ULPGC
Universidad de Las Palmas de Gran Canaria

Escuela de Ingeniería Informática

Clasificación Automática de Tumores Cerebrales Haciendo Uso de Redes Neuronales

Trabajo Fin de Grado

Grado en Ingeniería Informática

Daniel Reyes García

Tutor

Javier Sánchez Pérez

Las Palmas de Gran Canaria
Abril de 2022

Agradecimientos

Se ha reservado este apartado para agradecer a todas las personas que han hecho esto posible.

En primer lugar, agradezco a todos los profesores que he tenido, tanto en la universidad como fuera de ella, por haberme formado académicamente y haberme inculcado unos principios morales y racionales. De no ser por ellos no sería la persona que soy hoy en día.

También a mi familia, mis padres, los cuales me han educado y han hecho de mí una persona noble y bondadosa. Siempre contaré la historia de como ellos, al enterarse de que iba a iniciar esta carrera, se llevaban las manos a la cabeza y comentaban que cómo iba a hacer Ingeniería Informática con mis pésimos conocimientos matemáticos. Cualquiera en mi caso podría guardar algún rencor, pero no es mi caso, ya que yo compartía esta preocupación. Tenía ese mismo miedo, sé que lo hacían por mi bien y se los agradezco mucho, pero como he demostrado, con esfuerzo y dedicación lo he logrado. A mi abuela Mercedes, la cual me apoyaba de manera incondicional, y creía fielmente que lo lograría, aunque por desgracia, yo no lo logré a tiempo. Una severa enfermedad se la llevó antes; solo espero que aun así siga orgullosa de mí.

Quería ofrecer mi más sincera gratitud a Javier Sánchez Pérez por tutorizar este proyecto y estar en todo momento pendiente y dispuesto a explicar y resolver cualquier duda.

También a mis compañeros y amigos, los cuales han sido un pilar importante en mi desarrollo personal, ya que, a lo largo de este periodo, he crecido no solo a nivel académico, sino también como persona, y eso se debe a todo lo que he vivido junto a ellos. Entre estos compañeros quería agradecer en especial a María de los Ángeles Rodríguez Felipe por haberme apoyado incondicionalmente en este último tramo de la carrera.

Por último, quería darle las gracias a una persona muy especial para mí, una persona que admiro y quiero mucho, esta persona, la cual ha sido el faro de esperanza que me ha guiado en este largo viaje y que me ha dado fuerzas para seguir adelante, y no rendirme. Aquella de la que más me alegro de haber podido compartir esta experiencia, mi mayor pilar emocional.

Mi gran amiga, Sara L. Almonacid Uribe

A la que le deseo lo mejor.

Por todo esto y mucho más.

Gracias.

Daniel Reyes García

Índice

Índice de Figuras

Índice de Tablas

Resumen

Este proyecto, denominado “Clasificación Automática de Tumores Cerebrales Haciendo Uso de Redes Neuronales”, tiene como objetivo crear, usar, analizar y comparar distintas arquitecturas de redes neuronales para lograr clasificar tumores cerebrales.

Haciendo uso de imágenes de resonancia magnética, estas redes neuronales son capaces de catalogar o discernir entre tres tipos de tumores, además de cerebros sin masas tumorales.

Este proyecto trata de solucionar, principalmente, un problema existente en los países en vía de desarrollo y es la falta de personal cualificado, y es que, para realizar esta tarea es necesario disponer de profesionales en neurología o neurocirugía. Los tumores cerebrales son complejos de identificar debido a las anomalías existentes en cuanto al tamaño y localización.

Abstract

This project, called “Clasificación Automática de Tumores Cerebrales Haciendo Uso de Redes Neuronales”, consists of creating, using, analyzing and comparing different architectures of neural networks to classify brain tumors.

Using magnetic resonance imaging, these neural networks can classify between three types of tumors, in addition to brains without tumor masses.

This project mainly tries to solve an existing problem in developing countries, which is the lack of qualified personnel, since it is necessary to have professionals in neurology or neurosurgery for carrying out this task. Brain tumors are complex to identify due to existing anomalies in terms of size and location.

Prólogo

El Trabajo Fin de Grado (TFG) presente consiste en un análisis en profundidad de los resultados obtenidos de las distintas arquitecturas de redes neuronales aplicadas en este proyecto. Este trabajo ha sido escrito como parte de los requisitos de graduación para el grado de Ingeniería Informática del programa formativo de la Universidad de las Palmas de Gran Canaria. El periodo de investigación y realización de este proyecto tiene como fecha de inicio marzo de 2021.

Hoy en día, la inteligencia artificial, el aprendizaje automático y el aprendizaje profundo se están incorporando a la vida cotidiana y a las empresas de manera asidua. Cada vez son más los productos y aplicaciones que ofrecen estas tecnologías, y es que esta se encuentra incluso en algo tan común como los dispositivos móviles. Es una realidad que estas tecnologías se han labrado un hueco en muchos sectores, entre ellos el de la salud. El uso de recursos multimedia ha permitido a estas tecnologías desempeñar una gran cantidad de tareas en el ámbito de la salud, entre las que se pueden encontrar: diagnósticos generales, diagnósticos predictivos, análisis de la conducta y segmentación de imágenes. La automatización de estas tareas ha supuesto un gran avance en el sector de la *e-Salud (cibermedicina – e-health)*, término que hace referencia a la salud cibernética y que engloba desde las anteriores tareas hasta la asistencia sanitaria online. La e-Salud es una disciplina muy reciente. Se remonta a finales de la década de los noventa y ha avanzado a paso de gigante a causa de la evolución tecnológica que se ha vivido en la última década. Además, esta disciplina médica facilita la búsqueda de información a los profesionales y permite conectar a pacientes de todo el mundo con médicos sin necesidad de estar en el mismo país, rompiendo así los impedimentos que puedan suponer las barreras geográficas. Estas herramientas tienen mucha aceptación, siendo muchas de ellas gratis o con un precio muy asequible.

Desde el punto de vista del autor, el presente trabajo constituye un valioso aporte a la e-Salud y, para ser más precisos, a los campos del diagnóstico asistido por computador y aprendizaje profundo. Este proyecto trata de aportar una solución al diagnóstico de tumores cerebrales, labor que se basó en el uso de modelos de aprendizaje profundo. Los datos recopilados en este trabajo de grado, y que podrá ver expuestos a lo largo de esta memoria, a su vez serán de utilidad en futuras investigaciones. Hay que recordar que estas herramientas siguen evolucionando hoy en día, quedando aún mucho trabajo por delante. Estas tienen como objetivo brindar apoyo y beneficios a un amplio número de sectores, incluido el de la sanidad.

Espero que disfrute de la lectura.

Daniel Reyes García

Las Palmas de Gran Canaria, abril de 2022.

Capítulo 1
Introducción

Uno de los grandes tópicos del **diagnóstico por imágenes** es la clasificación de tumores cerebrales, en donde es necesario determinar de manera precisa y rápida, qué tipo de tumor presenta el paciente, para poder empezar un tratamiento adecuado y eficaz para poder luchar esta patología. Los trabajos recientes en los que se han empleado sistemas médicos asistidos por computador, que hacen uso de técnicas de aprendizaje profundo, han proporcionado mejoras en relación con la velocidad y precisión. Y es que en los últimos años la inteligencia artificial ha ido abriéndose camino en diversas áreas, y la medicina no está exenta de ello. Esto principalmente se debe a la llegada de la visión por computador, rama del aprendizaje automático que ha tenido un impacto significante en el área del diagnóstico y análisis de imagen, demostrando ser capaces de igualar en precisión a especialistas entrenados en este propósito. Además, estos sistemas pueden realizar dichas tareas de manera masiva e instantánea, lo cual es impensable para el ser humano, que se ve limitado por factores como la fatiga, el cansancio y la velocidad de asimilación y deducción de los datos propios de cada persona.

La visión por computador ha ido creciendo y mejorando a partir de las **redes neuronales convolucionales** (*CNNs – Convolutional Neural Networks*), que han marcado un antes y un después en el aprendizaje profundo. Estas redes son capaces de encontrar los patrones más complejos que oculta una imagen a partir de un entrenamiento previo, en el que se le proporcionan imágenes de aquello que se desea detectar o clasificar. Esto las vuelve una herramienta idónea para el desempeño de estas tareas, donde existe una complejidad muy elevada a la hora de determinar el tipo de tumor que sufre el paciente. Esta complejidad es producida por las grandes anomalías presentes en cuanto a la localización, forma e incluso a la posibilidad de que distintas patologías tengan un aspecto similar, siendo muy difíciles de identificar incluso para los mayores expertos en el campo de la neurología.

Pero esta no es la única tarea clínica en la que se puede emplear la visión por computador. Hoy en día, existe una gran cantidad de herramientas médicas que emplean sistemas digitales con los que se obtienen imágenes del interior del cuerpo del paciente. Ya sea visionado en tiempo real, como las ecografías y los TACs, o a partir de impresiones, como las resonancias magnéticas y los Rayos X. Es por esto por lo que se puede afirmar que la visión por computador tiene un amplio campo que cubrir dentro de la medicina. En el caso de los tumores cerebrales se emplean imágenes de resonancia magnética debido a que es una prueba médica no perjudicial para la salud y que ofrece imágenes claras del tejido interno.

En este proyecto se han desarrollado y empleado redes neuronales convolucionales para resolver este problema. Se pretenden clasificar tres patologías distintas de **tumores cerebrales**: gliomas, meningiomas y tumores pituitarios. Por otra parte, este proyecto cuenta con una cuarta clase que representa cerebros sin masas tumorales, llamada *no tumor*, que sirve como filtro para indicar si el paciente padece alguna patología. Estas redes han sido entrenadas, validadas y testeadas con un conjunto de datos formados por imágenes de resonancia magnética (*Magnetic Resonance Imaging – MRI*). Además, para profundizar en el análisis de estas arquitecturas se aplicó distintos pre-procesamientos a las imágenes, y en el caso de los modelos pre-entrenados se trabajó con una clase especial de aprendizaje profundo,

denominado aprendizaje profundo por transferencia (*Transfer Learning*), en el que se emplean los pesos almacenados por la red tras haber sido sometida a un entrenamiento con el conjunto de imágenes [1].

A lo largo de este documento se expondrán los resultados obtenidos en este proyecto que, a nuestro parecer, son de gran valor e interés para aquellos que quieran adentrarse en este problema, o problemas de índole similar. Este documento cuenta con información relativa a la viabilidad de algunas de las arquitecturas de aprendizaje profundo más famosas, los efectos que tienen sobre estas arquitecturas, el empleo de propiedades como el aumento de datos, el aprendizaje por transferencia o distintos preprocesamientos.

Algunas de estas arquitecturas lograron resultados excepcionales, superando con creces, el umbral de aceptación fijado en un 85% de precisión. Mientras que otras no lograron alcanzar el 30% de precisión, dando a entender que estas redes son, de cualquier manera, inviables en la resolución de este problema, o eso se podría pensar a priori. Más adelante se explicará todo esto en profundidad. Por último, para evaluar dichos resultados se empleó una serie de librerías destinadas a la realización de gráficos o a la extracción de valores útiles para el análisis del comportamiento de las redes, como *F1-Score*, *ROC-Curve, UAC*, *Accuracy*, *Loss*, *Precision* entre otras.

1.1 Motivación

Los tumores cerebrales son los que menor tasa de supervivencia poseen, llegando incluso, en el peor de los casos, a ser inferior al 10% [2]. Los motivos principales se deben a la complejidad y delicadeza de la zona a la que estos cuerpos afectan, así como a diagnósticos tardíos y, en algunos casos, erróneos. Un diagnóstico tardío conlleva un avance desmesurado de las células cancerígenas, que son invasivas en el caso de los tumores cancerígenos, o un crecimiento descomunal del cuerpo que imposibilita o dificulta el tratamiento directo. Esto hace que no sea factible realizar una intervención quirúrgica para extirpar el cuerpo en cuestión, ya que se pondría en riesgo la vida del paciente, llegando a ser la quimioterapia la única opción de lucha para estas personas.

Este proyecto trata de solventar este diagnóstico tardío o erróneo, ofreciendo una herramienta de apoyo para los hospitales y especialistas, permitiéndoles realizar esta tarea de manera precisa, rápida y masiva. Por otra parte, este trabajo también solventaría la falta de personal en países en vías de desarrollo y núcleos urbanos reducidos, que no cuentan con grandes hospitales y especialistas. Permite, a su vez, reducir la cantidad de trabajo monótono, consiguiendo que el personal sanitario pueda descansar y realizar un mayor número de intervenciones. La reducción de tareas recurrentes conllevaría la disminución de bajas por estrés tomadas por los especialistas y redundaría en mayores beneficios económicos.

Cabe destacar que las labores de investigación realizadas en este proyecto, y plasmadas en esta memoria, permitirán profundizar más en las ventajas que conlleva el uso de estas tecnologías. Permitirá saber a otros investigadores cuales, de las presentes tecnologías, son viables y pueden emplearse en la solución de problemas similares, y cuales no, permitiéndoles avanzar con mayor velocidad en sus trabajos futuros.

1.2 Objetivo

El objetivo principal que se propone en este trabajo es el de realizar un clasificador de tumores cerebrales a través de técnicas de aprendizaje profundo o, lo que es lo mismo, a partir del uso de modelos de aprendizaje profundo (*Deep Learning Models*).

El objetivo que el alumno se plantea lograr es la creación de una red neuronal que logre resolver esta tarea, poniendo a prueba distintas arquitecturas y comparando el rendimiento de estas con el fin de encontrar la mejor solución.

Para esto, el alumno hará uso de distintas arquitecturas formadas por redes neuronales convolucionales. Estas son las mejores para realizar este tipo de trabajos, ya que esta tecnología simula el poder receptivo de las neuronas que forman la corteza visual del cerebro humano. Cada neurona artificial tiene la capacidad de emular el campo receptivo, dotando así de la capacidad de "ver" a estas arquitecturas computacionales.

1.3 Problema existente

El presente proyecto trata de solucionar principalmente la falta de personal cualificado, este es un problema recurrente en algunos países en vías de desarrollo y pueblos alejados de grandes ciudades que no disponen de grandes hospitales, sino que disponen de centros de salud. No obstante, esta herramienta no solo podría ser usada en estos casos, sino también podría ser empleada por expertos para acelerar y automatizar el proceso de identificación de tumores cerebrales, de manera que estas brinden apoyo y, en ningún caso, intenten sustituir el papel que tienen estos expertos.

Para realizar esta tarea por lo general es necesario disponer de profesionales del campo de la neurología o neurocirugía, ya que los tumores cerebrales son complejos de identificar dado las existentes anomalías que estos cuerpos presentan en relación con sus tamaños, ubicaciones y posibles similitudes con otras patologías que pueden presentarse en dicha zona de estudio.

1.4 Organización del documento

A lo largo de esta sección se expondrá un breve resumen del contenido plasmado en cada uno de los capítulos que componen esta memoria. Estos capítulos serán encontrados en el mismo orden que el que precisado en este apartado. El presente documento contará con:

- En el **Capítulo 2** verá reflejado un extenso estado del arte, en el que encontrará otros artículos relacionados y una breve introducción a las tecnologías empleadas en este proyecto académico.
- En el **Capítulo 3** se expondrán, de manera resumida, las tecnologías empleadas. Esta información engloba los entornos de trabajo, aplicaciones y librerías empleadas en este proyecto.
- En el **Capítulo 4** se describen la metodología y la planificación empleadas en este proyecto. En el apartado referente a la planificación se proporciona tanto la planificación inicial, aportada en el documento TFT-01, como la planificación final, la cual contiene información fidedigna de los pasos llevados a cabo y del tiempo destinado para cada uno de ellos.

- El **Capítulo 5** introducirá al lector definiciones propias de la rama de las ciencias de la salud abarcadas en este proyecto, es decir, cuestiones relacionadas con los tumores y el MRI.
- En el **Capítulo 6** se muestra la información relevante al conjunto de datos empleado y la competición de Kaggle del que se extrae estos datos.
- El **Capítulo 7** detalla la información que concierne a los distintos métodos empleados para la clasificación de tumores.
- En el **Capítulo 8** se analizarán y mostrarán los resultados obtenidos en este proyecto. Para validar los resultados, se han empleado diferentes métricas que han sido acompañadas de imágenes y tablas para facilitar su lectura y comprensión. Como sección final se ha agregado un apartado comparativo.
- En el **Capítulo 9** se detallan las conclusiones que se obtienen a partir de la información dada en los capítulos anteriores. En estas se comenta la solución final y general del proyecto a partir de las ideas del autor. Además, se ofrecen ideas de posibles modificaciones que podrían aportar beneficios a este proyecto u otros proyectos similares.
- **Competencias específicas**. Apartado obligatorio en el que el autor expresa como su proyecto ha llevado a cabo una serie de competencias o requerimientos propias de cada una de las ramas de la informática, impartidas por la Universidad de Las Palmas De Gran Canaria en el Grado de Ingeniería Informática.
- **Problemas encontrados**. Sección que abordará las dificultades halladas durante la realización de este proyecto.
- **Trabajos Complementarios**. En este se mostrarán los trabajos adicionales realizados por el autor. Por un lado, se explica la creación de una aplicación web que permite visualizar los resultados del proyecto y, por otro, se desarrolla un método para segmentar tumores cerebrales, que podría ser una continuación del proyecto actual.

Capítulo 2
Estado del arte

La tarea de clasificación de tumores cerebrales no es sencilla, y es que, como se precisa en la introducción, los tumores cerebrales presentan anomalías que dificultan su diagnóstico, tarea que es crucial en la lucha de la mayoría de las enfermedades y patologías. Determinar qué patología o enfermedad padece el paciente, y el estadio en el que esta se encuentra en el momento en el que se diagnostica, permite a los especialistas comenzar a trabajar o preparar un tratamiento para combatirlo de manera eficaz. Sabiendo esto, se debe entender que las herramientas de diagnóstico asistido por computador deberán cumplir ciertos requisitos en cuanto a precisión y fiabilidad se refiere. Ya que se pone en riesgo la vida de personas, un diagnóstico erróneo conllevaría a un casi seguro tratamiento erróneo, que en el peor de los casos podría costarle la vida al paciente. La clasificación de tumores cerebrales basada en *CAD (Computer-Aided Diagnosis),* en tumores benignos y malignos, es un tema ampliamente investigado. La clasificación de gliomas, que es un tipo de tumor maligno que se aloja en esta zona de estudio, es uno de los problemas más recurrentes de investigación en este dominio. Los sistemas CAD mencionados anteriormente se basan en imágenes de resonancia magnética (MRI) del cerebro. Esto se debe, principalmente, a la capacidad de la resonancia magnética para proporcionar un mayor contraste entre los distintos tipos de tejidos que componen el cerebro en comparación con otras pruebas similares.

Los algoritmos de aprendizaje profundo se han utilizado ampliamente en el análisis de imágenes médicas, como puede ser en la detección de cuerpos extraños en mamografías [3] [4] [5], con la finalidad de luchar el cáncer de mama y otro tipo de patologías que afectan a esta zona. Otros ejemplos pueden ser la detección de carcinomas bronquiales y cánceres pulmonares [6], la clasificación de tumores cerebrales, el diagnóstico dermatológico [7] e, incluso, problemas más recientes como el diagnóstico de COVID-19 [8], entre muchos otros. Como se observa por las fechas de estos proyectos, durante estos últimos años ha habido un incremento significativo en el interés del análisis de imágenes médicas empleando redes neuronales artificiales y algoritmos de aprendizaje profundo.

Otra técnica de interés en el análisis de tumores cerebrales en el que se emplean sistemas CAD es la segmentación de imágenes lineales [9] [10] [11] [12] [13] [14]. Esta técnica consiste en extraer de una imagen aquello que es de interés, de manera que se elimina el resto de materia contenida en la imagen y solamente queda el objeto en cuestión. En el caso de los tumores cerebrales a partir de imágenes de resonancia magnética, se extrae el cuerpo a estudiar de la zona afectada del cerebro del paciente que figura en la secuencia de resonancia magnética, es decir, la patología o cuerpo extraño que padece el paciente. De esta forma conseguimos precisar la localización, forma y tamaño de este cuerpo. Esta técnica logra a su vez reducir el ruido, al eliminar todo aquello que no es de interés para el estudio, y permite redimensionar las imágenes, acelerando así el proceso de entrenamiento y validación de las redes. Para realizar esta tarea se emplean herramientas como:

- **BET** [15] [16]**:** *Brain Extraction Tool* o BET, es utilizada para eliminar, tejido no cerebral de las imágenes, es decir tejidos que no son fundamentales para el estudio, como el cráneo y el cuello.
- **FLIRT** [17]: *FMRIB's Linear Image Registration Tool*, es una herramienta robusta y precisa, empleada en el registro lineal de imágenes cerebrales. Puede utilizarse para el registro inter e intramodal con imágenes 2D o 3D.
- **FreeSurfer** [9] [18]: Esta es una librería software destinada al procesamiento de imágenes cerebrales, desarrollada originalmente por Bruce Fischl, Anders Dale, Martin Sereno y Doug Greve. Esta herramienta es capaz de extraer segmentaciones por regiones y datos volumétricos del cerebro.
- **UNET**: Esta es una arquitectura de red neuronal convolucional, que se emplea en la segmentación de imágenes biomédicas. Un ejemplo de uso es [13]**.**

2.1 Trabajos anteriores

Existe una gran cantidad de proyectos relacionados con el diagnostico de tumores cerebrales, alguno de ellos como el de Justin S. Paul et al. [19]. Este grupo investigó la aplicación de las redes neuronales convolucionales sobre un conjunto de imágenes MRI de tipo *T1 weigthed*. Este conjunto contaba con tres clases de tumores, meningioma, glioma y tumores pituitarios. Aplicaron tres métodos de clasificación sobre el conjunto de imágenes:

- Redes neuronales convolucional (CNN).
- Redes neuronales completamente conectadas (*Fully Connected*).
- *Random Forest*.

La tasa de precisión más elevada hallada por este equipo fue de 90.26%, y fue hallada al emplear la red convolucional como método de clasificación. Esta red contaba con una arquitectura bastante simple, formada por dos capas *Conv2D*, cada una con un filtro de salida de 64 y un tamaño de núcleo 5×5. Estas capas a su vez eran seguidas por capas de submuestreo *MaxPooling2D*. Por último, poseían dos capas densas de 800 neuronas y una capa de salida con tres neuronas con activación *Softmax*.

En el artículo de P. Afshar et al. [20] se alimentó con imágenes de resonancia magnética, de dos maneras distintas, una Red de Cápsulas (Capsule Net) con arquitectura predeterminada. El primer entrenamiento contaba con las imágenes MRI sin procesar, y el segundo se realizó empleando estas mismas imágenes tras haberlas segmentado linealmente. La precisión alcanzada al catalogar fue de 86.56% para la arquitectura que empleó tumores segmentados y de 72.13% para aquella que empleó imágenes cerebrales sin procesar. La fase de aprendizaje se detuvo después de diez épocas y no quedó claro si su red pudiera haber logrado mejores resultados o si alcanzó su precisión final en la fase inicial. Es por esto por lo que idearon un segundo artículo [21] en el que la posición del tumor se introdujo en una Red de Cápsulas junto con las imágenes de resonancia magnética. Finalmente P. Afshar et al. alcanzaron una tasa de precisión del 90.89%.

Yang et al. [22] demostraron la capacidad del aprendizaje por transferencia para trabajar con conjuntos de datos más pequeños. En este proyecto hicieron uso de las redes AlexNet y GoogLeNet para clasificar gliomas a partir de imágenes de resonancia magnética. En términos de medidas de rendimiento observadas (precisión de la validación, precisión del testeo y prueba

AUC), GoogLeNet demostró ser superior a AlexNet para esta tarea, con resultados del 86.70%, 90.90%, y 93.90% frente al 86.60%, 85.50%, y 89.50%.

Pashaei et al. [23] diseñaron una arquitectura de CNN para extraer características de las imágenes por resonancia magnética en su proyecto. El modelo en cuestión contaba con cinco capas de aprendizaje y un tamaño de los filtros constate para cada una de las capas de 3x3. Con este modelo, Pashaei et al. lograrían alcanzar una ratio de precisión del 81%.

Phaye et al. [24] realizaron un estudio en el que examinaron algunas de las diferentes arquitecturas de redes neuronales capsulares, llegando a la conclusión de que CapsNet es una arquitectura apropiada para la resolución de este problema. Sus investigaciones darían como fruto una red con una precisión del 95.03%, siendo esta la mejor precisión que se ha obtenido hasta el momento.

Ismael y Abdel-Qader [25] realizaron una tarea más basada en la estadística. Extrajeron características de las imágenes de resonancia magnética y entrenaron un modelo con 270 entradas, una capa oculta con 90 neuronas y una capa de salida con tres neuronas, alcanzando una precisión del 91.90%. Por otra parte, Tahir et al. [12] indagaron en varios métodos de preprocesamiento con la finalidad de mejorar los resultados de los modelos estadísticos empleados en la clasificación. Se utilizó un clasificador SVM (*Support Vector Machines*) para comparar su modelo. La precisión más alta informada por este autor fue de 86% de precisión.

Dado que las imágenes por resonancia magnética se pueden interpretar como una secuencia de imágenes, Yufan Zhou et al. [10] optaron por emplear una red basada en la arquitectura LSTM *(Long-Short Term Memory)*. Estas redes neuronales son una extensión de las redes neuronales recurrentes, que permiten aprender hechos importantes que han pasado en instantes de tiempo anteriores haciendo uso de una memoria ampliada. Esta permite que los modelos secuenciales puedan recordar sus entradas durante un mayor periodo de tiempo. Volviendo al proyecto de Zhou et al., estos investigadores emplearon la arquitectura LSTM para resolver su problema de etiquetado del córtex. Extrajeron características de la vista axial utilizando un codificador automático y clasificaron las exploraciones. Los datos recopilados muestran que consiguieron una precisión del 92.13%.

Uno de los proyectos más completos es el de Navid Ghassemi, Afshin Shoeibi y Modjtaba Rouhani [26], ya que realizaron comparaciones con distintos optimizadores, ratios de aprendizaje, valores de *Dropout* e incluso hicieron uso de redes generativas adversativas para realizar un pre-entreno. Finalmente, este grupo alcanzó precisiones de 93.01% y 95.60%.

Como última mención tenemos el proyecto de S. Deepak y P.M. Ameer [27], los cuales lograron resultados extraordinarios de 97.10%, empleando una red basada en la arquitectura GoogleNet modificada. Esta red se entrenó previamente utilizando el conjunto de entrenamiento (después del preprocesamiento). Ajustaron heurísticamente los hiperparámetros de la red para facilitar la convergencia de la función de pérdida durante el entrenamiento.

2.2 Modelos pre-entrenados

En el campo del aprendizaje profundo, es común hacer uso de redes previamente diseñadas y entrenadas por otros investigadores. A estas redes se les conoce como redes pre-entrenadas. Estos modelos se usan con la finalidad de ahorrar tiempo o como tema de investigación. Comprobar la versatilidad de estas redes en la resolución de problemas diferentes a los que fueron ideados y diseñados inicialmente, se ha vuelto todo un tópico entre los investigadores y apasionados en este campo.

El presente apartado corresponde a los modelos pre-entrenados empleados en este TFG. Para el desarrollo de este proyecto, se ha puesto a prueba algunas de las arquitecturas preestablecidas ofrecidas por Keras. Estas redes neuronales, conocidas como *Keras Applications*, son modelos de aprendizaje profundo que han sido previamente entrenados a partir del conjunto de datos ImageNet. Los pesos obtenidos en el entrenamiento también están disponibles permitiendo de esta manera, emplear la propiedad transfer learning. En este trabajo se han hecho de los modelos contenidos en la Tabla 2.1, y se han puesto a prueba con distintas configuraciones para contrastar sus comportamientos ante las distintas pruebas.

Modelos	Tamaño (MB)	Top-1 Precisión	Top-5 Precisión	Parámetros	Profundidad	Tiempo (ms) por inferencia de paso (CPU)	Tiempo (ms) por inferencia de paso (GPU)
Xception	88	79.0%	94.5%	22.9M	81	109.4	8.1
VGG16	528	71.3%	90.1%	138.4M	16	69.5	4.2
VGG19	549	71.3%	90.0%	143.7M	19	84.8	4.4
ResNet50	98	74.9%	92.1%	25.6M	107	58.2	4.6
ResNet50V2	98	76.0%	93.0%	25.6M	103	45.6	4.4
ResNet101	171	76.4%	92.8%	44.7M	209	89.6	5.2
ResNet101V2	171	77.2%	93.8%	44.7M	205	72.7	5.4
ResNet152	232	76.6%	93.1%	60.4M	311	127.4	6.5
ResNet152V2	232	78.0%	94.2%	60.4M	307	107.5	6.6

Tabla 2.1: Keras Applications. Fuente original: Keras [28].

2.2.1 VGG

La arquitectura de red VGG fue presentada por Simonyan y Zisserman en su artículo de 2014 [29]. Esta red se caracteriza por su simplicidad, ya que utiliza solo capas convolucionales con un campo receptivo de 3 × 3 (matriz de pesos de tres pixeles de alto y tres pixeles de ancho) y capas de submuestreo de tipo *MaxPooling* (véase esta en Figura 2-1). Estas capas se encuentran apiladas una encima de otra en profundidad creciente. Este modelo tiene dos versiones: VGG-16 y VGG-19. Estos números representan las cantidades de capas de pesos (capas Convolucionales y Densas) que dispone la red.

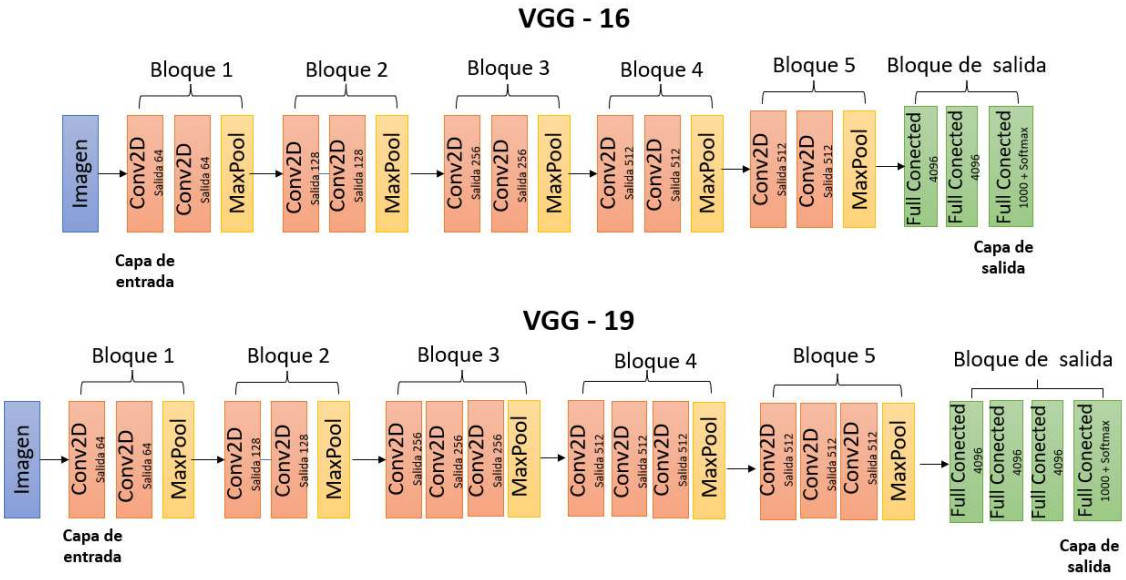

Figura 2-1: Representación de las arquitecturas VGGs.

2.2.2 ResNet

A diferencia de las arquitecturas de red secuenciales tradicionales como VGG, ResNet, Figura 2-2, es una "arquitectura exótica" que se basa en módulos de microarquitectura. Introducido por primera vez por Kaiming He. [30], esta emplea **módulos residuales** (ver Figura 2-4) de ahí su nombre ResNet o *Residual Network*.

Las redes neuronales residuales, se basan en las construcciones que forman las células piramidales de la corteza cerebral. Simulan esta parte del cerebro mediante el uso de conexiones de saltos o atajos para saltar entre capas. Existen varias versiones de la arquitectura ResNet, según la cantidad de capas de peso empleadas, siendo las más conocidas las que tienen 50, 101 y 152. Por otra parte, esta tecnología cuenta con una versión denominada coloquialmente como ResNetV2. Por lo tanto, existe un total de seis modelos procedentes de esta arquitectura en Keras. Estas son: ResNet-50, ResNet-101, ResNet-152, ResNetV2-50, ResNetV2-101 y ResNetV2-152. A lo largo de este documento se referenciará a esta agrupación por el nombre de ResNets.

Diferencias entre ResNet y ResNetV2 [31]:

- ResNet adopta una estructura **Conv - BN - ReLU** (*Convolutional – Batch Normalization – Rectified Linear Unit*) mientras que ResNetV2 adopta una estructura **BN - ReLU - Conv**. Esto se ve representado en la Figura 2-3.
- ResNetV2 hace uso de la preactivación, lo cual le da ventaja ya que puede fortalecer la regularización y aumentar la velocidad de convergencia del modelo.
- Distinto tamaño en los bloques de características.

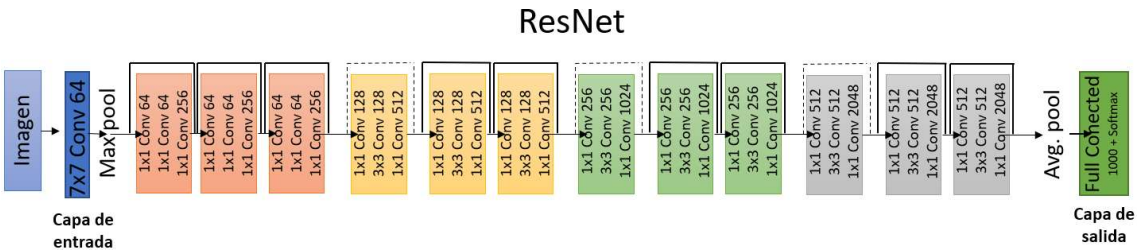

Figura 2-2: Representación de la arquitectura ResNet.

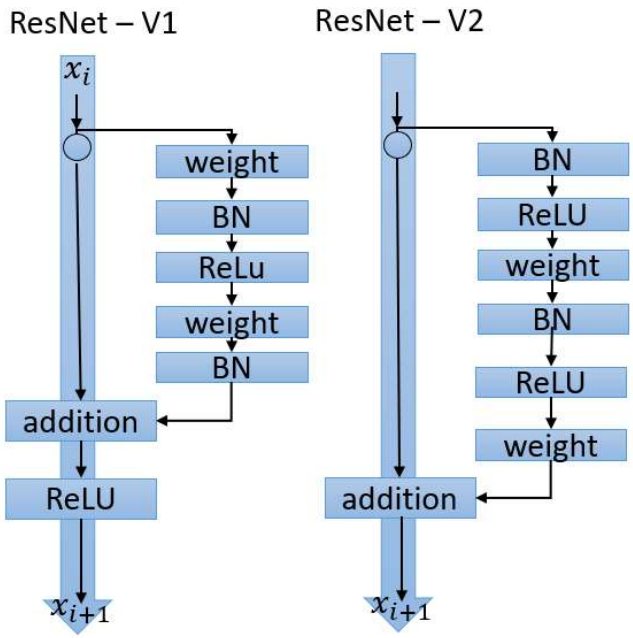

Figura 2-3: Diferencias entre ResNet-V1 y ResNet-V2.

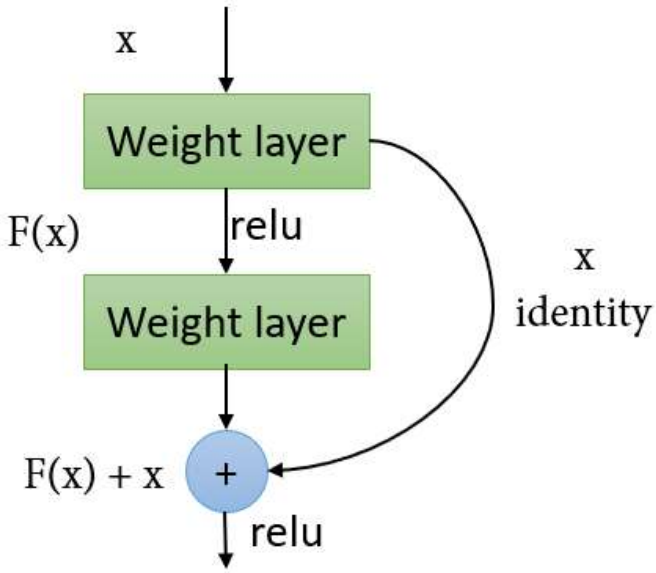

Figura 2-4: Salto residual ResNet.

2.2.3 Xception

Xception [32], o *Extreme Inception* (ver en Figura 2-5), fue propuesto por François Chollet, el creador y responsable del mantenimiento de la biblioteca Keras. Xception, es en sí una extensión de una arquitectura llamada *Inception*, de ahí la gran similitud entre los nombres, y es que François Chollet la denominó como:

> *"Una interpretación de los módulos de Inception en redes neuronales convolucionales como un paso intermedio entre la convolución regular y la operación de convolución separable en profundidad."*
>
> *François Chollet* [32]

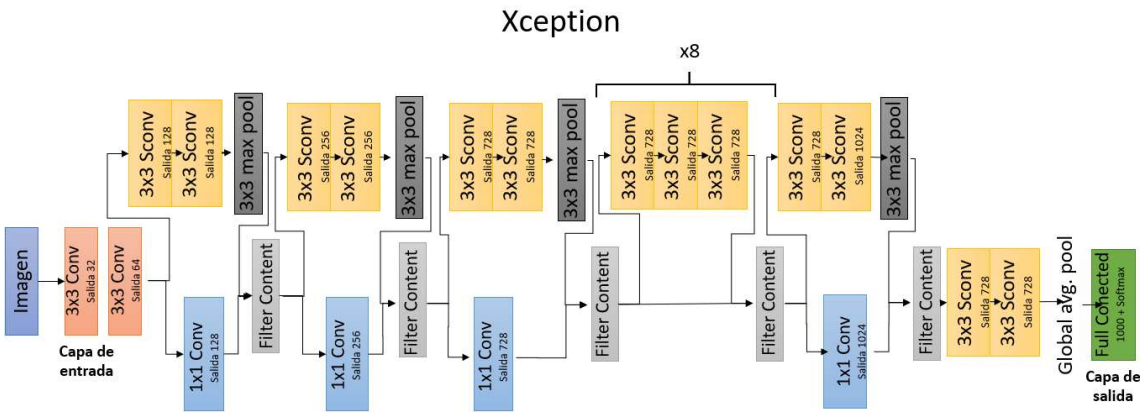

Figura 2-5: Representación del modelo Xception.

Capítulo 3
Tumores cerebrales

En este capítulo se introducirán definiciones propias de la rama de la ciencia de la salud que han sido abarcadas en este proyecto. Siendo más precisos, se hablará de información relativa a los tumores y resonancias magnéticas. En este apartado se expondrá inicialmente la definición de tumor y los tipos de tumores existentes. Tras esta, se tratarán en profundidad los tumores empleados en este proyecto, procurando así dar explicación a cuestiones como sus composiciones, orígenes, localizaciones y riesgos que conllevan padecer cada una de estas patologías. Por último, se llevará a cabo una tarea de definición que abarcará las resonancias magnéticas dando a conocer, de manera resumida, la historia detrás de esta tecnología, las partes que conforman las máquinas de resonancia magnética y el fenómeno físico por el cual estas máquinas son capaces de producir imágenes tan precisas.

3.1 Definición y tipos de tumores

Un tumor es una masa de tejido cuyas células sufren un crecimiento anormal y sin tener ninguna función fisiológica, produciendo hinchazón e incluso distención de la zona afectada; estas células tienden a propagarse y a invadir otras partes del cuerpo. Existen dos tipos de tumores; ver Figura 3-1:

3.1.1 Tumores Benignos

También conocidos como **neoplasias**, son tumores no cancerosos, por lo que no crecen de forma desproporcionada ni agresiva. Tampoco se expanden ni invaden zonas adyacentes, estas se encuentran rodeada de una capa fibrosa que no permite la propagación de las células que las componen por el cuerpo. No tiene consecuencias graves para el organismo, aunque un crecimiento desmesurado puede poner en riesgo la salud del paciente. Para denominar estos tumores se hace uso del sufijo "-oma" (tumor) acompañado por un prefijo que denomina la zona o tejido afectado.

3.1.2 Tumores Maligno

Este tipo de tumor es perjudicial para las personas, ya que puede extenderse por otras partes del cuerpo, pudiendo incluso causar la muerte del portador. Son también conocidos como cánceres o tumores cancerosos. Las células cancerosas que forman estos cuerpos son invasivas y dañinas: estas se reproducen e invaden los tejidos y órganos adyacentes a la zona inicialmente afectada. Otro de los factores que produce mayor riesgo de muerte en este tipo de tumores es la agresividad que tienen a la hora de expandirse y crecer de manera desmesurada.

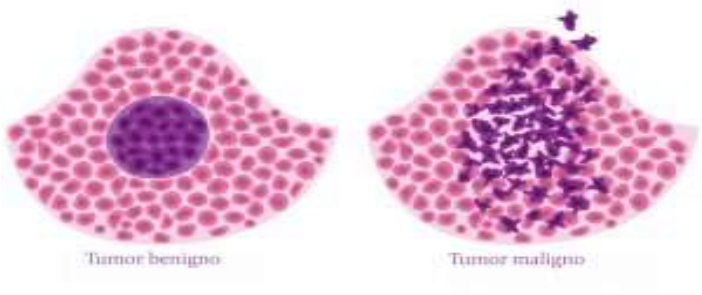

Figura 3-1: Tipos de tumores. Fuente original: Saúde [33].

3.2 Clasificación de tumores cerebrales

Los tumores de los que se dispone para la realización de este proyecto son tumores cerebrales. El conjunto de datos, o *dataset,* del que se dispuso contenía cuatro categorías o clases, en la que tres de ellas eran patologías:

3.2.1 Glioma

Se trata de un tipo de neoplasia[1] y se conocen como glioma [34] debido a que se forma a partir de **células gliales o células del soporte viscoso**. Este soporte viscoso rodea las células nerviosas, siendo un apoyo en el funcionamiento de estas. Los gliomas se desarrollan u originan en el cerebro y/o en la médula espinal. Pueden ser clasificados según tres características:

La primera por el tipo de célula por las que está formada el tumor a estudiar:

- **Ependimomas**: formado por células ependimarias (ependimarias o tancitos). Estas células forman parte del conjunto de células neurogliales del tejido nervioso, tienen como misión secretar y conservar la composición química del líquido cefalorraquídeo. Por último, hay que añadir que estas células pueden ser encontradas en los recubrimientos del conducto ependimario y los ventrículos cerebrales.
- **Astrocitomas**: es el más común entre los gliomas y están formados por astrocitos, que son las principales células gliales. Estas asumen una gran cantidad de funciones claves relacionadas con la actividad nerviosa.
- **Oligodendrogliomas**: formado por oligodendrocitos, que son un tipo de células de la familia macroglia. Estas son más pequeñas que los astrocitos y son cortas o poco prolongadas. Son las células más abundantes del cerebro junto a las neuronas.

La segunda propiedad por la que se puede clasificar estos es por el grado de evaluación patológica, pudiendo ser:

- De **bajo grado**, si son gliomas fáciles de distinguir o de fácil diferenciación, además de benignos.
- De **alto grado** si estos son difíciles de diferenciar y son malignos.

Por último, también pueden clasificarse por la ubicación en la que se encuentran estos cuerpos extraños:

- **Supratentoriales** o **infratentoriales,** dependiendo de si se encuentran encima o debajo del tentorio (tienda del cerebelo). El tentorio se encuentra ubicado en la parte posterior de la cavidad craneal. Se trata de un tabique transversal que separa la fosa cerebral de la fosa cerebelosa. Delimitan el foramen oval de Pacchioni, que es una amplia abertura a través de la cual pasa el mesencéfalo. Esta delimitación se produce entre la parte del centro y la delantera.

[1] ***Neoplasia****: tumor cuyas células se multiplican a una velocidad anormal.*

3.2.2 Meningioma

El meningioma [35] es el tumor cerebral más común. Surge de las **meninges**, que son unas membranas que sirven como recubrimiento tanto al cerebro como a la medula espinal, por lo que este tipo de tumor afecta al sistema nervioso central. Existen tres grados de meningiomas:

- De **primer grado**, conocido como grado I o tumor de bajo grado. Las células que forman el tumor tienen una lenta velocidad de crecimiento.
- De **grado dos**, conocido como grado II o meningioma atípico de grado intermedio. Tiende a reaparecer tras su extirpación a causa de la gran velocidad de propagación que tienen sus células.
- **Grado tres (grado III)** o meningioma anaplásico maligno (canceroso). Este tipo de tumor es muy peligroso para aquellos que lo poseen. Esto se debe a su gran rapidez de propagación y crecimiento.

Las causas de la aparición del meningioma son inciertas hasta ahora. Solo se sabe que se produce una alteración de las células que componen esta membrana que hacen que se multipliquen de manera veloz y sin control. Esta gran reproducción de las células es lo que provoca la formación del tumor.

3.2.3 Tumores Pituitarios

Conocidos como **tumores de la glándula pituitaria** [36] [37], tumor pituitario, tumor de la hipófisis o tumor hipofisario. Estos tumores son bastante comunes y aparecen en la glándula pituitaria, glándula conocida como "glándula de control maestro". Es la encargada de producir las hormonas relacionadas con las funciones de otras glándulas y el crecimiento. Por lo general, este tipo de tumor no produce riesgo alguno a la salud de quien lo porta, pero puede causar trastornos endocrinos, es decir, trastornos en la secreción hormonal. Estas perturbaciones vienen acompañadas de un prefijo que puede ser hiper, si la secreción es mayor de lo normal o hipo si es menor, y un sufijo dependiendo de la glándula afectada. Un ejemplo sería el hipertiroidismo, que es una alta segregación de hormonas generadas por la tiroides. A causa de los desequilibrios hormonales que estos generan, entre estos desórdenes, se encuentra el síndrome de Cushing y el hipertiroidismo.

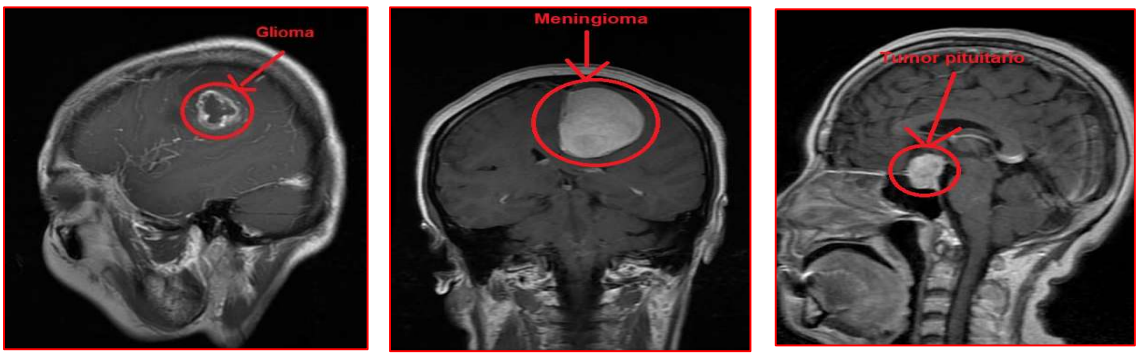

Figura 3-2: Glioma / Meningioma / Tumor Pituitario.

3.3 Imágenes de Resonancia Magnética (MRI)

Las **Imágenes de Resonancia Magnética** [38] [39] son una de las pruebas de neurocirugía y neurología más usadas. Esta provee de imágenes con exquisito detalle de cerebros, médulas espinales, ojos y oídos, entre otros. El MRI es una técnica no invasiva e indolora que utiliza el fenómeno de la resonancia magnética nuclear para obtener información detallada sobre la forma, tamaño, localización exacta, propagación (en el caso de que se pueda extender) y composición del cuerpo a analizar.

Los MRI ayudan a los médicos en la lucha contra el cáncer, permitiéndole a estos profesionales planificar la manera en la que deben abordar el problema, ya sea por cirugía, en la que se realizaría una extirpación el cuerpo no deseado, o por radiación, como la quimioterapia.

Las resonancias magnéticas no se usan exclusivamente en la zona cervical y el cerebro. Estas se pueden emplear en extremidades y zona abdominal, aunque su uso es más común en la zona superior del cuerpo.

3.3.1 Historia del MRI

Herman Carr [40] fue un físico estadounidense nacido el 28 de noviembre de 1924 y fallecido el 9 de abril de 2008. Es principalmente conocido por ser pionero en el campo de la resonancia magnética, hasta el punto de ser el creador de la primera MRI de una sola dimensión.

Herman informó en su tesis doctoral de Harvard que había producido la que en un futuro se conocería como la primera imagen de resonancia magnética. Descubrió que al aplicar un campo magnético variable era capaz de producir una imagen unidimensional, introduciendo así el uso de **gradientes de campo magnético** para la resonancia magnética.

Más tarde, en 1960, en la Unión Soviética, Vladislav Ivanov [41], físico e ingeniero de la URSS nacido en 1936 que falleció a los 71 años, presentó ante el Comité Estatal de la URSS de Invenciones y Descubrimientos en Leningrado, el primer dispositivo para realizar estas imágenes, aunque este producto no fue aprobado hasta 1970.

Como bien se explicó anteriormente, las imágenes de resonancia magnética emplean de un fenómeno llamado **Resonancia Magnética Nuclear** (RMN). Este es un método para producir imágenes detalladas de los órganos y tejidos sin la necesidad de usar rayos X o radiación "ionizante". Es por esto último por lo que se le denomina no invasiva a esta prueba. Otros métodos similares como las radiografías emplean este tipo de radiación por lo que son, en cierta medida, perjudiciales para la salud. Si alguien se expusiera durante un periodo prolongado, sería posible ver efectos adversos en la salud de esa persona provenientes de la radiación.

3.3.2 Funcionamiento del MRI

Las MRI funcionan a partir del **RMN** nombrado con anterioridad. Este fenómeno físico consiste en las propiedades que consiguen los átomos tras ser excitados por señales de radiofrecuencia dentro de un intenso campo magnético. Estas propiedades de las que se habla son: la absorción y emisión de energía. Esta energía se recoge con unas antenas que envían de manera inmediata la señal a un ordenador que se encargara de procesar la información. Cada tejido tiene distintas formas de responder ante este fenómeno, por lo que se generan distintos tipos de señales. Tras el procesamiento de la información, la computadora crea las imágenes y las muestra en el monitor de esta para poder ser tratadas por un especialista.

3.3.3 Partes de un equipo de resonancia magnética

Un equipo de resonancia magnética, como el representado en la Figura 3-3, se compone de los siguientes elementos [42]:

- **Un imán**, de un tamaño superior al de la cabeza del paciente. Esto se debe a que este último deberá introducir su cabeza dentro de él. Este imán será el encargado de generar el potente y necesario campo magnético del que se habló en el apartado anterior.
- Un **emisor** u **bobina de radiofrecuencia**, encargado de generar y emitir señales de radiofrecuencia.
- **Las bobinas de gradiente**, que regulan el campo magnético, permitiendo realizar variaciones en él.
- **Una antena**, que recoge las señales emitidas por el cuerpo del paciente al exponerse a esta prueba médica.
- **Un ordenador**, encargado de procesar las señales recibidas y de convertir estas en imágenes de resonancia magnética.
- **Una camilla**, en la que se tumbará el paciente, ya que este procedimiento se realiza tumbado.

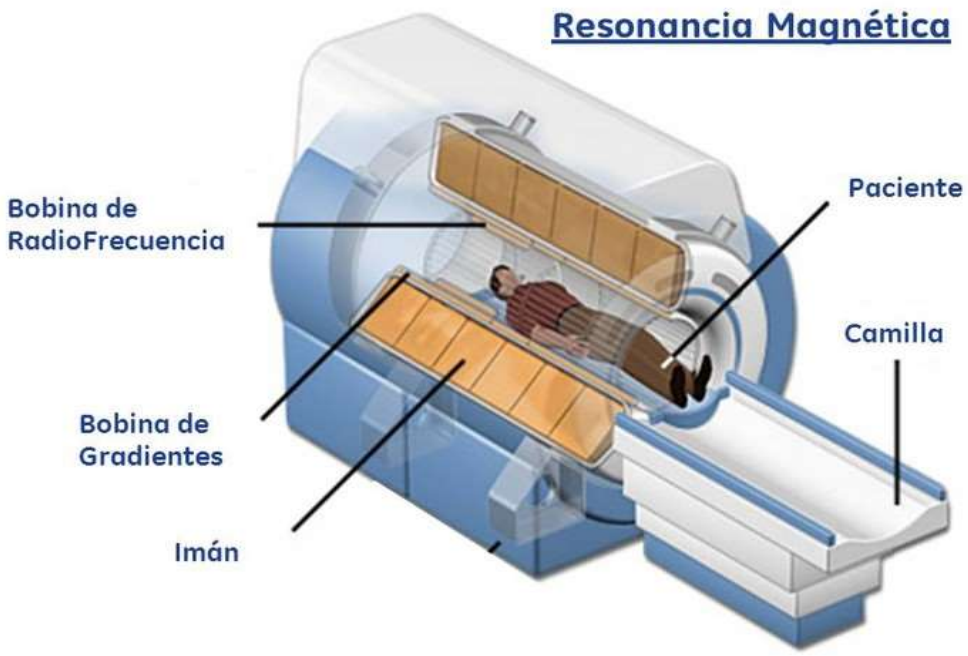

Figura 3-3: Partes de una máquina de R.M. Fuente original: Xavier Pardell [42].

Capítulo 4
Metodología y planificación del proyecto

En este capítulo se abordará la metodología empleada en este proyecto y las planificaciones elaboradas al inicio y final de este, detallando las tareas realizadas a lo largo de este periodo y los tiempos destinados a cada una de estas tareas.

4.1 Metodología

En este desarrollo de proyecto se tomó un enfoque **metodológico de ciencia de datos** (*Data Science Methodology*). Esta metodología, empleada en ciencia de datos, busca encontrar soluciones a problemas específicos.

La metodología de ciencia de datos es un marco iterativo, al igual que *SCRUM*. La principal diferencia es que este cuenta con una primera fase no iterativa, en la que se realizan tareas de análisis, recopilación y entendimiento de los datos.

En este proyecto, la primera fase del desarrollo estaría formada por la búsqueda de un conjunto de datos que se adaptase a las necesidades del problema, un aprendizaje base sobre los tumores y tipos de tumores a tratar. Por otro lado, también se incluye un estudio sobre los modelos predictivos y tecnologías que se han empleado en este proyecto, y en otros proyectos similares, definidos en el estado del arte.

Tras esta tarea de recopilación, se entraría en la zona iterativa de este marco de trabajo, el cual corresponde a la tarea práctica de este proyecto. A partir del conjunto de datos recopilado con anterioridad se realiza una tarea de modelaje, en este caso, la creación de un primer modelo predictivo. Este modelo predictivo es entrenado y evaluado, obteniendo así los resultados de la precisión del modelo asociados a la resolución de muestro problema principal. A estos resultados se les conoce como *feedback* debido a que estos servirán de guía y realimentarán el proceso iterativo, determinando los futuros pasos o comportamientos que se llevarán a cabo con la idea de mejorar los resultados. Se realizan las siguientes tareas:

A) Ajustar los hiperparámetros.

B) Añadir o reducir el número de capas del modelo.

C) Cambiar la tecnología o estrategia empleada.

D) Crear otro modelo con el que comparar los resultados.

E) Realizar un despliegue del modelo.

En el caso de este proyecto, hay que destacar dos tipos de evaluaciones, las básicas y las determinantes. Las básicas son aquellas de las que, de manera individual, se obtiene *feedback* de un entrenamiento y se decide por qué camino seguir, realizando pequeños ajustes en los hiperparámetros y capas. Las determinantes son las decisiones tomadas que afectaban totalmente al estado del proyecto. Determinar qué tecnologías emplear, qué tipos de entrenamientos realizar, refactorizaciones, etc., son cambios a gran escala y con efectos relevantes sobre el proyecto.

4.2 Planificación

En la Tabla 4.1 se puede visionar la planificación inicial. Esta planificación corresponde al apartado "Plan de trabajo" del documento TFT-01, documento que es necesario para la realización de este proyecto y que se realiza con anterioridad al comienzo de este. Es por esto por lo que la información plasmada en esta tabla es muy genérica, sin mucho detalle y no completamente fiable.

Fases	**Duración Estimada** (horas)	**Tareas** (nombre y descripción, obligatorio al menos una por fase)
Estudio previo / Análisis	60	Tarea 1.1: Recopilación de información y aprendizaje de visión por computador y las librerías necesarias.
		Tarea 1.2: Estudio básico sobre el campo que exploraré (medicina, neurocirugía).
Diseño / Desarrollo / Implementación	80	Tarea 2.1: Desarrollo de los distintos modelos de aprendizaje
Evaluación / Validación / Prueba	120	Tarea 3.1: Comprobación de los resultados otorgados por las distintas redes neuronales
		Tarea 3.1: Comprobación de los resultados otorgados por las distintas redes neuronales
Documentación / Presentación	40	Tarea 4.1: Desarrollo de un documento explicativo.
		Tarea 4.2: Práctica de la exposición
		Tarea 4.3: Exposición.

Tabla 4.1: Planificación inicial (TFT-01).

Por otra parte, la Tabla 4.2 muestra las distintas etapas por las que ha pasado el proyecto junto a su duración aproximada. En esta segunda tabla se ha obviado la presentación, ya que no es una tarea que se pueda precisar. Esta planificación contiene un mayor desarrollo, en el que se precisan las redes y propiedades empleadas, separadas en fases y ordenadas por orden cronológico.

Fases	Duración Estimada (horas)	Tareas (nombre y descripción, obligatorio al menos una por fase)
Estudio	67	Estudio previo (53 horas aprox.)
		Toma de contacto con las distintas tecnologías (4 horas aprox.)
		Estado del arte (10 horas aprox.)
Fase 1: Red propuesta	18	Modelaje y entrenamiento de una primera red neuronal con resultados superiores al 60% (2h aprox.)
		Ajuste de los hiperparámetros hasta alcanzar resultados superiores al 90% (16 horas aproximadamente).
Fase 2: VGGs	6	Modelaje de la red VGG (esta tarea nos llevó escasos minutos, ya que la implementación es sencilla)
		Entrenamientos de este modelo (aproximadamente tomó 5h 30m – 5h 50m)
Fase 3: ResNet, ResNetV2 y Xception	9	Creación de las distintas redes (estimado 1 hora)
		Entrenamientos (estimado 8 horas)
Fase 4: Transfer Learning y RGB	18	Configurar las funciones para trabajar con imágenes RGB y Transfer Learning (1 hora 30 min. aprox.)
		Reentreno de los modelos con estas configuraciones (se estima una duración de 16 horas, 30 min)
Fase 5: Data Augmentation	25	Prueba de múltiples Data Aug. (5 horas aprox.)
		Entrenamiento sobre todos los modelos (estimado 20 h)
Fase 6: Análisis y esquematización de los resultados	25	Creación de funciones para analizar los datos (25 horas aprox.)
Documentación	220	Desarrollo de la memoria y de varios blogs con conceptos teóricos (30 horas aprox.)
		Análisis exhaustivo (190 horas aprox.)
Añadidos	40	Desarrollo de una web (30 horas aproximadamente)
		Desarrollo de un segmentador lineal (10 horas aproximadamente)

Tabla 4.2: Planificación final.

4.3 Desarrollo del proyecto

A continuación, se procederá a detallar el desarrollo del proyecto y el tiempo designado para cada una de las tareas referenciadas en la planificación de la Tabla 4.2.

4.3.1 Estudio

La primera fase de este proyecto, que tomó alrededor de 67 horas, está constituida por un estudio previo, una toma de contacto con las distintas tecnologías y el estado del arte, como investigación y búsqueda de proyectos similares. Esto forma parte de aquel proceso no iterativo de la metodología de ciencia de datos que han sido explicado en el apartado 4.1 de esta documentación.

Estudio previo: Una vez escogido el conjunto de datos a emplear, se procedió a realizar un estudio sobre los conceptos médicos abordados en este trabajo (tipos de tumores a clasificar, MRI, historia del MRI, su funcionamiento etc.). Esta tarea se estima que duró en torno a 10 horas. Por otra parte, también se empleó una gran cantidad de tiempo en estudiar los conceptos relacionados con la IA o la parte más relacionada con la informática de este proyecto (funcionamiento de las redes neuronales convolucionales, aprendizaje profundo, inteligencia artificial, etc.). El tiempo aportado para esto es de unas 53 horas. En estas horas se encuentran los cursos realizados, *podcasts* escuchados, documentación leída y videos visionados por el autor de este proyecto.

Toma de contacto: siguiendo este estudio, también se realizó una corta tarea de investigación sobre el empleo de estas herramientas. Es decir, información relativa a la parte práctica de este proyecto. ¿Cómo crear, entrenar, validar y probar un modelo? ¿cómo realizar graficas o extraer métricas a partir de la información obtenida?, etc. Como el alumno ya poseía ciertos conocimientos sobre la mayoría de las tecnologías empleadas, esta tarea no necesitó de una gran cantidad de tiempo. Esto se debió a que el autor había realizado proyectos similares con anterioridad. En total se estima que esta tarea tomó cerca de 4 horas para llevarse a cabo.

Estado del arte: el estado del arte es un punto fundamental en este tipo de artículos ya que permite a nuevos competidores e investigadores comparar o conocer la viabilidad de sus métodos, arquitecturas y/o estrategias. Esto les da la oportunidad de conocer otras formas de abordar y resolver el problema al que se enfrentan, y avanzar de manera más rápida en este, ya que disponen de información fidedigna proveniente de otros autores. Además, el estado del arte permite marcar un punto de inicio y un objetivo a cumplir y/o superar, a nuevos y no tan nuevos investigadores. A esta tarea se le designó un tiempo estimado de aproximadamente 10 horas.

4.3.2 Red propuesta

Esta etapa, corresponde a la creación y ajuste de los resultados, de un primer modelo predictivo customizado o propuesto. Es decir, un modelo cuyo desarrolló se realizó desde cero apilando capas convolucionales Conv2D, y capas de submuestreo MaxPooling2D. Esta primera red procesaba imágenes monocromáticas y conseguía alrededor de un 60% de tasa de precisión. Esta red serviría finalmente como base del resto de redes propuestas, siendo los resultados de esta la que permitió llevar esta arquitectura a su punto actual. La red inicial fue una red anterior, cuyo estudio fue de gran beneficio para la realización del ajuste del resto de las redes.

Para llegar a ajustar los resultados de esta primera red, se llevó a cabo una serie de modificaciones. En esta red inicial se probaron distintos hiperparámetros, resoluciones de imágenes y optimizadores. Tras cada cambio, se realizó un entrenamiento en el que los resultados eran empleados como *feedback*, permitiendo al autor tomar decisiones con las que dirigir de forma correcta esta red. Estas modificaciones no cambian el rumbo del proyecto, por lo que son cambios básicos, en la que la toma de decisiones se hizo de manera individual. Esta tarea cíclica se repitió hasta logra alcanzar una red cuyo aprendizaje era adecuado, "limpio" de desajuste, y sus resultados superiores al 90% de precisión.

4.3.3 VGGs

Tras una evaluación exhaustiva por parte del tutor de este TFG, se determinó que se debería tener en cuenta otros modelos de aprendizaje profundo, cuyas arquitecturas estuviesen basadas en las redes convolucionales. Ya se había logrado alcanzar una precisión superior al 90% en la red customizada por lo que mejorar esta red sería ya una tarea complicada.

Se optó inicialmente por la arquitectura pre-entreneda VGG y sus variantes. Tras modelar y entrenar cada una de las redes que conforman esta arquitectura, se obtuvieron resultados catastróficos, y es que la red no era capaz de aprender, quedándose en un perpetuo estado de congelación o *stand by*. La manera de emplear estas arquitecturas es muy sencilla, solo basta con importar la librería de Keras, ajustar los parámetros de entrada y salida para que se adecúe a las imágenes entrantes, y compilarlas. Por lo que no se tenía forma alguna de ajustar más los resultados sin realizar modificaciones sobre la arquitectura.

Tras realizarse varios intentos, con la finalidad de contrastar los resultados, se optó por almacenar los resultados para así poder realizar comparativas con el resto de las arquitecturas y dejar evidencias de este hallazgo para futuros estudios. En total, se tomó alrededor de 6 horas en esta fase del proyecto.

4.3.4 ResNets

Esta arquitectura fue escogida de manera conjunta por el autor y el tutor de este proyecto, tras un informe del autor sobre los resultados obtenido de la anterior arquitectura (VGG).

La tarea de modelaje de esta red es idéntica a la de VGG, por lo que el tiempo empleado en el modelaje es despreciable. Sin embargo, el tiempo de entrenamiento es muy elevado ya que ResNet cuenta con una mayor cantidad de parámetros y versiones. Tras conseguir unos resultados algo más prometedores con ResNet, se procedió a evaluar la última tecnología que se emplearía en este proyecto.

4.3.5 Xception

Xception demostró ser una buena elección desde el primer momento. Se calcula que los entrenamientos de ResNet tomaron alrededor de 8 horas y media en ser realizados. En cambio, para Xception se emplearían unos 30 minutos debido a que no presenta versión alguna, a diferencia de VGG o ResNet que cuentan con múltiples versiones. Al igual que el resto de las arquitecturas pre-entrenadas, el tiempo de modelaje es despreciable debido a que se encuentran prácticamente listas para su uso desde que se realiza su importación.

4.3.6 Aprendizaje por transferencia e imágenes a color

En un intento de enriquecer este proyecto, el alumno tomó la decisión de emplear una tecnología hasta entonces nueva para él. Esta es la conocida como aprendizaje por transferencia o *transfer learning*. La red escogida inicialmente para someterse a este aprendizaje fue la red VGG-16. La razón tras la selección de este modelo fue el resultado poco alentador que se obtuvo en la etapa anterior. Esta red, junto a sus "hermanas", habían logrado, hasta el momento, resultados catastróficos, logrando precisiones inferiores al 30% tanto en el conjunto de entrenamiento, como en el de validación y testeo.

Para poder emplear este aprendizaje, se realizaron cambios en la función encargada del procesamiento de imágenes y en el método encargado de generar las redes provenientes de la arquitectura VGG. Este primer cambio es causado por la necesidad de contar con imágenes a color para poder hacer uso del aprendizaje por transferencia. Hasta ese preciso instante, todos los entrenamientos se habían realizado haciendo uso, a modo de información entrante, de imágenes con un solo canal de color o gama de colores gris.

Para sorpresa de todos, la red VGG-16, haciendo uso del aprendizaje por transferencia, lograba aprender de manera adecuada, llegando a obtener resultados superiores al 90% de precisión. Algo inaudito hasta ese momento. Por lo tanto, se procedió a realizar esta misma tarea con el resto de las redes pre-entrenadas propuestas en este proyecto, añadiendo así una segunda tanda de resultados. Se aprovechó la necesidad de realizar un segundo procesamiento de imágenes para modificar el número de canales de estas, de uno a tres, o lo que es lo mismo de tonos grises a formato de color RGB. Se decidió agregar una tercera tanda de modelos que emplearon, como información entrante, estas imágenes a color.

Esta fase tomó muchas horas a causa de la gran cantidad de entrenamientos que se agregaron, fue tanto el procesamiento ejercido que se superó la cuota de recursos ofrecida por Google Colab en reiteradas ocasiones. En total, se estimaron unas 18 horas, de las cuales 16 se emplearon en los entrenamientos. Las 2 horas restantes se emplearían en las modificaciones necesarias para emplear ambas propiedades y ajustar las funciones encargadas en la generación de los modelos.

4.3.7 Aumento de datos

Al igual que en el punto anterior, esta técnica de ampliación de datos, comúnmente conocida cómo *Data Augmentation*, fue agregada con el fin de enriquecer la información detalla en este proyecto. Aumentando la variedad de las imágenes se logra reducir el desajuste de las redes, ya que no se les permite "memorizar" los patrones que estas contienen. El aumento de datos somete a las imágenes a modificaciones que pueden alterar el brillo, la rotación o el enfoque, entre otros parámetros. Las siguientes características fueron las escogidas y empleadas en este proyecto:

- ***rotation_range* = 10**: rango de rotación que va de 0 a 10%.
- ***zoom_range* = [0.5,1.0]**: rango de zoom de 0.5 a 1.0.
- ***width_shift_range* = 0.15**: rango de desplazamiento vertical de 0 a 0.15.
- ***height_shift_range* = 0.15**: rango de desplazamiento horizontal de 0 a 0.15.
- ***horizontal_flip* = True**: volteo de imágenes horizontales.

Previo a esta configuración final, se realizaron pruebas con diferentes configuraciones. Al igual que con los modelos propuestos, estos no se encuentran registrados ni almacenados, debido a la imposibilidad de almacenar y analizar absolutamente todo. Esta fase supuso unas 25 horas.

4.3.8 Análisis y esquematización de los resultados

Este punto engloba el desarrollo total de la parte analítica de este proyecto integrada en el fichero tfg-analytics.ipynb. En este fichero se muestran gráficas y métricas relacionadas con:

- La cantidad de imágenes empleadas por lotes.
- La cantidad de imágenes por etiqueta o clase.
- El aprendizaje de los modelos presentados, con las siguientes métricas:
    - Accuracy.
    - Loss.
    - Precision.
    - F1-Score.
    - Recall.
- Roc Curve y UAC.
- Matrices de confusión.
- Predicciones.

Estas métricas y gráficas podrán ser apreciadas junto a una breve introducción continuando esta lectura. El tiempo designado a esta tarea es de unas 25 horas.

Capítulo 5
Tecnologías empleadas

A continuación, se abarcan las herramientas empleadas en este proyecto. Este apartado contará con las librerías, entornos de trabajo, aplicaciones e *IDL's* de los que se han hecho uso a lo largo de este programa.

La presente información aborda el modelaje y entrenamiento de las redes, los análisis desarrollados sobre estas y la aplicación web ideada en este proyecto. Para la realización de este trabajo se ha hecho uso de tecnologías muy comunes en este tipo de proyectos, y es que se podría incluso decir que estas son los pilares o las bases para la implementación de herramientas de visión por computador.

Python [43] es el lenguaje de programación escogido para realizar este proyecto. la razón principal tras esta elección no reside en la popularidad que este ha tomado en los últimos años, sino que Python es, hoy en día, el lenguaje de programación con mayor cantidad de librerías *de* código abierto para aprendizaje profundo y computación avanzada. Esto hace de este el lenguaje por excelencia para la implementación de modelos inteligentes. En este desarrollo, Python es el ápice o epicentro de todo el trabajo, ya que este lenguaje se empleó en todo el desarrollo incluyendo la aplicación web que cuenta con un lado servidor desarrollado en este lenguaje.

5.1 Librerías utilizadas

Un ejemplo de estas librerías de código abierto de la que se ha hablado en el punto anterior es **Numpy** (*Numerical Python*) [44] [45]. Numpy es posiblemente uno de los paquetes de Python más importantes para el aprendizaje automático. Esta es empleada en la computación de datos y permite crear matrices multidimensionales y vectores de gran tamaño, además de poseer grandes colecciones de funciones matemáticas de alto nivel. En la ciencia de la computación se utilizan con gran frecuencia operaciones matriciales. Estas operaciones pueden ser bastante pesadas desde el punto de vista computacional, por lo que implementarlas de manera equivocada puede provocar un uso ineficiente de la memoria. Las matrices Numpy son una clase especial de matrices que realizan estas operaciones en cuestión de milisegundos. Numpy está implementada principalmente en C y tiene un papel muy importante en el *NLP (Natural Language Processing)*, donde el volumen de datos es tan amplio que en una simple matriz se puede tener millones de números.

TensorFlow [46] [47], es otra de estas librerías de código abierto y fue desarrollada por Google Brain Team. Estos se definen así mismo como *"un equipo que se centra en realizar investigaciones fundamentales para avanzar en áreas claves de la IA y crear una mejor comprensión teórica del DL"*. Tensorflow es usada para el desarrollo o la implementación de proyectos de aprendizaje automático y es el sustituto predilecto de DistBelief, que era un sistema de aprendizaje automático basado en redes neuronales de aprendizaje profundo.

Esta se ha empleado en el proyecto para:

- Activar y hacer uso de las GPU's que ofrece Google Colab.

- Importar la interfaz Keras.

Keras [48] es una librería de código abierto escrita en Python para el desarrollo de redes neuronales. Esta se puede ejecutar en TensorFlow, Microsoft Cognitive Toolkit o Theanos. En este caso es ejecutada sobre TensorFlow y se emplea en la construcción de modelos formados por redes neuronales. Tanto la red propuesta, como las pre-entrenadas han sido implementadas haciendo uso de esta librería. En el caso de la red propuesta se ha debido importar previamente las capas que componen esta (capas Conv2D, MaxPooling2D, Dense, entre otras) Además, también se emplea para agregar a los entrenamientos y a las redes funciones de optimización, pérdidas, etc.

OpenCV [49] es una biblioteca de funciones de programación empleada principalmente en el campo de la visión por computador en tiempo real. Fue desarrollado originalmente por Intel, y más tarde fue apoyado por Willow Garage y por Itseez. Esta librería es multiplataforma y de código abierto. En este proyecto, la mayoría de los procesamientos de imágenes se han implementado a partir de esta.

Matplotlib [50] es una biblioteca de trazado para el lenguaje de programación Python. Proporciona una API orientada a objetos para incrustar gráficos en aplicaciones. Esta tecnología permitió representar gráficamente los resultados de los modelos presentados y de las imágenes contenidas en el conjunto de datos.

Scikit-learn [51] es una biblioteca gratuita de software para aprendizaje automático. Cuenta con varios algoritmos de clasificación, regresión y agrupación. En el presente trabajo se ha hecho uso de esta con el objetivo de dividir las imágenes en lotes, extraer métricas como F1-score, ROC-curve, Recall, entre otras. Por otro lado, también se empleó Scikit-image o skimage. Este último es un paquete de Python de código abierto diseñado para el preprocesamiento de imágenes, que es bastante fácil de comprender y usar. Este ofrece utilidades similares a las de OpenCV, dando la posibilidad de leer, mostrar y realizar tratamientos específicos a las imágenes.

Otras librerías empleadas en esta sección, y que merecen ser mencionadas son: **Pickle**, cuya función principal ha sido la de volcar, almacenar y cargar la información de los procesos de validación y entrenamiento; y **os,** que ofrece funciones del sistema, como, por ejemplo, tomar la ruta del directorio actual o concatenar estas.

5.2 Entornos de desarrollo

Como IDE se ha utilizado **Google Colaboratory** [52]**,** (Google Colab). Este junto a Jupyter Notebook se encuentran entre los entornos de programación más utilizados para la realización de proyectos relacionados con el mundo de la inteligencia artificial. Esto se debe a la facilidad con la que se pueden configurar ambos entornos. Finalmente, se decidió tomar Colab sobre Jupyter. Esta decisión fue tomada tras una comparación entre ambos entornos y a pesar de las limitaciones impuestas por Google, creemos que los servicios ofrecidos por Colab lo hacen una opción más atractiva que Jupyter. Colab ofrece *CPU's*, *GPU's*, *RAM's* y almacenamiento en línea. Hay que recordar que estas tareas son a nivel computacional muy estresantes para los equipos, por lo que, si no se cuenta con un equipo especializado, o con características muy elevadas, es mejor optar por esta opción en vez de sobrecargar el computador del que se dispone.

Anaconda [53] es un framework que se emplea en ciencias de la computación. Ofrece un marco en el que poder trabajar con los lenguajes Python y R. Su principal objetivo es simplificar la administración e implementación de paquetes. Anaconda es de código abierto y se

encuentra disponible en Windows, Linux y macOs. Fue desarrollada en julio de 2012 por Anaconda Inc. que anteriormente era conocida como Continuum Analytics. En este proyecto se usa como gestor de paquetes y como soporte para la aplicación web, ya que es la encargada de ofrecer soporte y servicio haciendo el papel de servidor.

Por otra parte, **Pycharm** [54] es un entorno de desarrollo integrado o IDE, principalmente pensado para programar en el lenguaje Python. Fue desarrollado por la empresa checa JetBrains, famosa principalmente por haber desarrollado herramientas colaborativas, administradores de paquetes y proyectos, y otros IDE como:

- *IntelliJ IDEA* para *Java*.
- *GoLang* para *Go*.
- *RubyMine* para *Ruby*.
- *WebStorm* para *JavaScript*.

Entre otras muchas herramientas, este se empleó como IDE para llevar a cabo la tarea adicional de la aplicación web, por la comodidad programática que esta brinda a sus usuarios. A diferencia de Google Colab o Jupyter, este programa emplea una interfaz directa, formada por un archivo de texto plano y no una codificación por celdas que vuelve más tediosa la escritura de código.

5.3 Otras tecnologías

Google Drive [55] es un servicio online de alojamiento de archivos o, lo que es lo mismo, una *Cloud*. Este servicio web es utilizado para almacenar las imágenes y guardar la información relativa a los modelos y sus calificaciones. Se escogió esta herramienta por la compatibilidad ofrecida por Google entre sus herramientas (Colab y Drive). Esta compatibilidad permite, tras un sencillo montaje, tomar desde Colab las imágenes del conjunto de datos que se hallan alojadas en este servicio. Además, permite almacenar la información recopilada directamente desde Colab.

Flask [56] es un micro marco de trabajo desarrollado por Armin Ronacher. Se utiliza para el desarrollo web en Python. En este proyecto se ha usado Flask para el desarrollo de una aplicación web clasificadora de tumores, ya que esta no formaba parte de las tareas a realizar en este trabajo. Se llevó a cabo una implementación sencilla y con funcionalidad mínima para mostrar un posible producto real.

Kaggle [57] es un sitio web que alberga una comunidad de profesionales e interesados en los campos de la ciencia de datos y la inteligencia artificial. En Kaggle los usuarios pueden compartir sus soluciones a problemas, o agregar sus propios problemas para que otros usuarios traten de resolverlos haciendo uso de estas tecnologías. Los problemas pueden agregarse de manera que los interesados en el tema toman los datos y tratan de solucionarlo sin ningún tipo de recompensa más allá del reconocimiento y la satisfacción que genera haber formado parte de ello, o, como competiciones donde se ofrecen suculentos premios a los competidores que alcancen los mejores resultados. De esta página es de donde se ha extraído la idea del proyecto y el conjunto de datos empleado.

Capítulo 6
Base de datos de tumores cerebrales

A continuación, se expondrá información relativa a los datos empleados en este proyecto. Información como el origen del conjunto de datos, la cantidad de imágenes ubicadas en este conjunto, los tipos de imágenes empleados, la forma en la que se encuentran divididas las imágenes y otras muchas cuestiones. Por otra parte, se realizará esta misma tarea con la información relativa a la competición Kaggle de la que sale este problema, dando así respuestas a ciertas cuestiones de interés y realizando una breve comparativa con otros autores/competidores.

6.1 Base de datos Brain Tumor Classification (MRI)

Las imágenes fueron extraídas de una competición de Kaggle [58]. En el apartado metada de esta competición se pueden encontrar los nombres de los autores, Sartaj Bhuvaji, Ankita Kadam, Prajakta Bhumkar, Sameer Dedge y Swati Kanchan, la fecha de creación, siendo esta el 25 de mayo de 2020 y una licencia de dominio público o renuncia de los derechos de autor, lo cual permite al que quiera hacer libre uso de estos datos.

Por otra parte, en las discusiones, Sartaj, el principal autor de este trabajo comenta que las imágenes las tomaron de otro conjunto de datos [59], cuyo autor es Navoneel Chakrabarty. En la propia competición explican que se vieron obligados a realizar limpieza de datos a causa de la existencia de imágenes con etiquetado erróneo. Tras esta limpieza consultaron con un especialista la validez del conjunto: el doctor encargado en examinar y analizar las imágenes MRI les proporciono además un certificado de veracidad.

6.2 Cantidad y tipos de imágenes

Se cuenta con un conjunto de datos inicialmente separado en dos carpetas, la primera de entrenamiento y la segunda de testeo o comprobación. Estas carpetas contienen a su vez cuatro subdirectorios con los nombres de las clases (*glioma_tumor, no_tumor, meningioma_tumor, pituitary_tumor*). Estos subdirectorios contienen las imágenes MRI de la misma variedad, aunque el autor de este proyecto decidió agrupar las imágenes de testeo y entrenamiento en un mismo directorio al que llamo “Total” con sus respectivos subdirectorios separados.

La finalidad de esta agrupación era la de poder distribuir las imágenes en lotes (entrenamiento, validación y testeo) por porcentajes de la manera que quisiera el autor. Esta decisión también fue tomada debido a que la metodología de extracción de datos usada en este proyecto dio mejores resultados que la convencional.

Inicialmente el conjunto de datos contaba con:

- 826 imágenes de entrenamiento y 100 de testeo de gliomas.
- 822 imágenes de entrenamiento y 115 de testeo de meningiomas.
- 395 imágenes de entrenamiento y 105 de testeo de cerebros sanos.
- 827 imágenes de entrenamiento y 74 de testeo de tumores de la glándula pituitaria.

Los cuales fueron reagrupados por el autor de la siguiente manera:

- 926 imágenes de tumores de tipo glioma.
- 937 imágenes de tumores de tipo meningioma.
- 500 imágenes de cerebros sin masas tumorales.
- 911 imágenes de tumores pituitarios.

A pesar de que todas son fotografías de resonancias magnéticas, se poseen distintos tipos de resonancias, basándonos en los planos (axial, sagital y coronal) y basándonos en las físicas del MRI (T1, T2 y Flair) [60].

Basado en los planos

En esta categoría se encuentran distintos tipos de imágenes, según el plano desde el que se tomó la foto:

- **Plano axial o transversal**: El plano axial divide horizontalmente el cuerpo, separándolo en dos mitades, mitad superior e inferior, se encuentra de manera paralela al suelo.
- **Plano sagital o medio**: Este plano se encuentra de manera perpendicular al suelo, divide el cuerpo en dos mitades, mitad derecha e izquierda.
- **Plano coronal o frontal**: Este, al igual que el anterior, se encuentra de manera perpendicular al suelo, pero separa la mitad frontal de la trasera o, lo que es lo mismo, la mitad anterior de la posterior.

A continuación, se ofrecerá, en la Figura 6-1, un ejemplo de cada uno de estos MRI basados en el tipo de corte, disponibles en este proyecto.

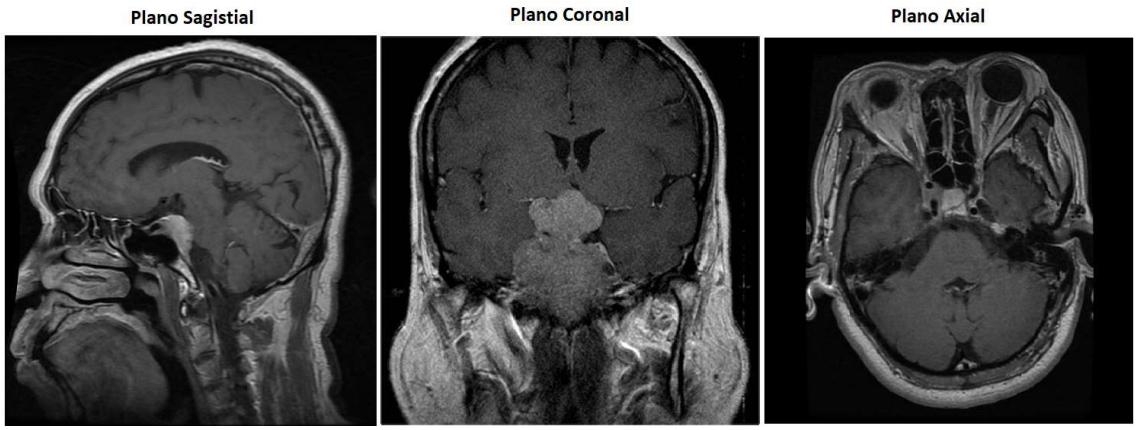

Figura 6-1: MRI basado en los planos.

Basado en las físicas

En este punto se pueden encontrar otros tres tipos de MRI, basados en las **propiedades magnéticas** de las que se documentó en el punto 4.1. Dos de los tipos empleados son exploraciones ponderadas, conocidas comúnmente como T1[2] Weighted y T2[3] Weighted (abreviadas como T1 y T2). Estas son las secuencias de resonancias magnéticas más comunes y pertenecen al grupo de *spin echo* [61] [62].

Por último, se tiene las Flair (*Fluid-Attenuated Inversion Recovery* o Recuperación de la Inversión Atenuada de Fluido), que es una secuencia de resonancia magnética perteneciente al grupo *inversion recovery*, que tienen una TE[4] y TR[5] mucho mayor que el de las T2, de tal manera que se consigue oscurecer el líquido LCR (líquido cefalorraquídeo, que es el líquido que baña el encéfalo y la medula espinal) y destacar la anomalía. Esta secuencia es muy sensible a la patología y facilita la diferenciación del cuerpo extraño frente al resto de capas y el LCR. Fue desarrollada por el Doctor Graeme Bydder y puede aplicarse tanto en resonancias bidimensionales como tridimensionales (2D Flair y 3D Flair).

Tipo de resonancia	TR (ms)	TE (ms)
T1-Weighted (TR y TE corto)	500	14
T2-Weighted (TR y TE largo)	4000	90
Flair (TR y TE muy largo)	9000	114

Tabla 6.1: Tiempos de exposición. Fuente original: David C. Preston [60].

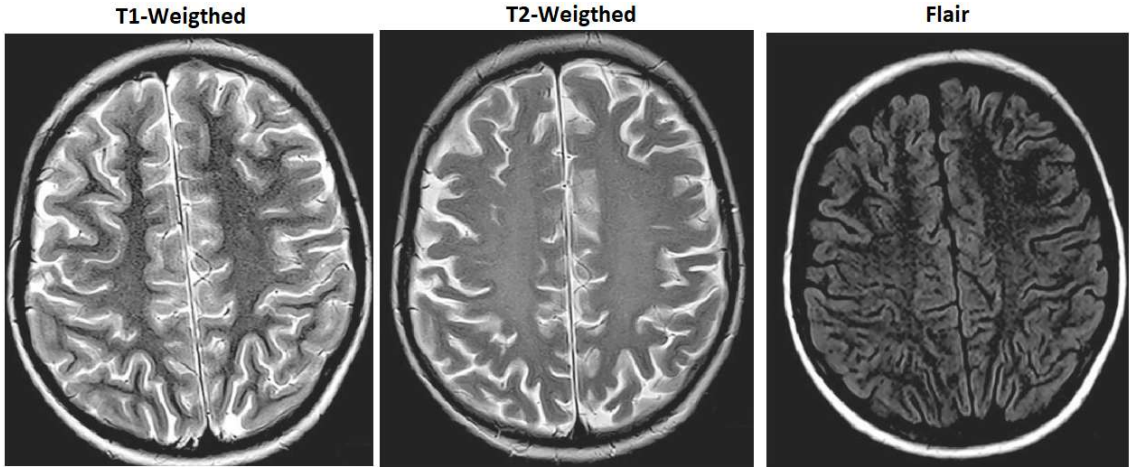

Figura 6-2: MRI basado en los principios físicos.

[2] ***T1*** [60]*: Esta se forma a partir de tiempos cortos de TE (Time echo) y TR, (Repetition Time) este tipo de resonancia muestra mayoritariamente el tejido, no obstante, datan de menos información sobre la anomalía que el resto.*

[3] ***T2*** [60]*: Estas son lo contrario a la anterior el tiempo de TE y TR deben ser largos, lo que se consigue con una mayor exposición es extraer propiedades de manera más clara, en estas además de tejido se puede ver el líquido céfalo raquídeo, de manera que se pueda discernir entre estos*

[4] ***TE*** [60]*: Es el tiempo entre la entrega del pulso de radiofrecuencia y la recepción de la señal de eco.*

[5] ***TR*** [60]*: Es la cantidad de tiempo entre secuencias de pulsos sucesivas aplicadas al mismo corte.*

6.3 Usuarios objetivos y razón de uso de este conjunto

Cualquier persona podría hacer uso de este conjunto de datos, ya que, como se explicó en el apartado 6.1, este conjunto se encuentra disponible en Kaggle y los autores renunciaron a sus derechos sobre el mismo, por lo que se puede hacer libre uso de este. No obstante, las carencias existentes en este conjunto dificultan su recomendación para uso comercial y/o profesional. Este conjunto es muy limitado en cuanto a cantidad y variabilidad de imágenes por clase. Es por esto por lo que no se debería emplear este conjunto de datos en una herramienta destinada al uso médico, ya que estas carencias podrían poner en riesgo la salud de los pacientes. Actualmente, esta base de datos es empleada con fines académicos y/o competitivos, como es este proyecto, y esta es la opción más adecuada para este conjunto.

Por otra parte, la existencia de otras patologías que afectan a la misma zona de estudio haría pensar que la cantidad de clases a diagnosticar no es del todo acertada, ya que no solo existen tres tipos de tumores cerebrales. En ese aspecto, este conjunto de datos es de los más adecuados de los vistos hasta el momento. La mayoría de las competiciones y bases de datos relacionados con el diagnóstico de tumores cerebrales son binarios, es decir, se basan únicamente en detectar si el paciente padece o no alguna patología.

El principal motivo por el que se empleó este conjunto de datos es que este ofrecía la posibilidad de discernir entre diferentes patologías. No solo detectar si se encuentra o no un tumor o cuerpo extraño en el MRI. Es decir, este conjunto permite realizar una clasificación categórica en vez de una binaria. Esto, sumado a lo que se explicó en la sección 6.1 sobre los problemas de veracidad de los datos en otras competiciones, fue por lo que se decidió tomarlo sobre el resto de los candidatos. Además, esta competición contaba con muy pocas soluciones, ya que, a lo largo de un año, tan solo contaba con una veintena de soluciones subidas. Esto es muy poco en comparación a otras competiciones que alcanzan una mayor popularidad en la misma plataforma, permitiendo así poner como objetivo personal batir los resultados del resto de competidores. Otro conjunto muy recomendado y empleado por otros autores es [63].

6.4 Estado actual de la competición

Respecto a la competición, como bien se indicó en el punto anterior, a un año de su fecha de inicio, apenas existían soluciones a este problema, y la única tarea solicitada por los autores era la del desarrollo de un modelo capaz de clasificar las imágenes MRI en las cuatro clases aportadas. Hoy en día, esto ha cambiado de manera significativa, actualmente existen más de un centenar de códigos que buscan solventar este problema. En función a las tareas establecidas, se han agregado dos más, la primera, crear un segmentador lineal, mientras que la segunda se trata de realizar una aplicación Flask o *Django*, que emplee el modelo clasificador.

6.5 Comparativa

Los competidores que en aquel entonces habían alcanzado resultados extraordinarios, realizaban una clasificación binaria, en la que empleaban la clase “no_tumor” y una clase de “tumor”, como la solución de Karelia Mohit [64]. Aquellos que se atrevían con las cuatro clases obtenían valores entre 60% y 80% en el mejor de los casos, como J. Dipit [65], que logró un valor próximo a 80%. Actualmente, la situación es muy diferente, y es que existen una gran cantidad de usuarios que han logrado alcanzar resultados superiores al 90% empleando las cuatro clases como Antony Sandesh [66], que haciendo uso de una red basada en Effnet ha logra unos resultados de 96.59%. Otro que emplea Effnet y logra resultados de 93.89% es Maruf Adewole

[67], aunque cabe destacar que su evaluación sobre el modelo de testeo está muy por debajo de lo esperado de una red de ese calibre: 75% es la puntuación extraída para ser precisos. Por último, Yasin Soylu [68], cuya mejor red, de nuevo una red Effnet, alcanzó los 96.02%.

6.6 Procesamiento de imágenes.

Las imágenes son obtenidas y separadas de tal manera que por cada foto real se obtienen dos imágenes preprocesadas de 128x128xN, donde N será tres si las imágenes son convertidas a color, o uno sí las imágenes son convertidas a escala de grises. Además, se dividieron estas imágenes en tres **lotes**, entrenamiento, validación y testeo:

- **Conjunto de entrenamiento**: es el lote usado para entrenar y ajustar los pesos de la red neuronal. Estos valores son ajustados por los algoritmos de optimización. El objetivo es minimizar las pérdidas y aumentar la precisión del modelo a la hora de aprender los patrones.
- **Conjunto de validación**: este conjunto se usa también durante el entrenamiento, pero debe estar separado, ya que se usa para afinar los hiperparámetros y comprobar cómo está aprendiendo la red. Se podría decir que estas imágenes son problemas que la red debe resolver en cada iteración tras haber realizado un aprendizaje, es decir, cada vez que termina una época de entrenamiento se inicia esta prueba en la que se comprueba su aprendizaje.
- **Conjunto de testeo**: el último conjunto que queda por ver es el de testeo. Este contiene imágenes que la red no conoce y son usadas para comprobar el aprendizaje tras el entrenamiento. Esta tiene como finalidad demostrar la manera en que trabaja la red neuronal con imágenes que no conoce. Por decirlo de alguna manera, es el resultado final que se tomará como juicio de la red neuronal.

En este proyecto se han dividido estas imágenes de tal manera que el 72% se emplean en el entrenamiento, el 20% en la validación y el 8% restante en el lote de testeo. Véase la división de las imágenes por clases y por lotes en las figuras, Figura 6-3 y Figura 6-4.

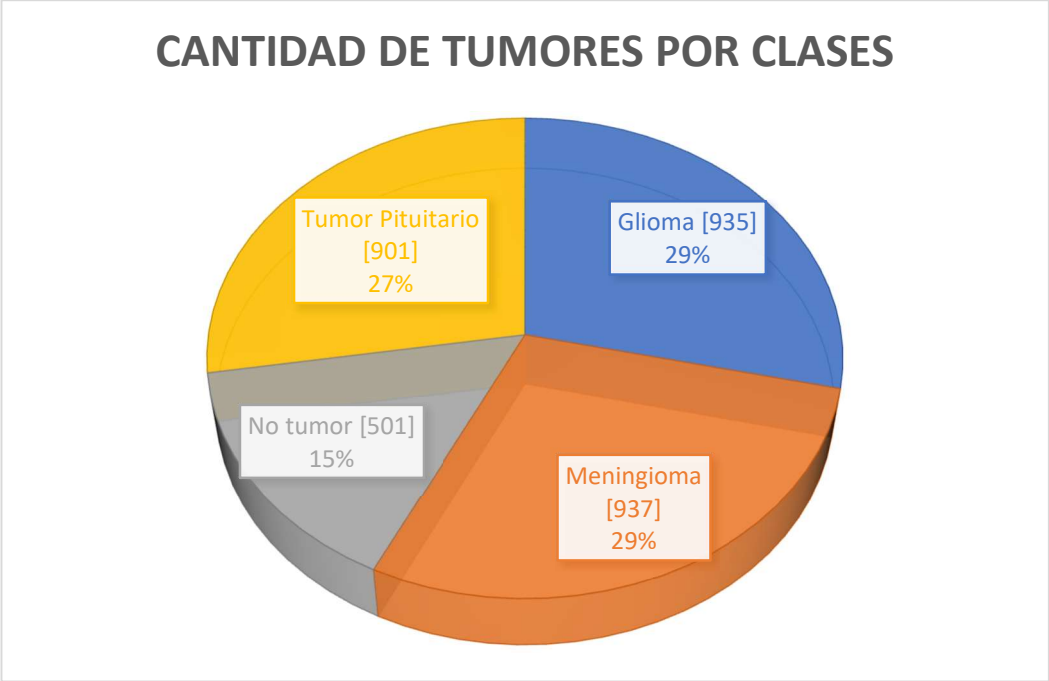

Figura 6-3: Cantidad de tumores por clases.

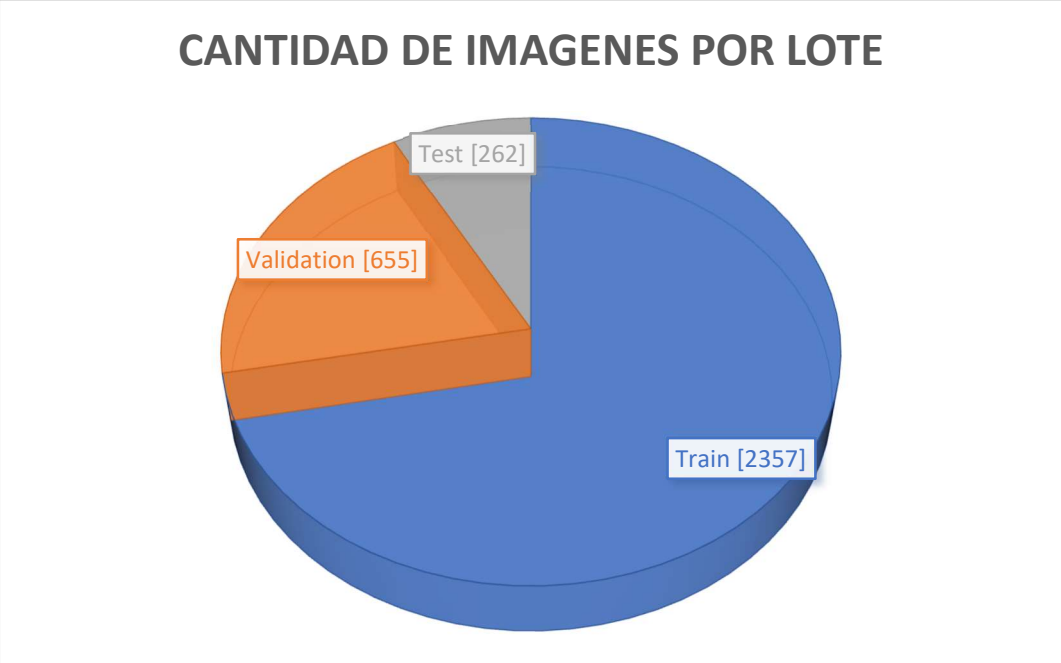

Figura 6-4: Total de imágenes por lotes.

Capítulo 7
Métodos para la clasificación de tumores

En este apartado se precisarán los métodos de clasificación de tumores usados en este proyecto. Dentro de estos métodos pueden encontrarse modelos propuestos por el autor, modelos pre-entrenados de aprendizaje profundo, una variación de aprendizaje denominada aprendizaje por transferencia que ha sido empleada sobre los modelos propuestos, distintos preprocesamientos y configuraciones, y el uso del aumento de datos sobre los conjuntos de entrenamiento y validación. Para llevar a cabo los procesos de entrenamiento y validación, ha de usarse una función de compilación que es ofrecida por Keras. Los parámetros empleados en la compilación, a partir del punto en el que se logró alcanzar resultados excelentes por parte de la red propuesta, ha sido constante. Este cuenta con un optimizador Adam y un algoritmo de cálculo de pérdidas (*Categorical Cossentropy*).

Adam

El optimizador Adam [69] [70] [71], o *Adaptative Moment Estimation*, es una combinación de las heurísticas de los optimizadores *RMSProp* y *Momentum*, por lo que se puede ver como una versión extendida del descenso de gradiente estocástico. Adam se presentó por primera vez en una famosa conferencia para investigadores de aprendizaje profundo llamada ICLR 2015. Este optimizador se ha vuelto muy popular entre los investigadores por su fácil configuración. Los hiperparámetros empleados en este optimizador son muy intuitivos y fáciles de comprender:

- **LR (Learning Rate)**: Hace referencia a la tasa de aprendizaje.
- $\boldsymbol{\beta_1}$ **(Beta_1)**: Tasa de caída exponencial para las estimaciones del primer momento.
- $\boldsymbol{\beta_2}$ **(Beta_2)**: Tasa de caída exponencial para las estimaciones del segundo momento.
- $\boldsymbol{\varepsilon}$ **(épsilon)**: Una pequeña constante para la estabilidad numérica.
- **AMSGrad**: Variable que indica si se debe aplicar la variante AMSGrad de este algoritmo.

Categorical Cossentropy

La Entropía Cruzada Categórica [72] [73] , o *Categorical Crossentropy*, es una función de pérdida. Permite comprender, a nivel humano, lo lejos que se encuentra la red de clasificar correctamente, durante los procesos de entrenamiento y validación, en un instante concreto. Este error es a su vez empleado por la red para mejorar los pesos de la matriz.

Esta función de pérdida se utiliza en tareas de clasificación multiclase y es en sí una de las más empleadas. Esta función, además, cuenta con una función parecida que es utilizada en la resolución de problemas binarios. Esta es conocida como Entropía Cruzada Binaria o *Binary Crossentropy*. Estas dos funciones se expresan, matemáticamente hablando, de la siguiente forma [73]:

- **Categorical Cross Entropy Loss:**

$$L\,(\theta) = -\sum_{i=1}^{k} y_i \log(\hat{}y_i)$$

- **Binary Cross Entropy Loss:**

$$L\,(\theta) = -\frac{1}{k}\sum_{i=1}^{k} y_i \log(\hat{}y_i) + (1 - y_i)\,.\log\,(1 - \hat{}y_i)$$

7.1 Modelo propuesto

La red propuesta o customizada, cuenta con una configuración secuencial, es decir, un apilamiento de capas. Esta red está formada por una capa inicial de tipo convolucional Conv2D, que hace la función de capa de entrada, conectando sus neuronas directamente con los *pixeles* de la foto. En Keras, para crear una capa convolucional, se emplea la clase Conv2D. Esta capa crea un núcleo de convolución que produce un tensor de salida a partir de la convolución de la capa. Este núcleo se puede ver como una matriz bidimensional, que se desplaza por el ancho y largo de la imagen realizando una operación de producto escalar entre los pixeles de la foto y los pesos contenidos en la matriz.

La primera capa convolucional cuenta con un *output-filter*[6] de 64, un núcleo de 2x2, un *Stride*[7] por defecto de 1x1, *Padding*[8] tipo *Same*, que aplica la misma cantidad de pixeles al marco en todas las direcciones, y activación ReLU[9]. Excepto por el número de filtros de salida, el resto de los parámetros será constante para cada una de las capas convolucionales empleadas. Las capas convolucionales que forman esta red están ordenadas de manera que la primera cuenta con 64 filtros, la segunda con 128, la tercera 256, la cuarta 128 y la quinta 512. Además, a cada capa convolucional le sigue una capa de submuestreo MaxPooling2D con un núcleo de 2x2 y una función reguladora Dropout de 0.2, o lo que es lo mismo, una ratio de apagado o caída de unidades de entrada del 20%. Finalmente, esta red está conformada por una capa de "aplanamiento" o Flatten, cuya función es transformar la información matricial en información vectorial, una capa densa de 1024 filtros de salida que actúa como capa conectiva, una función reguladora Dropout con una ratio de 0.5 y una capa de salida de cuatro neuronas con activación *Softmax* empleada para extraer la respuesta del modelo. Esta configuración es visible en la Tabla 7.1 y en las figuras, Figura 7-1 y Figura 7-2 .

[6] ***Output-Filter (Filtro de salida):*** *Tamaño de la figura de salida, determinará el número de neuronas que deberá enlazar siguiente capa.*

[7] ***Stride****: esta tupla referencia la cantidad de pixeles que se desplaza la ventana que recopila los resultados.*

[8] ***Padding:*** *Esta propiedad agrega un marco alrededor de la foto con la que ajusta el contenido y proporciona más patrones.*

[9] ***ReLU:*** *La función de activación lineal rectificada o Rectified Linear Unit (ReLU para abreviar) es una función lineal por partes que generará la entrada directamente si es positiva; de lo contrario, generará cero. ReLU se ha convertido en la función de activación predeterminada para muchos tipos de redes neuronales porque facilita el aprendizaje durante el logrando mejorar el rendimiento de los modelos que lo emplean.*

```
def create_custom_models(input_shape):
  model = Sequential()

  model.add(Conv2D(filters = 64, kernel_size = (2,2),
          padding = 'Same', activation ='relu',
          input_shape = input_shape))
  model.add(MaxPool2D(pool_size=(2,2)))
  model.add(Dropout(0.2))

  model.add(Conv2D(filters = 128, kernel_size = (2,2),
          padding = 'Same', activation ='relu'))
  model.add(MaxPool2D(pool_size=(2,2)))
  model.add(Dropout(0.2))

  model.add(Conv2D(filters = 256, kernel_size = (2,2),
          padding = 'Same', activation ='relu'))
  model.add(MaxPool2D(pool_size=(2,2)))
  model.add(Dropout(0.2))

  model.add(Conv2D(filters = 128, kernel_size = (2,2),
          padding = 'Same', activation ='relu'))
  model.add(MaxPool2D(pool_size=(2,2)))
  model.add(Dropout(0.2))

  model.add(Conv2D(filters = 512, kernel_size = (2,2),
          padding = 'Same', activation ='relu'))
  model.add(MaxPool2D(pool_size=(2,2)))
  model.add(Dropout(0.2))

  model.add(Flatten())

  model.add(Dense(1024, activation = "relu"))
  model.add(Dropout(0.5))

  model.add(Dense(4, activation = "softmax"))

  model.compile(
      optimizer = Adam(lr=0.001, beta_1=0.9, beta_2=0.999),
      loss = "categorical_crossentropy", metrics=["accuracy"])
  return model
```

Tabla 7.1: Función generadora de modelos propuestos.

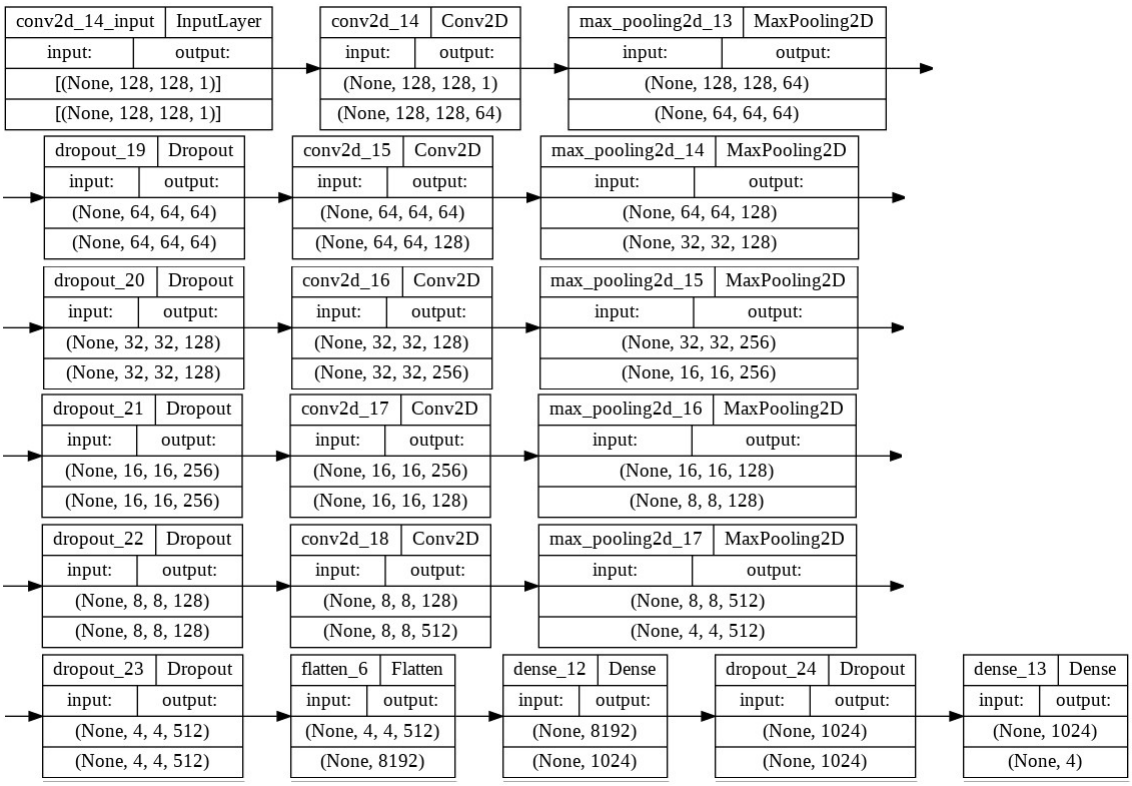

Figura 7-1: Diagrama Red Custom GS.

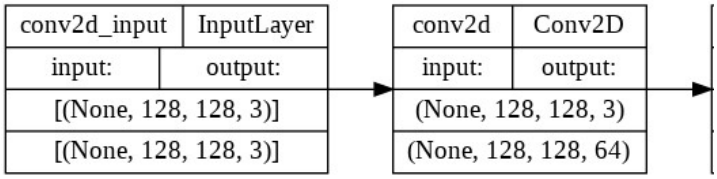

Figura 7-2: Entrada Custom RGB.

7.2 Modelos pre-entrenados

Los modelos pre-entrenados cuentan con dos configuraciones, dependiendo de si emplean o no el aprendizaje por transferencia. La primera es una configuración básica, generada al realizar la importación de los modelos; ver Tabla 7.2. A este modelo recién importado solamente se le extrae los pesos, y se le ajusta tanto la entrada como la salida para adaptarla a las necesidades del conjunto de datos.

```
Model = VGG16(include_top=True, weights=None, input_shape=input_shape, classes=4)
```

Tabla 7.2: Función VGG16 sin aprendizaje por transferencia.

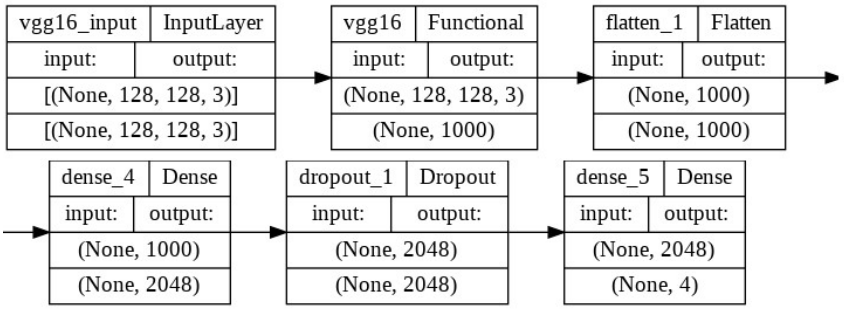

Figura 7-3: Diagrama VGG-16 RGB.

La segunda emplea el aprendizaje por transferencia; ver Tabla 7.3. Será necesario hacer uso de los pesos obtenidos por el modelo tras su entrenamiento original con ImageNet. Para esto se introduce el parámetro *weights* como ImageNet. Al mismo tiempo será requerido extraer la cabecera de la red, ya que la entrada de esta cuenta con una configuración adaptada al conjunto ImageNet. Para esto se debe deshabilitar el parámetro *include_top*.

Una vez realizada ambas tareas, se extraerán las capas entrenables, para reducir a cero la cantidad de parámetros entrenables y se deberá apilar las capas para adecuar la salida de la red, ya que esta contará con una capa de salida de mil clases. En este caso se le incorporó una capa aplanadora, una capa conectiva de 2048 filtros de salida, una función reguladora Dropout con una ratio de 30% y una capa de salida de cuatro neuronas con activación *Softmax* empleada para extraer la respuesta del modelo.

```
vgg_conv = VGG19(weights= "imagenet", include_top=False, input_shape=input_shape)
vgg_conv.trainable = False
model = Sequential()
model.add(vgg_conv)
model.add(Flatten())
model.add(Dense(2048, activation='relu'))
model.add(Dropout(0.3))
model.add(Dense(4, activation='softmax'))
```

Tabla 7.3: Código necesario para generar una red VGG-16 TFL.

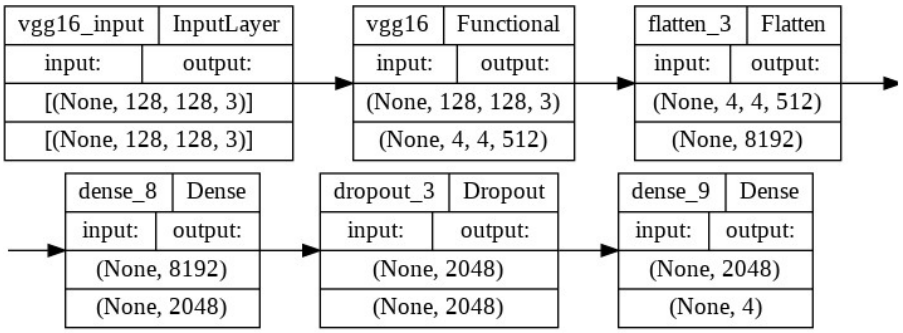

Figura 7-4: Diagrama VGG-16 TFL.

Las diferencias entre estas dos configuraciones se ofrecen en la Figura 7-3 y en la Tabla 7.3.

Estas diferencias yacen a partir de la capa funcional, hacia adelante o hacia abajo, si a profundidad se refiere. Esta capa fue adaptada al retirar las capas entrenables y propaga sus efectos en las capas flaten_3 y dense_8.

7.3 Modificación de la entrada

A cada uno de los métodos además hay que sumarle el uso de dos configuraciones distintas de entrada. En la primera se emplean imágenes con gama de colores grises, que fue la configuración o el procesamiento de datos inicial. Más tarde, al introducir el aprendizaje profundo, se requirió un segundo procesamiento sobre las imágenes, transformándolas a RGB. Aprovechando que ya se contaba con este segundo procesamiento, se añadieron dos series de entrenamientos, el primero haciendo uso de imágenes a color y aprendizaje por transferencia, mientras que el segundo únicamente tenía la configuración de imágenes a color.

Para facilitar la creación de los modelos, se generaron funciones para cada una de las arquitecturas a diseñar. Estas tienen gran similitud con el código plasmado en la Tabla 7.4. Usando como parámetros la versión del modelo, si es que contaba con más de uno, las dimensiones de la entrada (alto, ancho y número de canales que disponen las imágenes) y una variable booleana que determina si se debía o no emplear el aprendizaje por transferencia.

```
def create_resnet_model(resnet_version, input_shape, transfer_learning=False):
 if transfer_learning and input_shape[-1] == 3:
  if resnet_version == 50:
   resnet_model = tf.keras.applications.ResNet50(
      weights="imagenet", include_top=False, input_shape=input_shape)
  elif resnet_version == 101:
   resnet_model = tf.keras.applications.ResNet101(
      weights="imagenet", include_top=False, input_shape=input_shape)
  else:
   resnet_model = tf.keras.applications.ResNet152(
      weights="imagenet", include_top=False, input_shape=input_shape)

  resnet_model.trainable = False
  model = Sequential()

  model.add(resnet_model)

  # Y le añadimos las capas que queramos
  model.add(Flatten())
  model.add(Dense(2048, activation='relu'))
  model.add(Dropout(0.3))
  model.add(Dense(4, activation='softmax'))
 else:
  if resnet_version == 50:
   model = tf.keras.applications.ResNet50(
      weights=None, input_shape=input_shape, classes=4)
  elif resnet_version == 101:
   model = tf.keras.applications.ResNet101(
      weights=None, input_shape=input_shape, classes=4)
  else:
   model = tf.keras.applications.ResNet152(
      weights=None, input_shape=input_shape, classes=4)

 model.compile(optimizer = Adam(lr=0.001, beta_1=0.9, beta_2=0.999),
           loss = "categorical_crossentropy", metrics=["accuracy"])
 return model
```

Tabla 7.4: Ejemplo de función generadora de modelos.

7.4 Aumento de datos

Por último, se agregaron modelos que emplearían el aumento de datos. Este fue comentado en el punto 3.3.7 de esta memoria, pero de forma resumida esta función de procesamiento contaría con la siguiente configuración:

- **rotation_range** = 10: rango de rotación que va de 0 a 10%.
- **zoom_range** = [0.5,1.0]: rango de zoom de 0.5 a 1.0.
- **width_shift_range** = 0.15: rango de desplazamiento vertical de 0 a 0.15.
- **height_shift_range** = 0.15: rango de desplazamiento horizontal de 0 a 0.15.
- **horizontal_flip** = True: volteo de imágenes horizontales.

Los efectos de estas propiedades pueden verse reflejado en la Figura 7-5. Poniendo un ejemplo de su funcionamiento, la primera ilustración situada en la esquina superior izquierda muestra un desplazamiento lateral izquierdo, mientras que la de la esquina contraria presenta ampliación (zoom) y una leve rotación.

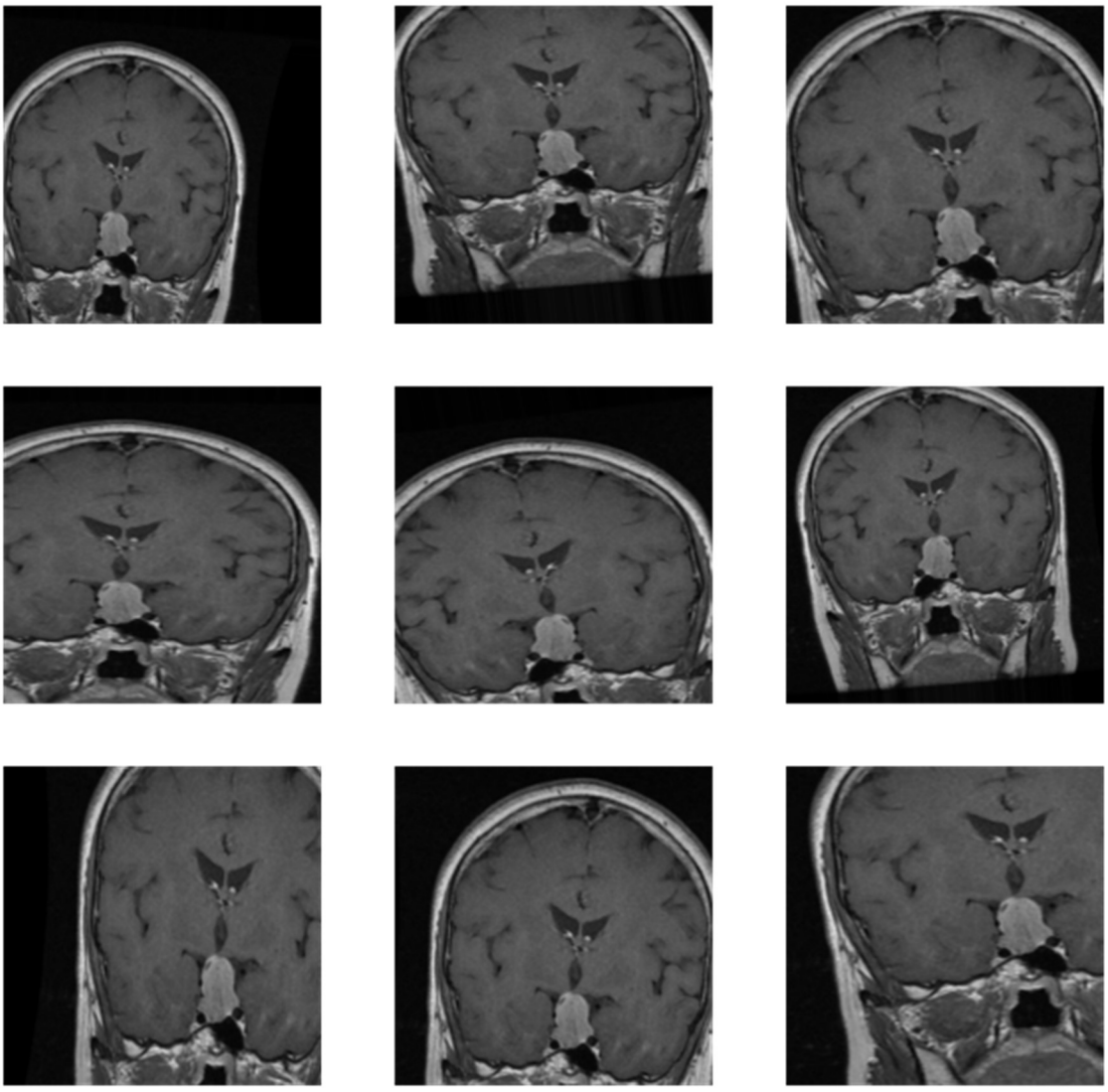

Figura 7-5: aumento de datos sobre tumor pituitario.

Capítulo 8
Resultados experimentales

En este capítulo se expondrá, de manera general, los resultados obtenidos por los cincuenta y ocho modelos distintos realizados en este proyecto. Además, se proporcionarán distintas métricas y gráficas referente a, en opinión del autor, tras una exhaustiva tarea analítica, los cinco mejores modelos.

La información plasmada en este punto se encuentra distribuida entre los ficheros de entrenamientos y análisis denominados tfg-train.ipynb y tfg-analytics.ipynb. Ambos ficheros podrán ser encontrados en el siguiente repositorio[10] de GitHub, junto a un enlace al blog personal del autor, y la aplicación web desarrollada por el mismo. El contenido de este capítulo abarca los entrenamientos y validaciones obtenidos en el fichero de entrenamiento y los puntos *Confusion Matrix*, *Loss and Accuracy Graphics*, *Results (f1_weigthed, test_accuracy and test_loss)* y *Another Results (ROC, f1 and recall)* del fichero de análisis.

8.1 Estudio de la precisión de los métodos

Este estudio de la precisión[11] comprende los cincuenta y ocho procesos de aprendizaje realizados en este proyecto. Para facilitar su lectura se han dividido los datos en dos tablas de veintinueve: en la primera se ofrecerán los resultados de los modelos que no emplearon el aumento de datos, mientras que en la segunda se ofrecerán los que lo emplearon. Los valores comprendidos en las columnas de *Training* y *Validation* han sido extraídos de los campos *accuracy* y *val_accuracy*, de la última época de cada una de las redes tras haberlas sometido a los procesos de entrenamiento y validación. En referencia a los resultados bajo el identificador "Test", estos son adquiridos de manera inmediata tras evaluar con la función *evaluate* de Keras, cada uno de los modelos con las imágenes del conjunto de testeo.

Los nombres de los modelos que se encuentran a continuación están formados por una concatenación de arquitectura y procesamiento empleado. Por ejemplo, el nombre de la red customizada que emplea imágenes a color es Custom RGB. Siendo RGB, el procesamiento en el que las imágenes se transforman a la gama de colores que soportan los formatos digitales convencionales.

Ambas tablas se encuentran ordenadas de manera ascendente según los resultados del conjunto de testeo, es decir, los modelos que encabezan esta tabla son aquellos que lograron obtener un menor porcentaje de acierto a la hora de hacer uso del lote de prueba. A medida que se baja por esta tabla los resultados se irán incrementando, llegando finalmente a los modelos con mejores resultados de cada una de las agrupaciones. O lo que es lo mismo, los modelos más prometedores de este trabajo.

[10] ***Repositorio**: https://github.com/danielreyes9756/TFG*

[11] ***precisión**: precisión (en inglés, accuracy) se refiere a lo cerca que se encuentra la predicción de una red del resultado real o de una medición del valor verdadero. En términos estadísticos, la precisión está relacionada con el sesgo de una estimación. La precisión de una red vendría a ser, de forma general, la cercanía de la red a resolver perfectamente el problema.*

Para evaluar el rendimiento de las arquitecturas se hará uso de las gráficas de precisión o *accuracy*. En ellas se muestra representada en una escala continua, la evolución de las precisiones de los entrenos y validaciones, época tras época. Estas gráficas presentan dos líneas: La primera de color azul, que representa la precisión del entrenamiento y se le denomina accuracy; la segunda, de color naranja, representa el proceso de validación o el val_accuracy. La gráfica cuenta con una leyenda que muestra la asociación anteriormente nombrada, y ejes de ordenada para facilitar la lectura. Los ejes de estos esquemas muestran:

- **Eje X:** las épocas en múltiplos de diez, iniciando desde el cero.
- **Eje Y:** los valores obtenidos, que se encuentran divididos según su crecimiento y en formato decimal.

Sin aumento de datos			
Modelos	**Training**	**Validation**	**Test**
VGG-16 GS	0.2762	0.2672	0.2481
VGG-19 GS	0.2898	0.2672	0.2481
VGG-19 RGB	0.3061	0.2672	0.2481
VGG-16 RGB	0.2749	0.2687	0.2939
ResNet101V2 GS	0.9912	0.6901	0.6756
ResNet152 GS	0.9574	0.7420	0.7328
ResNet101V2 RGB	0.9864	0.7053	0.7328
ResNet50 GS	0.9934	0.7450	0.7710
ResNet50 TFL	0.8108	0.7908	0.7786
ResNet152 TFL	0.8563	0.8076	0.8015
ResNet152V2 GS	0.9600	0.8672	0.8130
ResNet50V2 GS	0.9828	0.8443	0.8282
ResNet101 TFL	0.8551	0.8412	0.8359
ResNet50 RGB	0.9893	0.8260	0.8397
ResNet152 RGB	0.9804	0.8412	0.8473
ResNet50V2 RGB	0.9892	0.8489	0.8588
ResNet152V2 RGB	0.9782	0.8641	0.8588
ResNet101 RGB	0.9904	0.8153	0.8664
ResNet101 GS	0.9881	0.8794	0.8702
ResNet50V2 TFL	0.9688	0.8824	0.8779
ResNet152V2 TFL	0.9827	0.8855	0.8779
ResNet101V2 TFL	0.9922	0.9008	0.8931
VGG-16 TFL	0.9759	0.9145	0.9046
Xception GS	0.9949	0.9252	0.9048
Custom RGB	0.9827	0.9298	0.9122
Custom GS	0.9753	0.9130	0.9237
VGG-19 TFL	0.9540	0.9130	0.9237
Xception RGB	0.9988	0.9435	0.9274
Xception TFL	**0.9982**	**0.9573**	**0.9656**

Tabla 8.1: Precisión de los distintos métodos de clasificación (sin DA).

VGGs

Iniciando desde la parte superior se encuentra una agrupación formada por los modelos de la arquitectura VGG que no emplearon el aprendizaje por transferencia. Estos modelos cuentan con un rendimiento inferior al 30%, valor que queda muy distante del umbral de aceptación. Hay que recordar que es la precisión el factor que permite destacar la validez o la capacidad cognitiva de los modelos inteligentes, ya que como se vio en la introducción de este documento "estas herramientas deberán cumplir ciertos requisitos en cuanto a precisión y fiabilidad se refiere".

Los procesos de entrenamiento y validación de cada uno de estos modelos muestran un comportamiento poco deseado. Se puede observar, durante el proceso de entrenamiento de estos modelos, una alta inestabilidad. Esta se encuentra representada como variaciones indiscriminadas, las cuales, a priori, no son alarmantes. No obstante, se producen con mucha frecuencia, esto junto a los resultados obtenidos es muestra de un mal proceso de aprendizaje. Época tras época los resultados de estas redes rondan entre los 0.26-0.28 de precisión mínimas, y los 0.30-0.32 de precisión máxima. Estas franjas de valores dependen de la red estudiada.

Por otra parte, la validación queda en un estado de congelamiento o *stand by*, en el que los resultados extraídos son iguales, o con muy baja variación, teniendo así una tendencia constante, que se refleja como una línea recta en el paso del tiempo. Es posible ver ambos comportamientos en la Figura 8-1.

A pesar de los datos contrastados con anterioridad, dentro de esta arquitectura existen dos modelos que destacan por sus resultados, que discierne totalmente de lo previamente visto. Estos son los modelos que emplearon el aprendizaje por transferencia de datos. Estas se encuentran contenidas entre las diez mejores redes de este conjunto, colocándose en las posiciones tres y siete comenzado desde la parte inferior de la tabla. Se debe recordar que estos resultados se encuentran ordenados de manera descendente.

VGG-16 TFL alcanza un grado de acierto o precisión del 90.46%, mientras que VGG-19 TFL logra establecer una tasa de precisión del 92.37%. A priori se puede observar un correcto comportamiento en las gráficas contenidas en la Figura 8-2. Cabe destacar que tanto el accuracy como el val_accuracy varían de manera que los resultados crecen y decrecen entre épocas. No obstante, estos cambios vuelven a no ser alarmantes ya que se encuentran dentro de un baremo que podría denominarse como "normal".

El presente hallazgo ha permitido al autor y al tutor de este proyecto plantear una hipótesis y es la siguiente: este acontecimiento se debería principalmente a la alta variación de parámetros de entrenamiento existente entre las redes que emplean el aprendizaje por transferencia y las que no. VGG-16 cuenta con, aproximadamente, 139 millones de parámetros entrenables y VGG-19 con unos 143 millones. Mientras tanto, las versiones que emplean el aprendizaje profundo por transferencia de datos cuentan con un total de 0 parámetros entrenables. Esto lleva a pensar que estas arquitecturas están desarrolladas para resolver problemas de gran volumen de tamaño y que este problema en concreto "se les queda corto". Más adelante se podrá ver si esta hipótesis se cumple estrictamente o si, por lo contrario, es un planteamiento equívoco.

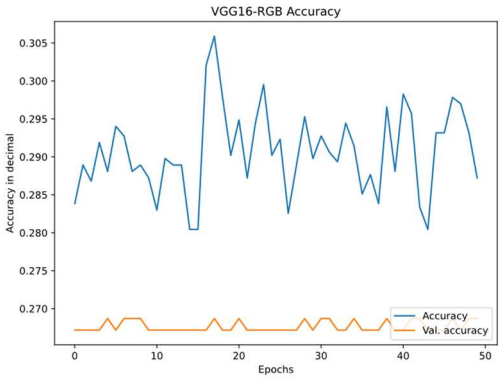

Figura 8-1: Entrenamiento y validación modelo VGG-16.

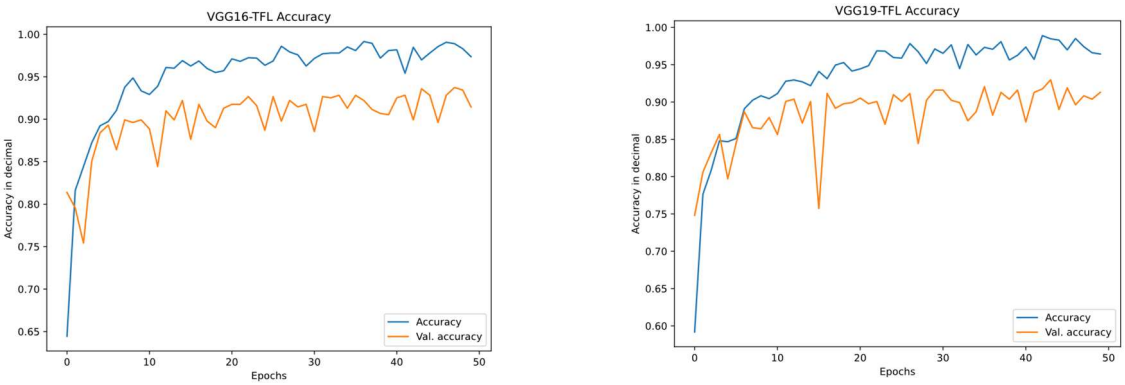

Figura 8-2: Entrenamientos y validaciones, VGG-19 TFL y VGG-16 TFL.

ResNets

Las siguientes arquitecturas que conforman esta tabla son las arquitecturas ResNets. Este conjunto posee un amplio abanico de resultados, tomando precisiones entre 69.01% y 89.31%. A pesar de esto, estas redes comparten ciertas similitudes en cuanto a comportamiento se refiere, y es que, a pesar de la diversidad de resultados, estas redes realizan un gran desempeño a la hora de entrenar y el factor por el que se ven limitadas es la validación. Estas redes efectúan a su vez un comportamiento algo diversificado, pero del que se pueden extraer algunas conductas comunes. Como la alta inestabilidad por parte de ambos conjuntos, reflejado con caídas muy prologadas, estas son más fáciles de apreciar en aquellas que desempeñaron un peor papel, y superposiciones entre los resultados en las redes que emplearon el aprendizaje por transferencia. Véase la actuación de esta arquitectura para los casos más extremos. (el peor y el mejor caso).

En el peor caso, Figura 8-3 , la precisión es muy estable, rondado a partir de la época diez, valores entre 97% y 99%. Por otra parte, la precisión del conjunto de validación es más errática, pudiendo recordar incluso a una cordillera por su forma. Se precisa dentro de esta figura, descensos que van, inicialmente, desde valores superiores al 80% de precisión, hasta valores próximos al 60%. Al comparar los resultados de la Tabla 8.1 se puede observar que el val_accuracy se encuentra muy próximo al resultado de la evaluación del conjunto de testeo, siendo la precisión de la validación 69.01% y el la del 67.56%. Algo similar puede verse en la Figura 8-4, el mejor caso, no obstante, en esta el ruido propinado por el val_accuracy es menos contundente a pesar de tener una mayor frecuencia de aparición. Los resultados varían entre el

86.00% y el 90.00% pero de una manera más errática o azarosa. Al igual que en el caso anterior el val_accuracy es muy próximo al resultado del testeo, 90.07% y 89.31%. En referencia al accuracy, aunque menos estable, vuelve a rondar en todo momento precisiones del 96.00% y 99.00%, y con una buena convergencia al 100% época tras época.

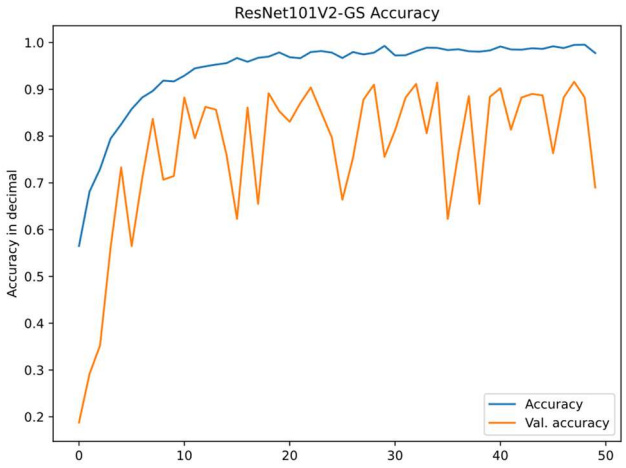

Figura 8-3: Entrenamiento y validación ResNet101V2-GS.

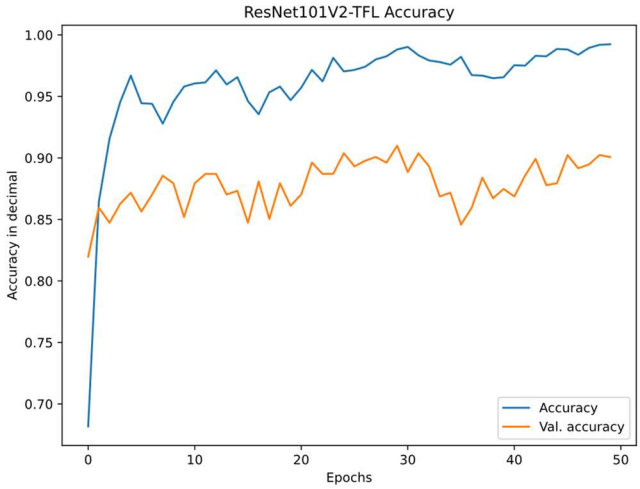

Figura 8-4: Entrenamiento y validación ResNet101V2-TFL.

Xception

Xception es una de las arquitecturas prodigio de este proyecto, esta logra unos resultados abrumadores. Llegando a obtener como resultado mínimo, dentro de esta agrupación, un 90.48%. Estas redes toman la sexta, segunda y primera posición de este grupo, con los siguientes resultados, 90.48%, 92.74% y 96.56%. Época tras época, estos modelos muestras un alto grado de rendimiento, y es que, estos alcanzan a lograr una taza de acierto superior al 90% antes del ciclo veinte, en el caso del val_accuracy, y antes del diez, en cuanto a accuracy se refiere.

Un punto negativo por destacar en estos modelos, y podría decirse que el único, son las prolongadas pérdidas de precisión que se producen a lo largo de estos procesos. En general no sucede con tanta frecuencia, pero como se puede visionar en las imágenes, Figura 8-5, Figura 8-6 y Figura 8-7, estas pérdidas son muy preocupantes, debido a la alta diferencia entre el punto más elevado y el más hundido. Se puede observar, próximo al periodo veinte de la validación en la Figura 8-7, un pico que tiende de un valor próximo al 90% de tasa de precisión, hasta un valor

que roza el 20% de precisión sobre esta misma métrica. No obstante, tras esta caída se estabiliza nuevamente.

La principal hipótesis sobre el porqué de este acontecimiento se basaría en un problema con el tamaño de los lotes. Estos podrían ser muy pequeños, por lo que la red estaría aprendiéndose los patrones intrínsecos de las imágenes que forman este lote a gran velocidad. Al producirse un cambio de lote en la que las imágenes fueran totalmente diferentes a las de la agrupación anterior, la red intentaría aplicar los conocimientos adquiridos previamente sin buenos resultados. Siendo incapaz de reconocerlas de manera instantánea y necesitando de un periodo de adaptación.

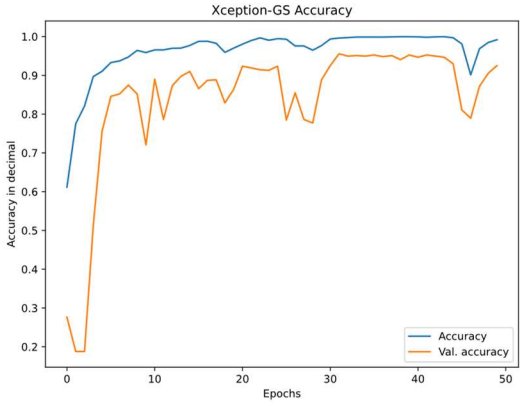

Figura 8-5: Entrenamiento y validación Xception-GS.

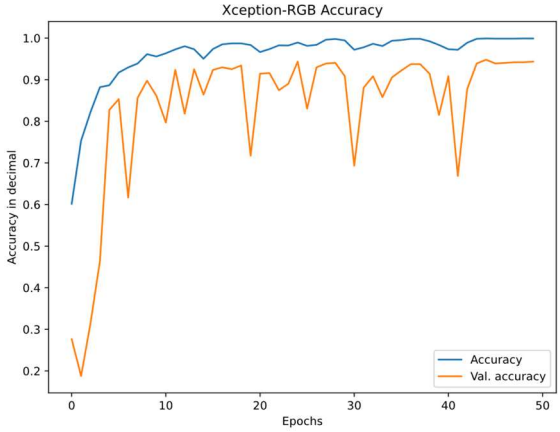

Figura 8-6: Entrenamiento y validación Xception-RGB.

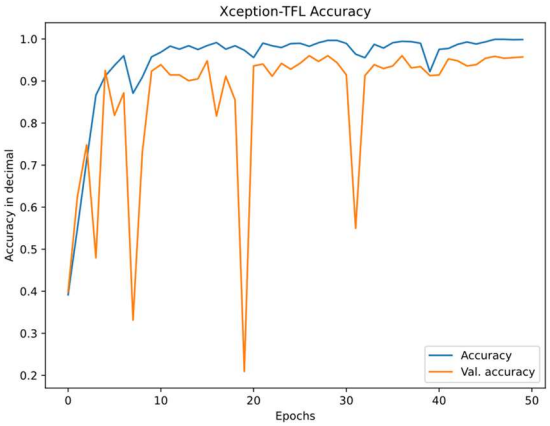

Figura 8-7: Entrenamiento y validación Xception-TFL.

Modelos customizados

Los modelos customizados, al igual que los modelos presentados con anterioridad, logran resultados extraordinarios. Las redes de este último conjunto reciben los puestos cuarto y quinto.

Siendo el cuarto Custom RGB con una precisión final (precisión del testeo) del 91.22%, y el quinto Custom GS con un 92.37%. Probando así que, no solo las arquitecturas preconstruidas son eficaces a la hora de solucionar problemas de este calibre.

Estas redes además reflejan una gran labor investigativa, ya que, a pesar de no conseguir los mejores resultados, logra ser la arquitectura más estable hasta ahora. Estas redes muestran una alta cognición, ya que preservan de manera adecuada resultados alcanzados y ambos procesos, visibles en la Figura 8-8, forman una convergencia con una tendencia correcta, equilibrada y a la par, indicando así que ambos procesos se ejercen adecuadamente. La convergencia en la precisión inicia desde algo más de un 40%, mientras que la de la validación se encuentra poco por encima del 60%, estas toman resultados por encima del 90% entre la primera y segunda decena de iteraciones.

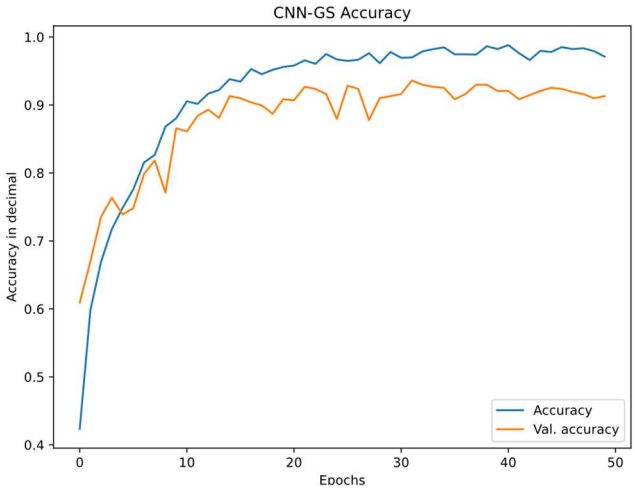

Figura 8-8: Entrenamiento y validación Custom GS.

Con aumento de datos			
Modelos	**Training**	**Validation**	**Test**
VGG-16 GS	0.2745	0.2672	0.2481
VGG-19 GS	0.2972	0.2672	0.2481
VGG-16 RGB	0.2970	0.2672	0.2481
VGG-19 RGB	0.2982	0.2672	0.2481
ResNet152 GS	0.7556	0.3939	0.4237
ResNet152 TFL	0.4507	0.4397	0.4541
ResNet152 RGB	0.8278	0.4947	0.4657
ResNet101 TFL	0.4600	0.5481	0.5038
ResNet50 TFL	0.4107	0.5496	0.5802
ResNet152V2 RGB	0.8637	0.6107	0.5878
ResNet50 GS	0.8926	0.7237	0.7290
ResNet101 RGB	0.8163	0.7374	0.7290
ResNet101V2 GS	0.9859	0.7511	0.7519
Xception RGB	0.9417	0.7573	0.7557
Custom GS	0.7990	0.8076	0.7939
ResNet152V2 TFL	0.8287	0.8244	0.8053
ResNet50 RGB	0.8513	0.8122	0.8092
ResNet50V2 TFL	0.8530	0.8198	0.8130
VGG-19 TFL	0.7738	0.8000	0.8206
Xception TFL	0.8112	0.8183	0.8320
VGG-16 TFL	0.7999	0.8458	0.8321
ResNet101 GS	0.8082	0.8305	0.8473
Custom RGB	0.8010	0.8229	0.8511
ResNet101V2 TFL	0.8338	0.8290	0.8511
ResNet152V2 GS	0.8854	0.8290	0.8550
ResNet101V2 RGB	0.9876	0.8611	0.8817
ResNet50V2 RGB	0.9956	0.9160	0.8893
ResNet50V2 GS	0.9916	0.9053	0.9198
Xception GS	**0.9425**	**0.9008**	**0.9275**

Tabla 8.2: Precisión de los distintos métodos de clasificación (con DA).

VGGs

Al igual que sus equivalentes sin aumento de datos, las redes VGG que no emplearon el aprendizaje por transferencia, vuelven a encontrarse en la cima de esta agrupación, obteniendo nuevamente valores inferiores a 30% de tasa de precisión. En esta agrupación incluso puede verse cierta **bajada de rendimiento general** y un cierto aumento de ruido por parte de la precisión. Más adelante se verá que esta bajada de rendimiento no es algo único de estas redes, sino que es un patrón común de esta configuración y que son pocas las excepciones.

En cuanto a las versiones que emplearon el aprendizaje por transferencia, se debe destacar una vez más su evaluación. Estas tienden a evolucionar y alcanzar resultados alto. No obstante, esta vez no alcanzan el umbral de aceptación, ya que obtienen un 82.06% y un 83.21%. Y tan solo VGG-16 TFL se posiciona entre las diez mejores redes.

Estos resultados sustentan la hipótesis planteada anteriormente, ya que, de nuevo, las redes bases vuelven a atorarse durante sus procesos de validación. Mientras que, sus entrenos, se desplazan erráticamente entre una tasa de precisión mínima superior al 20% y una tasa de precisión máxima inferior al 31%. En cambió cómo se ha explicado con anterioridad las redes que emplearon la propiedad del aprendizaje por transferencia, cuya cantidad de parámetros entrenables vuelve a ser inferior, efectúan un trabajo en condiciones, a pesar de la evidente reducción de valor. Véase ambos comportamientos en sus figuras correspondientes, Figura 8-9, modelo sin aprendizaje por transferencia, y Figura 8-10, modelo que hizo uso de este aprendizaje.

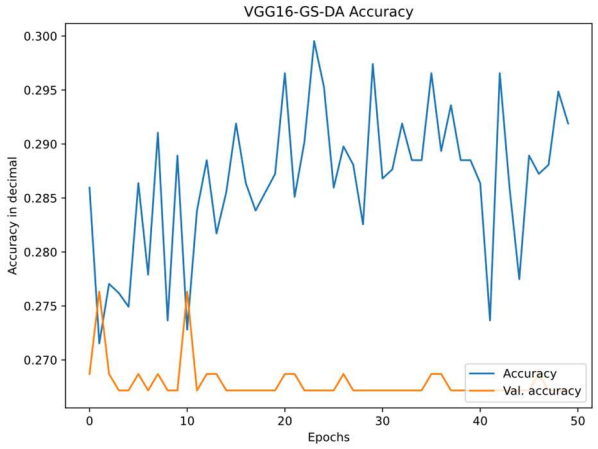

Figura 8-9: Entrenamiento y validación VGG16 GS-DA.

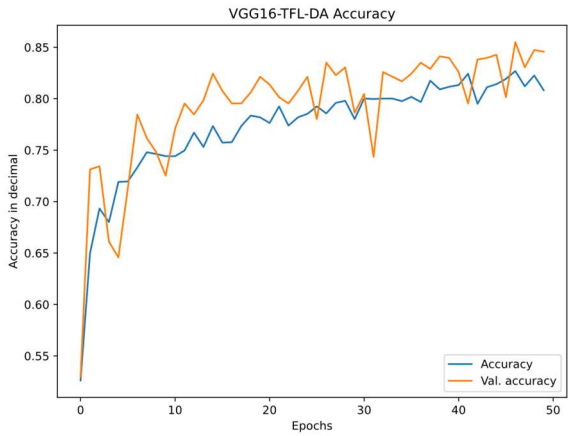

Figura 8-10: Entrenamiento y validación VGG16 TFL-DA.

ResNets

Resulta pelicular el comportamiento de las redes ResNets en esta agrupación, ya que, como se puede observar, existe cierta disparidad en los resultados. En la primera tabla podían verse como una comuna, que albergaba desde la posición quinta, empezando por la parte superior, hasta la vigésima primera. En cambio, en esta segunda tabla, Tabla 8.2, a pesar de que la mayoría de las redes se encuentran en las mismas posiciones, y con resultados muy inferiores, hay una agrupación que se halla dentro de las diez mejores redes. Siendo estas las únicas redes de esta arquitectura "aceptables" debido a sus resultados iguales o superiores a 85% de precisión. Estas son las redes ResNet101GS, las versiones RGB y TFL de ResNet 101V2, ResNet50V2 RGB y GS, y ResNet152V2 RGB.

A pesar de ello estas redes vuelven a efectuar una actuación similar a la previamente vista. Llegando a apreciarse en sus gráficas (Figura 8-11, Figura 8-12 y Figura 8-13), altos niveles de inestabilidad por parte del conjunto de validación, una mayor eficiencia a la hora de entrenar, resultados limitados por la validación, y superposiciones de los resultados en las redes que emplearon el aprendizaje por transferencia.

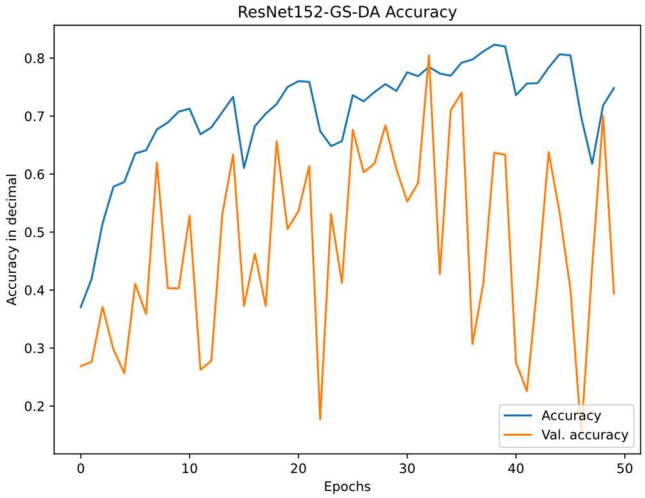

Figura 8-11: Entrenamiento y validación ResNet152GS-DA.

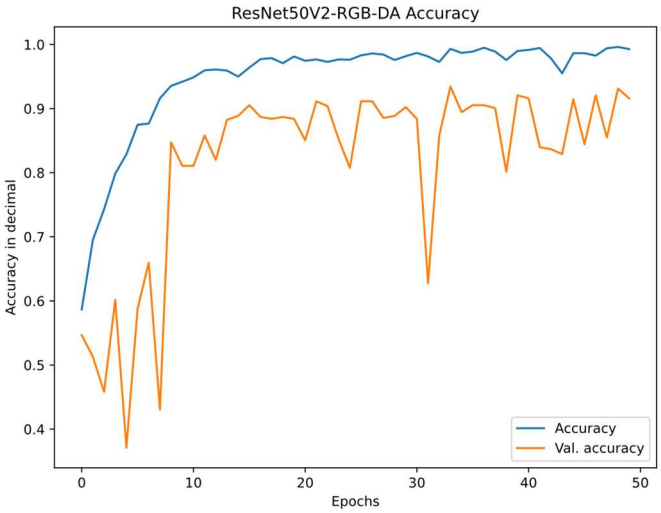

Figura 8-12: Entrenamiento y validación ResNet50V2RGB-DA.

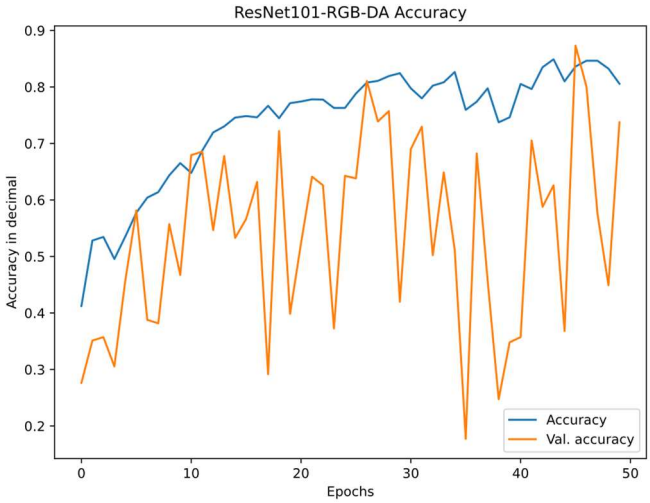

Figura 8-13: Entrenamiento y validación ResNet101 RGB-DA.

Xception

Xception vuelve a tomar el primer puesto en este ranking, a pesar de la notoria perdida de score general. Esto se puede ver reflejado principalmente en las redes Xception TFL y Xception RGB, esta primera, que alcanzaba a solucionar el problema con un 96.56% de precisión, lo cual hacía que se volviera la mejor red del grupo anterior. En esta ocasión se conforma con un 83.20%. La red Xception RGB que albergaba el segundo puesto con un 92.74%, obtiene en este caso un 75.57%. Xception GS es una de las excepciones de las que se habló junto a las redes ResNet anteriormente nombradas, batiendo su score anterior de 90.48%, con un 92.75% de precisión en esta ocasión. Estas redes, a pesar de tener una tendencia similar a las anteriormente vista de esta arquitectura, son mucho más ruidosas, las caídas prolongadas que se podían ver antes, que eran preocupantes, pero poco frecuentes, ahora se vuelven más prolongadas y frecuentes, esto se puede observar al analizar sus gráficas (Figura 8-14, Figura 8-15 y Figura 8-16).

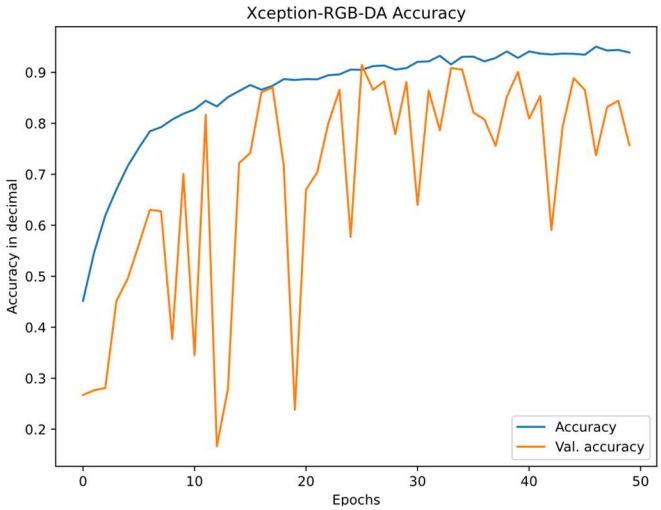

Figura 8-14: Entrenamiento y validación Xception RGB-DA.

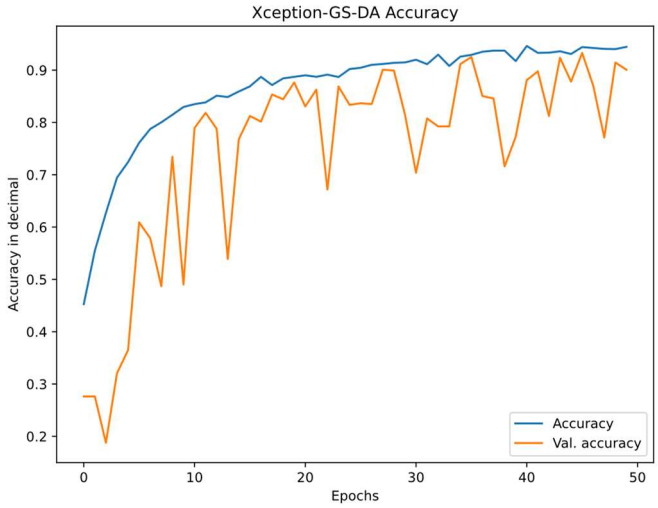

Figura 8-15: Entrenamiento y validación Xception GS-DA.

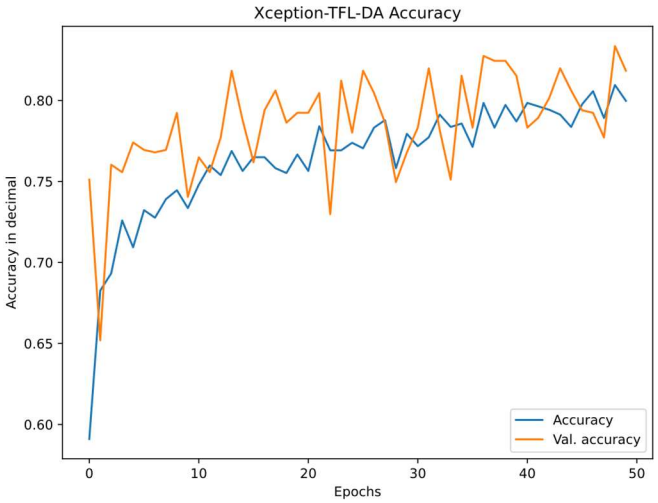

Figura 8-16: Entrenamiento y validación Xception TFL-DA.

Modelos customizados

Las redes customizadas se ven peculiarmente estresadas al emplear esta configuración, el crecimiento de la precisión del entrenamiento se vuelve más lento y progresivo, aunque tiende de manera correcta. Esta conducta se traduce en una línea que se aproxima más a una diagonal que a una curva, véase esta en las siguientes figuras, Figura 8-17 y Figura 8-18. En cuanto a la validación se vuelve más caótica, y se presenta un alto grado de sobreajuste. Es a partir de esta información por la que se deduce este notorio estrés comentado en el párrafo superior. En cuanto a los resultados, como era de esperar, son inferiores a los obtenidos por sus similares, 79.39% y 85.11%.

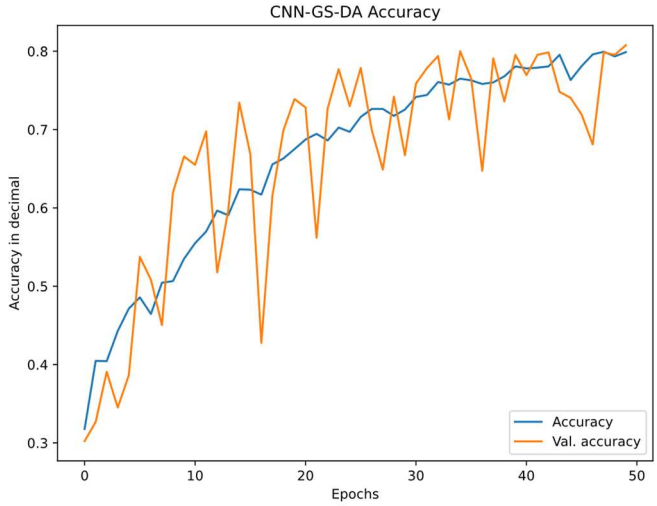

Figura 8-17: Entrenamiento y validación Custom GS-DA.

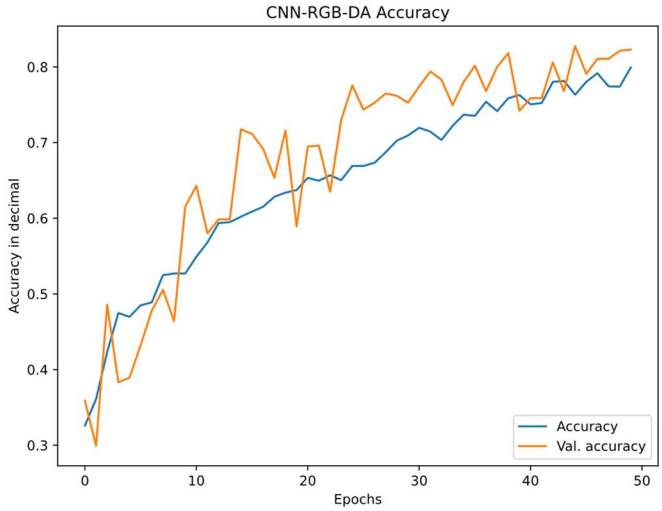

Figura 8-18: Entrenamiento y validación Custom RGB-DA.

Tras este análisis se ha concluido, observando los resultados y las características de los modelos precedentes, que las redes VGG's que no emplearon la propiedad del aprendizaje por transferencia y las redes ResNet que no alcanzaron el umbral, son poco fiables e inviables de usar para la resolución de esta tarea. Es por ello por lo que quedarían totalmente descartadas.

También, se descartaría en su totalidad emplear este aumento de datos en concreto, puede que los desplazamientos y las rotaciones, en un problema en el que la localización es un punto clave y para tener en cuenta en el diagnostico, no sea del todo acertado. Sobre todo, teniendo en cuenta que las MRI se realizan de una forma específica y con los pacientes en unas posiciones concretas, por lo que no se pueden dar volteos verticales y horizontales, rotaciones, etc.

De las redes que alcanzaron buenos resultados, las arquitecturas propuestas y Xception son sin lugar a duda las más interesantes y atractivas, ya sea por su estabilidad o por sus resultados próximos a la perfección. Las ResNets son muy ruidosas, en comparación al resto, esto las vuelve en cierto modo, impredecibles. Aunque esto no suponga un problema en este proyecto, lo puede ser para aplicaciones de continuo aprendizaje. Por lo que, a ojos del autor, esto es una razón suficiente como para descartarla para las siguientes fases de análisis. Si bien Xception presenta inestabilidad, esta se produce con menos frecuencia.

Los métodos más precisos de acuerdo con estos datos y a partir de este análisis final, son:

- Custom GS
- VGG-19 TFL
- Xception RGB
- Xception DA GS
- Xception TFL

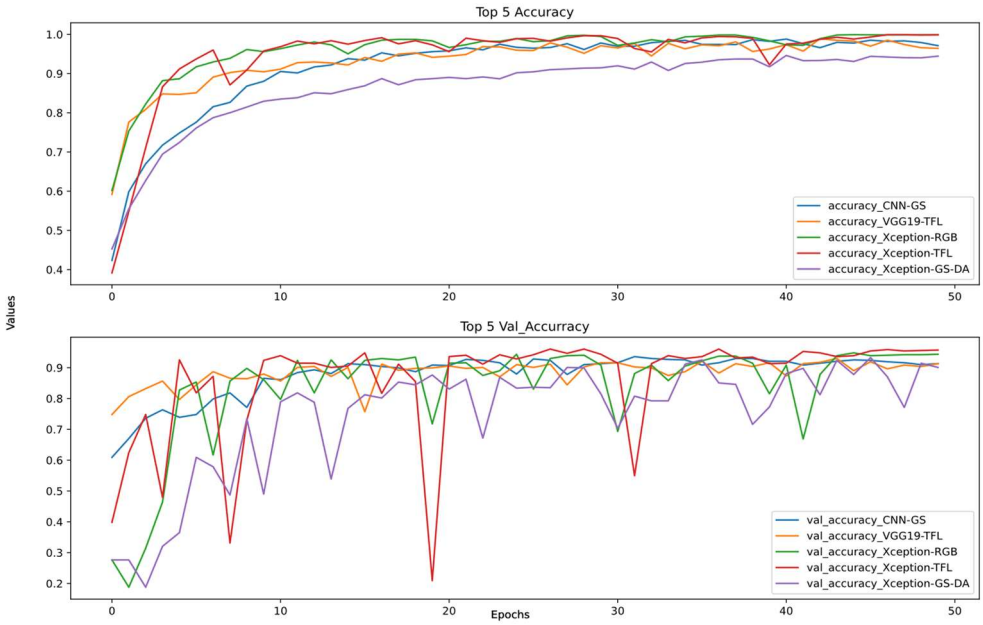

Figura 8-19: Enntrenamiento y validación cinco mejores modelos.

Como se observa en la ilustración anterior, Figura 8-19, todos estos modelos presentan un comportamiento adecuado a la hora de entrenar, logrando una curva, cuyas principales características son una tendencia convergente, que tiende al 1.0, y un bajo grado de ruido (alta estabilidad).

En la validación, a pesar de sus altos rendimientos y resultados, sucede algo similar. Se aprecia claramente esta tendencia y convergencia, aunque esta segunda de una manera menos gradual, ya que alcanzan resultados óptimos de manera precoz. Seguramente a causa de la limitación de tamaño proveniente del lote de validación, ya que esta cuenta con muy pocas imágenes. A parte de ello puede encontrarse un alto grado ruido proveniente, sobre todo, de los modelos Xception, que como se vio anteriormente, son algo inestables a la hora de ejercer este proceso.

Estas irregularidades durante el proceso de validación no se encuentran únicamente en los modelos Xception. VGG-19 TFL también cuenta con este problema, aunque de forma más reducida. Estos desajustes son propios del conocido como, sobreajuste u *overfitting*. Este desajuste, que se ve presentado en la Figura 8-20, es causado por un aprendizaje forzado, en el que la maquina ha aprendido a deducir, o a diagnosticar en este caso, por la falta de variabilidad del conjunto. Al realizarse los cambios de lotes, estos aplicarían el aprendizaje previo sin existo alguno. Y tardarían un periodo en adaptarse y aprender a resolver este nuevo lote, este periodo

es el punto en el que se produce una depresión y alce de los resultados. Volviendo a las gráficas anteriores, podrá ver que esta se encuentra presente, en mayor o menor medida, en los modelos presentados.

Por último, la red propuesta, entre los modelos presentados, podría ser el método de clasificación con mayor nivel de naturalidad a la hora de entrenar y validar, esto debido a la estabilidad que presenta, dato que se venía prediciendo con anterioridad.

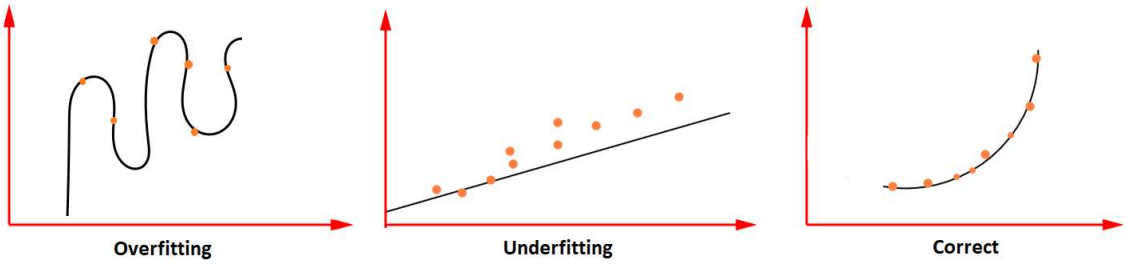

Figura 8-20: Representación gráfica de los desajustes.

8.2 Evaluación de la clasificación de tumores

En este punto se evaluará de forma gráfica los resultados de los cinco mejores modelos, al clasificar los tumores del conjunto de testeo. Para ello se emplearán las matrices de confusión.

Las matrices de confusión determinan la cantidad de fallos y aciertos, por clases, que produce una modelo inteligente al someterlas a la tarea de predecir o evaluar las imágenes de un conjunto, en este caso, el conjunto de prueba. Permitiendo así, precisar, que clase o clases están provocando mayor equivocación a los modelos. Es decir que clases son más costosas de resolver para la red. En estas matrices ambos ejes, X e Y, representan las clases del conjunto de datos y es por ello por lo que cuentan con sus etiquetas.

La forma de comprender estas matrices es tomando el eje X como la clase en la que se realizará el estudio. Mientras que la Y será tomada como la resolución o respuesta del modelo. El valor encerrado en cada uno de los cuadrados precisará la cantidad de imágenes, de esa clase otorgada por el eje X, tras evaluarlas como la clase referenciada por el eje Y. Cuando las etiquetas de ambos ejes son iguales serán comprendidos cómo aciertos, por otra parte, si difieren serán fallos. Quedando así, en teoría, una matriz con una diagonal descendente como la de la Figura 8-21. Esta va desde el punto (0, 0) hasta el punto (N-1, N-1), siendo N el número de clases. Este sería el comportamiento deseado, no obstante, al depender del aprendizaje de los modelos, puede no darse esta diagonal, esto sucede por ejemplo con los modelos VGGs que no emplearon el aprendizaje por transferencia. Como se visiono en el apartado anterior, estos modelos eran incapaces de aprender, y esto se refleja en su matriz de confusión, Figura 8-22, de la siguiente manera. La red considera que todos los datos entrantes son meningiomas, estos resultados son incoherentes, ya que setenta y siete de las imágenes entrantes son gliomas, cuarenta son cerebros sin patologías y ochenta son tumores pituitarios.

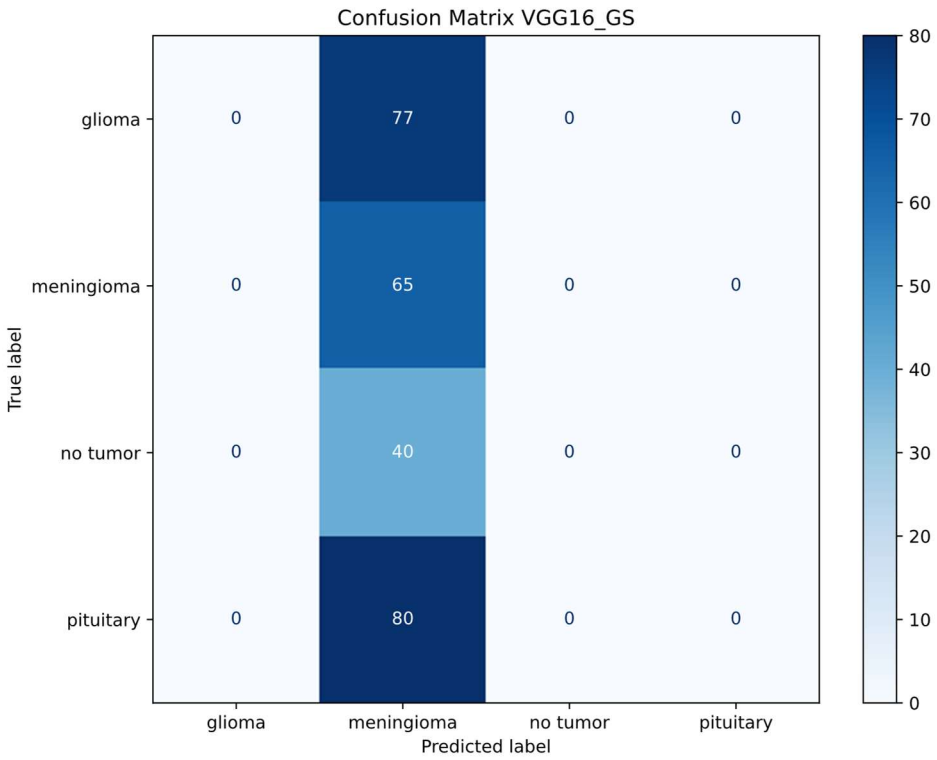

Figura 8-21: Ejemplo de matriz de confusión con erróneo comportamiento.

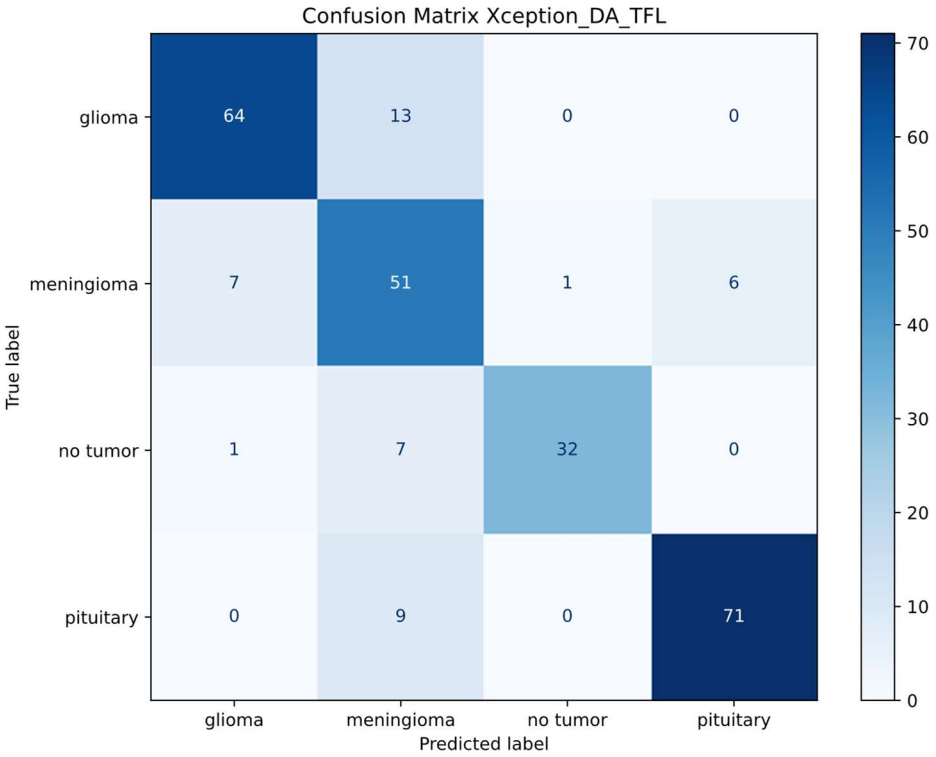

Figura 8-22: Ejemplo de matriz de confusión con correcto comportamiento.

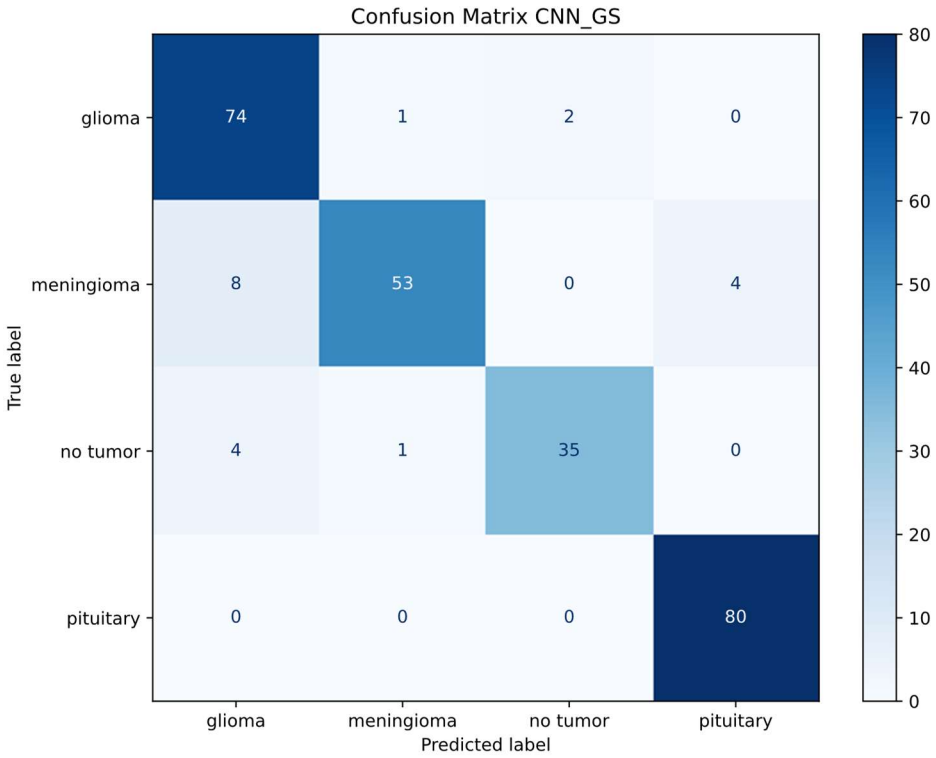

Figura 8-23: Matriz de confusión Custom GS.

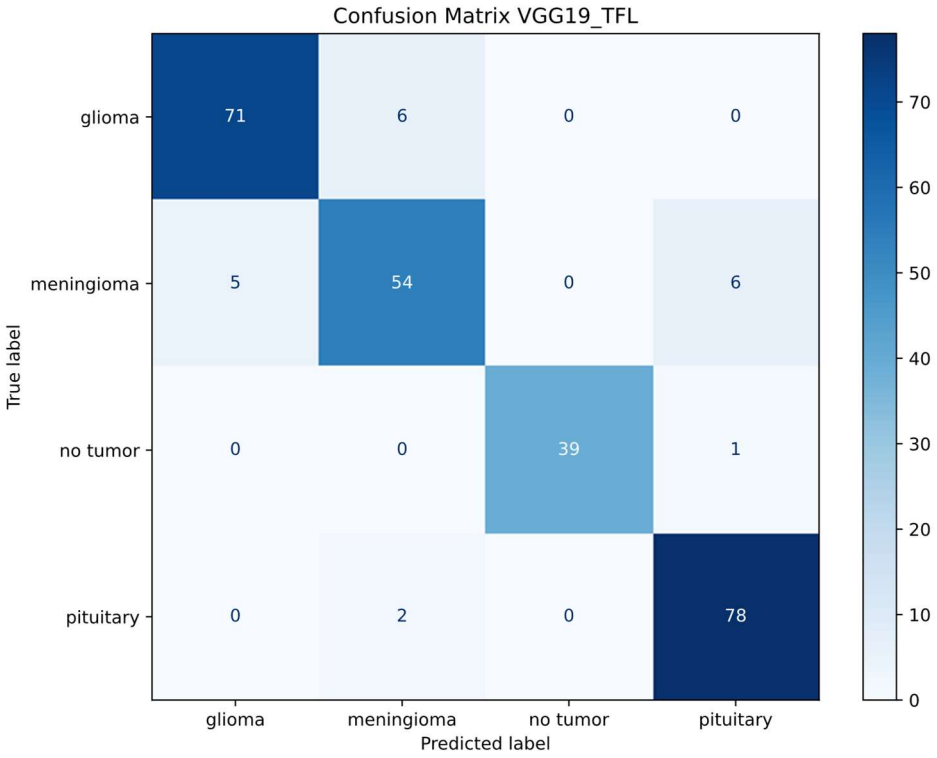

Figura 8-24: Matriz de confusión VGG-19 TFL.

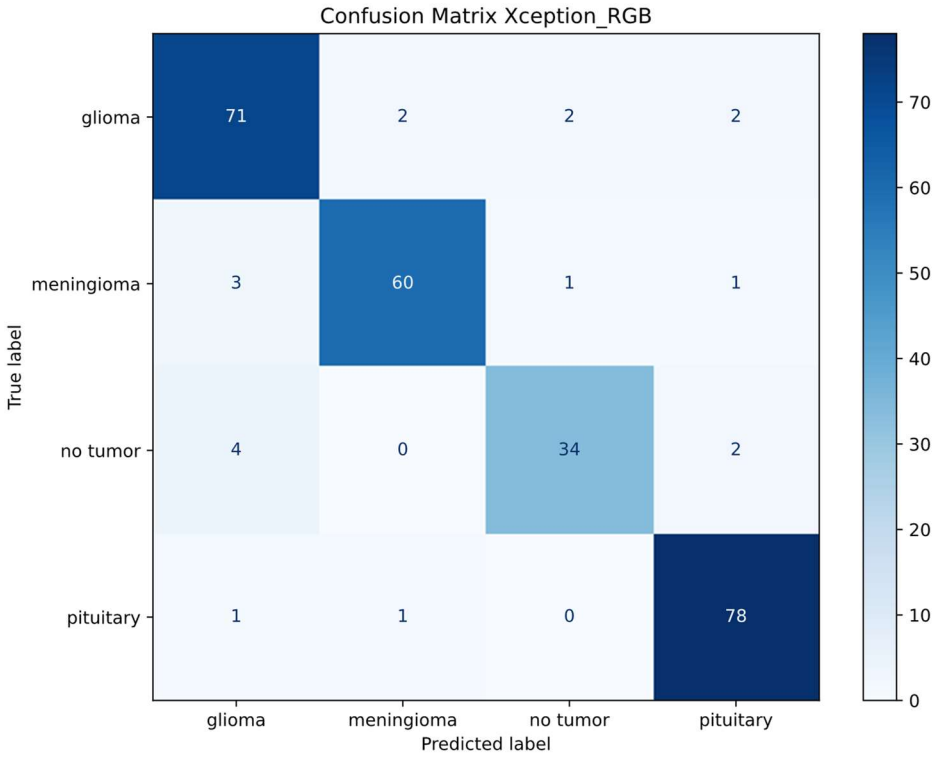

Figura 8-25: Matriz de confusión Xception RGB.

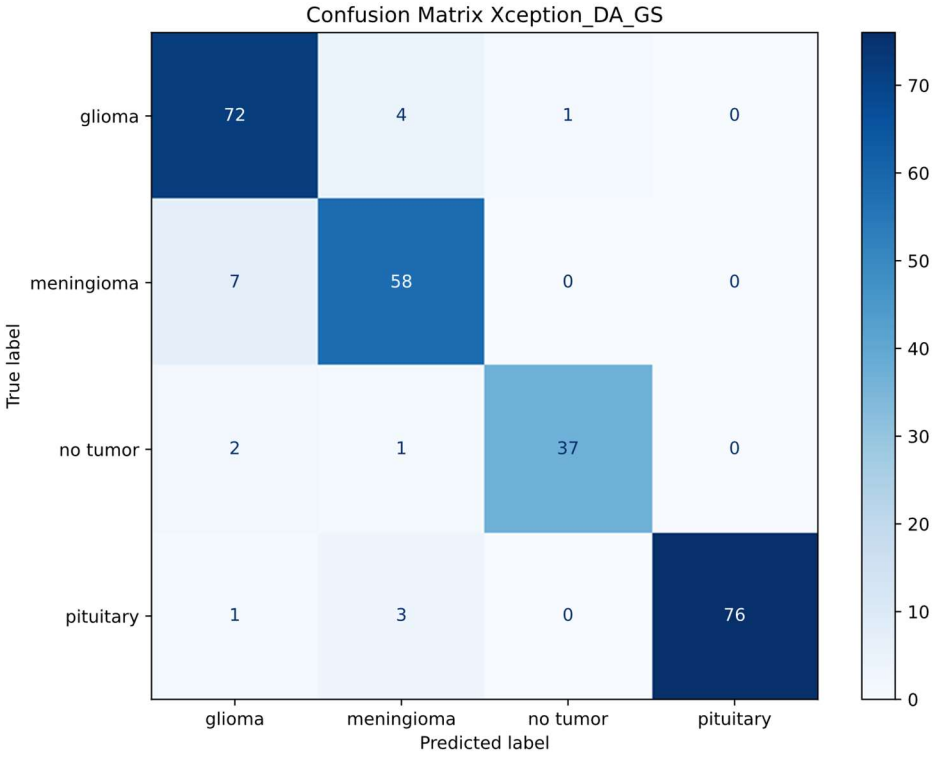

Figura 8-26: Matriz de confusión Xception DA-GS.

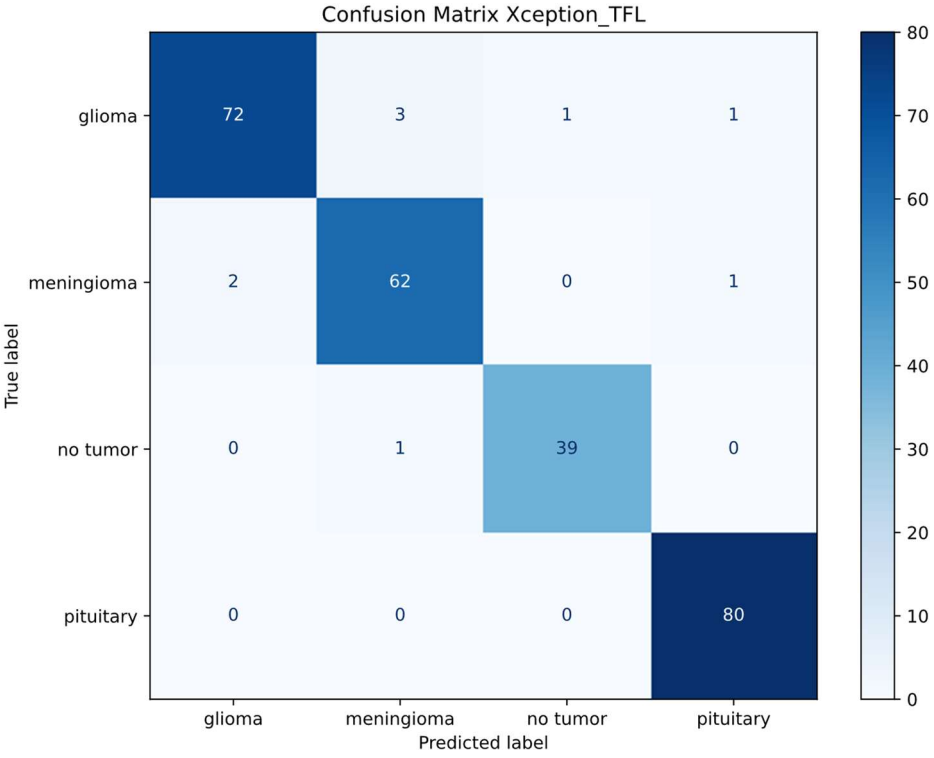

Figura 8-27: Matriz de confusión Xception TFL.

Número de aciertos					
Modelos	**Glioma**	**Meningioma**	**No Tumor**	**Pituitario**	**Total**
VGG 19 TFL	71	54	39	78	242
CNN GS	74	53	35	80	242
Xception RGB	71	60	34	78	243
Xception DA GS	72	58	37	76	243
Xception TFL	72	62	39	80	253

Tabla 8.3: Número de aciertos por clases

Porcentaje de aciertos					
Modelos	**Glioma**	**Meningioma**	**No Tumor**	**pituitario**	**Total**
VGG 19 TFL	92.21%	83.08%	97.50%	97.50%	92.37%
CNN GS	96.10%	81.54%	87.50%	100.00%	92.37%
Xception RGB	92.21%	92.31%	85.00%	97.50%	92.75%
Xception DA GS	93.51%	89.23%	92.50%	95.00%	92.75%
Xception TFL	93.51%	95.38%	97.50%	100.00%	96.56%

Tabla 8.4: Porcentaje de aciertos por clases

Número de fallos					
Modelos	**Glioma**	**Meningioma**	**No Tumor**	**Pituitario**	**Total**
VGG 19 TFL	Meningioma: 6 No Tumor: 0 Pituitario: 0 Total: 6	Glioma: 5 No Tumor: 0 Pituitario: 6 Total: 11	Glioma: 0 Meningioma: 0 Pituitario: 1 Total: 1	Glioma: 0 Meningioma: 2 No Tumor: 0 Total: 2	20
CNN GS	Meningioma: 1 No Tumor: 2 Pituitario: 0 Total: 3	Glioma: 8 No Tumor: 0 Pituitario: 4 Total: 12	Glioma: 4 Meningioma: 1 Pituitario: 0 Total: 5	Glioma: 0 Meningioma: 0 No Tumor: 0 Total: 0	20
Xception RGB	Meningioma: 2 No Tumor: 2 Pituitario: 2 Total: 6	Glioma: 3 No Tumor: 1 Pituitario: 1 Total: 5	Glioma: 4 Meningioma: 0 Pituitario: 2 Total: 6	Glioma: 1 Meningioma: 1 No Tumor: 0 Total: 2	19
Xception DA GS	Meningioma: 4 No Tumor: 1 Pituitario: 0 Total: 5	Glioma: 7 No Tumor: 0 Pituitario: 0 Total: 7	Glioma: 2 Meningioma: 1 Pituitario: 0 Total: 3	Glioma: 1 Meningioma: 3 No Tumor: 0 Total: 4	19
Xception TFL	Meningioma: 3 No Tumor: 1 Pituitario: 1 Total: 5	Glioma: 2 No Tumor: 0 Pituitario: 1 Total: 3	Glioma: 0 Meningioma: 1 Pituitario: 0 Total: 1	Glioma: 0 Meningioma: 0 No Tumor: 0 Total: 0	9

Tabla 8.5: Número de fallos por clases

Las ilustraciones Figura 8-23, Figura 8-24, Figura 8-25, Figura 8-26 y Figura 8-27, muestran los resultados proporcionados por las cinco mejores redes de este proyecto. A su vez para facilitar la lectura se ha almacenado esta información en tres tablas.

La primera muestra los números de aciertos por clases, Tabla 8.3, la segunda muestra los errores producidos, Tabla 8.4, y la tercera el porcentaje de acierto, Tabla 8.5. Los datos contenidos en cada una de estas tablas sirven para corroboran los resultados obtenidos por cada una de las redes de manera individual.

Modelos	Entrenamiento	Validación	Testeo
Custom GS	0.9753	0.9130	0.9237
VGG-19 TFL	0.9540	0.9130	0.9237
Xception RGB	0.9988	0.9435	0.9274
Xception GS	0.9425	0.9008	0.9275
Xception TFL	0.9982	0.9573	0.9656

Tabla 8.6: Comparativa de datos.

La red **VGG 19 TFL**, cuenta con una precisión total de 92.37%.

- La clase glioma produce 71 aciertos y 6 diagnósticos erróneos. Esto equivale a un 92.21% de acierto, por parte de esta red hacia esta clase, los 6 diagnósticos erróneos son confundidos con meningiomas.
- En cuanto a la clase meningioma el error es más elevado. Este predice correctamente 54 de las 65 imágenes, errando así un total de 11. Cinco de estas con la clase glioma y el resto con la clase tumor de pituitaria. La tasa de acierto para esta etiqueta es de 83.08%.
- La clase no tumor, o la clase que referencia cerebros sin patologías, obtiene un 97.50% de diagnósticos correctos. Por lo que 39 de las 40 imágenes son diagnosticadas de forma adecuada. Esta red únicamente erra uno de los diagnósticos afirmando que se trata de un tumor pituitario.
- La clase pituitaria también obtiene un 97.50%, esto se traduce como dos diagnósticos erróneos y 78 aciertos. Ambos diagnósticos equívocos, son tratados como meningiomas.

De igual manera la red **CNN GS**, o red propuesta empleando imágenes a escala gris como entrada, logra una tasa de acierto total del 92.37%.

- Este modelo logra un pleno en la clase tumor pituitario, consiguiendo diagnosticar correctamente todos los casos.
- La clase no tumor genera un total de 5 diagnósticos erróneos, de los cuales 4 son confundidos con gliomas y uno por meningioma. Esto se traduce en una tasa de acierto del 87.50%.
- La clase meningioma revela un comportamiento similar al del anterior modelo, siendo la clase con menor puntuación, 81.54%. Esta red produce 12 predicciones erróneas para este subconjunto, de los cuales 8 son confundidos con gliomas y 4 con tumores pituitarios.
- Por último, la etiqueta glioma alcanza un 96.10% de acierto, esto debido a que solo realiza 3 diagnósticos erróneos, de los cuales dos son no tumor y uno meningiomas.

Xception RGB logra un 92.75% de acierto.

- 92.21% por parte de la clase glioma, la cual falla un total de 6 diagnósticos y de forma equitativa, dos por cada clase.
- El subconjunto de imágenes con etiqueta meningioma logra obtener un 92.31% de éxito, fallando únicamente cinco diagnósticos.
- El subconjunto de cerebros sin patologías se ve drásticamente afectado en esta ocasión logrando alcanzar tan solo un 85.00%.
- 97.50% obtuvo la red al catalogar los tumores pituitarios, tan solo dos diagnósticos erróneos de los ochenta.

Xception DA GS iguala en resultados a Xception RGB, no obstante, esto resultados difieren a nivel de clases.

- 93.51% por parte de la clase glioma, un fallo menos que Xception RGB.
- 89.23% se ve reducido de forma drástica frente a Xception RGB.
- El subconjunto de cerebros sanos logra alcanzar el 92.50% de acierto, bastante superior a los datos reflejados por Xception RGB sobre esta misma clase.
- 95.00% obtuvo la red al catalogar los tumores pituitarios, esto equivale a cuatro diagnósticos incorrectos.

Por último, **Xception TFL**, red que alcanza una tasa de acierto del 96.56%, además de lograr porcentajes superiores al 90% en todas las clases.

- Esta red casi logra un diagnóstico perfecto. La clase que más fallos genera es la clase glioma y tan solo 5 de los 77 diagnósticos son erróneos.
- Seguido de esto, la clase meningioma registra un total de 3 diagnósticos equívocos.
- La clase no tumor por otra parte diagnostica únicamente uno de los MRI de forma errónea.
- En cuanto a la clase tumor pituitario al igual que CNN GS diagnostica de forma correcta la totalidad de los casos.

Con todo esto se deduce que, los meningiomas son los tipos de tumores más problemáticos o complicados de diagnosticar por estos modelos, los únicos que superan el 90% de precisión con esta clase son Xception RGB y Xception TFL. Por otra parte, los modelos CNN GS y Xception RGB tienen algún que otro problema al catalogar si un cerebro no padece ninguna patología.

Por último, cabe destacar el gran trabajo logrado por Xception TFL, la cual supera con creces el umbral mínimo de aceptación en todas y cada una de las clases a diagnosticar, además de ser el modelo que mejores resultados obtiene en cada una de las clases, a excepción del modelo CNN GS al diagnosticar gliomas.

8.3 Evaluación robusta del rendimiento

A continuación, se procederá a evaluar de forma robusta los rendimientos de los mejores modelos. Para comprender mejor las métricas que se emplearán en este punto se ha agregado una breve introducción en la que se pretende explicar en qué casos se usa y para que se emplean, cada una de ellas. Pero antes se debe explicar ciertos conceptos:

- **TP (True Positives):** este valor representa la cantidad de veces que la red acierta de manera real. Es decir, la cantidad de etiquetas predichas que concuerdan con las etiquetas reales.

- **TN (True Negatives):** este concepto está relacionado con el anterior, y es que, si TP es la cualidad que determina la cantidad de veces que una red acierta de manera correcta. Esta representa la cantidad de veces que una red falla de manera correcta.

- **FP (False Positive):** la cantidad de falsos positivos muestran las veces que una red produce un diagnóstico equivoco de una etiqueta real, esta afirma de manera fehaciente, que aquello que "ve" es correcto sin serlo.

- **FN (False Negative)**: por último, los falsos negativos son la cantidad de veces que una red afirma que lo que está "viendo" no es de la clase a estudiar, siendo esta parte de esa clase y por tanto esta predicción incorrecta.

Para analizar el funcionamiento de las redes neuronales se ha optado por extraer las siguientes métricas [74] [75] [76]:

8.3.1 Precisión (Precision)

La precisión es una métrica de rendimiento que se aplica a los datos recuperados de una colección, corpus o espacios de muestras. La precisión se entiende como la fracción de documentos recuperados que son relevantes para la consulta dividido entre el total del conjunto a buscar. Por ejemplo, para una búsqueda de texto, en la que se desea hallar el número de palabras que contenga la palabra casa, la precisión sería el número de resultados correctos, palabras que contengan casa, dividido entre el total. La precisión (precision) se define como:

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

Esta cuenta con un denominador que en algunos casos puede ser igual a cero, lo que hace imposible calcular una estimación de la métrica. Es por ello por lo que esta métrica junto a las siguientes es consideradas robustas. La precisión sería *NaN*[12] o cero para cualquier etiqueta que no es hipotetizada por el modelo cuando realiza predicciones en el conjunto de testeo o lo que es lo mismo, el número de positivos verdaderos (TP) es cero.

[12] ***NaN**: En computación, NaN, que significa Not a Number, es un tipo de datos numérico que se puede interpretar como un valor que no está definido o no se puede representar, especialmente en la aritmética de coma flotante.*

8.3.2 Recuperación (Recall)

La recuperación, al igual que la precisión, es una métrica de rendimiento que se aplica a los datos recuperados de una colección, corpus o espacio de muestra. De esta métrica se obtiene la proporción de observaciones positivas predichas correctamente con respecto a todas las observaciones en la clase real. Esta métrica permite interpretar la cantidad de clases que el modelo es capaz de diagnosticar de manera correcta, por lo que se suele emplear en el marketing, el recall se define como:

$$Recall\ = \frac{TP}{TP + FN}$$

En el recall el denominador que provoca que esta métrica sea imposible de calcular es (TP + FN). Esta aritmética simple, sería igual a cero para cualquier etiqueta que no está presente en el conjunto de prueba, o lo que es lo mismo, ninguna muestra en el al conjunto de prueba es etiquetada con su etiqueta real.

8.3.3 Puntuación F1 (F1-score)

También conocido como F-score, es una medida de la precisión de un modelo ante un conjunto de datos determinado, se trata del promedio ponderado de la precisión y recuperación. Por tanto, esta puntuación tiene en cuenta tanto los falsos positivos como los falsos negativos. Esta métrica se emplea principalmente para evaluar sistemas de recuperación de información, y también en modelos de aprendizaje profundo u automático, en particular aquellos destinados al procesamiento natural del lenguaje (*NLP*). Esta se representa con la siguiente formula.

$$F1 - Score\ =\ 2 * \frac{(Recall\ *\ Precision)}{Recall\ +\ Precision}$$

En el F1-score, los denominadores que imposibilitan el cálculo de esta métrica son, la Precision y el Recall, ya que para una Precision o Recall NaN, provoca un F1 NaN.

8.3.4 Apoyo (Support)

El apoyo, indica la cantidad de imágenes empleadas en cada uno de los procesos nombrados anteriormente. Para este proyecto se ha empleado, 77 meningiomas, 65 gliomas, 40 sin tumores y 80 tumores pituitarios, estas son las imágenes contenidas en el conjunto de prueba.

8.3.5 Curva ROC (ROC Curve)

Curva ROC [74] [77] o curva característica de funcionamiento del receptor, (ROC Curve - *Receiver Operating Characteristic Curve*) es un método estadístico para determinar la precisión de los diagnósticos sobre el conjunto de prueba. Esta grafica se emplea para evaluar la capacidad discriminativa del modelo, es decir, su capacidad de discernir entre las distintas clases. Este tipo de métrica se conoce como Curva ROC extendida, en ella se muestran las curvas formadas por las distintas clases, por lo que su uso se adecua a este problema de carácter categórico. Los ejes de este gráfico adoptan valores entre 0 y 1, delimitando un cuadrado de área igual a 1. Por otra parte, un modelo se considera no-discriminativo si su curva ROC coincide con la línea de no-discriminación o AUC, esta poseería un AUC igual a 0.50, esta curva ROC, por ende, se superpone con la línea de no-discriminación. Al igual que con la precisión a medida que el AUC se acerca al 100%, mayor será su capacidad discriminativa (véase estos comportamientos en la Figura 8-28).

$$ROC\ Curve\ = \frac{Recall}{(\frac{FP}{FP + TN})}$$

Estas gráficas contienen cuatro líneas continuas, cada una de estas líneas referencian la relación TP/TN de una clase, además de dos líneas discontinuas denominadas ***micro-average*** **y** ***macro-average*** [78].

El **micro-average** o micro promedio, es la suma de todos los verdaderos positivos dividiendo entre la suma de todos los verdaderos positivos y todos los falsos positivos. Básicamente, se debe dividir el número de predicciones correctamente identificadas por el número total de predicciones. Esta medida permite comprobar si existe algún desequilibrio de clases.

$$Micro - Avg.\ = \sum_{i=0}^{n-1} \frac{Tp_i}{Tp_i + Fp_i}$$

El **macro-average** o macro promedio en cambio, es la suma de todos los verdaderos positivos y divido entre la cantidad de clases. Se podría decir que este es el promedio medio de la actuación general de la red.

$$Macro - Avg.\ = \frac{\sum_{i=0}^{n-1} Tp_i}{n}$$

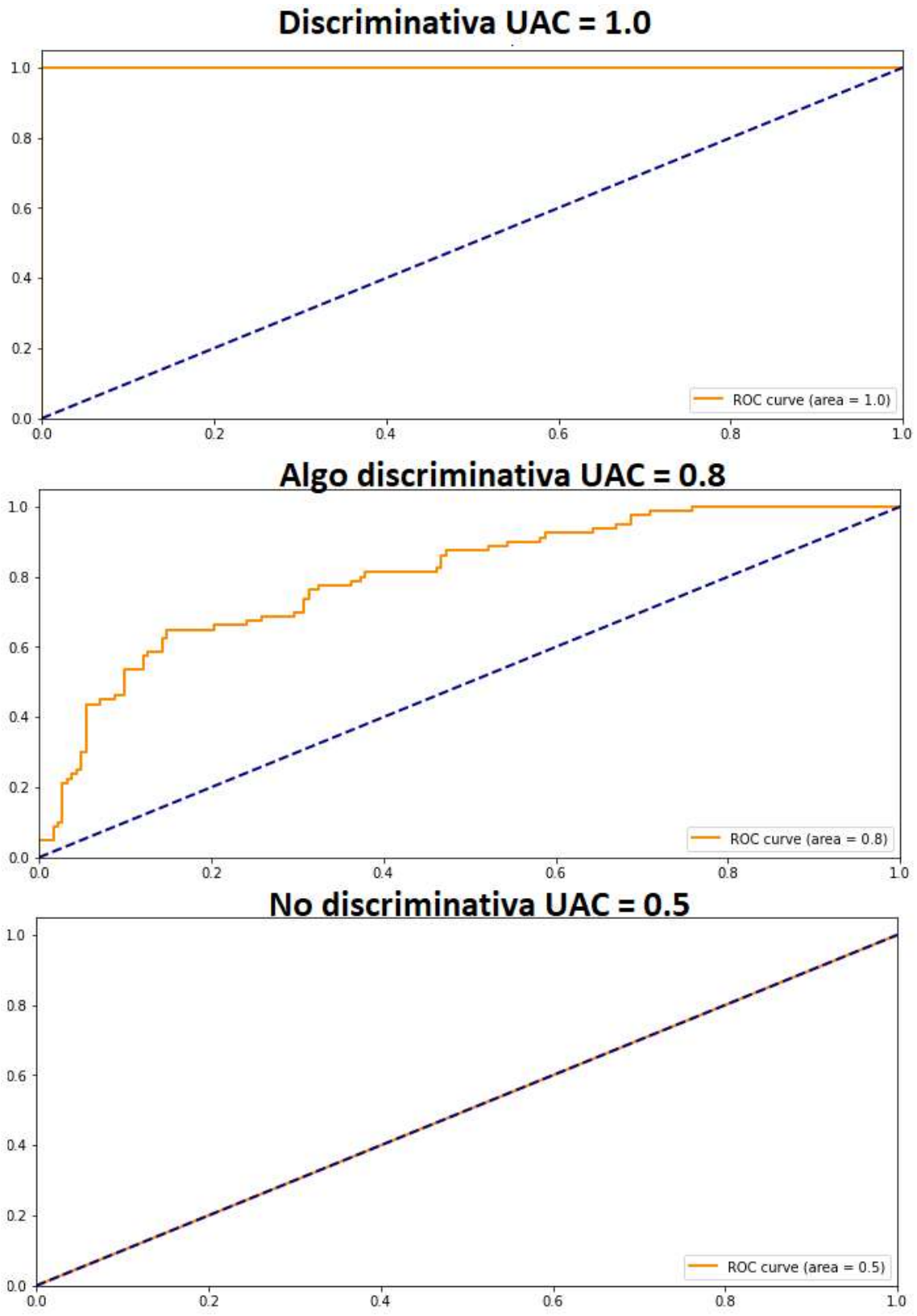

Figura 8-28: Tipos de discriminaciones curva ROC y UAC.

Model	labels	precision	recall	f1-score	support
VGG-19 TFL	0	0.93	0.92	0.93	77
	1	0.87	0.83	0.85	65
	2	1.00	0.97	0.99	40
	3	0.92	0.97	0.95	80
Xception RGB	0	0.90	0.92	0.91	77
	1	0.95	0.92	0.94	65
	2	0.92	0.85	0.88	40
	3	0.94	0.97	0.96	80
Custom GS	0	0.86	0.96	0.91	77
	1	0.96	0.82	0.88	65
	2	0.95	0.88	0.91	40
	3	0.95	1.00	0.98	80
Xception TFL	0	0.97	0.94	0.95	77
	1	0.94	0.95	0.95	65
	2	0.97	0.97	0.97	40
	3	0.98	1.00	0.99	80
Xception DA-GS	0	0.88	0.94	0.91	77
	1	0.88	0.89	0.89	65
	2	0.97	0.93	0.95	40
	3	1.00	0.95	0.97	80

Tabla 8.7: Precision / Recall / F1-Score / Support Mejores Modelos.

Como conclusión de estos datos, Tabla 8.7, que abarcan las precisiones, recuperaciones, puntuaciones F1 y apoyos de los cinco mejores modelos. Hay que puntualizar que a excepción del Recall hallado por Custom GS para los no tumores y el Recall obtenido por VGG-19 TFL para los meningiomas, el resto de los resultados se encuentra dentro del umbral de aceptación. Esto refleja en sí lo visto en los puntos previos. Estas tres métricas, formulan así las siguientes preguntas.

¿Cómo de precisos son los modelos presentados?

Esta primera pregunta es postulada por la **precisión**, y es a partir de las experiencias extraídas en este campo, la forma en la que se resuelve.

- **VGG-19 TFL:** 92.75% de precisión media.
- **Xception RGB**: 92,75% de precisión media.
- **Custom GS**: 93.00% de precisión media.
- **Xception TFL:** 96.50% de precisión media.
- **Xception Data Augmentation GS**: 93.25% de precisión media.

Todos los modelos presentados en este punto son al menos, un 92.75% de precisos.

¿Qué porcentaje de casos positivos identificaron de manera correcta cada uno de los presentes modelos?

Esta duda la formula la **recuperación**, y al igual que con la anterior, es a partir de ella por la que se puede resolver.

- **VGG-19 TFL:** 92.25% de recall medio.
- **Xception RGB:** 91.50% de recall medio.
- **Custom GS:** 91.50% de recall medio.
- **Xception TFL:** 96.50% de recall medio.
- **Xception Data Augmentation GS:** 94% de recall medio.

Las redes precisadas en este punto tienen como mínimo un 91.50% de discriminaciones positiva correctas.

¿Qué tan bueno son los presentes modelos?

El **f1-score**, determina que tan bueno es un modelo, por lo que, a partir de los resultados anteriores se responde esta duda.

- **VGG-19 TFL:** 93.00%. de f1-score.
- **Xception RGB:** 92.25% de f1-score.
- **Custom GS:** 92.00% de f1-score.
- **Xception TFL:** 96.50% de f1-score.
- **Xception Data Augmentation GS:** 93.00% de f1-score.

Los modelos empleados en este punto son como mínimo un 92.00% de fiabilidad.

Con estas tres preguntas concluimos que estos modelos poseen un alto nivel de precisión y de correcta discriminación. De manera que se puede afirmar que se cuenta con una buena serie de modelos inteligentes.

8.3.6 Análisis de las curvas ROC

Como anteriormente se explicó, las curvas ROC son un método estadístico que permite determinar con exactitud la discriminación a la hora de catalogar o diagnosticar empleando una escala continua.

En este análisis se determinará los puntos de corte y se evaluará las capacidades discriminativas de cada una de las redes y sus áreas, que de manera resumida se podría decir que ambos se encuentran en su máximo valor, que es uno. Lo cual hace que estén dentro del intervalo de “Test Excelente”, resultado que se encuentra en concordancia con la capacidad discriminativa. Además, se expresará de forma resumida, los resultados de estas gráficas. Pero antes una pequeña explicación sobre estos nuevos conceptos [74]:

- **Puntos de corte:** Este término hace referencia al punto en el que la curva alcanza su máximo valor.

- **Área:** El área referencia el alcance del valor promedio, es decir la tasa promedia de la sensibilidad/especificidad o lo que es lo mismo TPR/FPR, donde TPR y FPR son las ratios de verdaderos (T) y falsos (F) positivos.
- **AUC:** área bajo la curva (AUC - *Area Under the Curve*), mide el área bidimensional completa debajo de toda la curva ROC**.** Esta se puede interpretar como la línea divisoria entre las dos regiones que estiman la probabilidad de que el modelo clasifique un ejemplo de manera correcta o incorrecta. Este término va directamente ligado con el anterior.
- **Intervalos AUC:**

Intervalos	Interpretación del resultado
[0.5]	Aleatorio, es como lanzar una moneda.
[0.5, 0.6)	Test malo.
[0.6, 0.75)	Test regular.
[0.75, 0.9)	Test bueno.
[0.9, 0.97)	Test muy bueno.
[0.97, 1)	Test excelente.

Tabla 8.8: Interpretación de los intervalos UAC.

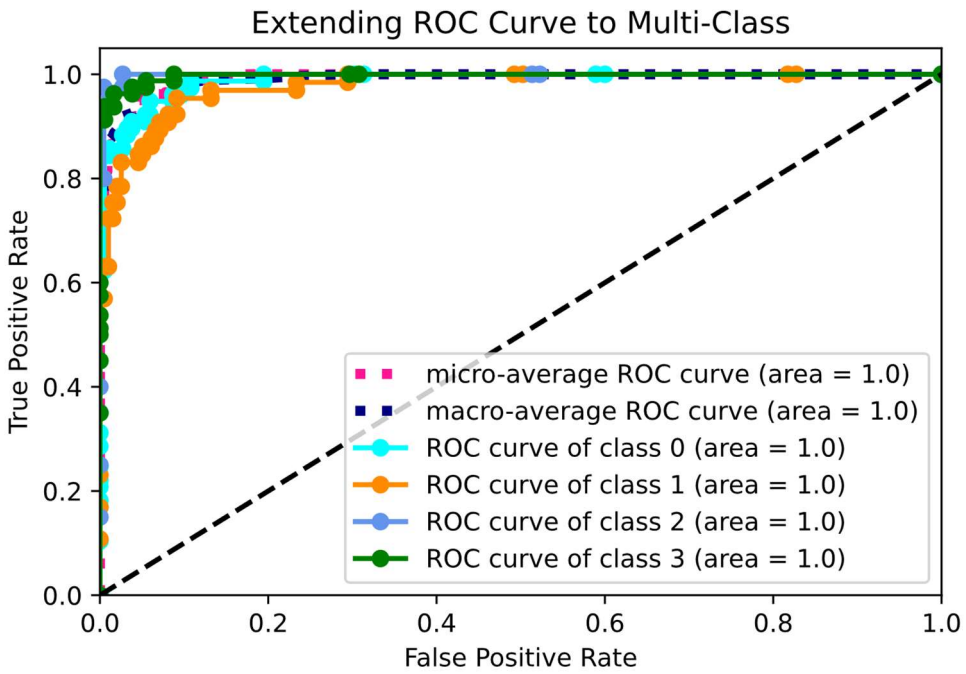

Figura 8-29: Curva ROC VGG-19 TFL.

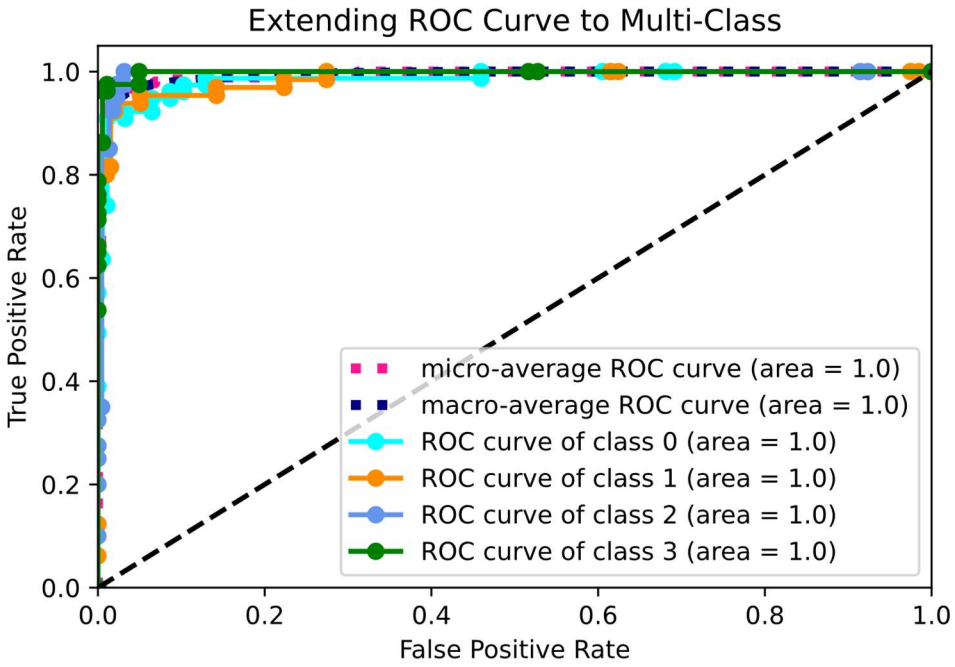

Figura 8-30: Curva ROC Xception RGB.

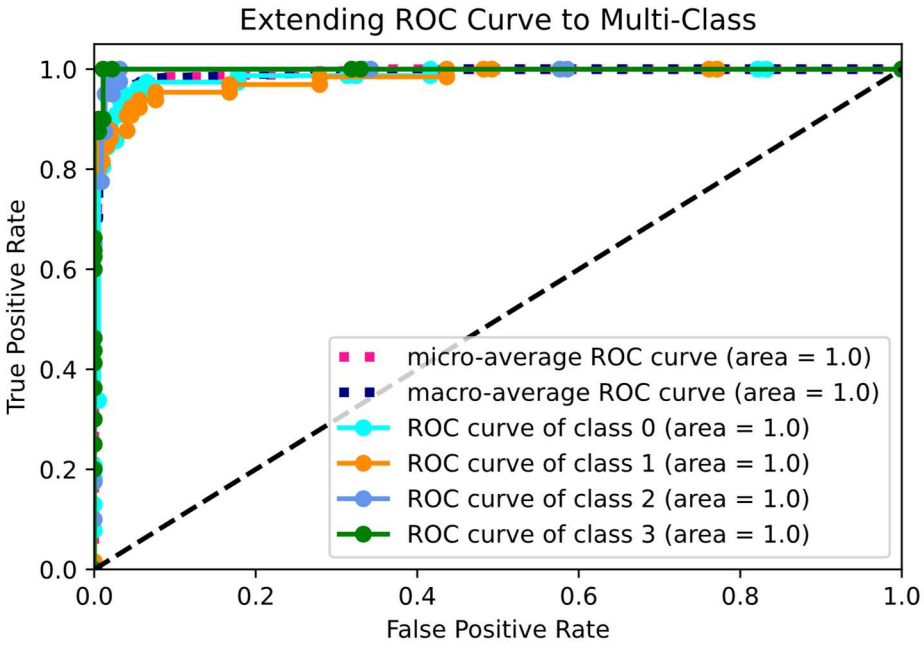

Figura 8-31: Curva ROC Custom GS.

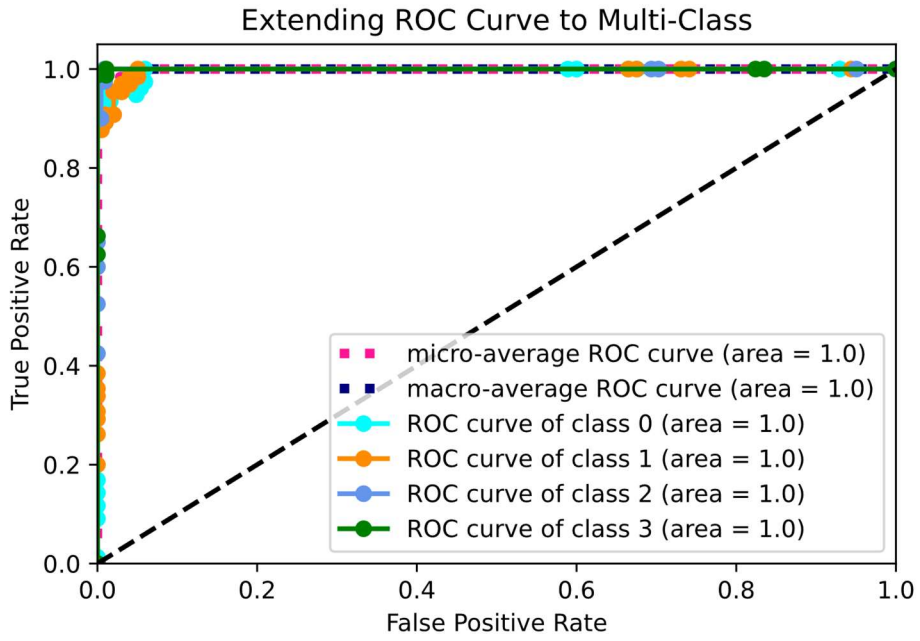

Figura 8-32: Curva ROC Xception TFL.

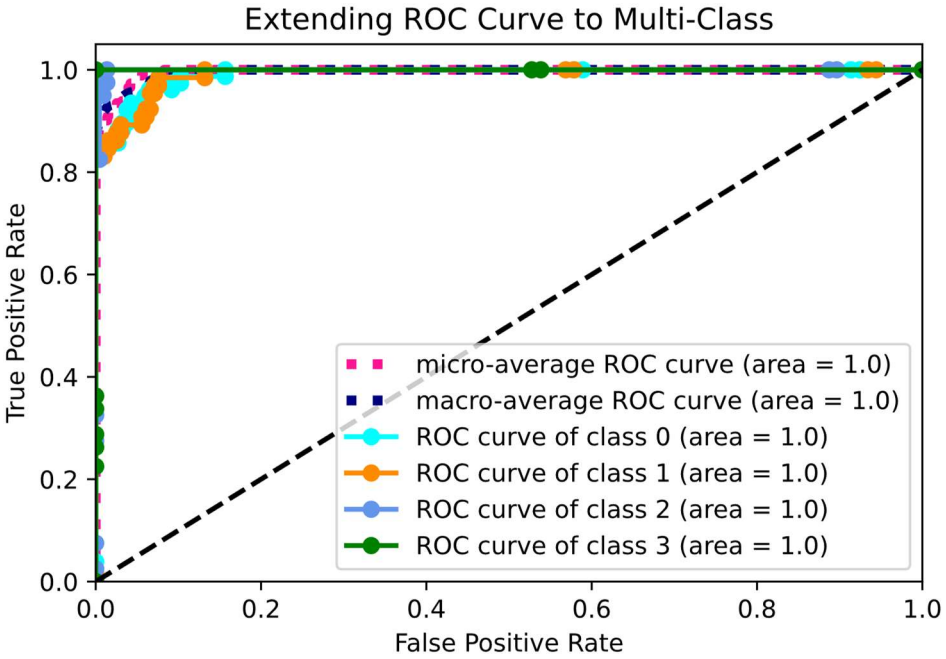

Figura 8-33: Curva ROC Xception Data Augmentation GS.

VGG-19 TFL

La curva ROC de la red VGG-19 TFL, presentada en la Figura 8-29, muestra la línea de color naranja (meningioma) como la clase con índice de TP inicial más bajo, esta se encuentra por debajo del 0.40. Próxima a esta se halla la curva correspondiente a los tumores pituitarios, representada con una línea verde, esta inicia su alzamiento en un valor próximo al 0.50. Por encima de estas dos, se encuentran la turquesa (glioma), con un valor inicial superior a 0.80 y la azul fuerte (no tumor), muy próxima al 1.00. Esta última es prácticamente una línea recta y constante, comportamiento que se estima como perfecto.

Es por estas tres primeras clases por las que existe una clara convergencia, y es la principal causante de la visible curvatura de las métricas micro y macro average. Estas curvaturas son correctas, simplemente significa que inicialmente existe un promedio TP/TN más bajo que uno, y que hay cierta resistencia a la tendencia, con el paso de las pruebas estas se normalizan. Los puntos de cortes y áreas para cada una de estas clases son:

Clases	Punto de corte	Áreas
0	Se encuentra entre 0.35 y 0.40 aproximadamente	1.00
1	Se encuentra entre 0.55 y 0.60 aproximadamente	1.00
2	Se encuentra en 0.10 Aproximadamente	1.00
3	Se encuentra entre 0.10 y 0.15 Aproximadamente	1.00

Tabla 8.9: Puntos de corte y áreas VGG-19 TFL.

Xception RGB

En la curva ROC de la Figura 8-30, esta vez la línea azul fuerte es la que tiene un menor índice de TP inicial, llegando a estar por debajo del 0.40. Seguida de esta se encuentra la turquesa que se encuentra muy próxima al 0.60, la naranja que inicia su recorrido en el 0.80 y la verde que se encuentra poco por encima de esta última. La curva ROC presenta una menor curvatura general, no obstante, todas las clases presentan algo de convergencia, a diferencia del caso anterior donde la clase representada con una línea verde era prácticamente recta.

Clases	Punto de corte	Áreas
0	Se encuentra entre 0.40 y 0.50 aproximadamente	1.00
1	Se encuentra entre 0.20 y 0.30 aproximadamente	1.00
2	Se encuentra entre 0.00 y 0.05 aproximadamente	1.00
3	Se encuentra entre 0.00 y 0.05 aproximadamente	1.00

Tabla 8.10: Puntos de corte y áreas Xception RGB.

Custom GS

Cierta mejora respecto a las anteriores curvas presenta los valores iniciales de la Curva ROC de la Figura 8-31. Esta es similar a la vista en el caso anterior, donde la clase representada como cero, comienza por debajo del 0.40, no obstante, el resto se encuentran agrupados entre 0.80 y 1.00. Cabe destacar que, a pesar de esta mejora inicial, la curva de la clase uno presenta cierta resistencia, es por ello por lo que su curva tarda en obtener el valor máximo. En cuanto a las clases dos y tres, estas toman el valor 1.00 casi desde su inicio, haciendo que estas sean prácticamente rectas.

Clases	Punto de corte	Áreas
0	Aproximadamente 0.4	1.00
1	Aproximadamente 0.4	1.00
2	Aproximadamente 0.2	1.00
3	Aproximadamente 0.0	1.00

Tabla 8.11: Puntos de corte y áreas Custom GS.

Xception TFL

La curva de la arquitectura Xception, empleando el aprendizaje por transferencia, Figura 8-32. Casi logra la absoluta perfección, todas las clases inician su recorrido entre el 0.90 y el 1.00. A pesar de no ser perfectas, estas líneas se tornan a 1.00 desde casi su inicio. Las curvas macro y micro también presentan una menor curvatura que en los casos previos.

Clases	Punto de corte	Áreas
0	Aproximadamente menos de 0.05	1.00
1	Aproximadamente menos de 0.05	1.00
2	Aproximadamente 0.05	1.00
3	Aproximadamente 0.05	1.00

Tabla 8.12: Puntos de corte y áreas Xception TFL.

Xception DA-GS

Aunque poco peor que la anterior, se encuentra muy próxima a ella, las curvas ROCs de las tres primeras clases inician con una relación TP/FP de 0.80 y tienden a 1.00 con rapidez. La última clase, que referencia a los tumores pituitarios, inicia en el 1.00 formando una línea recta perfecta, lo que significa que esta red realiza una perfecta tarea de discriminación cuando se trata de esta patología (ver Figura 8-33).

Clases	Punto de corte	Áreas
0	Aproximadamente 0.0	1.00
1	Aproximadamente 0.2	1.00
2	Aproximadamente 0.2	1.00
3	0.0	1.00

Tabla 8.13: Puntos de corte y áreas Xception DA GS.

Con la presente información propuesta en este punto, se deduce que los tumores pituitarios son los más sencillos de diagnosticar, ya que, en todos los casos, a excepción de la arquitectura VGG-19 TFL, esta, inicialmente se encuentra muy próxima al 1.00. A parte, aparentemente no existe ningún tipo adversidad que se oponga a que estas redes lo alcancen. Conforme a los resultados de estas métricas se puede afirmar que los modelos escogidos manejan perfectamente o casi perfectamente este problema. Ya que cuentan con una alta precisión, recall y f1-score.

8.4 Comparativa con otros autores

Referenciado al punto 2, los autores G. Navid y S. Deepak ofrecen en sus tesis información relativa a los resultados obtenidos por otros autores. Esta información es de gran valor ya que posibilita la realización de una comparativa con algunos de los autores nombrados en el estado del arte. Esta información se encuentra contenida en una serie de tablas que han sido adaptadas al agregar la información recopilada en esta tesis sobre los cinco modelos estudiados con anterioridad.

Método empleado	Número de imágenes	Mejor resultado en %	Empleo de segmentación	Métodos de evaluación empleados
Justin S. Paul et al. [19]	989 (solo axial)	84.52	No	5
Justin S. Paul et al. [19]	989 (solo axial)	90.26	No	5
P. Afshar et al. [20]	3064	86.56 (segmentadas) 72.13 (sin procesar)	Ambos	No se menciona
P. Afshar et al. [21]	3064	90.89	Solo cuadro delimitador	No se menciona
Phaye et al. [23]	3064	95.03	No se menciona	10
A. Pashaei et al. [23]	3064	93.68	No se menciona	Train/Val
Anaraki et al. [79]	989 (solo axial)	94.2	No	Train/Val
Abiwinanda et al. [80]	2100 (700 por cada tipo tumor)	84.19	No	Train/Val
Zhou et al. [10]	989 (solo axial)	92.13	No	Train/Val/Test
Tahir et al. [12]	3064	86.00	No	10
J. Cheng et al. [81]	3064	91.28	Sí	División por introducción
Ismael et al. [25]	3064	91.90	Sí	Train/Val
Navid Ghassemi et al. [26] **(introduced split)**	3064	93.01	No	División por introducción
Navid Ghassemi et al. [26] **(random split)**	3064	95.60	No	5
Custom RGB (propuesta)	3274	91.22	No	Train-Val-Test Acc/Loss. ROC Curve F1-Score Recall

Xception RGB (propuesta)	3274	92.74	No	Train-Val-Test Acc/Loss. ROC Curve F1-Score Recall
Custom GS (propuesta)	3274	92.37	No	Train-Val-Test Acc/Loss. ROC Curve F1-Score Recall
Xception TFL (propuesta)	3274	96.56	No	Train-Val-Test Acc/Loss. ROC Curve F1-Score Recall
Xception Data Augmentation GS (propuesta)	3274	90.48	No	Train-Val-Test Acc/Loss. ROC Curve F1-Score Recall

Tabla 8.14: Comparativa resultados generales. Fuente original: Navid G. et al. [26].

Autor y Referencia	**Precision**			**Recall**			**F1-Score**		
	M	**G**	**P**	**M**	**G**	**P**	**M**	**G**	**P**
J. Cheng et al.[81]	-	-	-	95.5	96.3	95.3	-	-	-
Ismael et al. [25]	-	-	-	96.0	96.3	95.7	-	-	-
A. Pashaei et al. [23]	94.5	91.0	98.3	-	-	-	-	-	-
S. Deepak et al. [27]	94.7	99.2	98.0	98.4	99.4	99.1	-	-	-
Navid Ghassemi (introduced split) [26]	87.6	94.1	95.0	-	-	-	86.1	94.5	95.6
Navid Ghassemi (random split) [26]	92.4	95.8	97.5	-	-	-	91.1	96.3	97.7
VGG-19 TFL (propuesta)	87.0	93.0	92.0	83.0	92.0	97.0	85.0	93.0	95.0
Xception RGB (propuesta)	95.0	90.0	94.0	92.0	92.0	97.0	94.0	91.0	96.0
Custom GS (propuesta)	96.0	86.0	95.0	82.0	96.0	100	88.0	91.0	99.0
Xception TFL (propuesta)	94.0	97.0	98.0	95.0	94.0	100	95.0	95.0	99.0
Xception Data Aug. GS (propuesta)	88.0	88.0	100	89.0	94.0	95.0	89.0	91.0	97.0

Tabla 8.15: Comparativa resultados por clases. Fuente original: S. Deepak et al. [27].

La Tabla 8.14, contiene la información relativa a los datos, configuraciones, técnicas empleadas y resultados en porcentajes, de los principales proyectos relacionados con el diagnostico de tumores cerebrales empleando técnicas de aprendizaje profundo. Estos datos contrastan y resumen la mayor parte de la información presentada en el estado del arte.

Al comparar la información relatada en esta tabla con la propuesta, que se encuentra al final de esta. Se puede visionar que los modelos, Custom RGB, Xception RGB, Custom GS y Xception TFL se encuentran al menos, en el elenco de mejores redes. Siendo únicamente superadas por Diverse Caps de Phaye et al. [24], GA + CNN de Anaraki et al. [79] y las redes GAN de Navid Ghassemi et al. [26]. En cuanto a Xception TFL, otra de las redes presentadas, y la joya de este proyecto, supera los resultados plasmados en esta cuadrícula, logrando alcanzar una precisión del 96.50%.

Por otra parte, la Tabla 8.15, contiene las precisiones, recalls y f1-scores obtenidas por estos autores, y al final las agregadas por el autor de esta tesis. Los datos de esta tabla se encuentran seccionadas en tres campos M, G y P. Estos campos o identificadores hacen referencia a los tipos de tumores diagnosticables en estos proyectos, meningiomas, gliomas y pituitarios. En este caso, el presente trabajo cuenta con una clase más, “no tumor”, por lo que, al resto no poseerlas, se ha decidido no hacer alusión a esta, en el apartado actual.

En esta agrupación de datos, se observa a simple vista, la carencia de alguna de las métricas por parte de los autores. Esto se debe a que en estos proyectos no se tuvieron en cuenta o no se especificó esta información. Es por ello por lo que aparecen algunos registros con guiones. Ciertamente, la falta de estos datos complica la comparativa, no obstante, se procederá a ello matizando únicamente en la información que se posee sobre cada uno de los trabajos.

De los presentes proyectos, solamente cuentan con la métrica **precisión** (precision) los trabajos [23], [26] y [27]. El baremo de resultados entre estos proyectos se encuentra entre los 87.60% y los 94.70%, en el caso de los meningiomas, 91.00% y 99.20% para los gliomas, y entre 95.05% y 98.30% los tumores pituitarios. En este proyecto, los resultados para esta primera patología se encuentran muy próximos. La menor precisión alcanzada entre las redes propuestas es de 87.00%, por la red VGG-19 TFL, que tan solo difiere con la introduced Split de Navid por un 0.06%. Por otra parte, el mayor resultado obtenido por las redes propuesta es 96.00%, resultado que se ha tomado directamente de la red Custom GS y es un 0.13% superior a la red de S. Deepak. En el caso de los gliomas, por lo general la tendencia es similar, los resultados vuelven a encontrarse dentro de la media establecida, salvo por Custom GS y Xception Data Augmentation GS, que quedan por debajo del 91.00%, valor obtenido por Pashaei. Por último, los tumores pituitarios, al igual que en los casos anteriores, los valores se aproximan mucho, logrando mínimos de 92.00% y máximos 100.00%, siendo este último valor, el más elevado de los resultados presentes y en sí el máximo alcanzable.

Los proyectos, [25], [27] y [81] son los únicos que extraen el **recall**, logrando en el caso del meningioma valores entre 95.50% y 98.40%, 96.30% y 99.40% por parte de los gliomas y 95.30% y 99.10% los tumores pituitarios. En este apartado se puede apreciar que los resultados logrados por las redes propuestas en este proyecto son por lo general, inferiores a las obtenidas en los proyectos nombrados, a excepción del campo que aborda los tumores pituitarios. Este último campo es de media, superior a los vistos previamente, logrando como valor mínimo un 95.00%, por parte de Xception Data Augmentation GS, dos recall’s de 97.00% por parte de las redes VGG-19 TFL y Xception RGB, y otros dos de 100.00% de tasa de recall, obtenidas por las dos redes restantes.

F1-score, solo se presenta en el proyecto [26] de Navid et al., la red que emplea el introduced split alcanza un 86.10%, mientras que la que emplea el random split logra un 91.10%, en el caso de los meningiomas, 94.50% y 96.30%, por parte de los gliomas y 95.60% y 97.70%, los tumores pituitarios. Las redes propuestas en este apartado logran superar con creces los obtenidos por otros investigadores en los campos meningiomas y tumores pituitarios, no obstante, en el campo con identificador "G" (glioma), solo una de las redes se adentra al baremo de estimación lograda por Navid et al.

8.5 Aportaciones científicas de este proyecto

A parte de los logros expresados previamente, este proyecto aporta información relevante para futuros investigadores interesados en este problema. Entre esta información se encuentran, las metodologías empleadas y sus desenvolvimientos en la resolución de este problema, redes no factibles para el problema presentado, la mejora que supone emplear el aprendizaje por transferencia, el uso de distintos procesamientos y cuál se adecua más, entre otras. Todo esto se ha presentado en este documento en forma de análisis incluyendo distintas gráficas y métricas, de forma esquematizada.

Por último, antes de concluir, se presentará una prueba de la potencia de estas tecnologías, realizando un diagnóstico sobre algunas de las imágenes del conjunto, Figura 8-34.

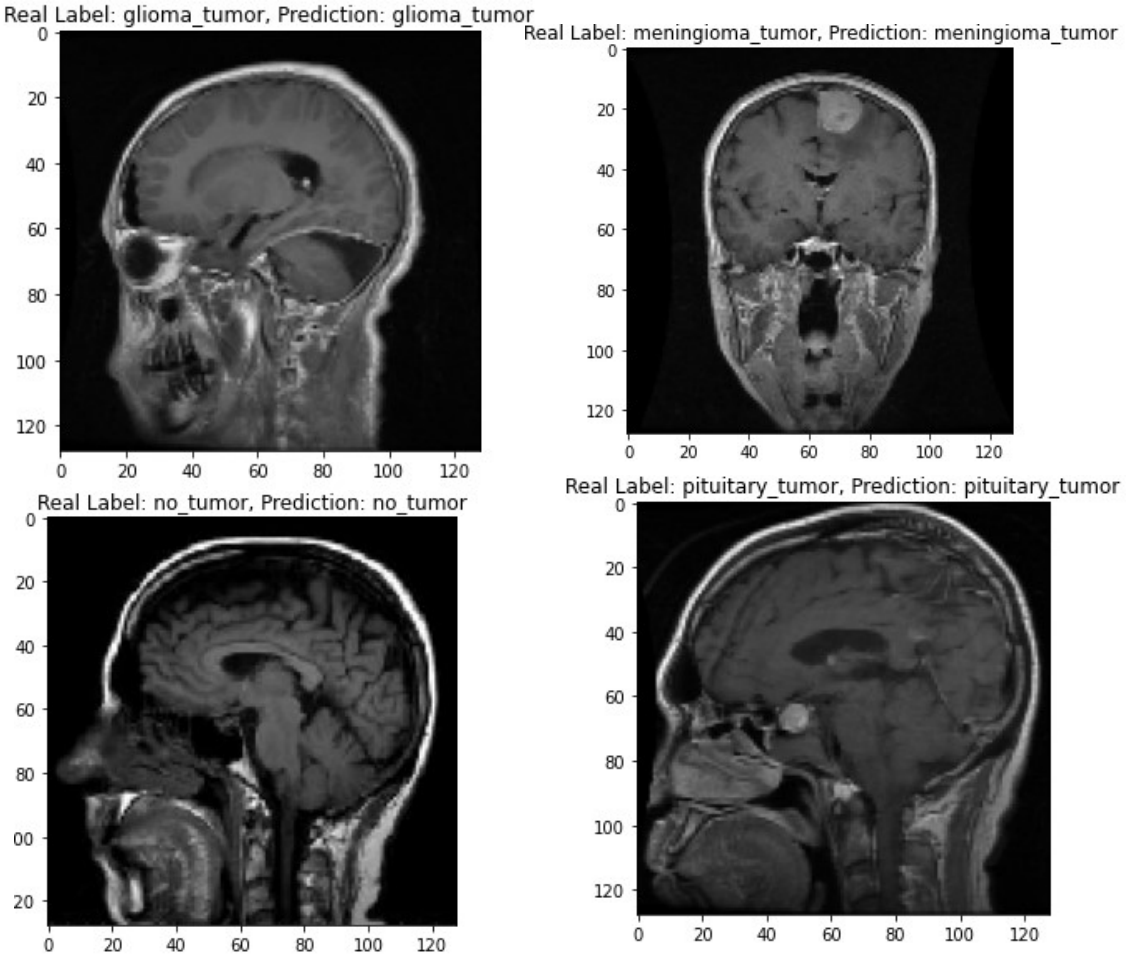

Figura 8-34: Predicciones.

Capítulo 9
Conclusiones

El presente trabajo contiene cincuenta y ocho modelos de aprendizaje profundo. De entre estos modelos, tan solo veinte son capaces de resolver el problema con más de un 85% de precisión. Aunque se debe aclarar que, a vista del autor, no todas son viables. Ejemplo de ello, los modelos de la arquitectura ResNet y ResNetV2, los cuales tienen cierta capacidad de mejora ya que logran obtener buenos resultados y tienen un aprendizaje muy prometedor. No obstante, la validación resulta muy ruidosa y preocupante. Esto junto a la gran diferencia de resultados entre modelos de una misma arquitectura, deja intranquilo e incluso con un mal sabor de boca al autor. Pues los resultados aparentan ser algo digno del azar. Comprobar el raciocinio de estos modelos empleando un lote más amplio y variado podría ser una tarea particularmente enriquecedora a nivel académico.

Algo similar sucedería con las redes VGG, que no emplearon el aprendizaje por transferencia, aparentemente son inviables, ya que no existe aprendizaje ninguno por parte de estas. No obstante, VGG es una de las arquitecturas más empleadas debido sus grandes logros y fama. Esta red se está empleando, por ende, de una manera inadecuada, ya que, como se expresó anteriormente, estas contarían con un alto número de parámetros de entrenamiento que les permitiría resolver eficientemente trabajos grandes. Pero que los condicionaría en trabajos pequeños como este. Esto fue deducido, a partir de la experiencia, al emplear el aprendizaje por transferencia se reduce a cero el número de parámetros, volviendo a esta funcionales nuevamente. Además, esta hipótesis se sustenta a su vez a partir del resto de arquitecturas empleadas, VGG es la red con mayor número de parámetros entrenables y todas a excepción de estas redes han ejecutado de forma satisfactoria la validación.

De entre los modelos presentados, los más destacados provienen de las arquitecturas Xception y customizada, ambas ofrecen resultados muy prometedores. El método que registró la mejor precisión a la hora de clasificar, en comparación con el resto de las redes de esta cosecha y de todos los trabajos relacionados o referenciados. Es la denominada como Xception Transfer Learning, esta logró una tasa de precisión (accuracy) del 96.50%. Estos resultados son extraordinarios y la red en sí puede ser empleada como punto de partida para futuras investigaciones.

Por otra parte, los procesamientos que llevaron a estas redes a su mayor punto son los procesamientos a color, tanto RGB simple, como RGB + aprendizaje por transferencia. Es por ello por lo que se debería optar directamente por estos y prescindir del procesamiento en escala de colores grises.

A pesar de los logros reportados en este documento, siguen existiendo la posibilidad de mejorar el desempeño de estas redes, una de las opciones es reduciendo el ruido de las imágenes. Para ello en futuras implementaciones, se tiene pensado crear un segmentador lineal de imágenes a partir de la arquitectura U-Net. Otras posibles actividades por realizar, que podrían enriquecer este proyecto, serían ampliar y extender el conjunto, esto con la finalidad de aumentar los casos y abarcar una mayor cantidad de patologías.

Anexo I Competencias específicas

Las **competencias** cubiertas en este proyecto fueron establecidas en el documento TFT-01, y son las siguientes:

TFG01: "Ejercicio original a realizar individualmente y presentar y defender ante un tribunal universitario, consistente en un proyecto en el ámbito de las tecnologías específicas de la Ingeniería en Informática de naturaleza profesional en el que se sinteticen e integren las competencias adquiridas en las enseñanzas".

Este proyecto académico se ha ejecutado de manera individual, y tiene como objetivos ser presentado y defendido ante un tribunal formado por docentes y académicos en la materia. Las tecnologías que se tratan en este proyecto son del ámbito tecnológico conocido comúnmente como IA, rama de la informática que se ha popularizado en los últimos años.

CP04: "Capacidad para conocer los fundamentos, paradigmas y técnicas propias de los sistemas inteligentes y analizar, diseñar y construir sistemas, servicios y aplicaciones informáticas que utilicen dichas técnicas en cualquier ámbito de aplicación".

A lo largo de este documento se ha plasmado de manera gráfica los resultados de las diferentes tecnologías introducidas al inicio de este, procesos de aprendizaje, técnicas para lidiar con el sobreajuste, diseño de modelos, entre otros.

CP05: "Capacidad para adquirir, obtener, formalizar y representar el conocimiento humano en una forma computable para la resolución de problemas mediante un sistema informático en cualquier ámbito de aplicación, particularmente los relacionados con aspectos de computación, percepción y actuación en ambientes o entornos inteligentes".

La representación del conocimiento humano es vista en los presentes modelos predictivos, los cuales han sido específicamente entrenados para desarrollar una tarea, en la que, por lo general los seres humanos necesitarían de personas especializadas y con estudios. Esto es posible a partir del reconocimiento de patrones, tarea fundamental que desempeña el cerebro humano.

CP07: "Capacidad para conocer y desarrollar técnicas de aprendizaje computacional y diseñar e implementar aplicaciones y sistemas que las utilicen, incluyendo las dedicadas a extracción automática de información y conocimiento a partir de grandes volúmenes de datos".

Se ha desarrollado de manera adicional, una aplicación web capaz de emplear las tecnologías de aprendizaje profundo que se han implementado a lo largo del proyecto con la finalidad de poner bajo perspectiva, como podría ser empleada está en un futuro. A parte a partir de una serie de funciones se ha sido capaces de representar los resultados logrados por cada uno de estos modelos.

Anexo II Problemas encontrados

Durante este periodo se ha producido una serie de incidencias. Desde fallos propios del alumno como realizar las validaciones con el lote de testeo, hasta fallos sin solución de librerías.

El primer fallo, relativo a la realización de las validaciones con el conjunto de testeo, se trata de un error por parte del autor, este pequeño error, que podría parecer insignificante, es más problemático de lo que parece. Ya que, además de estar desperdiciando una gran cantidad de imágenes, se está forzando a las redes a aprender un lote muy limitado, debido a que el conjunto de testeo es mucho más pequeño. Se ha destinado una gran cantidad de tiempo y esfuerzo en entrenamientos y validaciones, que finalmente, tuvieron que ser descartados. Este error se replicó en todos los entrenamientos debido al uso inadecuado del copiar y pegar. La solución fue cambiar el lote usado y realizar de nuevo, todos y cada uno de los entrenamientos y validaciones.

El segundo, es un fallo que el alumno ya experimentó en el pasado con otro proyecto de ámbito similar, en este caso se ha conseguido solucionarlo parcialmente. El error se produce al emplear modelos de Keras implementados en Python, en aplicaciones webs. Al internar realizar la predicción, estas se quedan "congeladas" de manera que siempre responden lo mismo, este mismo problema fue encontrado y reportado por el autor durante la realización de un proyecto grupal para la asignatura "Metodologías del Desarrollo Ágil". Donde el autor, junto a sus compañeros, se dispusieron a realizar una web de reconocimiento de flores, tras semanas sin poder hallar una solución, se vieron obligados a usar otra librería y red neuronal. Para este proyecto, se ha llegado a una solución parcial realizando un clonado del modelo por predicción, de manera que se insta a la red a reiniciarse, produciendo así predicciones más realistas, no obstante, tras unos pocos intentos el error vuelve a surgir.

Por último, se encuentra el problema de las limitaciones establecidas por Colab y Drive. Durante la realización de este proyecto se consumió los recursos otorgados por ambas herramientas en incontables ocasiones, esto provocó una ralentización en el proyecto, por otra parte, en Drive se tuvo que pagar la suscripción para poder almacenar todos los datos de los entrenamientos ya que Google Drive ofrece de manera gratuita 15GB, y este proyecto pesa más de 60GB. En cuanto a Colab, el préstamo de recursos (GPU, almacenamiento y RAM), tiene un límite, que es realmente fácil de superar con proyectos con tanta carga de trabajo como este.

Anexo III Trabajos Complementarios

En esta última etapa del proyecto, se decidió realizar una serie de tareas complementarias, es decir, tareas no contempladas en el TFT-01 y añadidos por el autor para enriquecer este trabajo.

La primera, un blog personal donde el autor explica los conceptos teóricos relacionados con el proyecto y datos de gran valor e interés para aquellos que quieran comprender o adentrarse un poco más en este campo. Estos blogs se realizaron en la plataforma web WIX [82] y su objetivo inicial era plasmar la información relevante que no pudo documentarse en esta memoria debido a las limitaciones establecidas.

Por otra parte, se desarrolló un segmentador lineal manual, cuyo funcionamiento puede ser visionado en las siguientes figuras, Figura 9-1, Figura 9-2 y Figura 9-3, esta no emplea ningún tipo de arquitectura inteligente, sino que se basa en un conjunto de funciones de procesamiento [83] de imágenes que sirven para extraer características de estas. Funciones como el desenfoque Gaussiano, con la finalidad de suavizar la imagen, un proceso de binarización donde se transforman las imágenes originales a imágenes binaria, es decir los pixeles que forman la foto pasan a ser negros o blancos (0's y 1's). Con este proceso se logra extraer una imagen más clara del cuerpo a identificar. Tras la binarización se emplean dos funciones denominadas dilatación y erosión. **Erode**, que es como se llama al método de erosión, genera erosiones sobre los límites del objeto en primer plano, eliminando así pequeñas zonas de ruido blanco. Por otra parte, **Dilate**, la función dilatadora, realiza un trabajo contrario al explicando anteriormente. Este sirve para acentuar las características del cuerpo. Este segmentador, al igual que el resto de las funciones de análisis, se encuentra en el fichero tfg-analitycs.ipynb, y su código se encuentra relatado en el apartado "Segmentador lineal manual", del blog personal.

Por último, se ha creado una web básica en Flask, cuyo back-end implementa una de las redes neuronales de este proyecto. Esta tarea se realizó con la finalidad de mostrar una aplicación real del uso de estas herramientas. Esta web se intentó implementar de múltiples formas sin logro alguno, hasta llegar a la versión actual, como se muestra en la Figura 9-4. Para ese ejemplo, se empleó como información entrante un MRI que contenía un tumor de tipo meningioma. La red predijo correctamente con un 98.37% de firmeza, es decir, está muy seguro de que lo que está "viendo" es un meningioma.

Para realizar esta web se ha hecho uso del microframework Flask, la librería Keras para cargar el modelo, procesar las imágenes y realizar las predicciones, la librería Numpy para redimensionar la matriz de pixeles y las librerías os y skimage para tomar las imágenes y realizar un redimensionamiento final que sirve para plasmar la imagen en la web de manera correcta. El entorno designado para esta tarea fue Anaconda, ya que brinda soporte y facilidades a la hora de instalar e importar las librerías. La información mostrada en la página se carga de forma dinámica desde el propio back-end. Esta información será volcada a los archivos ".html" que forman el front-end y cuenta con tres ficheros ".html" y un estilo totalmente basado en Bootstrap.

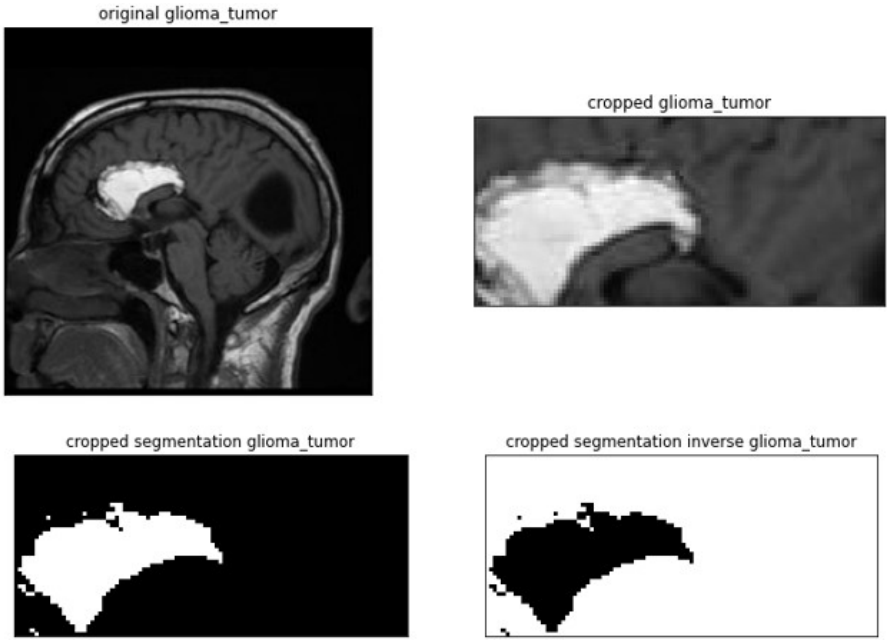

Figura 9-1: Segmentador lineal manual (Glioma).

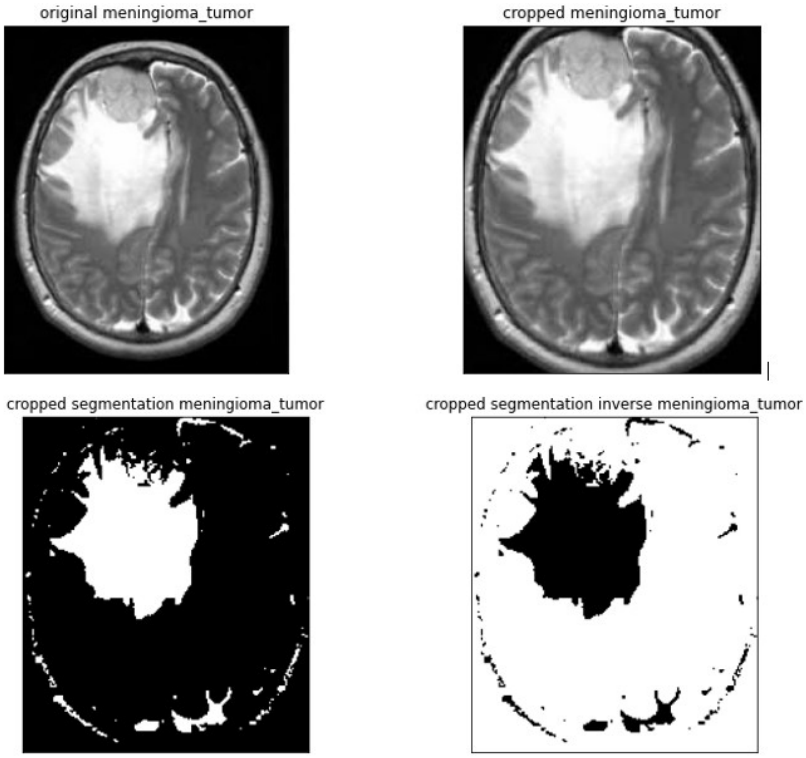

Figura 9-2: Segmentador lineal (Meningioma)

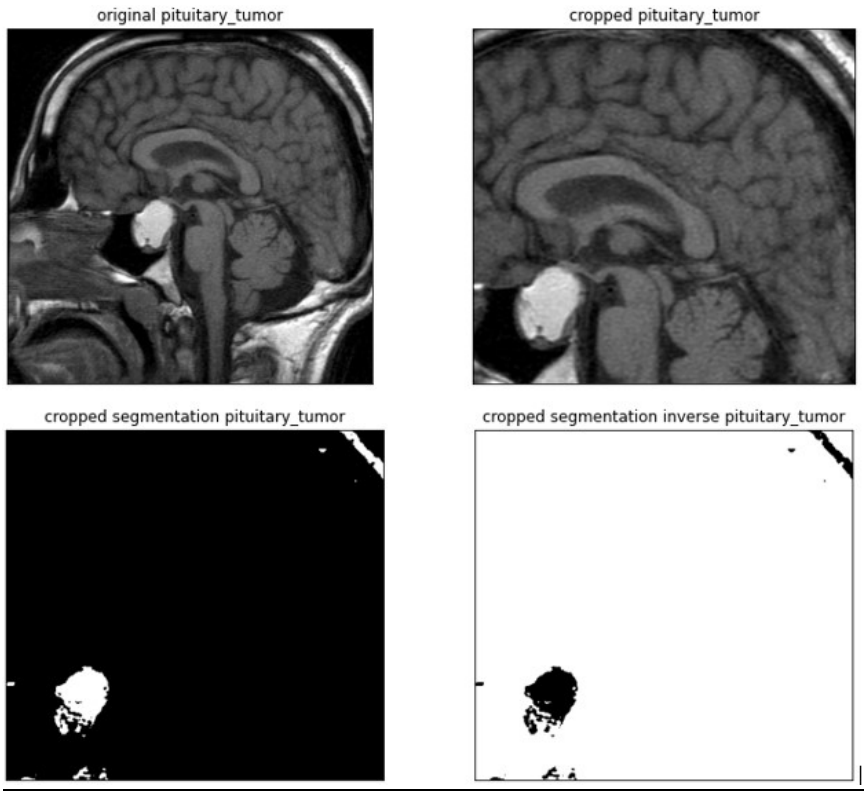

Figura 9-3: Segmentador lineal (Tumor pituitario).

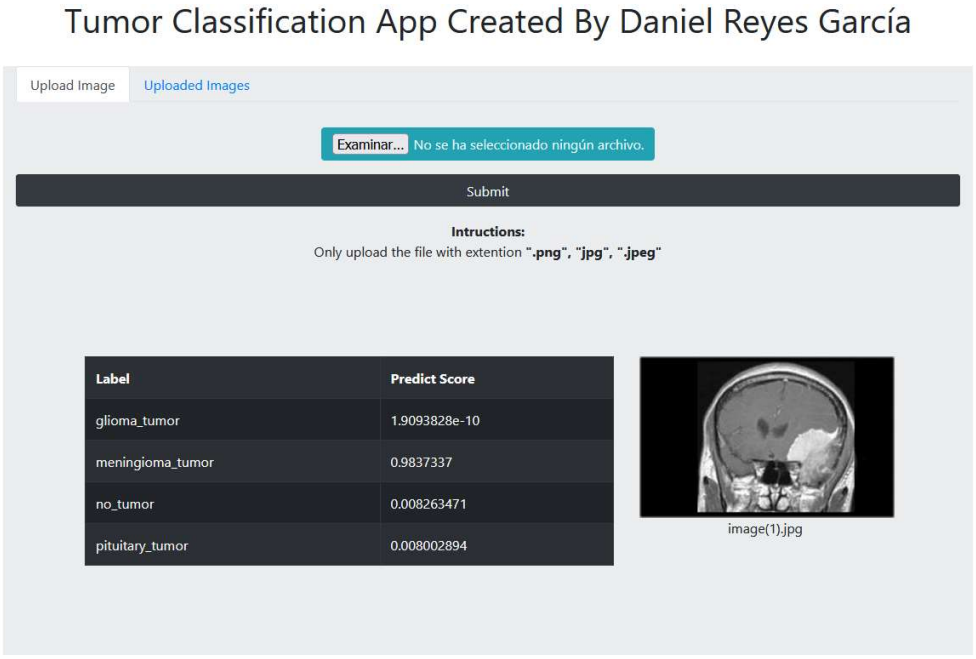

Figura 9-4: Web clasificadora de tumores.

Bibliografía

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li y L. Fei-FeiIEEE, «ImageNet: A large-scale hierarchical image database,» *2009 IEEE Conference on Computer Vision and Pattern Recognition,* pp. 248-255, 18 Agosto 2009.

[2] American Cancer Society, «Tasas de supervivencia de ciertos tumores de encéfalo y tumores de médula espinal en adultos,» American Cancer Society, 2020.

[3] E. Evain, C. Raynaud, C. Ciofolo-Veit, A. Popoff, T. Caramella, P. Kbaier, C. Balleyguier, S. Harguem-Zayani, H. Dapvril, L. Ceugnart, M. Monroc, F. Chamming's, I. Doutriaux-Dumoulin, I. Thomassin-Naggara, A. Haquin, M. Audrey, J. Orabona, T. Fourquet, I. Bousaid, N. Lassau y A. Olivier, «Breast nodule classification with two-dimensional ultrasound using Mask-RCNN ensemble aggregation,» Diagnostic and Interventional Imaging, vol. 102, nº 11, pp. 653-658, 01 Noviembre 2021.

[4] G. Fazio, F. Galioto, A. Ferlito, M. Coronella, S. Palmucci y A. Basile, «Cavitated pulmonary nodules in a female patient with breast cancer: Keep in mind Serratia marcescens' infections,» *Respiratory Medicine Case Reports,* vol. 33, p. 101441, 01 Enero 2021.

[5] J. Parashar, Sumiti y M. Rai, «Breast cancer images classification by clustering of ROI and mapping of features by CNN with XGBOOST learning,» *Materials Today: Proceedings,* vol. 55, nº 2, 19 Noviembre 2020.

[6] M. Both, J. Schultze, M. Reuter, B. Bewig, R. Hubner, I. Bobis, R. Noth, M. Heller y J. Biederer, «Fast T1- and T2-weighted pulmonary MR-imaging in patients with bronchial carcinoma,» *European Journal of Radiology,* vol. 53, nº 3, pp. 478-488, 01 Marzo 2005.

[7] S. Haggenmüller, R. C. Maron, A. Hekler, J. S. Utikal, C. Barata, R. L. Barnhill y H. Beltraminelli, «Skin cancer classification via convolutional neural networks: systematic review of studies involving human experts,» vol. 152, pp. 202-216, 08 Septiembre 2021.

[8] P. K. Gupta, M. K. Siddiqui, R. Morales-Menendez y V. Singh, «Application of deep learning for fast detection of COVID-19 in X-Rays using nCOVnet,» *Chaos, Solitons & Fractals,* vol. 138, p. 109944, 01 Septiembre 2020.

[9] A. R. Khan, L. Wang y M. F. Beg, «FreeSurfer-initiated fully-automated subcortical brain segmentation in MRI using Large Deformation Diffeomorphic Metric Mapping,» *NeuroImage,* vol. 41, nº 3, pp. 735-746, 01 Julio 2008.

[10] D. Zhang, G. Huang, Q. Zhang, J. Han, J. Han y Y. Yu, «Cross-modality deep feature learning for brain tumor segmentation,» *Pattern Recognition,* vol. 110, nº 0031-3203, p. 107562, 01 Febrero 2021.

[11] M. Havaei, A. Davy, D. Warde-Farley, A. Biard, A. Courville, Y. Bengio, C. Pal, P.-M. Jodoin y H. Larochelle, «Brain tumor segmentation with Deep Neural Networks,» *Medical Image Analysis,* vol. 35, pp. 18-31, 01 Enero 2017.

[12] S. I. Bilal Tahir, M. U. G. Khan, T. Saba, Z. Mehmood, A. Anjum y T. Mahmood, «Feature enhancement framework for brain tumor segmentation and classification,» *Microscopy Research and Technique,* vol. 82, nº 6, pp. 803-811, 15 February 2019.

[13] V. Rodríguez, «Brain Tumor segmentation with U-Net,» vincentblog, 13 Junio 2019.

[14] Y. Zhang, H. Liu y Q. Hu, «TransFuse: Fusing Transformers and CNNs for Medical Image Segmentation,» *CoRR,* vol. abs/2102.08005, nº 2, pp. 1-11, 16 Febrero 2021.

[15] S. M. Smith, «Fast robust automated brain extraction,» *Human Brain Mapping,* vol. 17, nº 3, pp. 143-155, 18 Septiembre 2002.

[16] S. M. Smith, «BET: Brain extraction tool,» *FMRIB TR00SMS2b, Oxford Centre for Functional Magnetic Resonance Imaging of the Brain), Department of Clinical Neurology, Oxford University, John Radcliffe Hospital, Headington, UK,* p. 25, 2000.

[17] M. Modat, D. M. Cash, P. Daga, G. P. Winston, J. S. Duncan y S. Ourselin, «Global image registration using a symmetric block-matching approach,» *Journal of Medical Imaging,* vol. 1, nº 2, pp. 1-6, 19 Septiembte 2014.

[18] B. Fisch, «FreeSurfer,» *NeuroImage,* vol. 62, nº 2, pp. 774-781, 15 Agosto 2012.

[19] J. S. Paul, A. J. Plassard, B. A. Landman y D. Fabbri, «Deep learning for brain tumor classification,» *Medical Imaging 2017: Biomedical Applications in Molecular, Structural, and Functional Imaging,* vol. 10137, pp. 253-268, 13 Marzo 2017.

[20] P. Afshar, A. Mohammadi y K. N. Plataniotis, «Brain Tumor Type Classification via Capsule Networks,» *2018 25th IEEE International Conference on Image Processing (ICIP),* nº 2381-8549, pp. 3129-3133, 06 Septiembre 2018.

[21] P. Afshar, K. N. Plataniotis y A. Mohammadi, «Capsule Networks for Brain Tumor Classification Based on MRI Images and Coarse Tumor Boundaries,» *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP),* nº 2379-190X, pp. 1368-1372, 17 Abil 2019.

[22] Y. Yang, L.-F. Yan, X. Zhang, Y. Han, H.-Y. Nan, Y.-C. Hu, B. Hu, S.-L. Yan, J. Zhang, D.-L. Cheng, X.-W. Ge, G.-B. Cui, D. Zhao y W. Wang, «Glioma Grading on Conventional MR Images: A Deep Learning Study With Transfer Learning,» *Frontiers in Neuroscience,* vol. 12, pp. 1-10, 15 Noviembre 2018.

[23] A. Pashaei, H. Sajedi y N. Jazayeri, «Brain Tumor Classification via Convolutional Neural Network and Extreme Learning Machines,» *IEEE Xplore,* pp. 314-319, 25-26 Octubre 2018.

[24] S. S. R. Phaye, A. Sikka, A. Dhall y D. Bathula, «Dense and Diverse Capsule Networks: Making the Capsules Learn Better,» *CoRR,* 13 Agosto 2018.

[25] M. R. Ismael y I. Abdel-Qader, «Brain Tumor Classification via Statistical Features and Back-Propagation Neural Network,» *2018 IEEE International Conference on Electro/Information Technology (EIT),* pp. 252-257, 22 Octubre 2018.

[26] N. Ghassemi, A. Shoeibi y M. Rouhani, «Deep neural network with generative adversarial networks pre-training for brain tumor classification based on MR images,» *Biomedical Signal Processing and Control,* vol. 57, p. 101678, 01 Marzo 2020.

[27] S.Deepak y P.M.Ameer, «Brain tumor classification using deep CNN features via transfer learning,» *Computers in Biology and Medicine,* vol. 111, p. 103345, 01 Octubre 2019.

[28] Keras, «Keras Applications,» Keras Documentation.

[29] K. Simonyan y A. Zisserman, «Very Deep Convolutional Networks for Large-Scale Image Recognition,» nº 6, pp. 1-14, 04 Septiembre 2014.

[30] H. Kaiming, X. Zhang, S. Ren y J. Sun, «Deep Residual Learning for Image Recognition,» vol. abs/1512.03385, 2015.

[31] «La diferencia entre Resnet V1 y V2,» programador clic, 2020.

[32] C. François, «Xception: Deep Learning with Depthwise Separable Convolutions,» *CoRR,* vol. abs/1610.02357, nº 3, 07 Octubre 2016.

[33] Saúde, «Há ligação entre tumor benigno e hematológico?,» revista ABRALE ON-LINE, 21 Febrero 2020.

[34] «Glioma,» Wikipedia, 2020.

[35] «Meningioma,» Wikipedia, 2019.

[36] «Pituitary Tumors,» Johns Hopkins Medicine.

[37] P. d. M. Clinic, «Tumores pituitarios,» Mayo Clinic, 30 Octubre 2021.

[38] ACUNSA, «Resonancia magnética, ¿cómo funciona y para qué sirve?,» Noticias ACUNSA, 2021.

[39] R. R. Edelman, «The History of MR Imaging as Seen through the Pages of Radiology,» *Radiology,* vol. 273, nº 2, pp. 181-200, 23 Octubre 2014.

[40] AHS Alumni Foundation, «Dr. Herman Y. Carr,» AHS Alumni Foundation, 2014.

[41] «Vladislav Ivanov (physicist),» Wikipedia, 2022.

[42] P. Xavier, «Resonancia Magnética \n Resonancia Magnética Nuclear (RMN),» pardell, 2016.

[43] «Python (programming language),» Wikipedia, 2019.

[44] «NumPy,» Wikipedia, 2019.

[45] NumPy, «NumPy,» NumPy.org, 2009.

[46] «TensorFlow,» Wikipedia, 2019.

[47] Tensorflow, «Machine learning educations,» TensorFlow.com.

[48] Keras, «Home - Keras Documentation,» Keras, 2019.

[49] A. Gupta, «Some Amazing Applications of OpenCV Library,» Analytics Vidhya, 2021.

[50] J. D. Hunter, «Matplotlib: A 2D graphics environment,» *Computing in Science & Engineering,* vol. 9, nº 3, p. 90-95, 18 Junio 2007.

[51] «scikit-learn,» Wikipedia, 2019.

[52] «Colab,» Wikipedia, 2022.

[53] «Anaconda (Python distribution),» Wikipedia, 2019.

[54] «PyCharm,» Wikipedia, 2020.

[55] «Google Drive,» Wikipedia, 2019.

[56] «Flask (web framework),» Wikipedia, 2019.

[57] A. Goldbloom, «Kaggle,» Kaggle and Google LLC, Abril 2010.

[58] S. Bhuvaji, «Brain Tumor Classification (MRI),» Kaggle, 2020.

[59] N. Chakrabarty, «Brain MRI Images for Brain Tumor Detection,» Kaggle, 2020.

[60] M. David C Preston, «Magnetic Resonance Imaging (MRI) of the Brain and Spine: Basics,» case, 07 Abril 2016.

[61] «MRI sequence,» Wikipedia, 2022.

[62] «Spin echo,» Wikipedia, 2020.

[63] J. Cheng, «brain tumor dataset,» 04 Febrero 2017.

[64] M. Karelia, «Brain Tumor Detection using CNN,» Kaggle, 2021.

[65] D. Jaywant, «Brain-Tumor-Classification,» Kaggle, 2021.

[66] A. Sandesh, «sujeeth's project,» Kaggle, 2021.

[67] M. Adewole, «Brain Tumor Classifier,» Kaggle, Noviembre 2021.

[68] Y. Soylu, «Brain Tumor Classification,» Kaggle, 2021.

[69] A. Ajagekar, «Adam,» Cornell University Computational Optimization Open Textbook - Optimization Wikipedia, 2021.

[70] V. Bushaev, «Adam — latest trends in deep learning optimization.,» Towards Data Science, 22 Octubre 2018.

[71] A. Kathuria, «Intro to optimization in deep learning: Momentum, RMSProp and Adam.,» PaperspaceBlog, 2018.

[72] Peltarion, «Categorical crossentropy,» Peltarion, 2022.

[73] K. Transfer, «How to choose cross-entropy loss function in Keras?,» Knowledge Transfer, 22 Mayo 2021.

[74] «Curva ROC,» Wikipedia, 2009.

[75] «Precision and recall,» Wikipedia, 2022.

[76] E. SOLUTIONS, «Accuracy, Precision, Recall & F1 Score: Interpretation of Performance Measures,» EXSILIO SOLUTIONS, 09 Septiembre 2016.

[77] Google, «Classification: ROC Curve and AUC,» Google - Machine Learning Crash Course, 2019.

[78] K. Transfer, «Micro and Macro Averages for imbalance multiclass classification,» Knowledge Transfer, 10 Julio 2021.

[79] A. Kabir Anaraki, M. Ayati y F. Kazemi, «Magnetic resonance imaging-based brain tumor grades classification and grading via convolutional neural networks and genetic algorithms,» *Biocybernetics and Biomedical Engineering,* vol. 39, nº 1, pp. 63-74, 01 Enero 2019.

[80] N. Abiwinanda, M. Hanif, S. Hesaputra, A. Handayani y T. Mengko, «Brain Tumor Classification Using Convolutional Neural Network,» *World Congress on Medical Physics and Biomedical Engineering 2018. IFMBE Proceedings,* vol. 68, nº 1, pp. 183-189, 30 Mayo 2018.

[81] C. J., H. W., C. S., Y. R., Y. W. y e. al., «Enhanced Performance of Brain Tumor Classification via Tumor Region Augmentation and Partition,» *PLOS ONE,* vol. 10, nº 10, pp. 1-13, 10 Agosto 2015.

[82] «Wix,» Wix.com, 2016.

[83] A. Rosebrock, «OpenCV Thresholding (cv2.threshold),» pyimagesearch, 28 Abril 2021.

[84] A. Rosebrock, «ImageNet: VGGNet, ResNet, Inception, and Xception with Keras,» pyimagesearch, 20 Marzo 2017.


ULPGC
Universidad de
Las Palmas de
Gran Canaria
Escuela de
Ingeniería
Informática