En este caso solamente trabajaremos con la partición de los datos que se propone en este trabajo. Además compararemos los resultados obtenidos en los casos en los que hemos aplicado *Data Augmentation*, con

los casos en los que no. Por último, compararemos las tasas de acierto (*accuracy*) obtenidas utilizando los dos métodos de clasificación que se han visto en los experimentos anteriores.

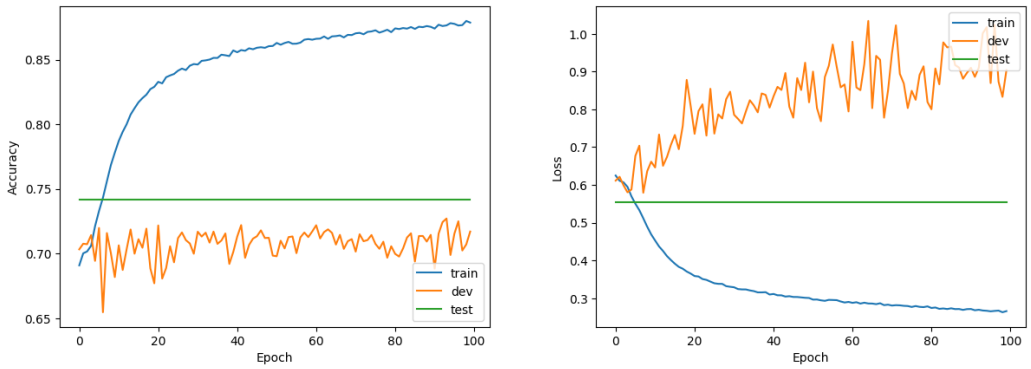
Para el clasificador C, utilizando data augmentation, obtenemos los resultados de *accuracy* y *loss* que se pueden observar en la Figura 5.22. En este caso se ha utilizado la clasificación estándar.



(a) **(b)**

Figura 5.22: En (a) podemos ver las gráficas de *accuracy* en entrenamiento, validación y test. En (b) podemos ver el *loss* también en las tres etapas. Los resultados se corresponden al Experimento 3 utilizando el clasificador C y la partición propuesta en este trabajo. En esta ocasión se ha utilizado data augmentation.

Para el clasificador D, bajo las mismas condiciones, obtenemos los resultados de *accuracy* y *loss* que se pueden observar en la Figura 5.23.



(a) **(b)**

Figura 5.23: En (a) podemos ver las gráficas de *accuracy* en entrenamiento, validación y test. En (b) podemos ver el *loss* también en las tres etapas. Los resultados se corresponden al Experimento 3 utilizando el clasificador D y la partición propuesta en este trabajo. En esta ocasión se ha utilizado data augmentation.

A modo de resumen presentamos los datos obtenidos para la *accuracy* en la etapa de test en la Tabla 5.6. En esta tabla se presentan también los resultados en la etapa de test de los mismos experimentos, pero sin aplicar *data augmentation* (DA) a los datos. Por otro lado, se recogen, en la Tabla 5.7, los datos para la *accuracy* en la etapa de test, pero aplicando esta vez la clasificación por votación.

Tabla 5.6: Resultados de *accuracy* en test para el experimento 3.

Clasificador	DA	Accuracy	Precision	Recall	F1-score
C	Sí	79,39 %	0,723157	0,907439	0,835042
C	No	66,87 %	0,723157	0,868739	0,789291
D	Sí	74,17 %	0,776109	0,897216	0,832280
D	No	70,68 %	0,728929	0,823125	0,800417

Tabla 5.7: Resultados de *accuracy* en test para el experimento 3 con clasificación por votación.

Clasificador	DA	Accuracy	Precision	Recall	F1-score
C	Sí	81,90 %	0,814606	0,966666	0,884146
C	No	71,42 %	0,714285	1	0,833333
D	Sí	83,33 %	0,824858	0,973333	0,891803
D	No	84,28 %	0,877419	0,906666	0,892966

5.6.1. Detalles, resultados y valoración del experimento

En este último experimento hemos obtenido los mejores resultados de entre todos los presentados en el trabajo. Esto lo relacionamos directamente con el preprocesado de los datos mediante la herramienta FLIRT. Como se ha visto en el Capítulo 4 la estructura cerebral se deforma por completo para parecerse a la estructura del cerebro de referencia seleccionado. Creemos que este efecto tiene un impacto positivo en el proceso de entrenamiento de la red. Ahora es más complicado para la red neuronal detectar el cerebro de un paciente en concreto y asignarle sistemáticamente la misma etiqueta. Con lo cual, podemos estar más seguros de que los aciertos que se producen probablemente sean gracias a algún biomarcador de la enfermedad.

En cuanto a la calidad de los clasificadores, destacamos el clasificador D entrenado sin *Data Augmentation* como el mejor. Con él, obtenemos un valores de precisión y *recall* cercanos a 0,9. Esto nos acerca más a poder implantar el clasificador para su uso en casos reales.

CAPÍTULO 6
Conclusiones y desarrollos futuros

La realización de este trabajo ha tenido como resultado conclusiones que nos ayudan a continuar con las investigaciones en el futuro. En este capítulo vamos a presentar dichas conclusiones y, además, repasaremos la consecución de los objetivos que tenía el trabajo en su inicio.

Uno de los objetivos que se estableció al inicio del trabajo fue el estudio de aproximaciones y soluciones recientemente publicadas y reproducir algunas de ellas para contrastar resultados. En el caso de este trabajo y tal y como se presenta en el Capítulo 2 nos hemos basado en varios artículos científicos de grupos que ya han trabajado con el mismo problema y que han publicado los resultados. Hemos tomado como base los experimentos que se realizan en ellos, para adaptarlos a la resolución de nuestro problema. Tras la realización de los experimentos, no se puede determinar si se ha conseguido igualar los resultados obtenidos en las mencionadas producciones científicas. Y no se puede determinar porque, como se ha expuesto a lo largo de esta memoria, no es posible determinar como se han llevado a cabo algunos aspectos importantes para replicar los experimentos, en concreto el particionado de los datos. Hemos demostrado que haciendo un particionado aleatorio igualamos e incluso mejoramos los resultados de los experimentos citados. Sin embargo, nos mantenemos firmes en la creencia de que este tipo de particionado no es correcto. Si imágenes de un mismo paciente, independientemente de si son o no de la misma sesión, alteran, de forma significativa, los resultados del clasificador.

Otro objetivo que se planteó fue el análisis y prueba de distintas técnicas de preprocesado para determinar cuales son las más apropiadas para el problema. Quedó demostrado que el problema es complejo en su resolución desde el punto de vista del análisis de imágen médica. Mientras en otros problemas, las redes neuronales convolucionales (CNN), no necesitan de grandes preprocesados de los datos para poder encontrar características de las mismas que puedan servir para la clasificación, el problema que nosotros tratamos de resolver sí que necesita de esta etapa de preprocesado. La razón de ello es que la estructura cerebral es muy compleja y varía de un cerebro a otro. Por tanto no se puede trabajar con las imágenes originales. Como sucede en gran parte de los artículos consultados, se hace necesaria la extracción del parénquima. Además, dado que la forma de cada cabeza es diferente, el cerebro también lo es. Esto nos produce la necesidad de llevar todos los cerebros a un mismo espacio donde poderlos comparar. Para ello, FLIRT deforma todos los cerebros con respecto a uno de referencia. Así pues, para poder abordar el problema de manera fiable, concluimos que es necesario el preprocesado de los datos utilizando, como mínimo, las herramientas BET y FLIRT.

Por último, se planteó como objetivo el diseño de varias topologías de redes neuronales profundas y evaluarlas con el *dataset* ADNI para compararlas con otras soluciones que también lo usen. Concluimos, tras todos los experimentos realizados, que la aplicación

de redes neuronales profundas en este problema es viable. Aún así, nos encontramos en una fase temprana que necesita seguir mejorando para llegar a cumplir el objetivo del problema planteado, a saber, implantar un clasificador binario que sea capaz de ayudar al diagnóstico médico de manera fiable.

En cuanto a las líneas de trabajo futuras, se propone encontrar maneras de mejorar los clasificadores binarios utilizando las imágenes obtenidas por resonancia magnética en dos dimensiones. Se propone utilizar imágenes en dos dimensiones ya que a la hora de aplicar el clasificador en un caso real, la mayoría de hospitales no cuentan con máquinas capaces de realizar estas pruebas en tres dimensiones.

Además, uno de los puntos clave está en conseguir clasificadores que únicamente utilicen imágenes obtenidas por resonancia magnética. Aunque existen soluciones propuestas por algunos autores que además se apoyan en imágenes obtenidas por emisión de positrones y tienen buenos resultados, creemos que hay que evitar su uso, pues estas imágenes se obtienen mediante un proceso caro, e invasivo para el paciente.

Finalmente, los clasificadores se podrían mejorar en el futuro haciéndolos capaces de detectar también los diferentes grados de deterioro cognitivo que pueda sufrir un paciente, es decir, incluyendo las etiquetas que, en los experimentos de este trabajo, no hemos considerado.

Bibliografía

[1] Claves sobre la enfermedad de Alzheimer Consultado en https://blog.fpmaragall.org/las-fases-de-la-enfermedad-de-alzheimer.

[2] Notas descriptivas OMS: Demencia. Diciembre de 2017. Consultado en https://www.who.int/es/news-room/fact-sheets/detail/dementia.

[3] Aderghal, Karim y Boissenin, M y Benois-Pineau, Jenny y Catheline, GwenaÃ«lle y Karim, Afdel
Classification of sMRI for AD Diagnosis with Convolutional Neuronal Networks: A Pilot 2-D-ϵ Study on ADNI *Lecture Notes in Computer Science*, 10.1007/978-3-319-51811-4_56, enero, 2017.

[4] Payan, Adrien y Montana, Giovanni
Predicting Alzheimer's disease: a neuroimaging study with 3D convolutional neural networks *arXiv e-prints*, 1502.02506, febrero, 2015.

[5] Tensorflow Consultado en https://en.wikipedia.org/wiki/TensorFlow?.

[6] Keras Consultado en https://en.wikipedia.org/wiki/Keras.

[7] V. Popescu y M. Battaglini y W.S. Hoogstrate y S.C.J. Verfaillie y I.C. Sluimer y R.A. van Schijndel y B.W. van Dijk y K.S. Cover y D.L. Knol y M. Jenkinson y F. Barkhof y N. de Stefano y H. Vrenken
Optimizing parameter choice for FSL-Brain Extraction Tool (BET) on 3D T1 images in multiple sclerosis *NeuroImage*, 1053-8119, 2012.

[8] S.M. Smith
Fast robust automated brain extraction *Human Brain Mapping*, 17(3):143-155, Noviembre 2002.

[9] Rachna Jain, Nikita Jain, Akshay Aggarwal, D. Jude Hemanth
Convolutional neural network based Alzheimer?s disease classification from magnetic resonance brain images *Cognitive Systems Research*, Diciembre 2018.

[10] H. Wang, Y. Shen, S. Wang, T. Xiao, L. Deng, X. Wang, X. Zhao
Ensemble of 3D densely connected convolutional network for diagnosis of mild cognitive impairment y Alzheimer's disease *Neurocomputing*, Diciembre 2018.

[11] The Most Intuitive y Easiest Guide for Convolutional Neural Network. Enero de 2019. Consultado en https://towardsdatascience.com/the-most-intuitive-y-easiest-guide-for-convolutional-neural-network-3607be47480.

[12] Rachna Jain y Nikita Jain y Akshay Aggarwal y D. Jude Hemanth
Convolutional neural network based Alzheimer's disease classification from magnetic resonance brain images *Cognitive Systems Research*, 2019.

[13] Karen Simonyan y yrew Zisserman
Very Deep Convolutional Networks for Large-Scale Image Recognition *arXiv preprint*, 2014.

[14] Jack Jr., Clifford R. y Bernstein, Matt A. y Fox, Nick C. y Thompson, Paul y Alexyer, Gene y Harvey, Danielle y Borowski, Bret y Britson, Paula J. y L. Whitwell, Jennifer y Ward, Chadwick y Dale, yers M. y Felmlee, Joel P. y Gunter, Jeffrey L. y Hill, Derek L.G. y Killiany, Ron y Schuff, Norbert y Fox-Bosetti, Sabrina y Lin, Chen y Studholme, Colin y DeCarli, Charles S. y Gunnar Krueger y Ward, Heidi A. y Metzger, Gregory J. y Scott, Katherine T. y Mallozzi, Richard y Blezek, Daniel y Levy, Joshua y Debbins, Josef P. y Fleisher, Adam S. y Albert, Marilyn y Green, Robert y Bartzokis, George y Glover, Gary y Mugler, John y Weiner, Michael W.
The Alzheimer's disease neuroimaging initiative (ADNI): MRI methods *Journal of Magnetic Resonance Imaging*, 2008.

[15] A detailed example of how to use data generators with Keras Consultado en https://stanford.edu/~shervine/blog/keras-how-to-generate-data-on-the-fly.

APÉNDICE A

Capas, funciones de activación y regularización en Keras

En este apéndice se describirán las diferentes capas que incluye Keras y que han sido utilizadas en los modelos presentados en el Capítulo 5. Separados en secciones encontraremos a continuación los fundamentos de cada una de las capas y figuras que nos ayudarán a comprender mejor su funcionamiento. Además se explicarán las principales funciones de activación y las técnicas de regularización utilizadas.

A.1 Capas Neuronales

Las capas neuronales se crean a través de las llamadas que proporciona Keras. Internamente Keras utiliza, en este caso, Tensorflow para crear el modelo.

A.1.1. Capa Convolucional

Para comprender el funcionamiento de las capas convolucionales, primero debemos pensar cómo nuestro cerebro es capaz de reconocer imágenes. Nuestro cerebro reconoce bordes y patrones en las imágenes y son éstos los que afectan a nuestra percepción. En resumen, nuestro cerebro analiza las imágenes para extraer patrones característicos de una imagen que nos permita clasificar los objetos que en ella aparecen. Esto sucede tan rápido que casi no nos damos cuenta de ello.

Las capas convolucionales, que dan lugar a las CNN[1] se basan en este principio para extraer conocimiento de las imágenes que se le dan como entrada a la red. En este caso, los patrones de los que hablábamos, se denominan *kernels* o filtros. En este punto, dados el filtro y la imagen de entrada, se puede proceder a la operación de convolución, que no es más que una operación matemática que, como resultado, expresa cómo la forma de un objeto se modifica por efecto de otro objeto.

Para ver cómo funciona una convolución, extraemos de [11] un ejemplo que queda reflejado en la Figura A.1. En la figura, tenemos un array unidimensional en el que aparecen varios valores 0 seguidos y en un momento dado, comienzan a aparecer valores 1. Lo que queremos saber es en qué posición del array se produce el cambio de valores 0 a valores 1. Para ello utilizamos el filtro sencillo { -1, 1 } que pasamos, por las diferentes posiciones del array hasta obtener el resultado mostrado. Un nuevo array, con un 1 marcando la posición donde se produce el cambio.

[1] Convolutional Neural Networks.

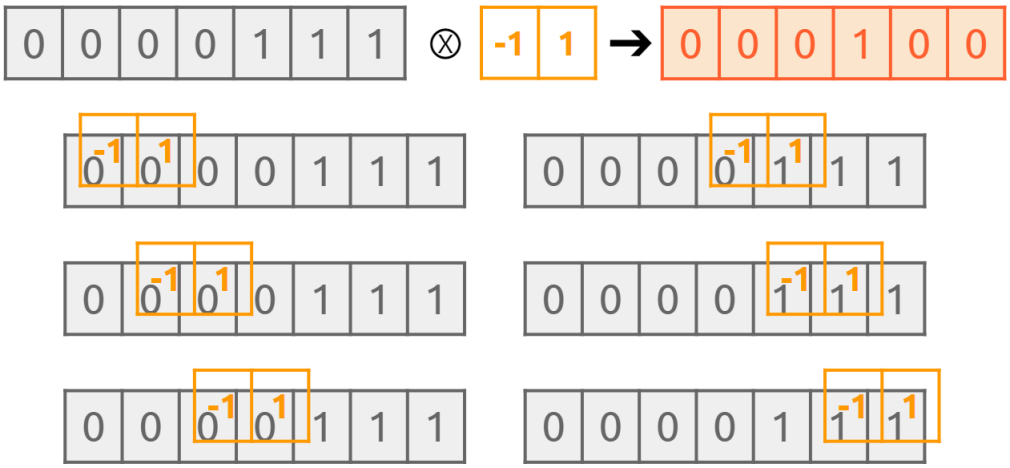
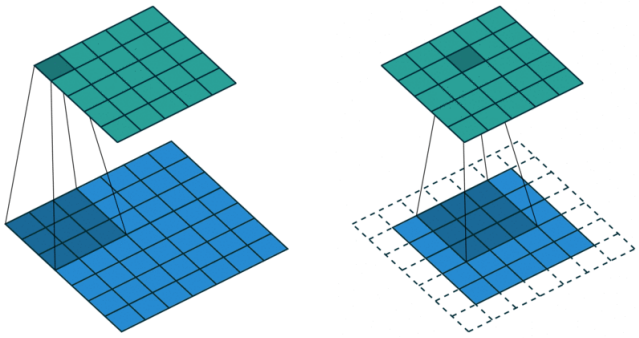


Figura A.1: En esta imagen se muestra como, de un array unidimensional, se puede extraer una característica utilizando la operación de convolución con el filtro adecuado. La imagen ha sido extraída de [11].

En cualquier caso, en las capas convolucionales de las CNN, los filtros no se preestablecen, sino que se inicializan con valores aleatorios y en el proceso de *backpropagation* se van modificando. Así pues, se obtienen filtros especializados en las partes más significativas de las imágenes de entrada para el reconocimiento de las zonas (o de los objetos) que más valor tienen para resolver el problema para el que estamos entrenando a la red. En este trabajo se han utilizado convoluciones 2D y 3D para trabajar con diferentes modelos de imagen médica. El funcionamiento es el mismo, solo que extrapolado a más dimensiones.

Por último, definiremos dos términos: *padding* y *stride*. Cuando estamos aplicando la operación de convolución tanto en 2D como en 3D a imágenes, los píxeles en las esquinas, tienen menos peso, porque se les aplican menos filtros. Esto puede provocar que, si existe información relevante en esos píxeles, la perdamos. Para evitar esto, usamos el *padding*. En Keras tiene dos valores principales: *valid* y *same*. *Valid* hace la reducción de tamaño tal y como se ha podido ver en el resultado de la Figura A.1. Sin embargo, *same* añade píxeles alrededor de la imagen original mediante la técnica del *zero-padding*, para que el resultado obtenido de la operación mantenga el mismo tamaño que la imagen de entrada. La diferencia se puede comprobar en la figura A.2



(a) **(b)**

Figura A.2: En (a) podemos ver como funciona la convolución cuando hacemos padding=valid. En (b) podemos ver como funciona la convolución cuando hacemos padding=same. La imagen ha sido extraída de [11].

El otro término que hemos mencionado es el de *stride*. El stride es el parámetro que determina, en el caso de las imágenes, cuántos píxeles se debe mover el filtro. Se compone de dos valores, alto y ancho, para el caso de imágenes 2D y de tres, alto, ancho y profundo para el caso en 3D.

A.1.2. Capa *MaxPool*

La técnica de *Pooling* se suele utilizar en conjunto con las capas convolucionales. Normalmente, usamos más de un filtro para analizar las imágenes en las capas convolucionales, con lo que, tras sucesivas convoluciones, tenemos un volumen ingente de datos que pueden estropear el entrenamiento de la red. Para ello, tras unas cuantas convoluciones, se suele utilizar *Pooling* para reducir el tamaño de los datos. Exactamente, no reducimos el tamaño de los datos aleatoriamente, sino siguiendo un criterio, es decir, de los resultados obtenidos eliminamos aquellos datos que no son significativos y que por tanto consideramos ruido, evitando así *overfitting* y consiguiendo una mayor velocidad de entrenamiento. Hay diferentes criterios para hacer *Pooling*, pero el que se utiliza en este trabajo es el *MaxPooling*.

El *MaxPooling* consiste en pasar una especie de ventana por la imagen y coger el máximo valor de todos los píxeles que caen dentro de la ventana. Como sucede en las capas convolucionales se puede ajustar el *stride*. En la figura A.3 se puede ver el resultado de aplicar *MaxPooling* a una imagen 2D.

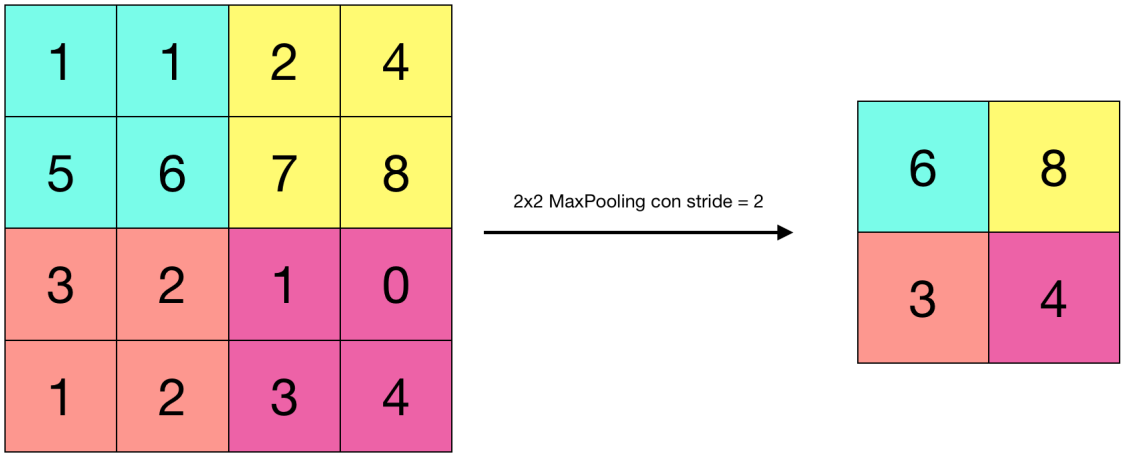


Figura A.3: En esta imagen se muestra cómo, aplicando MaxPooling con una ventana de 2x2 y *stride* = 2 a la imagen de la izquierda, se obtiene como resultado la imagen de la derecha.

A.1.3. Capa *Dense*

Las capas *Dense* en Keras crean lo que se conoce como capas *fully-connected*. A este tipo de capas, se le indica el número de neuronas que deben tener y se caracterizan por estar todas ellas conectadas a cada una de las neuronas de la capa inmediatamente siguiente. Se utilizan normalmente después de una CNN y son las encargadas de la parte de clasificación en diferentes clases de la red neuronal.

A.1.4. Operación Flatten

La conexión entre la salida de la última capa convolucional y la primera capa *fully-connected* debe cumplir una restricción, los datos de salida deben estar en un array unidimensional. Esta restricción se puede conseguir calculando minuciosamente las convoluciones con sus parámetros de *padding* y *stride* así como las operaciones de *pooling* correspondientes con el objetivo de reducir los datos sucesivamente hasta quedar en un array unidimensional. Sin embargo, esto no es necesario. Cuando los datos ya comienzan a tener un tamaño pequeño, podemos hacer servir la operacción *Flatten* que convierte el tamaño de esos datos en un array de una dimensión. Así, muchas veces se hace uso de esta operación para conectar correctamente la salida de la última capa convolucional con la primera capa *fully-connected.*

A.2 Funciones de activación

La unidad fundamental de las redes neuronales son las propias neuronas. Justo como ocurre en las neuronas de un cerebro, en las neuronas artificiales se dan conexiones entre ellas, unas más fuertes que otras, y se establecen caminos entre ellas capaces de activar diferentes funciones cerebrales. Por supuesto, en el cerebro, no están activadas todas las neuronas existentes a la vez. Dependiendo de la actividad que se realiza, se activan unas neuronas y se desactivan otras. En el caso de las redes neuronales, las neuronas no son más que unidades que hacen el cálculo que se puede ver en la Ecuación A.1

$$output = \sum (weight * input) + bias \tag{A.1}$$

El problema es que el *output* de la neurona no tiene por que ser un output binario que nos permite fácilmente determinar si la neurona está o no activada. Así pues, para establecer la relación entre los valores de salida de una neurona y su estado de activación, utilizamos las funciones de activación. En esta sección se explican las que se han utilizado en este trabajo.

A.2.1. *Softmax*

La función de activación *softmax* es muy utilizada en tareas de clasificación ya que transforma los valores de entrada, en un vector de probabilidades con tantas componentes como clases existan. Cada componente del vector expresa la probabilidad de pertenencia de la muestra de entrada a la clase que representa dicha componente.

Esta función de activación se suele utilizar en la última capa de neuronas de una red dedicada a clasificar las muestras en diversas clases. Esta última capa tiene tantas neuronas como clases y con la función *softmax* se transforma la salida de la capa en un vector de probabilidades. La manera de transformar los valores en el vector de probablididades resultante se puede ver en la Ecuación A.2

$$y = \begin{pmatrix} 2{,}0 \\ 1{,}0 \\ 0{,}1 \end{pmatrix} \rightarrow S(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}} \rightarrow P(y) = \begin{pmatrix} 0{,}7 \\ 0{,}2 \\ 0{,}1 \end{pmatrix} \tag{A.2}$$

A.2.2. *Rectified Linear Unit*

La función Rectified Linear Unit (ReLU) suele utilizarse sobre todo en las neuronas que forman parte de capas convolucionales. ReLU se define matemáticamente como $y = \text{máx}(0, x)$ que queda representado gráficamente en la Figura A.4

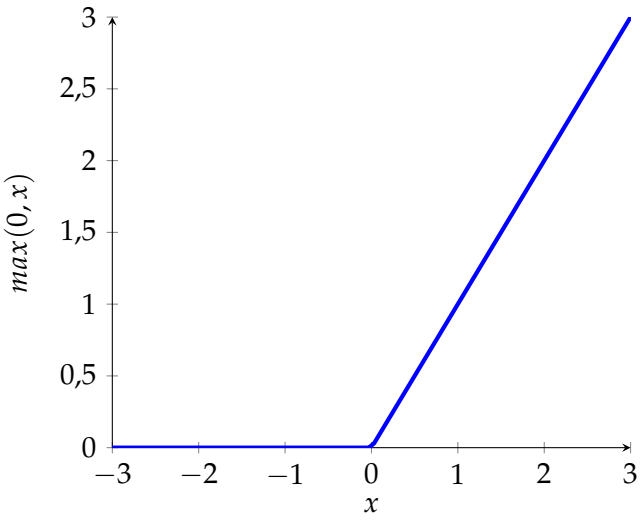


Figura A.4: Representación gráfica de la función de activación ReLU en el dominio [-3, 3].

Como se puede ver, una vez la neurona entra en la parte negativa, queda desactivada y es muy difícil volverla a activar. Es por esto que es la función de activación por antonomasia de las redes neuronales convolucionales. Aquellas características que no son discriminativas son, al final, descartadas mediante el apagado de las neuronas que las reconocen. Así, se acelera el entrenamiento y se centra este en la extracción de las características más discriminativas.

A.3 Regularización

En el campo de las redes neuronales se busca evitar el sobreentrenamiento o, en inglés, *overfitting*. El *overfitting* es un efecto que produce que la red sea capaz de predecir perfectamente las muestras que conforman el set de entrenamiento, pero no son capaces de predecir con la misma precisión las muestras de validación y test. Debemos evitar este efecto, puesto que lo que se busca es que las redes neuronales se comporten de manera correcta ante muestras que no han visto anteriormente, esto es, se busca que generalicen.

Para evitar el sobreentrenamiento de la red, se utilizan diversas técnicas de regularización. Lo que busca la regularización es modificar el algoritmo de aprendizaje para reducir el error de generalización, pero no el error de entrenamiento. A continuación se detallan algunas técnicas de regularización que se han usado en este trabajo.

A.3.1. *Dropout*

La técnica de *Dropout* es una técnica de regularización muy conocida a la par que fácil de implementar en Keras. Consiste en entrenar la red con algunas neuronas apagadas al azar. Para ello se le indica una probabilidad e internamente, se apagan las neuronas correspondientes aleatoriamente. De esta manera, conseguimos que las neuronas no memoricen los datos que por ellas pasan ya que aleatoriamente se activan y se desactivan pudiendo ver o no los datos de entrada.

A.3.2. Ruido gaussiano

La técnica del ruido gaussiano es tan sencilla como introducir ruido en los datos que entran a una capa neuronal. Para ello, se le indica la desviación típica del ruido que estará siempre centrado en cero. Con ello estamos variando los datos de entrada para que no siempre sean iguales, lo que ayuda a la red a generalizar su aprendizaje. La técnica de ruido gaussiano funciona mejor cuando las entradas son imágenes que cuando son datos numéricos, en este caso, datos de volumetría cerebral.

APÉNDICE B
Tablas de distribución de los *datasets*

En este apéndice se adjuntan las tablas de distribución, de los diferentes *datasets* obtenidos mediante diferentes técnicas de preprocesado. Se incluyen tablas de imágenes por etiqueta y set (tr, dev, te) así como tablas de distribución por edad.

Tabla B.1: Distribución de las imágenes del *dataset* original en 2D (ADNI 2D) por etiqueta y sexo.

H = hombre, M = mujer. Los porcentajes se acercan a la distribución habitual de los tres sets tr = 60 %, dev = 20 %, te = 20 %.

	LMCI		**AD**		**CN**		**EMCI**		**MCI**		**Total**	**%**
Set	M	H	M	H	M	H	M	H	M	H		
Entrenamiento (tr)	16958	14107	17478	20694	44733	46045	28753	33885	27772	46625	297050	61,63
Validación (dev)	5174	4296	5358	6336	14186	14582	9244	10944	8519	14971	93610	19,42
Test (te)	5036	4158	5086	6202	13754	14335	9085	10536	8342	14758	91292	18,94

Tabla B.2: Distribución de las imágenes del *dataset* en 2D con extracción de parénquima (ADNI 2D BET) por etiqueta y sexo.

H = hombre, M = mujer. Los porcentajes se acercan a la distribución habitual de los tres sets tr = 60 %, dev = 20 %, te = 20 %.

	LMCI		**AD**		**CN**		**EMCI**		**MCI**		**Total**	**%**
Set	M	H	M	H	M	H	M	H	M	H		
Entrenamiento (tr)	11163	9474	11498	13654	28834	30311	18971	22270	17566	30177	193918	61,66
Validación (dev)	3360	2871	3451	4177	9186	9594	6045	7098	5453	9619	60854	19,35
Test (te)	3269	2770	3326	4121	9089	9464	5956	6878	5318	9505	59696	18,98

Tabla B.3: Distribución de las imágenes del *dataset* en 2D con extracción de parénquima y normalización de forma (ADNI 2D FLIRT) por etiqueta y sexo.

H = hombre, M = mujer. Los porcentajes se acercan a la distribución habitual de los tres sets tr = 60 %, dev = 20 %, te = 20 %.

	LMCI		AD		CN		EMCI		MCI		Total	%
Set	M	H	M	H	M	H	M	H	M	H		
Entrenamiento (tr)	13652	11235	14744	17224	37365	37867	23085	26404	24224	39140	244940	61,70
Validación (dev)	4199	3407	4614	5250	11774	12055	7412	8476	7446	12444	77077	19,41
Test (te)	4038	3263	4205	5112	11528	11754	7265	8209	7267	12283	74924	18,87

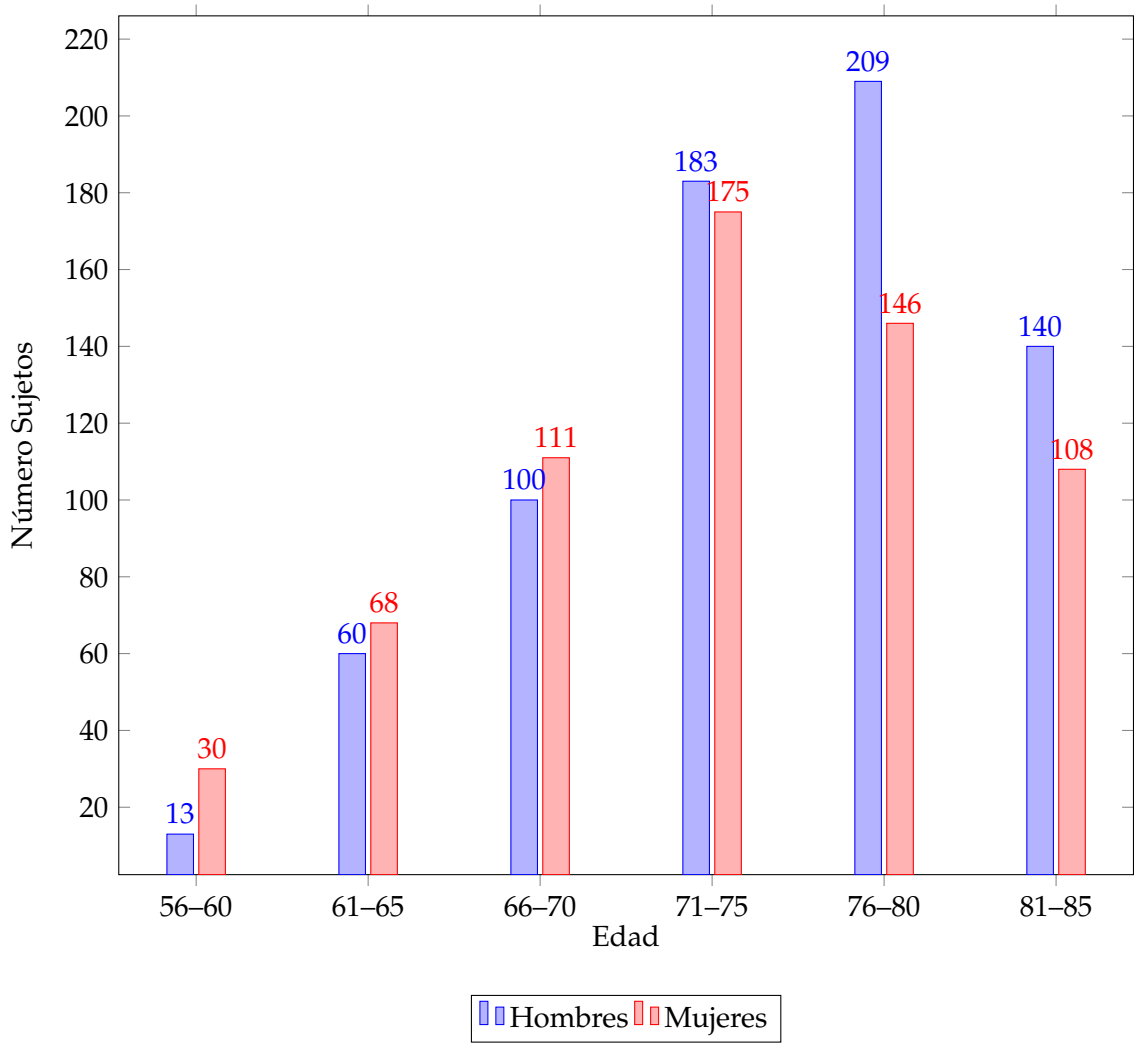


Figura B.1: Sujetos, de todas las etiquetas, por rango de edad cuyas sesiones conforman el *dataset* 2D (ADNI 2D).

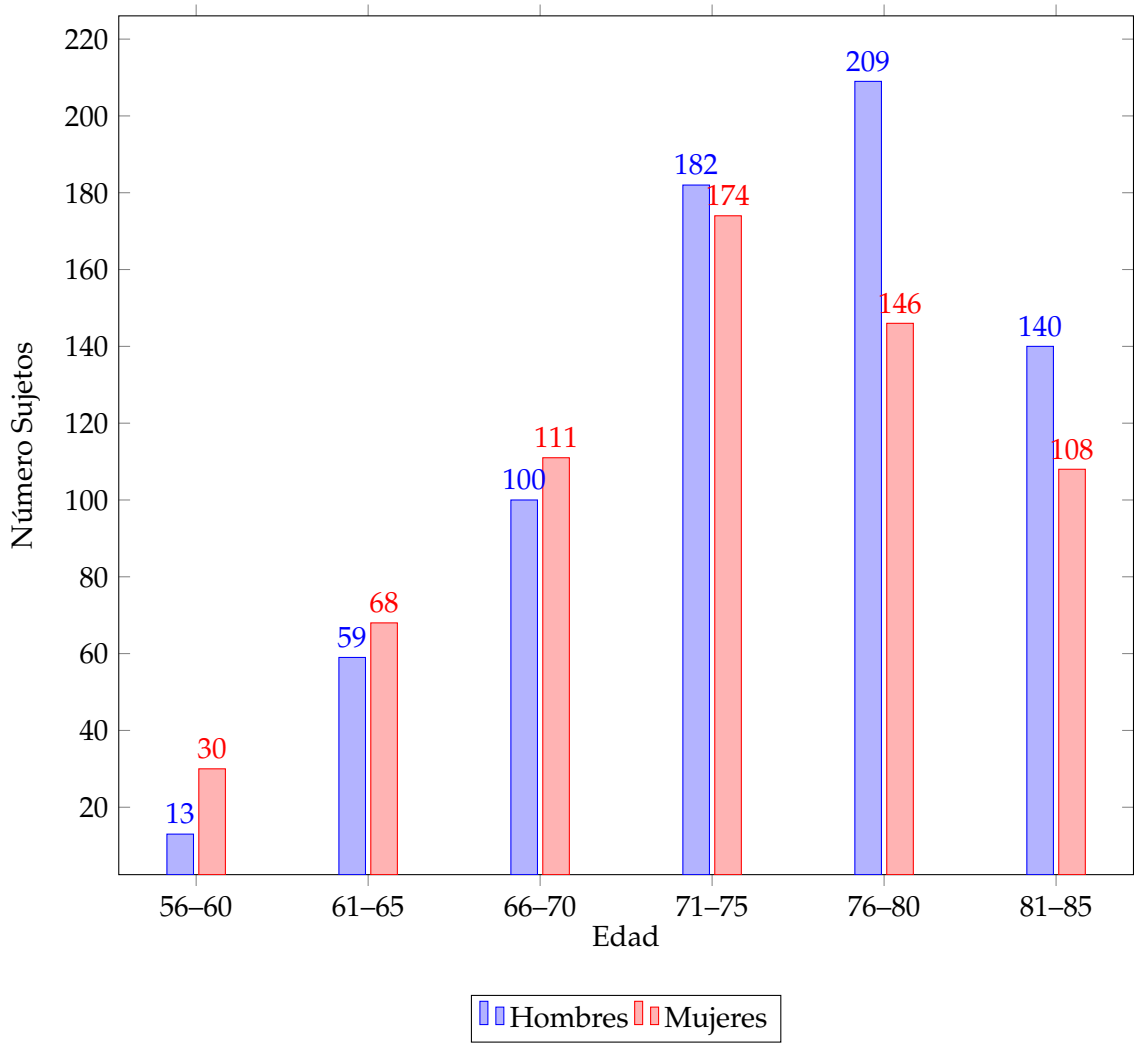


Figura B.2: Sujetos, de todas las etiquetas, por rango de edad cuyas sesiones conforman el *dataset* con extracción de parénquima (ADNI 2D BET).

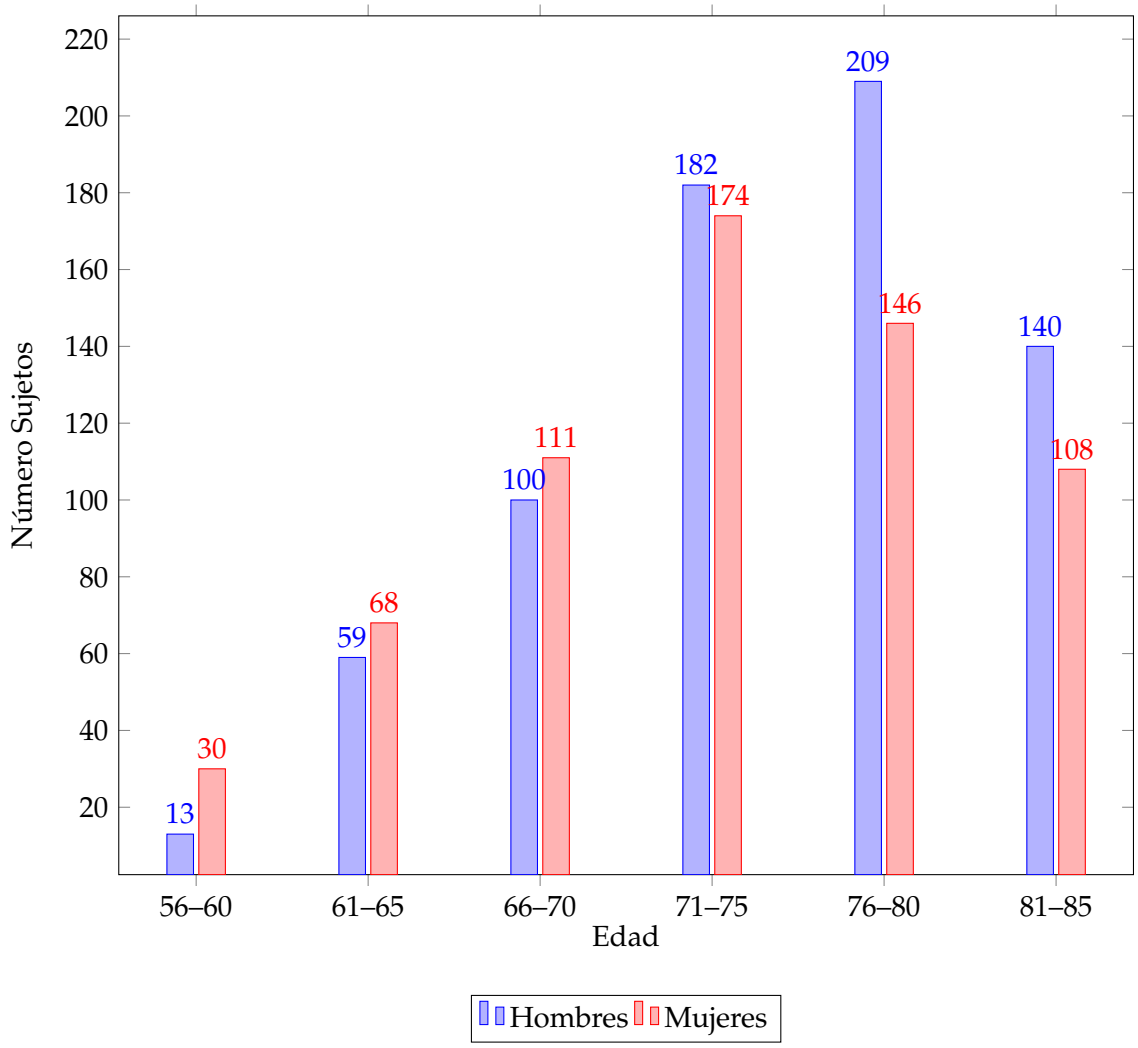


Figura B.3: Sujetos, de todas las etiquetas, por rango de edad cuyas sesiones conforman el *dataset* con extracción de parénquima (ADNI 2D FLIRT).