Universitat Oberta de Catalunya

uoc.edu

Detección y segmentación automática de lesiones en pacientes con esclerosis múltiple en imágenes de resonancia magnética

José Carlos García Pérez
Máster Universitario en Ciencia de Datos
Minería de datos y machine learning

Eloy Martínez de las Heras
Jordi Casas Roma

09/06/2019

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Detección y segmentación automática de lesiones en pacientes con esclerosis múltiple en imágenes de resonancia magnética*
Nombre del autor:	*José Carlos García Pérez*
Nombre del consultor/a:	*Eloy Martínez de las Heras*
Nombre del PRA:	*Jordi Casas Roma*
Fecha de entrega (mm/aaaa):	06/2019
Titulación:	*Máster Universitario en Ciencia de Datos*
Área del Trabajo Final:	*Minería de datos y machine learning*
Idioma del trabajo:	***Español***
Palabras clave:	*Redes neuronales convolucionales, deep learning, segmentación de lesiones cerebrales*

Resumen del Trabajo:

La esclerosis múltiple es una enfermedad crónica neurodegenerativa del sistema nervioso central. Las imágenes de resonancia magnética permiten visualizar y detectar lesiones cerebrales en pacientes de esta enfermedad, por lo que hacer un seguimiento de las mismas es muy importante tanto para el diagnóstico y monitorización de la enfermedad, como para evaluar el efecto del tratamiento.

Las redes neuronales convolucionales se han posicionado como una técnica muy prometedora para identificar y segmentar automáticamente estas lesiones. Este trabajo pretende diseñar e implementar una arquitectura de red neuronal convolucional para detectar y segmentar de forma eficiente las lesiones cerebrales en imágenes de resonancia magnética, realizando un ajuste de los hiperparámetros o modificando el *pipeline* propuesto en otras arquitecturas ya existentes.

Se evaluará el resultado del proceso según el coeficiente de similaridad de Dice con datos propios de pacientes propocionados por el Instituto de Investigaciones Biomédicas August Pi i Sunyer (IDIBAPS), lo cual permitirá cuantificar la precisión y la reproducibilidad del modelo.

Abstract:

Multiple sclerosis is a chronic neurodegenerative disease which affects the central nervous system. Magnetic resonance images allow the visualization and detection of brain lesions in patients with this disease, therefore monitoring such lesions is very important for both the diagnosis and monitoring of the disease, and to evaluate the effect of the treatment.

Convolutional neural networks are currently a very promising technique to identify and automatically segment these lesions. This work aims to design and implement a convolutional neural network architecture to efficiently detect and segment brain lesions in magnetic resonance images, fine-tuning the hyperparameters or modifying the proposed pipeline in other existing architectures.

The result of the process will be evaluated according to the Dice similarity coefficient on private data provided by the August Pi i Sunyer Biomedical Research Institute (IDIBAPS), which will allow to assess the accuracy and reproducibility of the model.

A Magda, Andrea y Marina, hacéis que
las cosas merezcan la pena

Agradecimientos

Mi más sincero agradecimiento a mi familia, por la paciencia y el apoyo recibido en todo momento.

A mi tutor de proyecto, el Dr. Eloy Martínez de las Heras, por su disponibilidad, cercanía y orientación a lo largo del desarrollo de este trabajo.

A la Dra. Sara Llufriu y al Instituto de Investigaciones Biomédicas August Pi i Sunyer, por poner a mi alcance una línea de trabajo apasionante.

A todos los investigadores que, con su esfuerzo y dedicación, luchan día a día para mejorar la vida de las personas.

Y por supuesto, a todos los pacientes.

Muchísimas gracias a todos.

Índice

Lista de figuras

Lista de tablas

1. Introducción

1.1 Contexto y justificación del trabajo

La esclerosis múltiple es una enfermedad crónica neurodegenerativa del sistema nervioso central y es una de las causas más importantes de discapacidad tanto física como cognitiva en adultos jóvenes (Esclerosis Múltiple España, n.d.). Los pacientes que la padecen presentan lesiones visibles en imágenes de resonancia magnética y su detección y seguimiento son muy importantes para el diagnóstico y monitorización de la enfermedad, así como para evaluar el efecto del tratamiento.

“Estas lesiones muestran una predisposición a localizarse en determinadas zonas anatómicas y a presentar una mayor carga lesional en regiones frontales y parietales. Para las lesiones hiperintensas visibles en T2, la presencia de la lesión es independiente al sustrato patológico o a la fase evolutiva. En cambio, las lesiones en T1 reflejan un sustrato patológico diferente en función si la lesión es crónica o activa”. (Martínez de las Heras, 2017, p. 53) (Figura 1).

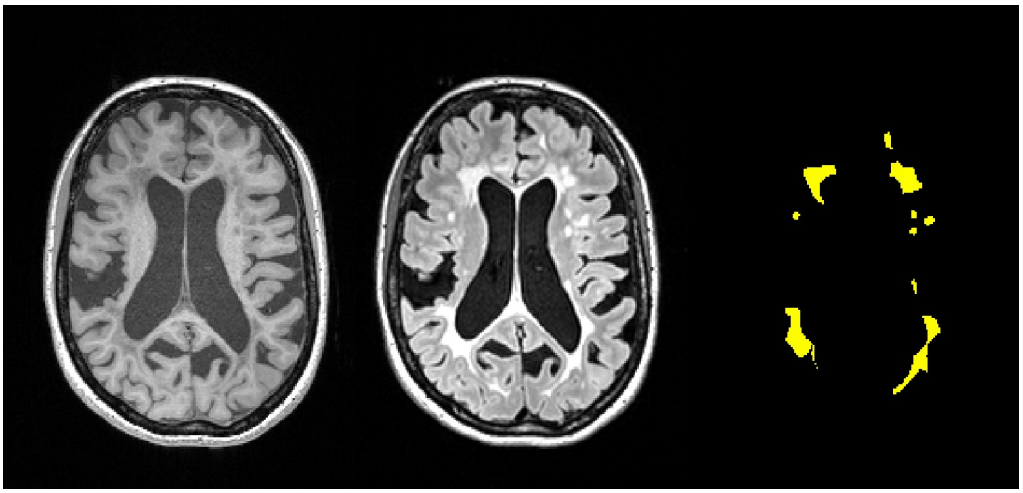

Figura 1: Tipo de secuencias usadas para la detección visual de lesiones por medio de resonancia magnética (T1 y FLAIR)

La detección de estas lesiones por parte de un especialista es un proceso lento y muy complejo. Sin embargo, gracias a los avances en las técnicas de aprendizaje automático, es posible identificar automáticamente la presencia de estas lesiones. En particular, las redes

neuronales convolucionales (*convolutional neural networks, CNN*) resultan muy prometedoras para la segmentación de imágenes, siendo el objetivo de este trabajo diseñar e implementar una arquitectura de red neuronal convolucional que permita detectar de forma eficiente las lesiones cerebrales en imágenes de resonancia magnética.

El interés en esta línea de proyecto viene por la posibilidad de aplicar la potencia de las técnicas de aprendizaje automático, en particular las redes neuronales convolucionales, al ámbito médico, ya que cualquier aporte en este campo, por pequeño que sea, puede resultar muy relevante tanto para mejorar el diagnóstico como monitorización de enfermedades, así como para evaluar el efecto de los tratamientos. El uso de estas tecnologías podrían, en definitiva, ayudar a mejorar la vida de las personas, lo cual resulta bastante motivador.

1.2 Objetivos del trabajo

El objetivo principal de este trabajo es diseñar e implementar una arquitectura de red neuronal convolucional que permita la detección de forma eficiente de lesiones cerebrales en imágenes de resonancia magnética de pacientes con esclerosis múltiple.

Este planteamiento lleva implícito otros objetivos, como el ajuste de hiperparámetros de otras arquitecturas ya propuestas o el uso de tareas de pre o postprocesamiento. El objetivo principal es evaluar el rendimiento realizando diferentes configuraciones de la red, hasta llegar a un conjunto de hiperparámetros que permitan definir una arquitectura que resuelva el problema de forma eficiente.

1.3 Enfoque y método seguido

Este trabajo se abordará principalmente siguiendo un enfoque iterativo, ya que dada la naturaleza del problema, habrá que volver a etapas anteriores si el resultado no es satisfactorio. Esta aproximación permitirá ir refinando y ajustando el trabajo hasta conseguir un resultado aceptable.

Es importante resaltar que el proyecto, en su conjunto, se encuadra en un proceso secuencial que consta de las siguientes fases:

1. **Análisis**

   Etapa en la que se analizará cómo se aborda actualmente el problema de la segmentación de imágenes de resonancia magnética, en particular en el caso de la detección de lesiones cerebrales.

2. **Diseño, implementación y evaluación**

   Esta etapa del proyecto es iterativa. Dado el objetivo que se persigue en este trabajo, si el resultado de la evaluación no es satisfactorio podría implicar un rediseño o ajustar los parámetros y configuraciones de la arquitectura de la red implementada. Esta etapa incluye las siguientes subtareas:

   - Obtención de los datos
   - Preparación de los datos: limpieza, transformación, etc.
   - Generación y entrenamiento del modelo
   - Evaluación de la máscara de lesión por medio de un "*gold standard*" (índice Dice)

3. **Documentación**

   Como paso final, debe generarse un documento que explique todos los aspectos del proyecto, incluyendo todos los pasos anteriores, además de los resultados, conclusiones y posibles líneas de trabajo futuras.

En el siguiente apartado se propone una planificación de acuerdo con este enfoque.

1.4 Planificación del trabajo

La siguiente planificación refleja las tareas del apartado anterior. Se han propuesto cuatro iteraciones dentro de la etapa de diseño, implementación y evaluación, a título orientativo. Al final de cada etapa se produce un entregable, bien en forma de documento o código fuente para reproducir el resultado obtenido.

Las fechas previstas de inicio y fin por cada etapa son las siguientes:

1. Análisis
    - Inicio: 04/03/19
    - Fin: 24/03/19
2. Diseño, implementación y evaluación
    - Inicio: 25/03/19
    - Fin: 19/05/19
3. Documentación
    - Inicio: 20/05/19
    - Fin: 09/06/19

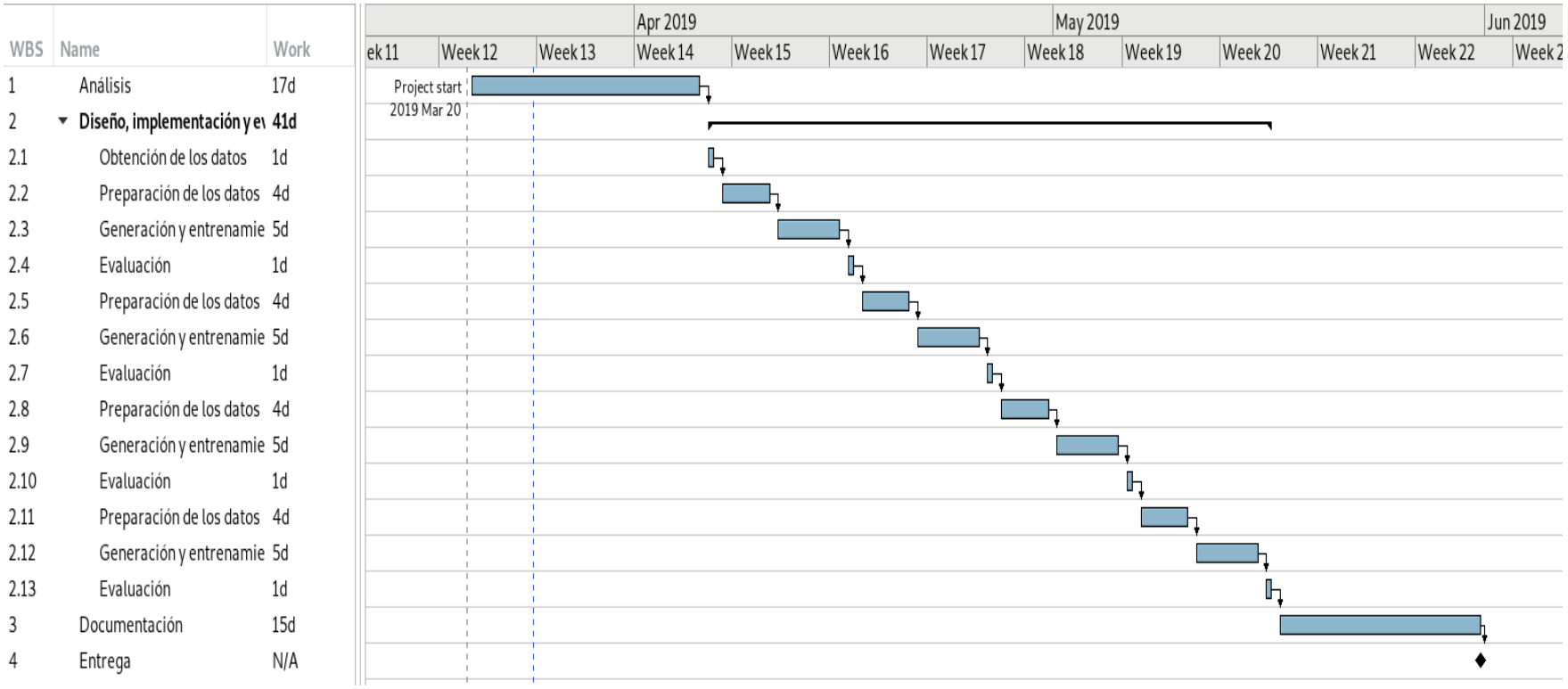

Figura 2: Planificación del trabajo

1.5 Breve sumario de productos obtenidos

Este trabajo se ha basado en la arquitectura FLEXCONN, propuesta por (Roy et al. 2018), sobre la cual se han evaluado diferentes funciones de coste, así como diferentes tamaños de parche para intentar mejorar el resultado de la segmentación.

Se han añadido módulos *inception* a la arquitectura original y se han aplicado mejoras mediante el uso de *callbacks* e incluyendo nuevas tareas de postprocesamiento, como la aplicación de una máscara con la región de interés para descartar los falsos positivos que se presentan fuera del parénquima cerebral.

Por lo tanto, como resultado de este trabajo se ha obtenido, por un lado, una evaluación exhaustiva de diferentes funciones de coste así como de algunos hiperparámetros que afectan a la calidad de la segmentación, y por otro, una modificación sobre la arquitectura FLEXCONN más una etapa adicional de postprocesamiento que permite reducir el número de falsos positivos.

1.6 Breve descripción de los otros capítulos de la memoria

La memoria está organizada en los siguientes capítulos, cuyo contenido se resume a continuación:

- **Capítulo 2. Estado del arte.** En este capítulo se repasan las técnicas más empleadas y que mejores resultados obtienen en la actualidad para realizar el proceso de segmentación, abarcando las etapas de preprocesamiento, segmentación automática, evaluación y postprocesamiento.

- **Capítulo 3. Metodología.** En este capítulo se presenta el entorno tecnológico usado, tanto hardware como software, así como las limitaciones encontradas.

  Igualmente, se definen las diferentes funciones de coste usadas en este trabajo para generar distintos modelos, así como los experimentos realizados con diferentes arquitecturas de redes neuronales convolucionales.

  Por último, se proponen algunas mejoras que podrían aumentar la precisión de la segmentación y obtener un mayor rendimiento.

- **Capítulo 4. Conclusiones y líneas de trabajo futuro.** Presenta las principales conclusiones obtenidas durante la elaboración de este trabajo, así como posibles líneas de trabajo futuro que pueden resultar prometedoras para mejorar los resultados conseguidos en este trabajo.

- **Capítulo 5. Glosario.** Glosario de términos utilizados a lo largo de este trabajo.

- **Capítulo 6. Bibliografía.** Presenta las referencias científicas de otros autores usadas para elaborar este trabajo.

- **Capítulo 7. Anexos.** Finalmente, se incluyen los resultados de los experimentos realizados con cada uno de los modelos generados, presentando la calidad de la segmentación tanto de manera individual como en promedio, según el coeficiente de Dice.

2. Estado del arte

El análisis de volumetría a partir de imágenes estructurales por resonancia magnética es una tarea crucial en el diagnóstico y monitorización de la esclerosis múltiple. La propia complejidad de las imágenes, así como la presencia de ruido y las diferencias de contraste e intensidad, hacen que las herramientas de segmentación presenten ciertas dificultades metodológicas para su análisis cuantitativo (Valverde Valverde 2016).

Anteriormente, el proceso de segmentación de lesiones en la esclerosis múltiple en imágenes por resonancia magnética era un proceso largo y tedioso, que se realizaba con herramientas semiautomatizadas. Esta metodología basada en la selección de vóxeles por umbralización o mediante detección de contornos, podía presentar una variabilidad significativa entre los diferentes operadores. Es por ello que, en los últimos años, debido a la implementación de nuevos procesadores más potentes y rápidos se han podido desarrollar técnicas de detección automática de lesiones capaces de identificar de forma más sencilla, rápida y precisa las lesiones características de la enfermedad de la EM.

2.1 Preprocesamiento

Para facilitar la detección automática de las lesiones en EM es necesario realizar un preprocesamiento de las imágenes de RM. Este tipo de imágenes contienen diferentes tipos de tejidos que pueden interferir en el proceso de segmentación, tales como piel, tejido óseo, parte del cuello o los globos oculares. A continuación se detallan los diferentes pasos del preprocesado necesarios para realizar la detección de lesiones en EM.

2.1.1 Skull-stripping / Brain extraction

Para eliminar los tejidos no cerebrales presentes en las imágenes de resonancia magnética se usan técnicas de *skull-stripping* o *brain-extraction*, basados en la morfología, en la intensidad de los píxeles de la imagen, en modelos de deformación de la superficie, en plantillas/atlas, o modelos híbridos (Kalavathi and Prasath 2016).

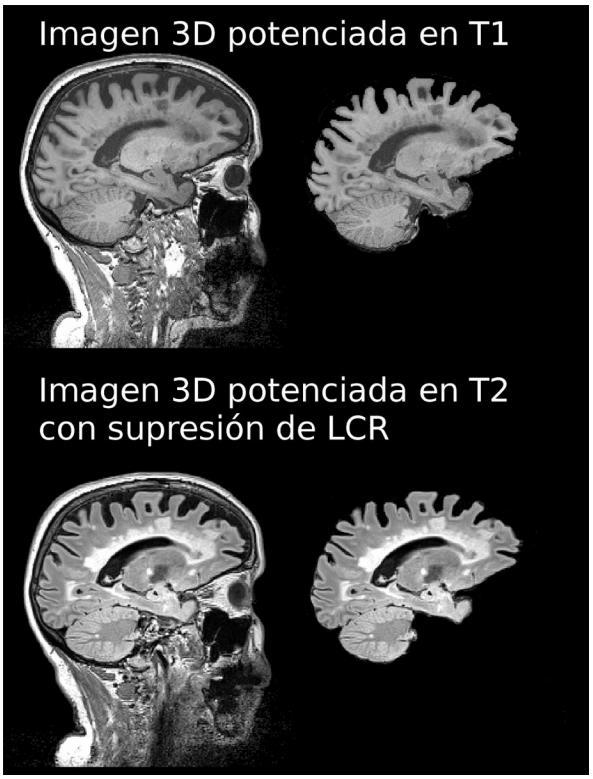

Figura 3: Ejemplo de skull-stripping en imagen 3D potenciada en T1 e imagen 3D potenciada en T2 con supresión del líquido cefalorraquídeo

Este paso preliminar facilita y mejora el proceso de segmentación. De acuerdo con (Akkus et al. 2017; Valverde Valverde 2016), las técnicas más usadas actualmente son: BET (Smith 2002), ROBEX (Iglesias et al. 2011), SPM (Ashburner and Friston 2005) y BEaST (Eskildsen et al. 2012).

2.1.2 Registro

Uno de los pasos más importantes en el proceso de segmentación automática es la etapa de registro. El registro de las imágenes permite ajustar imágenes de distinta modalidad a un mismo espacio anatómico e identificar el cambio de señal entre los diferentes tejidos cerebrales. Hay que recordar que las lesiones en EM son hiperintensas en T2/FLAIR e hipointensas en secuencias potenciadas en T1. Existen multitud de algoritmos específicamente ideados para realizar estas etapas de registro (Klein et al. 2009).

2.1.3 Intensity correction

Un problema muy común que surge a la hora de aplicar técnicas de segmentación automática en imágenes de resonancia magnética es la inhomogeneidad de campo magnético (*bias field or intensity inhomogeneity*, IIH), produciendo variaciones de intensidad en la imagen para un mismo tejido.

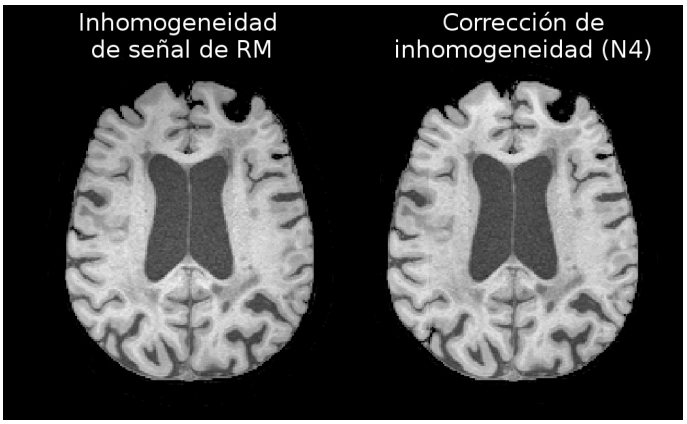

Figura 4: Corrección de inhomogeneidad de señal de RM con N4

Para corregir este problema existen diferentes algoritmos cuyos resultados dependen en gran medida de la selección de los parámetros de entrada (Hou 2006; Ganzetti, Wenderoth, and Mantini 2016).

Entre las técnicas de corrección más usadas actualmente están la N3 (Sled, Zijdenbos, and Evans 1998) y N4ITK (Tustison et al. 2010).

Una corrección de la inhomogeneidad de intensidad inadecuada afecta directamente al resultado de la segmentación, haciendo que el resultado obtenido sea menos preciso y fiable.

2.2 Segmentación automática

Son muchas las técnicas que se han propuesto para la segmentación automática de lesiones cerebrales en imágenes de RM. Normalmente estas técnicas siguen un enfoque supervisado, donde se entrena un modelo a partir de imágenes correctamente etiquetadas previamente (“máscaras de lesiones”) con la intención de aprender automáticamente las características de las lesiones en las distintas modalidades de imagen, o no supervisado, cuyo objetivo principal es agrupar los píxeles o vóxeles de las imágenes de entrada en función de algunas medidas de

similitud o distancia (Pouyan and Peyvandi 2015; Valverde Valverde 2016).

2.2.1 Técnicas supervisadas

La segmentación automática de imágenes de RM puede entenderse como un problema de clasificación binaria, donde la zona de interés tiene una etiqueta y el resto de la imagen tiene otra etiqueta distinta. Dependiendo del objetivo de la segmentación, puede tratarse de un problema de clasificación multiclase, donde cada clase corresponde con un tipo de tejido.

Dentro de las técnicas supervisadas que se han propuesto con cierto éxito, las más significativas son: kNN (Khalid 2011; Vrooman, Lijn, and Niessen 2013; Pouyan and Peyvandi 2015), SVM (Opbroek, Lijn, and Bruijne 2013; Pouyan and Peyvandi 2015), *Random Forests* (Mahapatra 2014; Serag et al. 2017), clasificadores bayesianos (*Bayesian Classifiers*) (Farzan 2014) y modelos de mezcla gaussiana (*Gaussian Mixture Models*) (Rajchl et al. 2016).

Gracias a los avances tanto en el hardware de procesamiento como en las técnicas de aprendizaje automático, las redes neuronales convolucionales (CNN) se están posicionando como una solución muy prometedora para abordar el problema de la segmentación (Valverde Valverde 2016; Kamnitsas et al. 2017; Bernal et al. 2018; Nair et al. 2018; Roy et al. 2018).

2.2.2 Técnicas no supervisadas

El problema de la segmentación automática también puede enfocarse como un problema de agrupación o como un problema de detección de *outliers*, de forma que el tejido lesionado quede separado del tejido de apariencia “normal”.

Los investigadores (Jain et al. 2015) han propuesto un modelo probabilístico capaz de detectar las lesiones como *outliers*, sin requerir datos de entrenamiento, pero usando conocimiento a priori sobre la localización y apariencia de las lesiones.

Otros métodos basados en la detección de *outliers* son los propuestos por (Schmidt et al. 2012) y (Bowles et al. 2017).

Técnicas de agrupación no supervisadas como *Fuzzy C-Means* también han sido propuestas (Caldairou et al. 2011; Banchpalliwar and Salankar 2016), combinaciones de técnicas de agrupación con modelos de

campos aleatorios de Markov (Saladi and Amutha Prabha 2018) o *k-Means* (Liu and Guo 2015; Banchpalliwar and Salankar 2016).

2.2.3 Deep learning

Las CNN están demostrando ser un método capaz de mejorar la precisión de otras técnicas en multitud de campos de aplicación. En particular, en cuanto a visión artificial se refiere, las CNN se han utilizado con éxito en el reconocimiento y detección de objetos, así como en la segmentación automática de imágenes (Bernal et al. 2018).

Este tipo de redes se caracterizan por el uso de una operación matemática llamada convolución, en al menos una de sus capas. Estas capas reciben el nombre de **capas convolucionales** (*convolutional layer*), y generan como salida unos mapas de características aplicando uno o más filtros a la entrada. Un mayor número de filtros será capaz de extraer un mayor número de características de la entrada.

La **capa de reducción o submuestreo** (*pooling layer*) reduce la dimensionalidad de los mapas de características generados por la capa convolucional, preservando las características más importantes. Existen diferentes funciones para realizar esta reducción, como *max pooling*, *min pooling* o *average pooling* (Scherer and Behnke 2010). Además de la reducción de la dimensionalidad, la reducción contribuye a conseguir una representación invariable a traslaciones de la entrada (Kauderer-Abrams 2017), aunque esto depende en cierto modo de la arquitectura de la red (Azulay and Weiss 2018).

La **capa completamente conectada o densa** (*fully-connected layer* o *dense layer*) normalmente se usa como capa de salida, para predecir la probabilidad de pertenecer a las distintas clases. Cada neurona de esta capa está conectada a todas las neuronas de la capa anterior.

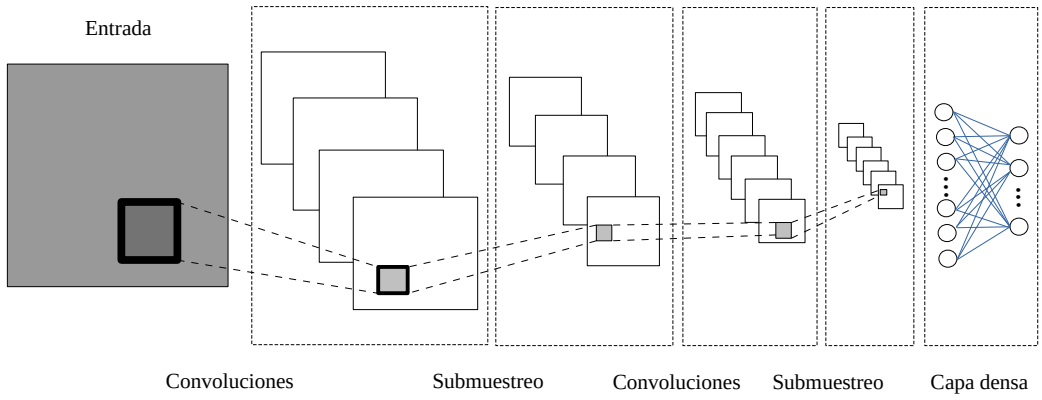

Figura 5: Arquitectura de una red neuronal convolucional típica

En el caso de la segmentación automática de imágenes de resonancia magnética, existen multitud de propuestas de arquitecturas específicas para abordar este problema (Bernal et al. 2018), como VoxResNet (*Voxelwise Residual Network*) (Chen et al. 2016), DeepMedic (Kamnitsas et al. 2017), FLEXCONN (*Fast Lesion EXtraction using COnvolutional Neural Networks*) (Roy et al. 2018) o U-Net (Ronneberger, Fischer, and Brox 2015; Salem et al. 2019).

Estas arquitecturas normalmente incluyen capas de interpolación (*upsampling layer*), cuyo principal objetivo es conseguir una salida de las mismas dimensiones que la entrada inicial.

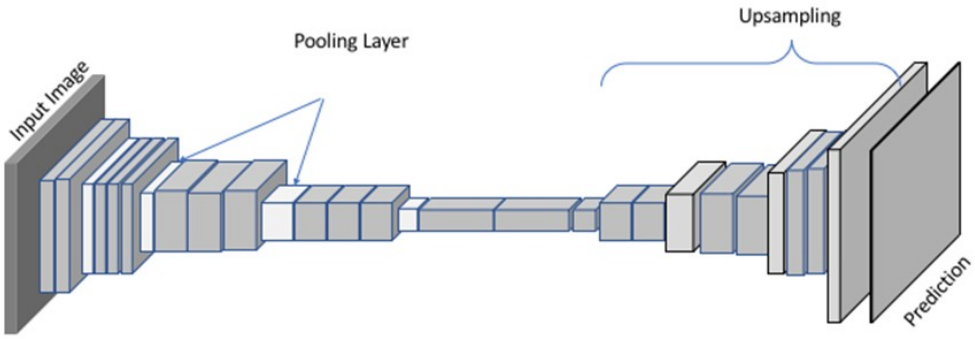

Figura 6: Arquitectura típica de red neuronal convolucional para segmentación, con capas de interpolación. Fuente: Abhinav Dadhich, Practical Computer Vision, 2018

2.3 Evaluación

Evaluar el resultado de la segmentación de imágenes de RM es una parte muy importante del proceso, de otra forma no hay manera de comparar con otros métodos ni de medir la precisión con respecto a la *ground truth*.

Sin una métrica adecuada no es posible ajustar las técnicas de segmentación y no sería posible medir su progreso.

Se han propuesto multitud de técnicas de evaluación que, dependiendo de su naturaleza, se pueden englobar en diferentes grupos, tal y como se define en el trabajo de (Taha and Hanbury 2015).

Los autores proponen seis grupos de métricas de evaluación:

- Métricas basadas en solapamiento
- Métricas basadas en volumen
- Métricas basadas en conteo de pares
- Métricas basadas en teoría de la información
- Métricas probabilísticas
- Métricas basadas en distancia espacial

En el caso de la segmentación de imágenes cerebrales de resonancia magnética, y en particular, en pacientes de esclerosis múltiple, las métricas basadas en solapamiento parecen ser las más adecuadas, habiéndose usado con éxito en múltiples estudios (Zou et al. 2004).

Estas métricas basadas en solapamiento utilizan cuatro medidas básicas, correspondientes a los verdaderos positivos (TP), verdaderos negativos (TN), falsos positivos (FP) y falsos negativos (FN).

De entre estas técnicas, el coeficiente o índice de similaridad de Dice (Dice 1945) es la métrica más usada en la actualidad para validar la segmentación de imágenes (Taha and Hanbury 2015).

Si T es la *ground truth* y S es la segmentación a evaluar, el índice Dice se define como:

$$DICE = \frac{2|S \cap T|}{|S| + |T|} = \frac{2\,TP}{(2\,TP + FP + FN)} \quad (1)$$

Por lo tanto, DICE(S, T) es igual a 0 cuando en ambos conjuntos no hay correspondencia y 1 cuando son idénticos. Valores más cercanos a 1 indican mayor precisión en la segmentación.

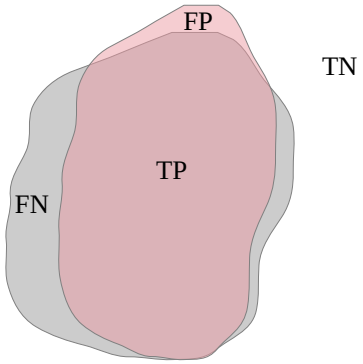

Figura 7: Solapamiento entre segmentación manual (gris) y segmentación automática (rojo), que permite evaluar la precisión de la segmentación y calcular el coeficiente de Dice.

El índice de similaridad de Dice será la métrica que se usará en este trabajo, tanto para medir la precisión como la reproducibilidad de la segmentación.

2.4 Postprocesamiento

Se han propuesto técnicas que permiten mejorar el resultado de la segmentación, aumentando la precisión según el índice de similaridad de Dice en un 1%. Técnicas como MSM y MBM parecen ser capaces de reducir la tasa de falsos negativos (Zhang et al. 2018).

Otros autores proponen el uso de campos condicionales aleatorios (CRF) como un paso de postprocesamiento para reducir la tasa de falsos positivos (Kamnitsas et al. 2017).

3. Metodología

Durante el desarrollo de este trabajo se ha seguido un enfoque iterativo, donde cada ciclo ha consistido fundamentalmente en las siguientes cuatro etapas:

- **Preprocesamiento**: carga y procesamiento de las imágenes previo al proceso de entrenamiento.
- **Entrenamiento**: generación de modelos de clasificación para realizar la segmentación, basado en redes neuronales convolucionales.
- **Postprocesamiento**: procesamiento posterior sobre el resultado de la segmentación, para intentar minimizar los falsos positivos.
- **Evaluación**: calidad de los resultados obtenidos, según el índice de similaridad de Dice.

El objetivo de cada iteración ha sido realizar experimentos añadiendo componentes a la arquitectura o modificando determinados hiperparámetros, teniendo siempre en cuenta tiempos razonables impuestos por las restricciones encontradas a nivel hardware, para seguir la planificación propuesta sin desviaciones.

A continuación se detalla el entorno tecnológico utilizado, así como las tareas realizadas en cada una de las iteraciones.

3.1 Entorno tecnológico

3.1.1 Hardware

La opción de usar un equipo “doméstico” no ha sido viable, ya que tras realizar las primeras pruebas de concepto sobre un equipo con procesador intel i7, 16 Gb de RAM y GPU NVIDIA® GeForce® GT-740M de 2 Gb, los tiempos de entrenamiento resultan excesivos, requiriéndose varios días de entrenamiento para generar algunos de los modelos propuestos en este trabajo.

El entrenamiento de las redes neuronales convolucionales propuestas para la solución de este problema es totalmente inviable usando sólo la CPU, por lo que es necesario usar unidades especializadas que permiten la paralelización de las operaciones con matrices necesarias para el entrenamiento de la red. Estas unidades especializadas son conocidas como GPU (*Graphical Processing Unit*), que, entre otras funciones, permiten entrenar eficientemente redes neuronales convolucionales y liberar de carga al procesador.

Por lo tanto, la opción elegida como infraestructura hardware ha sido la ofrecida por Google a través de Google Colab, que pone a disposición de los usuarios un entorno similar a Jupyter Notebook con 12 Gb de RAM y una GPU NVIDIA® Tesla K80 de 12 Gb. La cuenta usada en Google Colab durante el desarrollo de este proyecto es una cuenta privada de Google Business.

Una infraestructura hardware más potente hubiera ayudado a realizar más experimentos y a plantear otras arquitecturas, pero no ha sido posible disponer de una opción más razonable que Google Colab.

3.1.2 Software

Como se ha indicado en el punto anterior, se ha optado por usar Google Colab, un entorno similar a Jupyter Notebook, junto con el lenguaje de programación Python 3.6. Los siguientes módulos han resultado especialmente útiles:

3.1.2.1 Keras

Biblioteca de aprendizaje profundo para Python. Ofrece una capa por encima de TensorFlow, CNTK y Theano para facilitar su uso, ofreciendo una API consistente y homogénea independientemente del *framework* que se utilice por debajo.

3.1.2.2 TensorFlow-gpu

TensorFlow es una biblioteca especializada en aprendizaje automático liberada por Google, que permite aprovechar toda la potencia de la GPU. Para ello, requiere la instalación de los *drivers* específicos de NVIDIA, así como NVIDIA CUDA® Toolkit y cuDNN SDK.

3.1.2.3 NiBabel

NiBabel es un módulo de Python que permite la lectura / escritura de los formatos de archivos más usados en neuroimagen e imagen médica. En este trabajo se ha usado tanto para la lectura de los archivos *nii.gz* como para la generación del resultado de la segmentación.

3.1.2.4 Nipype

Es un módulo de Python que ofrece una interfaz común y uniforme para interactuar con otras herramientas de neuroimagen. En este trabajo se ha usado para evaluar la calidad de la segmentación.

3.2 Anonimización de datos

En cumplimiento del Reglamento General de Protección de Datos (RGPD), es muy importante que a partir de las imágenes de resonancia magnética no sea posible identificar al paciente.

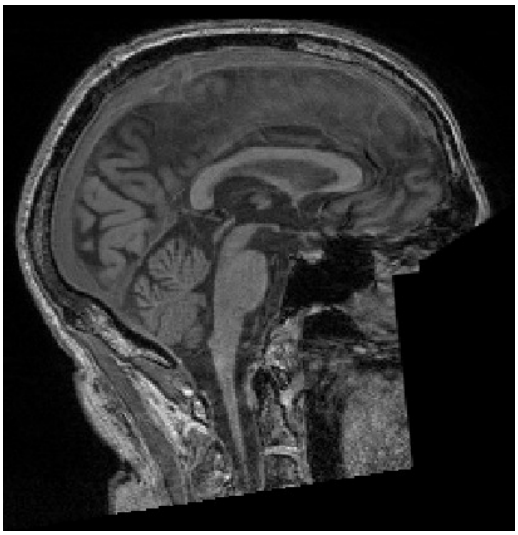

Figura 8: Imagen anonimizada

Por ello, los atlas proporcionados se han numerado de forma consecutiva, tal que no es posible conocer la identidad del paciente a partir del nombre del archivo.

Igualmente, se han anonimizado las imágenes mediante la conversión de DICOM a NifTI con la herramienta *dcm2nii (Rorden 2013)*, y se han

eliminado los rasgos faciales con la herramienta *pydeface* (poldracklab 2017).

3.3 Preprocesamiento

Este apartado es muy importante para facilitar que la herramienta FLEXCONN pueda llevar a cabo un proceso de aprendizaje supervisado de manera precisa.

Para ello, se han ajustado todas las imágenes de las diferentes modalidades (en este caso, T1 y FLAIR) a un espacio común mediante la herramienta de registro rígido FLIRT (Analysis Group, FMRIB, Oxford n.d.; Mark Jenkinson et al. 2002; M Jenkinson and Smith 2001; Greve and Fischl 2009).

Posteriormente, se ha realizado una corrección de las inhomogeneidades del campo magnético en ambas modalidades utilizando el software de ANTs (Tustison et al. 2010). De este modo, los vóxeles pertenecientes a la máscara de lesión ("*gold standard*") podrán ser asignados según su cambio de intensidad en las distintas modalidades de imagen y no por los efectos causados por las inhomogeneidades.

3.4 Funciones de coste

El proceso de aprendizaje de una red neuronal se basa en la optimización de un conjunto de pesos de entrada de cada neurona para minimizar una función de coste, que básicamente compara el resultado de la red con el resultado real. La red neuronal va ajustando los pesos de sus conexiones usando propagación hacia atrás (*backpropagation) (Hopfield 1984)*.

En este trabajo se han evaluado diferentes funciones de coste, las cuales se definen a continuación.

3.4.1 Error cuadrático medio (MSE)

Mide el promedio de la diferencia al cuadrado entre los valores observados y los valores predichos. Esta métrica penaliza mucho más los valores que están muy alejados del valor real.

$$MSE = \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{n} \quad (2)$$

3.4.2 Entropía cruzada binaria (BCE)

Mide a qué distancia del valor observado (0 o 1) está el valor predicho y calcula el promedio de estas desviaciones.

$$BCE = -\frac{1}{n}\sum_{i=1}^{n} y_i \cdot \log(p(y_i)) + (1 - y_i) \cdot \log(1 - p(y_i)) \quad (3)$$

Donde *y* es la clase (0 o 1) y *p(y)* es la probabilidad estimada de pertenecer a la clase.

3.4.3 Índice de similaridad de Dice

Mide el solapamiento entre los valores observados y los predichos. A priori, al tratarse de volúmenes binarios, esta función de coste resulta muy prometedora, tal y como se define en (Milletari, Navab, and Ahmadi 2016).

$$DICE = \frac{2|S \cap T|}{|S| + |T|} = \frac{2TP}{(2TP + FP + FN)} = \frac{2\sum_{i=1}^{n} s_i t_i}{\sum_{i=1}^{n} s^2_i + \sum_{i=1}^{n} t^2_i} \quad (4)$$

De esta forma, la función resulta diferenciable.

3.4.4 Distancia de Jaccard

Siguiendo la misma estrategia propuesta por (Milletari, Navab, and Ahmadi 2016), se puede hacer diferenciable la distancia de Jaccard, de forma que pueda ser usada como función de coste.

$$JACCARD = \frac{|S \cap T|}{|S \cup T|} = \frac{|S \cap T|}{|S| + |T| - |S \cap T|} = \frac{\sum_{i=1}^{n} s_i t_i}{(\sum_{i=1}^{n} s_i + \sum_{i=1}^{n} t_i) - \sum_{i=1}^{n} s_i t_i} \quad (5)$$

3.5 FLEXCONN

Este trabajo se basa extensivamente en la arquitectura FLEXCONN (*Fast Lesion EXtraction using COnvolutional Neural Networks*) propuesta por (Roy et al. 2018).

Esta arquitectura consiste en una red convolucional dividida en dos secuencias paralelas, una para T1 y otra para FLAIR.

Cada una de estas secuencias aplica cinco convoluciones con la siguiente configuración:

Convolución	Número de filtros	Tamaño	Padding	Función de activación
1	128	3x3	same	ReLU
2	64	5x5	same	ReLU
3	32	3x3	same	ReLU
4	16	5x5	same	ReLU
5	8	3x3	same	ReLU

Tabla 1: Configuración de las capas convolucionales de las secuencias paralelas en la red FLEXCONN

Las secuencias paralelas son concatenadas y, a continuación, se vuelven a aplicar una serie de convoluciones para producir la salida.

Hay ciertos parámetros que han sido fijados según los resultados obtenidos en (Roy et al. 2018), para los cuales no se ha realizado ningún ajuste en este trabajo. Estos parámetros son el número de filtros de las secuencias convolucionales y el umbral (*threshold*) final para determinar la segmentación, cuyo valor es 0.34. Como línea de trabajo futura se propone realizar más experimentos para ajustar estos parámetros según la resolución de las imágenes del conjunto de entrenamiento.

Debido a las limitaciones técnicas de la infraestructura, se ha trabajado con imágenes y parches 2D (*patches*) para optimizar la velocidad y el uso de memoria. Un parche es un conjunto de píxeles (o vóxeles), que normalmente forman un cuadrado (o cubo), y permite extraer información local al subdividir la imagen original en múltiples bloques del tamaño de parche especificado.

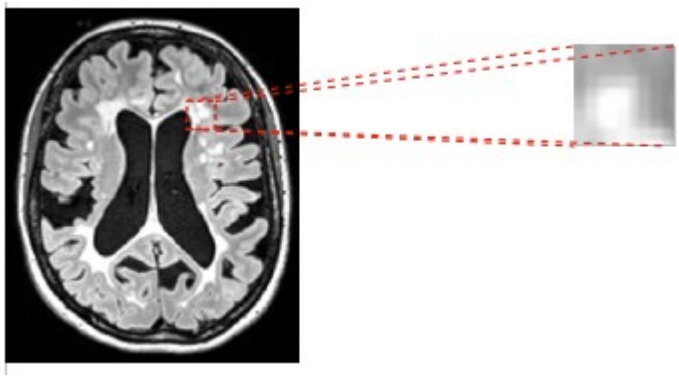

Figura 9: Parche de 25x25 en imagen FLAIR

De esta forma, si se divide una imagen de 100x100px en parches de 10x10px, se obtendrán un total de 100 parches, que potencialmente facilitarían la detección y el aprendizaje de características locales.

FLEXCONN propone parches 2D de 35x35, valor que se ha usado como punto de partida, aunque en este trabajo se han evaluado tamaños de parches distintos para ajustarlo a las características de las imágenes del conjunto de entrenamiento.

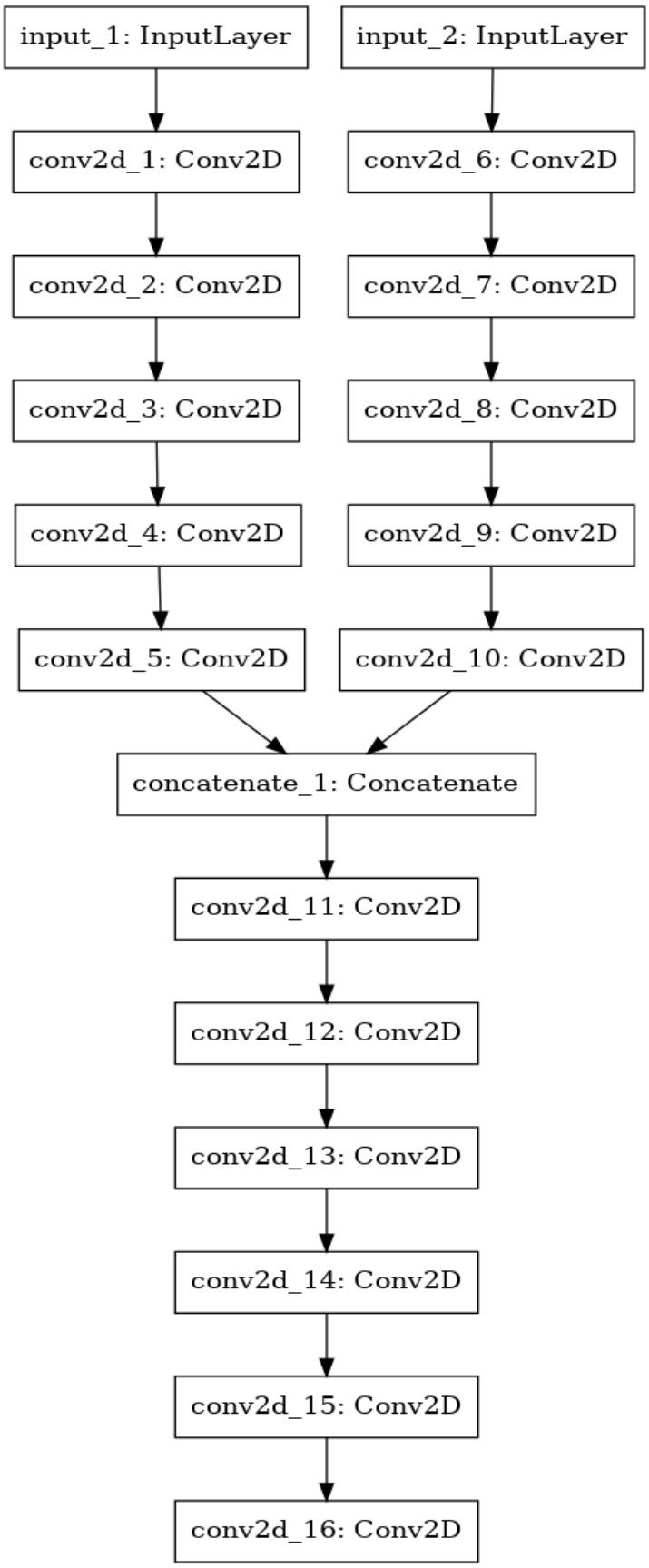

Figura 10: Arquitectura de FLEXCONN

3.5.1 Entrenamiento

Para el entrenamiento se han usado imágenes de 240x256x256 de 50 pacientes, con parches de 35x35, durante 10 épocas.

A lo largo del proceso de entrenamiento, el 80% de estos parches se ha usado para el propio entrenamiento y el 20% restante para validación.

El tiempo de entrenamiento por época es aproximadamente 500 segundos.

3.5.2 Postprocesamiento

Antes de determinar la segmentación final, se aplica un umbral para determinar el etiquetado como lesión, cuyo valor es 0.34. Igualmente, se eliminan las lesiones con un volumen inferior a 27 mm^3.

3.5.3 Evaluación

La evaluación se ha realizado usando las imágenes de 89 pacientes. Cada uno de ellos ha sido segmentado por cuatro modelos distintos que implementan la arquitectura descrita en este apartado, midiéndose la calidad de la segmentación según el índice de similaridad de Dice.

Cada uno de los modelos generados se diferencian entre sí en la función de coste utilizada (ecuaciones 2, 3, 4 y 5).

El siguiente diagrama de caja muestra el resultado de la segmentación en los conjuntos de entrenamiento y test por cada uno de los modelos generados.

Puede observarse que, salvo en el caso de la distancia de Jaccard, donde el índice de Dice medio es 0.18, los otros tres modelos son superiores, siendo el índice medio 0.22 en el caso del modelo BCE y 0.24 en el caso de MSE y DICE (ver Resultado de la evaluación con FLEXCONN).

La segmentación de un paciente con estos modelos tarda aproximadamente 10 segundos.

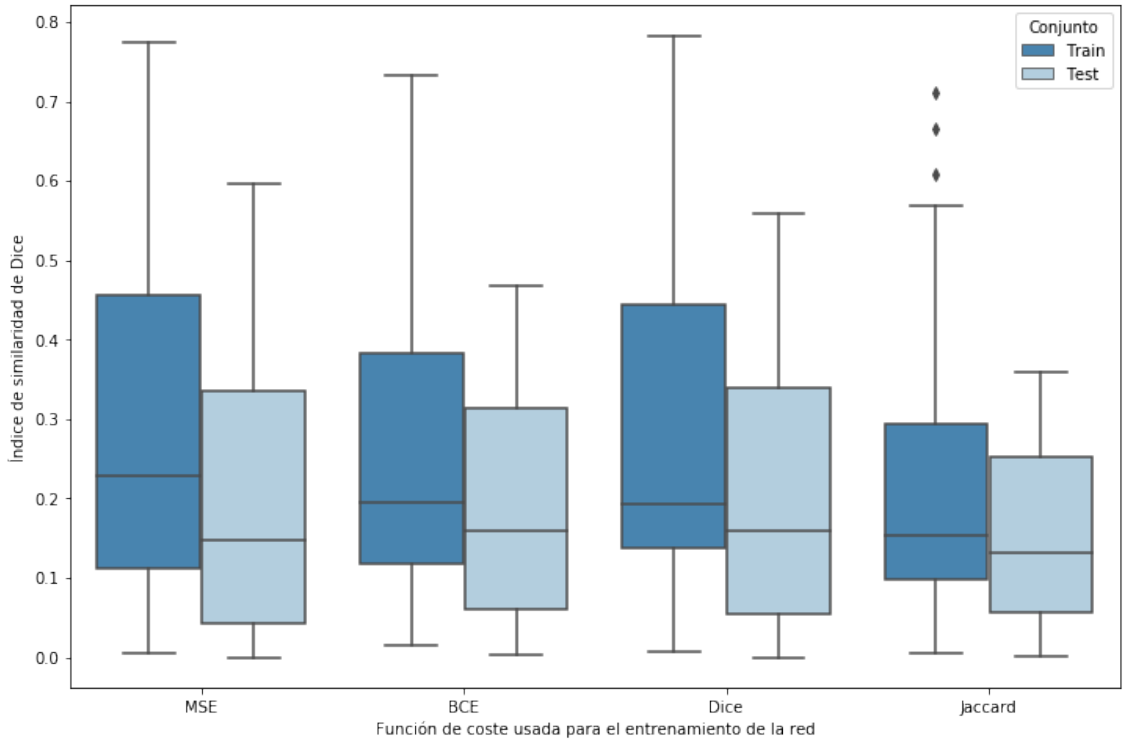

Figura 11: Índice de similaridad de Dice obtenido en los conjuntos de entrenamiento y test, para diferentes funciones de coste usadas durante el entrenamiento de la red FLEXCONN

	MSE	BCE	Dice	Jaccard
Dice promedio	0,246401	0,224072	0,183868	0,241331

Tabla 2: Índice Dice medio obtenido al evaluar distintas funciones de coste con la arquitectura FLEXCONN

3.6 Inception

El módulo *inception* (Szegedy et al. 2014) permite realizar varias convoluciones de forma paralela con diferentes filtros, de forma que no es necesario decidir a priori qué tipo de convolución aplicar. Será el propio modelo el que elija la más conveniente (Bosch Rué, Casas Roma, and Lozano Bagén 2018, p. 152).

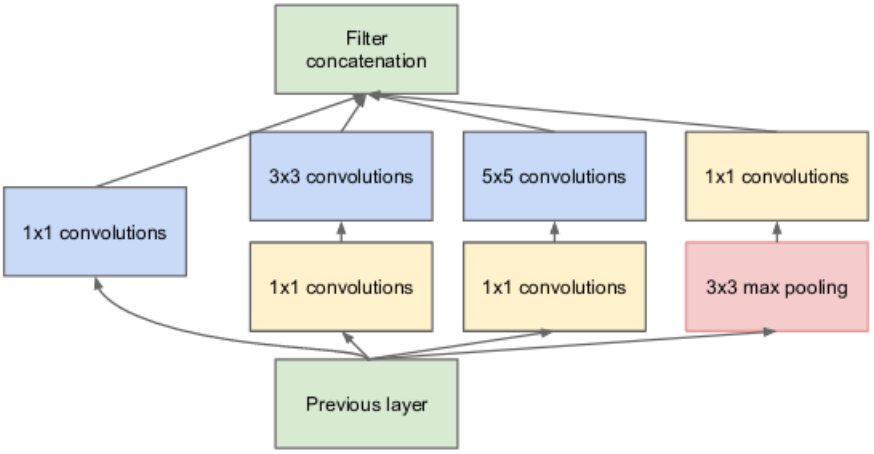

Figura 12: Módulo inception con reducción de dimensionalidad presentado en (Szegedy et al. 2014)

Para mejorar el resultado obtenido con FLEXCONN, se ha generado una nueva arquitectura incluyendo tres módulos *inception* en cada una de las dos secuencias paralelas de FLEXCONN, es decir, tres módulos *inception* en la secuencia para imágenes T1 y otros tres para la secuencia de imágenes FLAIR.

Esto conlleva un coste en el tiempo de entrenamiento y segmentación, pero la mejora en la calidad de la misma es considerable.

3.6.1 Entrenamiento

A pesar del incremento en el consumo de recursos, tanto en memoria como en tiempo de ejecución, para el entrenamiento de estos modelos se han usado las mismas imágenes de 50 pacientes, con parches de 35x35 durante 10 épocas. A lo largo del proceso de entrenamiento, el 80% de estos parches se ha usado para el propio entrenamiento y el 20% restante para validación.

El tiempo de entrenamiento por época es aproximadamente 2486 segundos, casi 5 veces más que en la arquitectura anterior.

3.6.2 Postprocesamiento

Al igual que en el caso anterior, se aplica un umbral de 0.34 antes de determinar la segmentación final, y se eliminan las lesiones con un volumen inferior a 27mm³.

3.6.3 Evaluación

El proceso de evaluación es idéntico al seguido en el caso anterior.

Como puede observarse en la gráfica, la inclusión de los módulos *inception* mejora los cuatro modelos.

El mejor resultado se obtiene con el modelo entrenado usando la distancia de Jaccard como función de coste, con un índice de Dice promedio de 0.35, mientras que el peor resultado se obtiene con el modelo entrenado usando BCE (0.30).

Los otros dos modelos arrojan prácticamente el mismo resultado, un índice Dice promedio de 0.31 (ver Resultado de la evaluación con módulos inception).

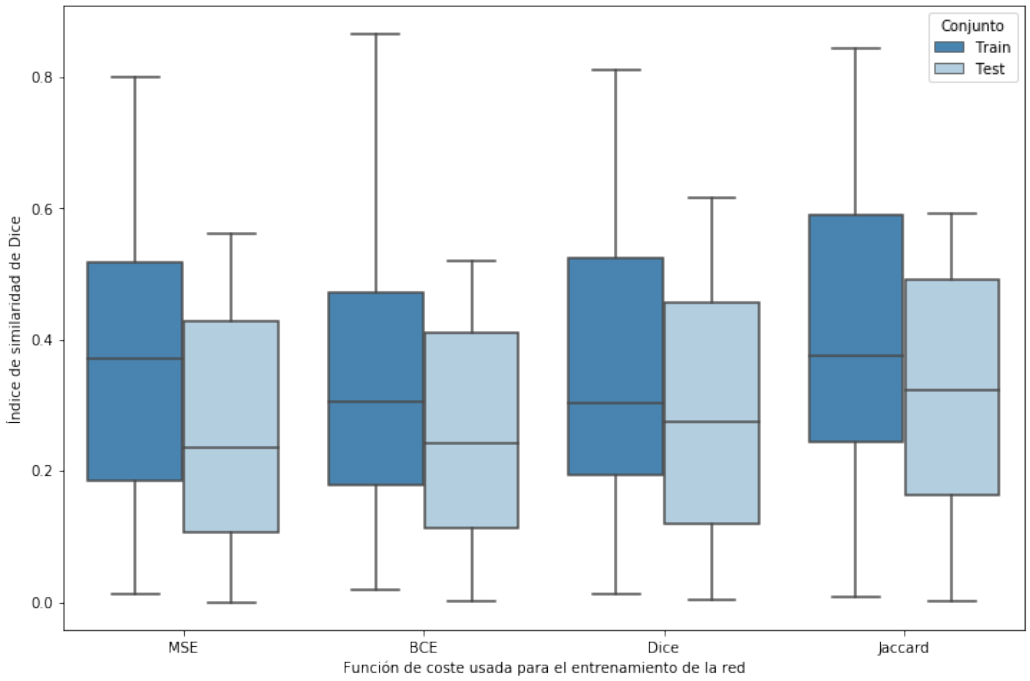

Figura 13: Índice de similaridad de Dice obtenido en los conjuntos de entrenamiento y test, para diferentes funciones de coste usadas durante el entrenamiento de la red FLEXCONN con módulos inception

	MSE	BCE	Dice	Jaccard
Dice promedio	0,311514	0,3002895	0,354699	0,315955

Tabla 3: Índice Dice medio obtenido al evaluar distintas funciones de coste con la arquitectura FLEXCONN más módulo inception

3.7 Mejoras

Se han realizado diversas modificaciones en el código de la herramienta FLEXCONN, puesto a disposición de la comunidad por (Roy et al. 2018), con la idea de implementar algunas mejoras que pudieran o bien incrementar la calidad del resultado, o bien reducir el consumo de recursos y el tiempo de entrenamiento.

3.7.1 Selección de cortes con presencia de lesiones

Cada una de las imágenes de resonancia magnética proporcionadas para este estudio consta de 256 cortes de 240x256. No todos los cortes presentan lesiones ni éstos son los mismos en cada paciente.

Dado que por cada corte de 240x256 se crean *n* parches de *XxY*, una posible mejora para reducir la cantidad de datos de entrada consiste en seleccionar únicamente aquellos cortes que presentan lesiones, reduciendo notablemente el número de cortes a procesar.

Por ejemplo, para el sujeto 1 se redujo el número de cortes a procesar de 256 a 63. Igualmente, para el sujeto 8 se redujo hasta 49. La proporción de cortes a procesar es bastante considerable en todos los pacientes.

Aunque esta tarea de preprocesamiento ayudará mínimamente a balancear los casos positivos y negativos, seguirá existiendo un desbalanceo de clases bastante notable.

3.7.2 Región de interés

En las primeras iteraciones se han detectado numerosos falsos positivos, identificando lesiones en localizaciones que no tienen sentido

para pacientes de esclerosis múltiple como, por ejemplo, en la calota craneal.

Esto provoca una reducción del índice de similaridad de Dice, en el que está basada la evaluación de la segmentación, y por tanto, en la calidad del modelo.

En la figura a continuación puede observarse cómo el modelo detecta lesiones en zonas donde no se presentan en pacientes de EM, que se consideran falsos positivos al estar fuera de la región de interés.

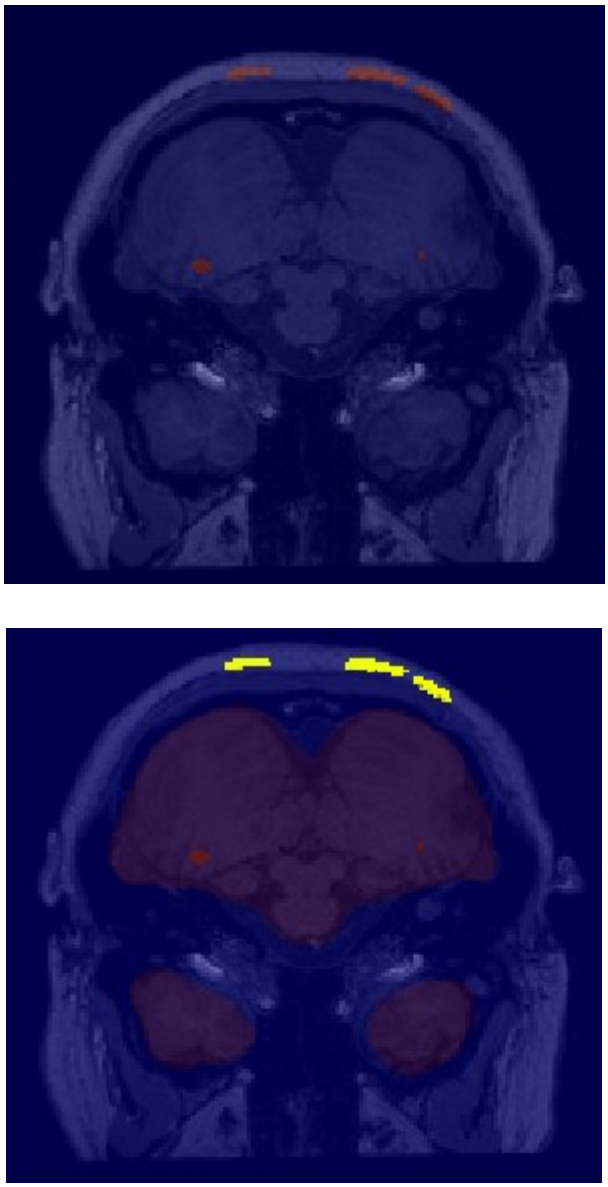

Figura 14: Eliminación de falsos positivos localizados en la calota craneal mediante una máscara de inclusión

Para mitigar este problema y reducir el número de falsos positivos se ha aplicado una máscara de inclusión perteneciente al parénquima cerebral.

Como se puede observar en la figura 14, la aplicación de una máscara perteneciente al parénquima cerebral (rojo translúcido) descarta aquellos falsos positivos detectados en la calota craneal (amarillo). Así pues, todas las lesiones detectadas fuera del área de interés, son descartadas.

3.7.3 Callbacks

Keras ofrece el uso de *callbacks* que permiten, entre otras cosas, controlar el proceso de aprendizaje. Estas funciones se disparan automáticamente durante el proceso de entrenamiento ante determinadas situaciones, y pueden ser utilizadas para ajustar determinados hiperparámetros o para acceder al estado intermedio del modelo durante el entrenamiento.

En este trabajo se han usado tres *callbacks* distintos, los cuales se detallan en los siguientes apartados.

3.7.3.1 Early stopping

El sobreajuste es un problema que surge cuando un modelo es capaz de predecir correctamente los datos de entrenamiento, pero no es capaz de generalizar con una precisión (o error) aceptables. Esto puede deberse a que el modelo ha sido sobreentrenado o sea tan complejo que no es capaz de dar una respuesta adecuada ante nuevos datos.

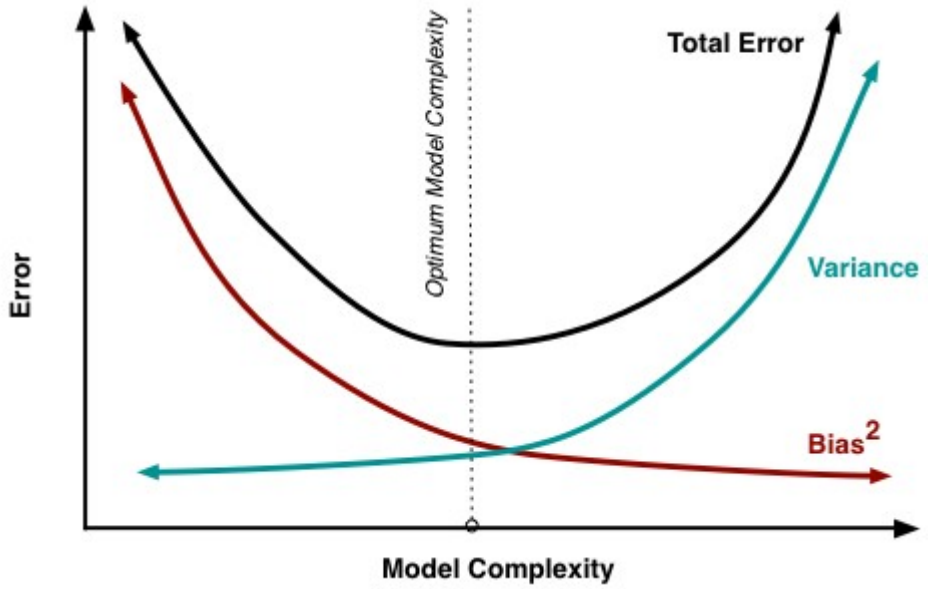

Figura 15: Equilibrio sesgo-varianza (Fortmann-Roe 2012)

A medida que la complejidad del modelo aumenta, el sesgo (*bias*) disminuye y la varianza (*variance*) aumenta. La clave está en generar un modelo con un nivel de complejidad suficiente de forma que no esté ni subajustado ni sobreajustado y, por lo tanto, generalice correctamente.

Para evitar el problema del sobreajuste suele desglosarse el conjunto de datos en un subconjunto de entrenamiento y otro subconjunto de test y/o validación, usar validación cruzada, o usar técnicas de regularización como L1, L2, *early stopping* o *dropout*, entre otras (Sharma et al. 2019).

Las redes neuronales son especialmente propensas a sobreajustarse, por lo que existen técnicas muy usadas que han dado buenos resultados con redes neuronales, como *dropout* (Srivastava et al. 2014) o *early stopping (Caruana et al. 2000)*.

Básicamente, *dropout* consiste en añadir capas que permiten la activación aleatoria de un porcentaje de neuronas. De esta forma, la red aprende características más robustas, ya que ésta necesita compensar durante el aprendizaje el hecho de que algunas neuronas están desactivadas.

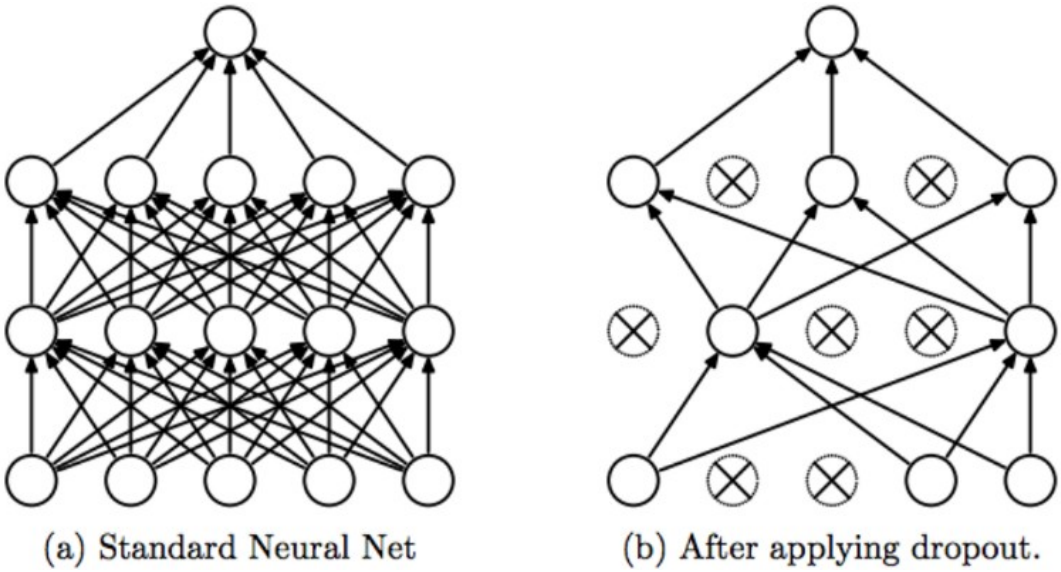

Figura 16: Ejemplo de dropout (Srivastava et al. 2014)

Por otro lado, *early stopping* evita que el modelo se sobreajuste evaluando el error de generalización en el conjunto de validación por cada época, de forma que si el coste aumenta o la precisión disminuye, el proceso de entrenamiento se detiene.

Keras proporciona un *callback* para añadir el método de regularización *early stopping*, que ha sido incluido como mejora en este trabajo.

3.7.3.2 Tasa de aprendizaje

La tasa de aprendizaje (*learning rate*) es probablemente uno de los hiperparámetros más importantes, ya que controla a qué velocidad se adapta la red a los datos de entrenamiento, siendo la duración del entrenamiento inversamente proporcional al valor de dicho parámetro. Valores más pequeños requieren más épocas de entrenamiento, ya que la tasa de aprendizaje afecta a cómo se actualizan los pesos de la red, mientras que un valor elevado podría llevar a un mínimo local no óptimo.

Keras proporciona el *callback ReduceLROnPlateau* que ajusta automáticamente la tasa de aprendizaje durante el entrenamiento si detecta que no hay mejora en un número determinado de épocas.

3.7.3.3 Checkpoints

El entrenamiento de los modelos generados en este trabajo lleva bastante tiempo en Google Colab, por lo que si se produce un error inesperado se pierde todo el progreso realizado. Cuando un proceso tarda mucho en ejecutarse, se suelen usar técnicas de *checkpoint* que permiten restaurar el trabajo desde donde se quedó.

Keras ofrece el *callback ModelCheckpoint*, que permite generar un archivo con el modelo tras cada época. Es posible configurar este *callback* para, por ejemplo, generar un archivo del modelo sólo si se produce una mejora, o generar únicamente un archivo con el mejor modelo.

En este trabajo se ha usado el *callback ModelCheckpoint* para generar un archivo del modelo tras cada época, de forma que en caso de error inesperado o pérdida de conexión del *runtime* de Google Colab, es posible continuar el entrenamiento del modelo desde la última época que finalizó con éxito.

3.7.4 Tamaño del parche

Se ha evaluado el efecto de usar tamaños distintos de parches: 21x21, 25x25, 31x31, 35x35, 41x41 y 45x45. Por cada tamaño de parche evaluado se han entrenado cuatro modelos basados en FLEXCONN más módulos *inception*, cada uno con una función de coste distinta.

Debido a las limitaciones hardware, estos modelos se han entrenado y evaluado usando imágenes de 32 pacientes y 10 épocas, aunque incluyendo todas las mejoras descritas.

A continuación se muestran los resultados obtenidos con las diferentes funciones de coste.

3.7.4.1 Error cuadrático medio

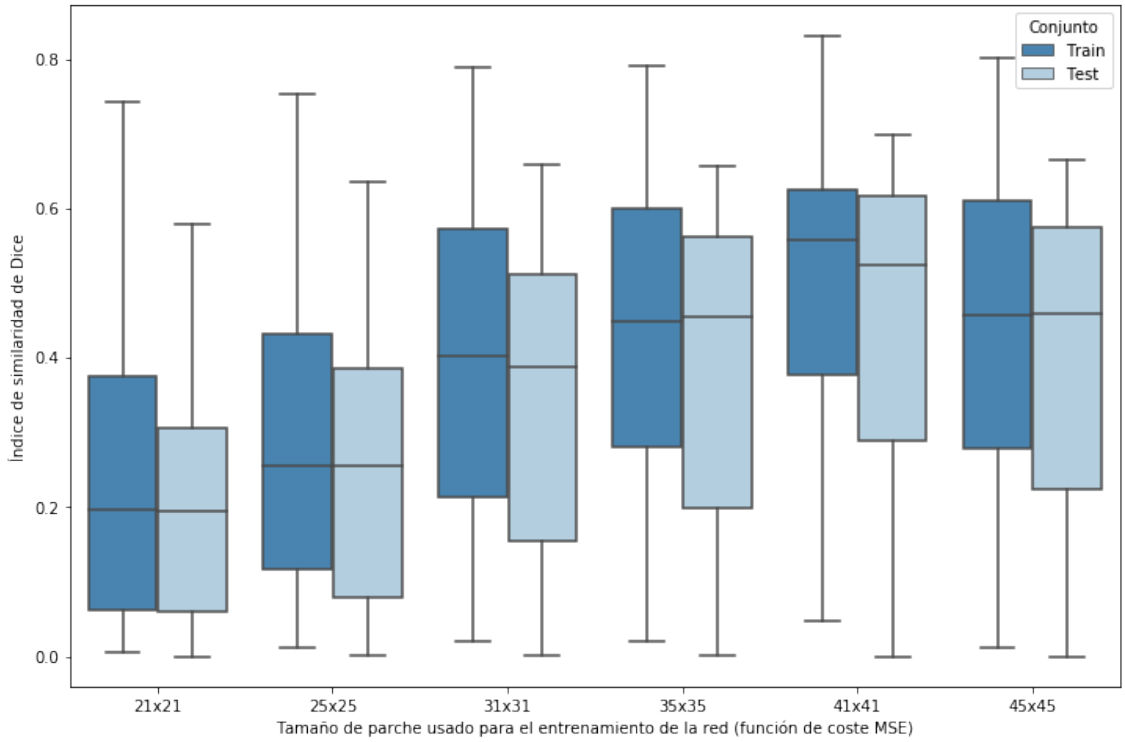

Figura 17: Resultado de la evaluación de diferentes valores de tamaño del parche usando el MSE como función de coste.

Según los valores evaluados, los tamaños 41x41 y 45x45 son los que arrojan los mejores resultados. Con tamaños menores que 31x31, los resultados son significativamente inferiores.

	21x21	25x25	31x31	35x35	41x41	45x45
Dice promedio	0,217801	0,267204	0,369538	0,404958	0,483109	0,413171

Tabla 4: Índice Dice medio obtenido al evaluar distintos tamaños de parche usando el MSE como función de coste.

3.7.4.2 Entropía cruzada binaria

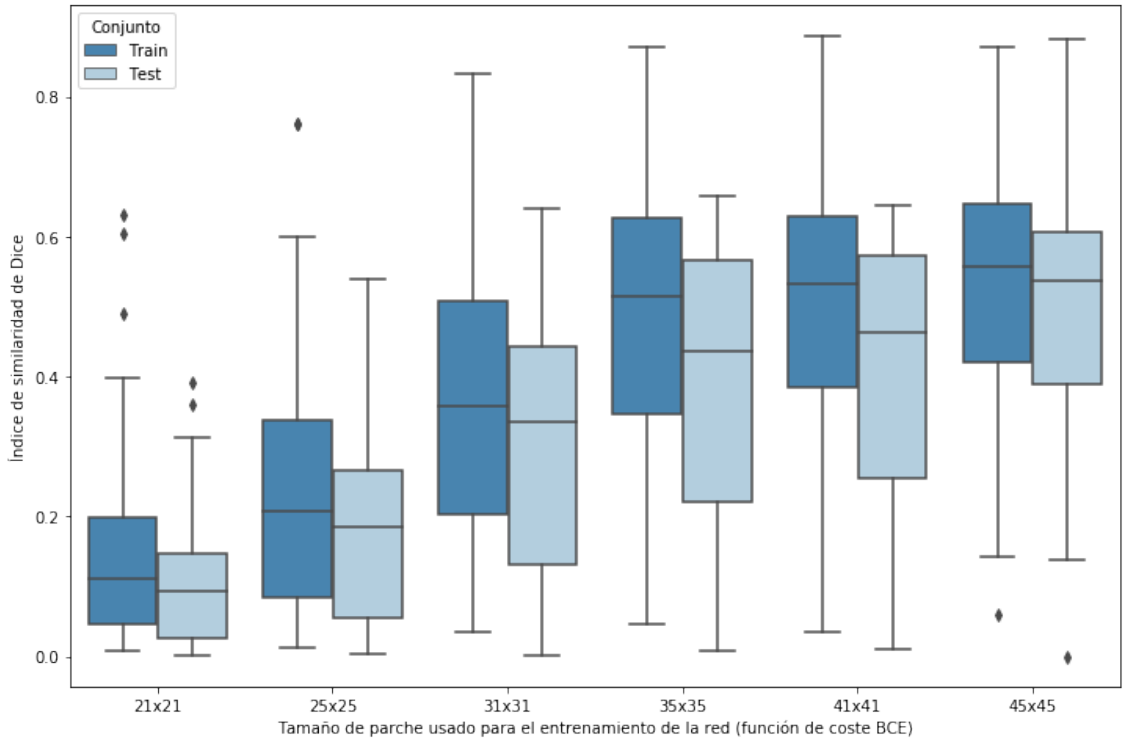

Figura 18: Resultado de la evaluación de diferentes valores de tamaño del parche usando la BCE como función de coste.

Los valores más adecuados son 45x45 y 41x41. La precisión es notablemente inferior con tamaños de parche menores que 35x35.

	21x21	25x25	31x31	35x35	41x41	45x45
Dice promedio	0,126867	0,209171	0,334156	0,434021	0,449753	0,505116

Tabla 5: Índice Dice medio obtenido al evaluar distintos tamaños de parche usando la BCE como función de coste.

3.7.4.3 Índice de similaridad de Dice

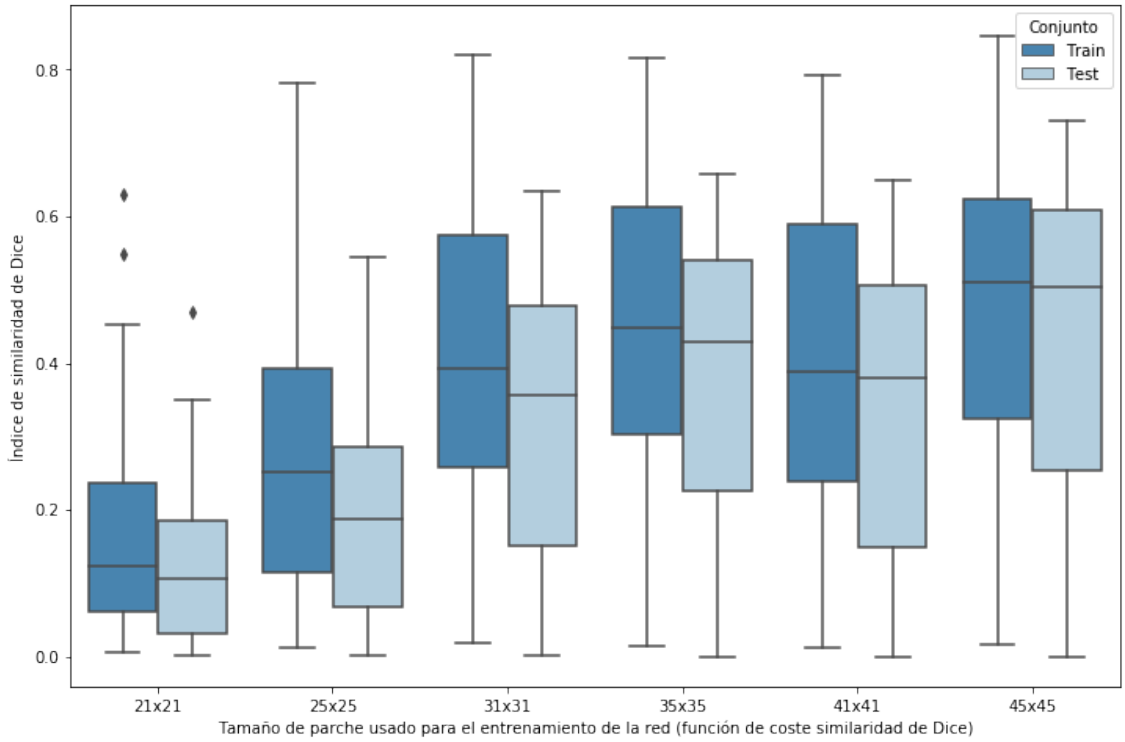

Figura 19: Resultado de la evaluación de diferentes valores de tamaño del parche usando el índice de similaridad de Dice como función de coste.

Al igual que en los casos anteriores, los tamaños inferiores a 35x35 producen un resultado notablemente inferior. El tamaño de parche más adecuado en este caso es 45x45.

	21x21	25x25	31x31	35x35	41x41	45x45
Dice promedio	0,141339	0,226679	0,362341	0,402701	0,369898	0,469224

Tabla 6: Índice Dice medio obtenido al evaluar distintos tamaños de parche usando el índice de similaridad de Dice como función de coste.

3.7.4.4 Distancia de Jaccard

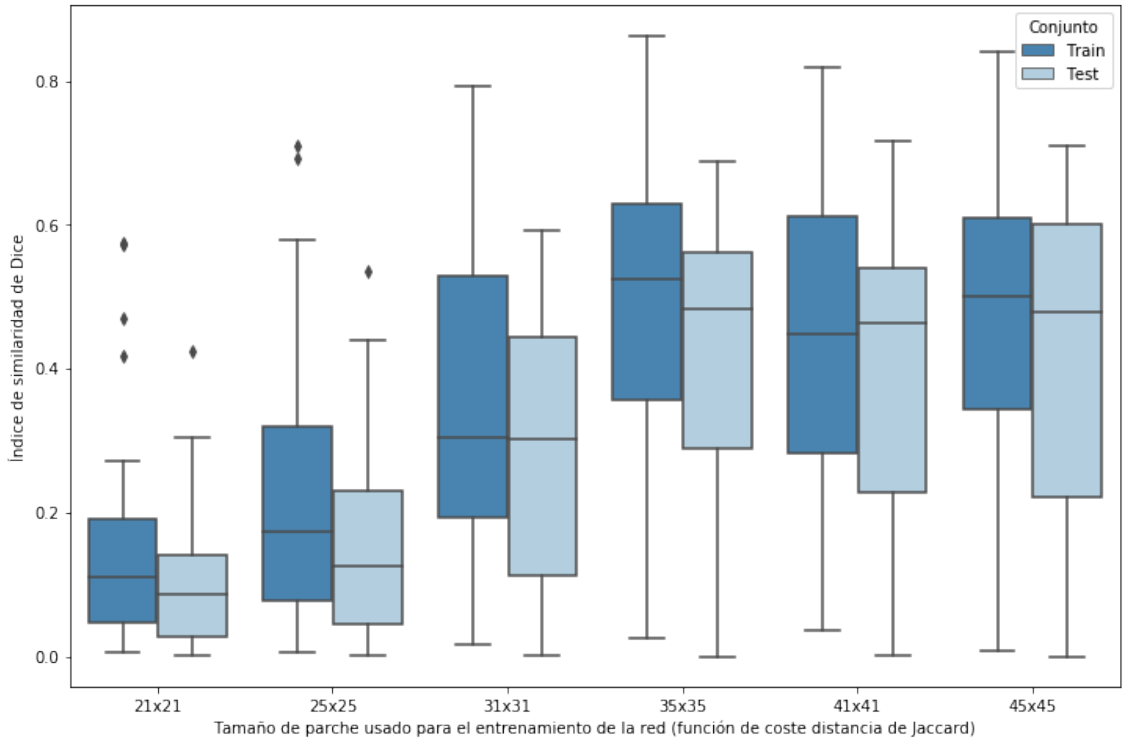

Figura 20: Resultado de la evaluación de diferentes valores de tamaño del parche usando la distancia de Jaccard como función de coste.

Nuevamente los tamaños que generan mejores resultados están entre 35x35 y 45x45. Parece claro que los tamaños 21x21, 25x25 y 31x31 resultan inadecuados independientemente de la función de coste usada.

	21x21	25x25	31x31	35x35	41x41	45x45
Dice promedio	0,121262	0,179993	0,319402	0,452508	0,426089	0,447093

Tabla 7: Índice Dice medio obtenido al evaluar distintos tamaños de parche usando la distancia de Jaccard como función de coste.

4.Conclusiones y líneas de trabajo futuro

Las redes neuronales convolucionales suponen una técnica muy prometedora para identificar y segmentar automáticamente las lesiones en EM, mejorando los resultados obtenidos con otro tipo de técnicas.

Sin embargo, el entrenamiento de estas redes con imágenes de resonancia magnética requiere un elevado consumo de recursos hardware, haciendo totalmente inviable su uso a menos que se cuente con la infraestructura adecuada.

Debido a estas limitaciones, en este proyecto se ha trabajado con imágenes 2D, siendo precisamente ésta una propuesta de trabajo futuro. Muy probablemente el uso de imágenes 3D contribuya a extraer información relevante que ayude a mejorar los resultados obtenidos.

Igualmente, el procesamiento de imágenes de alta resolución conlleva un elevado consumo de memoria, por lo que a pesar de contar con un buen conjunto de entrenamiento no ha podido usarse en su totatilidad. Es por ello que una de las mejoras propuestas en este trabajo es usar únicamente aquellos cortes que presenten lesiones, lo cual contribuye a reducir el tiempo de procesamiento a la vez que ayuda a balancear las clases, pero es posible que contribuya negativamente al proceso de aprendizaje al descartar información valiosa de los cortes sanos. Evaluar el efecto de balancear las clases sería otra propuesta de trabajo futuro.

Uno de los mayores problemas encontrados es la presencia de falsos positivos. La aplicación de una nueva etapa de postprocesamiento para aplicar una máscara con la región de interés mejoró notablemente el resultado. Aún así, se ha observado que donde más suele fallar la segmentación automática de las lesiones es en la región de la sustancia gris. Por ello, sería muy interesante desarrollar en un futuro una línea de trabajo para que la red neuronal aprendiera a descartar aquellas lesiones detectadas en esta región, reduciendo así notablemente el número de falsos positivos.

Es interesante que los resultados obtenidos sean de mayor precisión con parches de mayor tamaño, probablemente porque la CNN aprende características más robustas, considerando las lesiones más relevantes y excluyendo aquéllas de menor tamaño. Hay que recordar que el “*gold standard*” puede presentar errores y la CNN puede detectar lesiones que sean correctas y no calificadas como tal.

La transferencia de conocimiento (*transfer learning*) ha demostrado dar buenos resultados en diversos campos de aplicación (Tan et al. 2018). La posibilidad de trasladar el conocimiento aprendido en otros problemas de segmentación similares hacen que ésta sea una línea de trabajo futura muy prometedora.

En resumen, la planificación propuesta inicialmente ha facilitado y permitido el avance y el seguimiento de este proyecto, refinando los modelos iteración tras iteración. El mayor impedimento encontrado han sido las **limitaciones hardware**, que no han permitido la elaboración de modelos más complejos ni el ajuste fino de los hiperparámetros como se hubiera deseado.

Principalmente, podemos establecer las siguientes líneas futuras: (1) el ajuste de determinados hiperparámetros no evaluados en este proyecto, (2) el uso de otras arquitecturas, con redes residuales o módulos *xception*, y *transfer learning*, y (3) reducción del número de falsos positivos descartando las lesiones detectadas en la región de la sustancia gris.

A pesar de las limitaciones mencionadas, los resultados obtenidos a partir de la herramienta FLEXCONN así como de las mejoras incorporadas a la arquitectura original y a la etapa de postprocesamiento, sugieren que las redes neuronales convolucionales son a día de hoy una de las mejores opciones para la identificación y segmentación automática de lesiones en pacientes de esclerosis múltiple en imágenes de RM.

5. Glosario

ANN	*Artificial Neural Network*
BCE	*Binary Cross-Entropy*
BEaST	*Brain Extraction based on nonlocal Segmentation Technique*
BET	*Brain Extraction Tool*
CNN	*Convolutional Neural Network*
CPU	*Central Processing Unit*
EM	Esclerosis múltiple
FN	*False Negative*
FP	*False Positive*
FLAIR	*Fluid Attenuated Inversion Recovery*
FLEXCONN	*Fast Lesion EXtraction using COnvolutional Neural Networks*
GPU	*Graphical Processing Unit*
IIH	*Intensity Inhomogeneity*
KNN	*k Nearest Neighbors*
LCR	Líquido cefalorraquídeo
MBM	*Mean Binary masks*
MSE	*Mean Square Error*
MSM	*Mean Score Maps*

SVM	*Support Vector Machine*
TN	*True Negative*
TP	*True Positive*
RGPD	Reglamento General de Protección de Datos
RM	Resonancia magnética
ROBEX	*Robust Brain Extraction*
VoxResNet	*Voxelwise Residual Network*

6. Bibliografía

Akkus, Zeynettin, Alfiia Galimzianova, Assaf Hoogi, Daniel L Rubin, and Bradley J Erickson. 2017. “Deep Learning for Brain MRI Segmentation: State of the Art and Future Directions.” *Journal of Digital Imaging* 30 (4): 449–59. https://doi.org/10.1007/s10278-017-9983-4.

Analysis Group, FMRIB, Oxford, UK. n.d. “FLIRT.” Accessed May 16, 2019. https://fsl.fmrib.ox.ac.uk/fsl/fslwiki/FLIRT.

Ashburner, John, and Karl J. Friston. 2005. “Unified Segmentation.” *NeuroImage* 26 (3): 839–51. https://doi.org/10.1016/j.neuroimage.2005.02.018.

Azulay, Aharon, and Yair Weiss. 2018. “Why Do Deep Convolutional Networks Generalize so Poorly to Small Image Transformations?,” May. http://arxiv.org/abs/1805.12177.

Banchpalliwar, Ruchita A., and Dr. Suresh S. Salankar. 2016. “A Review on Brain MRI Image Segmentation Clustering Algorithm.” https://www.semanticscholar.org/paper/A-Review-on-Brain-MRI-Image-Segmentation-Clustering-Banchpalliwar-Salankar/5fea52a2524fab3baf5c8a4847e5323e4c890888.

Bernal, Jose, Kaisar Kushibar, Daniel S. Asfaw, Sergi Valverde, Arnau Oliver, Robert Martí, and Xavier Lladó. 2018. “Deep Convolutional Neural Networks for Brain Image Analysis on Magnetic Resonance Imaging: A Review.” *Artificial Intelligence in Medicine*, September. https://doi.org/10.1016/J.ARTMED.2018.08.008.

Bosch Rué, Anna, Jordi Casas Roma, and Toni Lozano Bagén. 2018. *Deep Learning. Principios y Fundamentos*. Edited by UOC. UOC.

Bowles, Christopher, Chen Qin, Ricardo Guerrero, Roger Gunn, Alexander Hammers, David Alexander Dickie, Maria Valdés Hernández, Joanna Wardlaw, and Daniel Rueckert. 2017. “Brain Lesion Segmentation through Image Synthesis and Outlier Detection.” *NeuroImage: Clinical* 16 (January): 643–58. https://doi.org/10.1016/J.NICL.2017.09.003.

Caldairou, Benoît, Nicolas Passat, Piotr A. Habas, Colin Studholme, and François Rousseau. 2011. “A Non-Local Fuzzy Segmentation Method: Application to Brain MRI.” *Pattern Recognition* 44 (9): 1916–27. https://doi.org/10.1016/J.PATCOG.2010.06.006.

Caruana, Rich, Rich Caruana, Steve Lawrence, and Lee Giles. 2000. “Overfitting in Neural Nets: Backpropagation, Conjugate Gradient, and Early Stopping.” *IN PROC. NEURAL INFORMATION PROCESSING SYSTEMS CONFERENCE*, 402--408. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.21.1952.

Chen, Hao, Qi Dou, Lequan Yu, and Pheng-Ann Heng. 2016. “VoxResNet: Deep Voxelwise Residual Networks for Volumetric Brain Segmentation,” August. http://arxiv.org/abs/1608.05895.

Dice, Lee R. 1945. “Measures of the Amount of Ecologic Association Between Species.” *Ecology* 26 (3): 297–302. https://doi.org/10.2307/1932409.

Esclerosis Múltiple España. n.d. “Qué Es La Esclerosis Múltiple · Esclerosis Múltiple España.” Accessed February 28, 2019. https://www.esclerosismultiple.com/esclerosis-multiple/que-es/.

Eskildsen, Simon F., Pierrick Coupé, Vladimir Fonov, José V. Manjón, Kelvin K. Leung, Nicolas Guizard, Shafik N. Wassef, Lasse Riis Østergaard, D. Louis Collins, and Alzheimer’s Disease Neuroimaging Initiative. 2012. “BEaST: Brain Extraction Based on Nonlocal Segmentation Technique.” *NeuroImage* 59 (3): 2362–73. https://doi.org/10.1016/j.neuroimage.2011.09.012.

Farzan, Ali. 2014. “Heuristically Improved Bayesian Segmentation of Brain MR Images.” *Science World Journal* 9 (3): 5–8. http://www.scienceworldjournal.org/article/view/14639.

Fortmann-Roe, S. 2012. “Understanding the Bias-Variance Trade-Off.”

Ganzetti, Marco, Nicole Wenderoth, and Dante Mantini. 2016. “Intensity Inhomogeneity Correction of Structural MR Images: A Data-Driven Approach to Define Input Algorithm Parameters.” *Frontiers in Neuroinformatics* 10: 10. https://doi.org/10.3389/fninf.2016.00010.

Greve, Douglas N., and Bruce Fischl. 2009. “Accurate and Robust Brain Image Alignment Using Boundary-Based Registration.” *NeuroImage* 48 (1): 63–72. https://doi.org/10.1016/j.neuroimage.2009.06.060.

Hopfield, J. J. 1984. “Neurons with Graded Response Have Collective Computational Properties like Those of Two-State Neurons.” *Proceedings of the National Academy of Sciences* 81 (10): 3088–92. https://doi.org/10.1073/pnas.81.10.3088.

Hou, Zujun. 2006. “A Review on MR Image Intensity Inhomogeneity Correction.” *International Journal of Biomedical Imaging* 2006: 1–11. https://doi.org/10.1155/IJBI/2006/49515.

Iglesias, J. E., Cheng-Yi Cheng-Yi Liu, P. M. Thompson, and Zhuowen Zhuowen Tu. 2011. “Robust Brain Extraction Across Datasets and Comparison With Publicly Available Methods.” *IEEE Transactions on Medical Imaging* 30 (9): 1617–34. https://doi.org/10.1109/TMI.2011.2138152.

Jain, Saurabh, Diana M. Sima, Annemie Ribbens, Melissa Cambron, Anke Maertens, Wim Van Hecke, Johan De Mey, et al. 2015. “Automatic Segmentation and Volumetry of Multiple Sclerosis Brain Lesions from MR Images.” *NeuroImage: Clinical* 8: 367–75. https://doi.org/10.1016/j.nicl.2015.05.003.

Jenkinson, M, and S Smith. 2001. “A Global Optimisation Method for Robust Affine Registration of Brain Images.” *Medical Image Analysis* 5 (2): 143–56. http://www.ncbi.nlm.nih.gov/pubmed/11516708.

Jenkinson, Mark, Peter Bannister, Michael Brady, and Stephen Smith. 2002. “Improved Optimization for the Robust and Accurate Linear Registration and Motion Correction of Brain Images.” *NeuroImage* 17 (2): 825–41. http://www.ncbi.nlm.nih.gov/pubmed/12377157.

Kalavathi, P, and V B Surya Prasath. 2016. “Methods on Skull Stripping of MRI Head Scan Images-a Review.” *Journal of Digital Imaging* 29 (3): 365–79. https://doi.org/10.1007/s10278-015-9847-8.

Kamnitsas, Konstantinos, Christian Ledig, Virginia F.J. Newcombe, Joanna P. Simpson, Andrew D. Kane, David K. Menon, Daniel Rueckert, and Ben Glocker. 2017. “Efficient Multi-Scale 3D CNN with Fully Connected CRF for Accurate Brain Lesion Segmentation.” *Medical Image Analysis* 36 (February): 61–78. https://doi.org/10.1016/J.MEDIA.2016.10.004.

Kauderer-Abrams, Eric. 2017. “Quantifying Translation-Invariance in Convolutional Neural Networks,” December. http://arxiv.org/abs/1801.01450.

Khalid, Noor Elaiza Abdul. 2011. “Brain Abnormalities Segmentation Using K-Nearest Neighbors (k-NN).” https://www.semanticscholar.org/paper/Brain-Abnormalities-Segmentation-using-K-Nearest-(-Khalid/34583aa2c65363ffd18ff53a19d0035f5f8f1cab.

Klein, Arno, Jesper Andersson, Babak A. Ardekani, John Ashburner, Brian Avants, Ming-Chang Chiang, Gary E. Christensen, et al. 2009. “Evaluation of 14 Nonlinear Deformation Algorithms Applied to Human Brain MRI Registration.” *NeuroImage* 46 (3): 786–802. https://doi.org/10.1016/j.neuroimage.2008.12.037.

Liu, Jianwei, and Lei Guo. 2015. “An Improved K-Means Algorithm for Brain MRI Image Segmentation.” In *Proceedings of the 3rd International Conference on Mechatronics, Robotics and Automation*. Paris, France: Atlantis Press. https://doi.org/10.2991/icmra-15.2015.210.

Mahapatra, Dwarikanath. 2014. “Analyzing Training Information From Random Forests for Improved Image Segmentation.” *IEEE Transactions on Image Processing* 23 (4): 1504–12. https://doi.org/10.1109/TIP.2014.2305073.

Martínez de las Heras, Eloy. 2017. “Desarrollo de Una Metodología de Reconstrucción de Fibras de Sustancia Blanca En Difusión Por Resonancia Magnética y Su Aplicación Al Estudio de La Relación Entre La Conectividad Estructural Cerebral y El Rendimiento Cognitivo En Pacientes Con Esclerosis .” Universitat de Barcelona. http://diposit.ub.edu/dspace/handle/2445/123751.

Milletari, Fausto, Nassir Navab, and Seyed-Ahmad Ahmadi. 2016. “V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation,” June. http://arxiv.org/abs/1606.04797.

Nair, Tanya, Doina Precup, Douglas L. Arnold, and Tal Arbel. 2018. “Exploring Uncertainty Measures in Deep Networks for Multiple Sclerosis Lesion Detection and Segmentation,” August. http://arxiv.org/abs/1808.01200.

Opbroek, Annegreet van, Fedde van der Lijn, and Marleen de Bruijne. 2013. “Automated Brain-Tissue Segmentation by Multi-Feature SVM Classification.” https://www.semanticscholar.org/paper/Automated-brain-tissue-segmentation-by-SVM-Opbroek-Lijn/7776d60461ffb1e8d34fcedfdab8886afd972a7d.

poldracklab. 2017. “GitHub - Poldracklab/Pydeface: Defacing Utility for MRI Images.” 2017-12-21. 2017. https://github.com/poldracklab/pydeface.

Pouyan, Ali, and Mohaddeseh Peyvandi. 2015. “Automatic Segmentation of Multiple Sclerosis Lesions in Brain MR Images.” *Journal of Biomedical Engineering and Medical Imaging* 2 (5): 21. https://doi.org/10.14738/jbemi.25.1560.

Rajchl, Martin, John S.H. Baxter, A. Jonathan McLeod, Jing Yuan, Wu Qiu, Terry M. Peters, and Ali R. Khan. 2016. “Hierarchical Max-Flow Segmentation Framework for Multi-Atlas Segmentation with Kohonen Self-Organizing Map Based Gaussian Mixture Modeling.” *Medical Image Analysis* 27 (January): 45–56. https://doi.org/10.1016/j.media.2015.05.005.

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. 2015. “U-Net: Convolutional Networks for Biomedical Image Segmentation,” May. http://arxiv.org/abs/1505.04597.

Rorden, Chris. 2013. “Dcm2nii DICOM to NIfTI Conversion Http://Www.Mccauslandcenter.Sc.Edu/Mricro/Mricron/Dcm2nii.Html.” 2013. https://people.cas.sc.edu/rorden/mricron/dcm2nii.html.

Roy, Snehashis, John A. Butman, Daniel S. Reich, Peter A. Calabresi, and Dzung L. Pham. 2018. “Multiple Sclerosis Lesion Segmentation from Brain MRI via Fully Convolutional Neural Networks,” March. http://arxiv.org/abs/1803.09172.

Saladi, Saritha, and N. Amutha Prabha. 2018. “MRI Brain Segmentation in Combination of Clustering Methods with Markov Random Field.” *International Journal of Imaging Systems and Technology* 28 (3): 207–16. https://doi.org/10.1002/ima.22271.

Salem, Mostafa, Sergi Valverde, Mariano Cabezas, Deborah Pareto, Arnau Oliver, Joaquim Salvi, Àlex Rovira, and Xavier Lladó. 2019. “Multiple Sclerosis Lesion Synthesis in MRI Using an Encoder-Decoder U-NET,” January. http://arxiv.org/abs/1901.05733.

Scherer, Dominik, M Andreas, and Sven Behnke. 2010. “Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition.” *In: Artificial Neural Networks (ICANN), 20th Int. Conf*. http://citeseerx.ist.psu.edu/viewdoc/citations;jsessionid=6847B2FB623D2E4D796B98A2E0872608?doi=10.1.1.592.3018.

Schmidt, Paul, Christian Gaser, Milan Arsic, Dorothea Buck, Annette Förschler, Achim Berthele, Muna Hoshi, et al. 2012. “An Automated Tool for Detection of FLAIR-Hyperintense White-Matter Lesions in Multiple Sclerosis.” *NeuroImage* 59 (4): 3774–83. https://doi.org/10.1016/j.neuroimage.2011.11.032.

Serag, Ahmed, Alastair G Wilkinson, Emma J Telford, Rozalia Pataky, Sarah A Sparrow, Devasuda Anblagan, Gillian Macnaught, Scott I Semple, and James P Boardman. 2017. “SEGMA: An Automatic SEGMentation Approach for Human Brain MRI Using Sliding Window and Random Forests.” *Frontiers in Neuroinformatics* 11: 2. https://doi.org/10.3389/fninf.2017.00002.

Sharma, Mayank, Aayush Yadav, Sumit Soman, and Jayadeva. 2019. “Effect of Various Regularizers on Model Complexities of Neural Networks in Presence of Input Noise,” January. http://arxiv.org/abs/1901.11458.

Sled, J.G., A.P. Zijdenbos, and A.C. Evans. 1998. “A Nonparametric Method for Automatic Correction of Intensity Nonuniformity in MRI Data.” *IEEE Transactions on Medical Imaging* 17 (1): 87–97. https://doi.org/10.1109/42.668698.

Smith, Stephen M. 2002. “Fast Robust Automated Brain Extraction.” *Human Brain Mapping* 17 (3): 143–55. https://doi.org/10.1002/hbm.10062.

Srivastava, Nitish, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. “Dropout: A Simple Way to Prevent Neural Networks from Overfitting.” *Journal of Machine Learning Research* 15: 1929–58. http://jmlr.org/papers/v15/srivastava14a.html.

Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2014. “Going Deeper with Convolutions,” September. http://arxiv.org/abs/1409.4842.

Taha, Abdel Aziz, and Allan Hanbury. 2015. “Metrics for Evaluating 3D Medical Image Segmentation: Analysis, Selection, and Tool.” *BMC Medical Imaging* 15 (August): 29. https://doi.org/10.1186/s12880-015-0068-x.

Tan, Chuanqi, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. 2018. “A Survey on Deep Transfer Learning,” August. http://arxiv.org/abs/1808.01974.

Tustison, Nicholas J, Brian B Avants, Philip A Cook, Yuanjie Zheng, Alexander Egan, Paul A Yushkevich, and James C Gee. 2010. “N4ITK: Improved N3 Bias Correction.” *IEEE Transactions on Medical Imaging* 29 (6): 1310–20. https://doi.org/10.1109/TMI.2010.2046908.

Valverde Valverde, Sergi. 2016. “Automated Brain Tissue Segmentation of Magnetic Resonance Images in Multiple Sclerosis.” *TDX (Tesis Doctorals En Xarxa)*, June. https://www.tdx.cat/handle/10803/386468.

Vrooman, Henri A., Fedde van der Lijn, and Wiro J. Niessen. 2013. “Auto-KNN : Brain Tissue Segmentation Using Automatically Trained k-Nearest-Neighbor Classification.” https://www.semanticscholar.org/paper/Auto-kNN-%3A-Brain-Tissue-Segmentation-using-Trained-Vrooman-Lijn/10e75df29ff9fed7e8d1d5cffa1b491f71d88c68.

Zhang, Yue, Wanli Chen, Yifan Chen, and Xiaoying Tang. 2018. “A Post-Processing Method to Improve the White Matter Hyperintensity Segmentation Accuracy for Randomly-Initialized U-Net,” July. https://arxiv.org/abs/1807.10600.

Zou, Kelly H, Simon K Warfield, Aditya Bharatha, Clare M C Tempany, Michael R Kaus, Steven J Haker, William M Wells, Ferenc A Jolesz, Ron Kikinis, and Ron Kikinis. 2004. “Statistical Validation of Image Segmentation Quality Based on a Spatial Overlap Index.” *Academic Radiology* 11 (2): 178–89. https://doi.org/10.1016/S1076-6332(03)00671-8.

7. Anexos

7.1 Resultado de la evaluación con FLEXCONN

Atlas	Error cuadrático medio	Entropía cruzada binaria	Distancia de Jaccard	Coeficiente de Dice
1	0.07969	0.07188	0.04546	0,045520
10	0,754530	0,733250	0,710780	0,740070
100	0,010030	0,010510	0,007950	0,011560
11	0,689650	0,670230	0,608720	0,653760
12	0,240010	0,242070	0,153020	0,197680
13	0,613370	0.57037	0,476050	0,565520
14	0,025470	0,032710	0.01545	0,024130
15	0,110900	0,190580	0,100480	0,191470
16	0,024080	0,030620	0,019860	0,022800
17	0,158510	0,188890	0,117850	0,179770
18	0,236080	0,291420	0,166730	0,232950
19	0,112410	0.11037	0,081910	0,126910
2	0.4199	0,315640	0.29347	0,412220
20	0,005790	0,015070	0,005820	0,006590
21	0,228990	0,257470	0,208220	0,246660
22	0,546900	0,390890	0,342700	0,443620
24	0,317630	0,317420	0,221540	0,233550
26	0,624040	0,300940	0,260010	0,492780
27	0,406730	0.26422	0,276420	0,322650
28	0,403920	0,383430	0.28756	0,361600
29	0,095260	0,139120	0,097360	0,145820
30	0,775180	0,644880	0,666090	0,781850

31	0,179010	0,139120	0,106000	0,137000
32	0,101240	0,113890	0,074160	0,177650
33	0.17027	0,162430	0,101280	0,189290
34	0,211730	0,193570	0.1202	0,184630
35	0,167570	0.14567	0,105640	0,193210
36	0,125440	0,100640	0,071120	0,097110
37	0,489200	0,396040	0,294040	0,531730
38	0,319600	0,168140	0,156420	0,158700
39	0,050590	0,093090	0,030180	0,055420
4	0,042690	0,034490	0,019130	0,043770
40	0,643640	0,663940	0.56769	0,638690
41	0,620670	0,531190	0,442570	0,601680
43	0,097030	0.11787	0,098390	0,130200
44	0,454960	0,514000	0,422540	0,586390
45	0,222730	0,189400	0,109230	0,187040
46	0,445390	0,292860	0,240420	0,310140
48	0,183080	0,194950	0,109140	0,164760
49	0.00544	0,014210	0.00452	0,006590
5	0,383660	0.28008	0,321450	0,437900
50	0,572370	0,474940	0,449630	0,527400
51	0,134830	0,015270	0,220160	0
52	0,124360	0,206980	0,116290	0,178350
53	0,391620	0,386870	0.33347	0.39274
54	0,504470	0,466850	0,358620	0,487470
55	0,329650	0,275990	0,186000	0,246820
56	0,012540	0.03254	0,013610	0,016200

57	0,108570	0,093870	0,056680	0,071270
58	0,326060	0,393870	0.33754	0,384600
59	0.18126	0,197200	0,111600	0,157300
6	0,213850	0,234510	0,140370	0,225010
60	0,154350	0,247660	0,147460	0,242970
61	0,027620	0,020200	0,013630	0,017370
62	0,451020	0,313650	0,255050	0,398400
63	0,034880	0.03535	0.02615	0,039460
64	0,026380	0,032830	0,025270	0,049500
65	0.18281	0,177580	0,132350	0,156450
66	0,087650	0,128850	0,089680	0,156450
67	0,383790	0,323220	0,211980	0.34727
68	0,097030	0,124650	0.06959	0,135590
69	0,335880	0,415060	0,279990	0,474260
7	0.39189	0,336760	0,263130	0,468670
70	0,229500	0.18589	0,133240	0,195330
71	0,067730	0,081500	0,048770	0,091950
72	0,487690	0,329450	0.34097	0,435880
73	0,273930	0,171180	0,131880	0,169740
74	0,135310	0,107320	0,087840	0.13996
75	0.01793	0,023140	0,018860	0,020040
76	0,141640	0,144000	0,100660	0,162480
77	0,365750	0,330880	0.31965	0,349670
78	0,255940	0,235580	0,169980	0,243930
79	0,002650	0,027800	0,105390	0,015040
8	0,042220	0,049450	0,026410	0,044710
80	0,008140	0,077980	0,345080	0,175060

81	0,104590	0,122560	0,066840	0,138910
82	0,204730	0,148350	0,131440	0,157960
83	0,436010	0,432520	0,331370	0.44312
84	0,097010	0,070940	0,056950	0,074570
85	0,000170	0,002420	0,000110	0,001240
86	0,043890	0,046610	0,031390	0,039780
87	0	0,197620	0,148200	0,055150
88	0,208350	0,136000	0,119590	0,217820
89	0,406610	0,312590	0,239780	0,314350
9	0,040390	0,057720	0,032790	0,054570
90	0,595740	0,412010	0.31471	0,557820
91	0.4051	0.33849	0.29882	0,451800
Dice promedio	0,246401	0,224072	0,183868	0,241331

Tabla 8: Resultado de la evaluación con FLEXCONN

7.2 Resultado de la evaluación con módulos *inception*

Atlas	Error cuadrático medio	Entropía cruzada binaria	Distancia de Jaccard	Coeficiente de Dice
1	0.11467	0.12597	0.12167	0.12195
10	0,801060	0,865310	0,835180	0,807140
100	0,013070	0,024830	0,053860	0,049150
11	0,735900	0,763110	0,773610	0,739980
12	0,370780	0,334090	0,383700	0,331930
13	0,501680	0.59646	0,611530	0,575470
14	0,043230	0,049860	0.05364	0,040470
15	0,245120	0,242020	0,314920	0,266400
16	0.05667	0,048210	0.05272	0,047820
17	0,185180	0,253240	0,290670	0,236190
18	0,376030	0,399080	0,468370	0,426080
19	0,170530	0.18883	0,230200	0.19318
2	0.55051	0.54861	0.56947	0,509020
20	0,017250	0,023370	0,028110	0,023060
21	0,354710	0,369560	0,401110	0,373100
22	0,556280	0,552050	0,585960	0,523690
24	0.38683	0,423030	0.50541	0,442850
26	0,518350	0,423760	0,638740	0,524460
27	0,468930	0.44141	0,456810	0,413020
28	0,565780	0.56498	0.60136	0,562560
29	0,143370	0,177980	0,192720	0,178450
30	0,776950	0,822160	0,843480	0,810500
31	0,273190	0,284310	0,308980	0,253020

32	0,222720	0,162830	0,267640	0,207800
33	0.26754	0,217570	0,314070	0.24814
34	0,342980	0.30175	0.31385	0,261000
35	0,389540	0.24251	0,289530	0,235730
36	0,202920	0,159600	0,241520	0,171800
37	0.48578	0,423400	0,613230	0,525130
38	0,465230	0,330580	0,375450	0,302170
39	0,065820	0,057130	0,061370	0,051100
4	0,054180	0.05935	0,078260	0.05873
40	0,731850	0,767390	0.79118	0,753600
41	0,619180	0,566420	0,618880	0,598060
43	0,183240	0.13186	0.24453	0,183340
44	0,599210	0,527350	0,588950	0,534040
45	0,307990	0,269170	0.29927	0.28493
46	0,397850	0,304330	0,426680	0,392530
48	0.34318	0.27849	0,307450	0,278100
49	0.01274	0,018310	0.00732	0,012380
5	0,449450	0.43831	0,525920	0,449950
50	0,522600	0,471940	0,624680	0,544230
51	0,178190	0,352800	0,489840	0,501700
52	0,162200	0,173880	0,267570	0,212840
53	0,538540	0,449390	0.50495	0.50576
54	0,471790	0.4274	0,545470	0,520600
55	0,431120	0,398120	0.42472	0,386610
56	0,037410	0.04102	0,050560	0,042340
57	0.19273	0,153810	0,217320	0,163700
58	0,518810	0,511900	0.56677	0.53603

59	0.27921	0,207580	0,279970	0,201190
6	0,414580	0,383400	0,343740	0,297590
60	0,328990	0,252110	0,370170	0,282890
61	0,029930	0.03624	0,048460	0,032230
62	0,513990	0,391320	0.5224	0,472130
63	0.05482	0.06079	0.07191	0,060070
64	0,063890	0,058650	0,101050	0,072430
65	0.33755	0,283060	0,300940	0,284940
66	0,214950	0,203190	0,289420	0,220710
67	0,401950	0,364280	0,410840	0.38186
68	0,254410	0.16567	0.24952	0,180510
69	0.52084	0,425020	0,542820	0,499010
7	0.45178	0,414270	0,517910	0,416260
70	0,283200	0.23008	0,332210	0.25159
71	0,119700	0,102990	0,153160	0,112640
72	0,414280	0,385640	0.48147	0,454400
73	0.33911	0,261640	0,341290	0,274430
74	0,202360	0.1679	0,190730	0.14654
75	0.04329	0,057850	0,066460	0,055950
76	0,160820	0,164450	0,231420	0,208970
77	0,476880	0,455490	0.5063	0,444760
78	0,309270	0,303710	0.37771	0,349050
79	0,040690	0.28444	0,370280	0,383120
8	0.05679	0,071000	0,071080	0,059090
80	0,153990	0,504330	0,592020	0,616540
81	0,192720	0,138650	0,190180	0,141060
82	0,216110	0,231570	0,266820	0.23578

83	0,521340	0,519480	0,577930	0.55282
84	0,151220	0,102420	0,113870	0,098500
85	0	0,000210	0,000150	0,002900
86	0,090780	0,061220	0,082350	0,065740
87	0,101350	0,413400	0,466980	0,457380
88	0,265190	0,222190	0,311690	0,273270
89	0,541670	0,477600	0,492860	0,456180
9	0,042970	0,054810	0,078290	0,046160
90	0,528780	0,436410	0.58159	0,517090
91	0.56146	0.47329	0.55363	0,470500
Dice promedio	0,311514	0,300295	0,354699	0,315955

Tabla 9: Resultado de la evaluación con módulos inception

7.3 Resultado de la evaluación con parches de 21x21

Atlas	Error cuadrático medio	Entropía cruzada binaria	Distancia de Jaccard	Coeficiente de Dice
1	0.060310	0.039440	0.036320	0.040420
10	0.703270	0.633050	0.574590	0.549490
100	0.024890	0.020830	0.031500	0.029690
11	0.741450	0.605410	0.572470	0.629930
12	0.228260	0.113200	0.107010	0.123520
13	0.375650	0.302220	0.256920	0.318720
14	0.038150	0.017040	0.014670	0.017370
15	0.126460	0.114550	0.133990	0.155960
16	0.025430	0.018950	0.017600	0.016030
17	0.062310	0.059110	0.060100	0.073850
18	0.391530	0.211480	0.213490	0.225330
19	0.051060	0.038320	0.035660	0.046970
2	0.374480	0.149150	0.152820	0.205990
20	0.009040	0.007940	0.006920	0.007410
21	0.235110	0.159030	0.145050	0.148620
22	0.392150	0.198730	0.216900	0.277200
24	0.479870	0.215470	0.240000	0.261590
26	0.261350	0.108710	0.110190	0.146030
27	0.223530	0.128680	0.115570	0.128440
28	0.426060	0.272370	0.272050	0.270490
29	0.048780	0.044990	0.047410	0.054180
30	0.504400	0.266270	0.258560	0.291290
31	0.106010	0.071160	0.065930	0.071800

32	0.084790	0.040990	0.045100	0.060880
33	0.107720	0.049910	0.047800	0.063560
34	0.130070	0.069030	0.062410	0.069450
35	0.123460	0.055660	0.057100	0.067470
36	0.106680	0.046930	0.058300	0.073020
37	0.323550	0.148250	0.119420	0.175970
38	0.204100	0.082370	0.069740	0.078850
39	0.013150	0.012910	0.012190	0.011690
4	0.018600	0.009450	0.009940	0.011600
40	0.608650	0.490390	0.470820	0.451570
41	0.586060	0.397740	0.417390	0.451730
43	0.030250	0.032620	0.036280	0.046670
44	0.392280	0.264120	0.109220	0.313470
45	0.174180	0.122210	0.109220	0.110780
46	0.178580	0.129910	0.132760	0.163520
48	0.196940	0.103240	0.090500	0.097080
49	0.005780	0.007050	0.005580	0.004930
5	0.204790	0.112080	0.119650	0.141780
50	0.327470	0.198190	0.190150	0.236040
51	0.470340	0.091200	0.027230	0.045160
52	0.072850	0.083750	0.089870	0.099250
53	0.578680	0.392530	0.423780	0.468570
54	0.354880	0.292840	0.304280	0.346220
55	0.278700	0.141810	0.124940	0.136160
56	0.015800	0.023050	0.023280	0.023120
57	0.199000	0.073780	0.069330	0.073960
58	0.397340	0.242550	0.288850	0.327900

59	0.224660	0.124080	0.118940	0.136460
6	0.180010	0.093730	0.068000	0.076890
60	0.215420	0.096130	0.106770	0.139430
61	0.017720	0.007390	0.007130	0.008680
62	0.258710	0.147630	0.136960	0.184270
63	0.024820	0.016450	0.020430	0.021530
64	0.012090	0.008730	0.011240	0.013150
65	0.313670	0.173430	0.181350	0.202320
66	0.089160	0.069770	0.084370	0.101990
67	0.260000	0.135330	0.153470	0.184880
68	0.105940	0.041440	0.037850	0.049370
69	0.270180	0.156640	0.184310	0.209140
7	0.192540	0.119200	0.118810	0.155770
70	0.168280	0.073500	0.081700	0.107820
71	0.039880	0.020920	0.020610	0.026100
72	0.314690	0.155910	0.140860	0.196040
73	0.243810	0.103870	0.098340	0.111800
74	0.094370	0.030760	0.026410	0.033960
75	0.022300	0.011100	0.008660	0.010860
76	0.059750	0.048240	0.054650	0.066610
77	0.325870	0.154750	0.181730	0.193820
78	0.191630	0.120220	0.148640	0.169090
79	0.525130	0.360150	0.249600	0.349400
8	0.008320	0.010270	0.009260	0.010030
80	0.490380	0.313110	0.134940	0.204980
81	0.046710	0.026180	0.027400	0.031970
82	0.196660	0.096070	0.111800	0.121140

83	0.387490	0.236610	0.267470	0.306870
84	0.078730	0.027780	0.025380	0.030070
85	0.000060	0.001790	0.001160	0.000780
86	0.060570	0.026580	0.030550	0.032480
87	0.474900	0.067470	0.035310	0.042130
88	0.059380	0.029960	0.034660	0.040000
89	0.426230	0.184810	0.202060	0.206390
9	0.002520	0.004570	0.004910	0.004750
90	0.248050	0.108390	0.088540	0.115270
91	0.243810	0.123810	0.134690	0.159500
Dice promedio	0.217801	0.126867	0.121262	0.141339

Tabla 10: Resultado de la evaluación con tamaño de parche 21x21

7.4 Resultado de la evaluación con parches de 25x25

Atlas	Error cuadrático medio	Entropía cruzada binaria	Distancia de Jaccard	Coeficiente de Dice
1	0.08327	0.0799	0.05338	0.09062
10	0,737370	0,762540	0,691410	0,695390
100	0,026770	0,025260	0,031730	0,037800
11	0,753240	0,763200	0,708750	0,781350
12	0,300830	0,210150	0,176090	0,264030
13	0.49231	0.50278	0.40717	0.4597
14	0,061310	0,031320	0.02382	0,039170
15	0,194320	0,223620	0,211420	0,259070
16	0.04	0,044140	0.02788	0,042290
17	0,115740	0,123150	0,082350	0,128580
18	0.42431	0.34707	0.32009	0.39639
19	0,079780	0.07998	0,047530	0.08278
2	0.42423	0.28094	0.25274	0,370880
20	0,032950	0,017550	0,011720	0,019980
21	0,323440	0,295300	0,202140	0,295080
22	0,455440	0,336800	0,337370	0,444110
24	0.49821	0,357530	0.38448	0,481480
26	0,337300	0,217380	0,206630	0,269630
27	0,264860	0.24273	0,173240	0,251150
28	0,511710	0.42542	0.37991	0,445190
29	0.09819	0.0849	0.05956	0.0959
30	0,562940	0,383070	0,357760	0,392260
31	0,134740	0,126890	0,091460	0,128610

32	0,131720	0,085170	0,077080	0,115000
33	0.18592	0,103020	0,078780	0.13029
34	0.18518	0.14357	0.10148	0.14742
35	0,172450	0.11434	0,089640	0,149270
36	0,132410	0,082640	0,086780	0,141240
37	0.43066	0,301380	0,237940	0,298490
38	0,248860	0,159300	0,124620	0,173180
39	0.02763	0.02122	0.0147	0.02421
4	0,027610	0.02105	0,016130	0.02521
40	0,650130	0,599730	0.57977	0,588370
41	0,598910	0,515800	0,488070	0,520490
43	0,054210	0.07858	0.05393	0,087100
44	0.49073	0.40372	0.36876	0.42963
45	0,271470	0,208560	0.15359	0.20919
46	0,247340	0,216460	0,220220	0,260460
48	0.26992	0.19316	0,143250	0,216380
49	0.0123	0,013110	0.00567	0,012070
5	0,254770	0.20715	0.18053	0.26208
50	0,382360	0,321310	0,309150	0,352530
51	0,635460	0,307470	0,076690	0,087890
52	0.11639	0,153610	0,123560	0,157890
53	0,586760	0,474250	0.53433	0.54473
54	0,425490	0.42361	0,412060	0,451550
55	0,349250	0.26217	0.19123	0,270900
56	0.02975	0.04165	0,027190	0,038570
57	0.25675	0,147090	0,123510	0,165110
58	0,457870	0,357090	0.38535	0.472240

59	0.30948	0,213150	0,178700	0,254390
6	0,253460	0,186490	0.11031	0.158110
60	0,294640	0,185080	0,171480	0,258460
61	0,034290	0.01655	0,014090	0,025630
62	0.35499	0.27592	0.23157	0,275320
63	0.03077	0.02554	0.02463	0,041080
64	0,026680	0,019820	0,014660	0,027610
65	0.38482	0,271950	0,274360	0,335870
66	0,139890	0,145980	0,118640	0.17661
67	0,332630	0,242690	0.22765	0.311570
68	0.17687	0.08709	0.06168	0,110610
69	0.36109	0,252220	0,250930	0,337000
7	0.26447	0,231640	0,189880	0,249140
70	0,215740	0.14179	0,127160	0.199260
71	0,063250	0,042050	0,032440	0,054280
72	0,386420	0,270090	0.24598	0,263640
73	0.3394	0.20011	0.16237	0,236850
74	0.16046	0.06511	0,046970	0.076440
75	0.03597	0,023300	0,016090	0,027080
76	0,091260	0,088510	0,086910	0,111670
77	0,383410	0,268040	0.25325	0,349690
78	0,244960	0,195560	0.19589	0,278030
79	0,451420	0.54047	0,438680	0.412120
8	0.02597	0,024060	0,010680	0,020500
80	0,484380	0,509880	0,306890	0,334160
81	0,075780	0,053280	0,038970	0,061940
82	0,227740	0.17505	0,160440	0.23557

83	0,488960	0,367650	0.36233	0.447950
84	0.10837	0,059200	0,045150	0,073090
85	0,000240	0,002850	0,000550	0,000510
86	0,067950	0,048890	0,043970	0,066280
87	0,600840	0,242230	0,091870	0,148840
88	0,093910	0,063030	0,049030	0.080030
89	0,419300	0,289710	0,292080	0,356660
9	0,011380	0,016190	0,005790	0.00945
90	0,394990	0,221470	0.14153	0,224010
91	0.32498	0.21743	0.19319	0.28866
Dice promedio	0,267204	0,209171	0,179993	0,226679

Tabla 11: Resultado de la evaluación con tamaño de parche 25x25

7.5 Resultado de la evaluación con parches de 31x31

Atlas	Error cuadrático medio	Entropía cruzada binaria	Distancia de Jaccard	Coeficiente de Dice
1	0.161750	0.176350	0.127620	0.142380
10	0.787090	0.822720	0.769180	0.797890
100	0.039550	0.048630	0.045710	0.062140
11	0.789380	0.834030	0.792220	0.820660
12	0.474300	0.418310	0.384820	0.402980
13	0.585990	0.700240	0.564250	0.636540
14	0.156970	0.106440	0.099290	0.081920
15	0.358600	0.432790	0.394430	0.497410
16	0.112020	0.101570	0.068700	0.089950
17	0.214710	0.288360	0.200120	0.279090
18	0.571950	0.508920	0.480670	0.491560
19	0.173740	0.198160	0.145520	0.185280
2	0.601100	0.516480	0.529460	0.537320
20	0.094660	0.065930	0.043540	0.054680
21	0.460190	0.482290	0.388990	0.438240
22	0.609530	0.578540	0.595550	0.601650
24	0.626470	0.571740	0.560210	0.573420
26	0.538420	0.357630	0.487310	0.563460
27	0.418150	0.369730	0.284420	0.366830
28	0.638920	0.610820	0.514070	0.626590
29	0.194910	0.205760	0.186340	0.252810
30	0.591110	0.475900	0.502250	0.536190
31	0.213170	0.204220	0.176040	0.213800

32	0.269120	0.213150	0.269540	0.393060
33	0.356620	0.229570	0.233840	0.336400
34	0.325320	0.244780	0.193760	0.257390
35	0.321360	0.241610	0.199940	0.301840
36	0.259270	0.175400	0.199950	0.279990
37	0.560960	0.505740	0.539630	0.618120
38	0.421720	0.276990	0.227830	0.314100
39	0.047050	0.060830	0.037170	0.062040
4	0.080200	0.059610	0.060690	0.068540
40	0.689830	0.660210	0.640020	0.671830
41	0.637850	0.590460	0.594820	0.605890
43	0.153970	0.163770	0.202220	0.319160
44	0.569940	0.548430	0.562680	0.615280
45	0.401550	0.368310	0.303120	0.392510
46	0.347930	0.351780	0.378880	0.457860
48	0.466720	0.350380	0.294680	0.390530
49	0.020100	0.034310	0.015920	0.017760
5	0.392660	0.377790	0.372040	0.421200
50	0.524630	0.479850	0.535770	0.602200
51	0.587130	0.470820	0.470370	0.475970
52	0.148670	0.250600	0.267620	0.320270
53	0.607350	0.553240	0.572400	0.543760
54	0.507450	0.546330	0.540590	0.579130
55	0.535370	0.427230	0.389750	0.419930
56	0.055290	0.082700	0.045590	0.080880
57	0.462900	0.264780	0.280580	0.272910
58	0.569450	0.526700	0.592670	0.594080

59	0.442770	0.354700	0.336370	0.352860
6	0.423400	0.386620	0.301100	0.294240
60	0.453320	0.326500	0.433160	0.429880
61	0.101900	0.048290	0.055350	0.057900
62	0.449130	0.428720	0.455750	0.482750
63	0.061650	0.052750	0.054650	0.067060
64	0.068130	0.049670	0.067360	0.072830
65	0.511270	0.443810	0.469080	0.459510
66	0.261780	0.277750	0.304260	0.340830
67	0.492830	0.419090	0.430650	0.462350
68	0.348550	0.201020	0.263210	0.260040
69	0.516980	0.440070	0.508280	0.539000
7	0.437800	0.433420	0.460040	0.481210
70	0.350340	0.281900	0.353340	0.369030
71	0.150990	0.095000	0.104670	0.141430
72	0.461130	0.402940	0.455160	0.433230
73	0.512020	0.345170	0.340200	0.371790
74	0.340670	0.151300	0.201470	0.206470
75	0.071810	0.047450	0.040090	0.048310
76	0.203130	0.261940	0.269420	0.360300
77	0.530000	0.427780	0.417730	0.525530
78	0.342480	0.375630	0.337000	0.415720
79	0.325450	0.518760	0.273200	0.349400
8	0.055940	0.071570	0.041190	0.070640
80	0.415750	0.639870	0.552370	0.633770
81	0.163670	0.136150	0.113450	0.180270
82	0.359320	0.314630	0.265920	0.318840

83	0.603320	0.577070	0.550240	0.618860
84	0.223990	0.130360	0.111610	0.140380
85	0.000580	0.002190	0.000120	0.000530
86	0.137330	0.102550	0.082290	0.107840
87	0.659200	0.560730	0.293040	0.424410
88	0.203720	0.176560	0.148040	0.227830
89	0.567730	0.454930	0.424210	0.480130
9	0.045020	0.070130	0.034230	0.086340
90	0.597860	0.489380	0.447630	0.526360
91	0.525770	0.444280	0.429320	0.520360
Dice promedio	0.369538	0.334156	0.319402	0.362341

Tabla 12: Resultado de la evaluación con tamaño de parche 31x31

7.6 Resultado de la evaluación con parches de 35x35

Atlas	Error cuadrático medio	Entropía cruzada binaria	Distancia de Jaccard	Coeficiente de Dice
1	0.19858	0.29975	0.23258	0.17066
10	0,779070	0,869900	0,847140	0,775080
100	0,050270	0,060760	0,094660	0,051590
11	0,790880	0,872220	0,861920	0,815000
12	0,529040	0,592320	0,532990	0,477430
13	0.60053	0.71215	0.64845	0.57981
14	0,169880	0,202420	0.16488	0,172570
15	0,373910	0,519000	0,540790	0,423090
16	0.12942	0,197120	0.16167	0,119160
17	0,280160	0,410640	0,472980	0,315870
18	0.57639	0.62586	0.61225	0.5534
19	0,212940	0.37203	0,347440	0.23419
2	0.64091	0.68935	0.71726	0,665340
20	0,079850	0,123750	0,114840	0,089080
21	0,485870	0,569540	0,527980	0,471390
22	0,661810	0,698320	0,730120	0,682160
24	0.64281	0,682530	0.67699	0,627380
26	0,566910	0,608350	0,713580	0,660080
27	0,461670	0.53784	0,510620	0,409820
28	0,648870	0.70723	0.68096	0,613410
29	0.23866	0.33856	0.36741	0.30316
30	0,617660	0,587490	0,568950	0,550800
31	0,281990	0,303810	0,279020	0,233720

32	0,341790	0,395760	0,557680	0,470260
33	0.44142	0,388250	0,468530	0.45077
34	0.34545	0.34615	0.32024	0.30091
35	0,421700	0.43948	0,398960	0,319990
36	0,327140	0,379170	0,435770	0,385560
37	0.5664	0,635520	0,626730	0,564190
38	0,457170	0,436920	0,398460	0,363080
39	0.07142	0.07956	0.10098	0.07875
4	0.10413	0.13245	0.1684	0.14357
40	0,693840	0,676770	0.68721	0,663490
41	0,643430	0,627470	0,619880	0,614350
43	0,180090	0.31692	0.35749	0,348270
44	0.61649	0.62729	0.62815	0.61191
45	0.44854	0.46845	0.4579	0.41064
46	0,394500	0,423170	0,449790	0,390960
48	0.49956	0.51453	0,524090	0,447120
49	0.0208	0,046710	0.02562	0,013580
5	0,443600	0.52855	0.57312	0.51452
50	0.54726	0.56687	0.64126	0.61588
51	0,581550	0,589710	0,480050	0,382460
52	0.1928	0,268660	0,365950	0,314900
53	0,655410	0,581800	0.55091	0.54212
54	0,563510	0.57298	0,600470	0,555020
55	0,560260	0.59035	0.55038	0,514040
56	0.06618	0.12299	0,094070	0,063230
57	0.48736	0,436650	0,433210	0,383880
58	0,626200	0,614500	0.68085	0.65378

59	0.48351	0,478940	0,470990	0,429560
6	0,484140	0,554810	0.48438	0.43195
60	0,505670	0,479940	0,579050	0,562700
61	0.10594	0.12391	0.15962	0.12065
62	0.5378	0.52151	0.56437	0,487490
63	0.08055	0.10433	0.12098	0,092930
64	0,089220	0,131060	0,211480	0,112210
65	0.57034	0,518080	0,564010	0,530660
66	0,304050	0,381600	0,435570	0.37945
67	0,524660	0,534480	0.56994	0.53924
68	0.4437	0.40559	0.47724	0,476290
69	0.57315	0,555480	0,643410	0,627050
7	0.49223	0,589190	0,688840	0,609010
70	0.42463	0.42934	0.53727	0.47919
71	0,190560	0,198440	0,283640	0,249810
72	0,481980	0,468980	0.51271	0,459560
73	0.52858	0.5053	0.5152	0,452090
74	0.42237	0.3441	0.41357	0.37701
75	0.10148	0,096110	0,093300	0,080550
76	0,272770	0,375320	0,540160	0,426330
77	0,572730	0,579570	0.63899	0,591770
78	0,437920	0,435870	0.50994	0,447100
79	0,333460	0.43084	0,308220	0.10927
8	0.07347	0,116130	0,112000	0,097630
80	0,463870	0,659810	0,545200	0,354240
81	0,216100	0,218400	0,306360	0,238210
82	0,373440	0.40923	0,422440	0.36769

83	0,626690	0,612830	0.67517	0.63894
84	0.23317	0,233700	0,217410	0,222800
85	0,000980	0,007390	0	0
86	0,160440	0,174470	0,179610	0,126630
87	0,650200	0,584390	0,513080	0,539580
88	0,288470	0,301370	0,393420	0.35045
89	0,586170	0,556600	0,555850	0,555290
9	0,078950	0,109460	0,246340	0.08766
90	0,638310	0,574100	0.65268	0,657760
91	0.56359	0.57063	0.62411	0.61481
Dice promedio	0,404958	0,434021	0,452508	0,402701

Tabla 13: Resultado de la evaluación con tamaño de parche 35x35

7.7 Resultado de la evaluación con parches de 41x41

Atlas	Error cuadrático medio	Entropía cruzada binaria	Distancia de Jaccard	Coeficiente de Dice
1	0.339660	0.282100	0.173350	0.114380
10	0.830240	0.887420	0.818740	0.792950
100	0.071510	0.08149	0.062350	0.057090
11	0.804800	0.864140	0.799770	0.769570
12	0.664080	0.559100	0.494290	0.380140
13	0.622140	0.709190	0.567680	0.588730
14	0.355850	0.181820	0.199310	0.122200
15	0.437070	0.541970	0.448800	0.504160
16	0.235860	0.177340	0.117720	0.072850
17	0.443270	0.414390	0.429850	0.249440
18	0.686260	0.646400	0.579920	0.485550
19	0.376750	0.347650	0.282050	0.170400
2	0.747350	0.662340	0.697490	0.580760
20	0.202530	0.133080	0.075580	0.042520
21	0.569340	0.572670	0.469290	0.400600
22	0.751910	0.692950	0.725670	0.616820
24	0.717180	0.668270	0.616320	0.552910
26	0.659210	0.678300	0.711690	0.666930
27	0.604050	0.562040	0.392390	0.322980
28	0.712600	0.710590	0.616980	0.571610
29	0.352490	0.384820	0.360640	0.238200
30	0.591270	0.581980	0.593120	0.598770
31	0.328730	0.343770	0.239480	0.201220

32	0.481540	0.481580	0.539470	0.422720
33	0.540890	0.476740	0.464320	0.387830
34	0.457370	0.402920	0.282750	0.255380
35	0.543390	0.466640	0.278480	0.232070
36	0.468760	0.417830	0.382320	0.344770
37	0.576760	0.620200	0.611500	0.608820
38	0.582970	0.489610	0.333450	0.300400
39	0.173650	0.103900	0.12257	0.069220
4	0.238060	0.142210	0.203890	0.101170
40	0.718130	0.701450	0.675750	0.684250
41	0.624620	0.629040	0.611590	0.635940
43	0.304100	0.336850	0.381760	0.323200
44	0.605230	0.610470	0.638540	0.623500
45	0.491510	0.506390	0.421550	0.367910
46	0.422290	0.455050	0.412620	0.428390
48	0.614900	0.532240	0.430810	0.381100
49	0.047820	0.034800	0.035630	0.012340
5	0.558290	0.550340	0.568510	0.447190
50	0.586190	0.572670	0.617750	0.627250
51	0.505750	0.543560	0.417840	0.378820
52	0.223010	0.285130	0.333610	0.341260
53	0.610720	0.565880	0.539060	0.576750
54	0.586700	0.601740	0.578330	0.589700
55	0.656030	0.574680	0.524640	0.403190
56	0.109510	0.138380	0.070180	0.051570
57	0.634630	0.432330	0.408230	0.307570
58	0.688880	0.637600	0.715810	0.649200

59	0.557000	0.467460	0.421380	0.344050
6	0.618020	0.524360	0.467710	0.313820
60	0.578290	0.492120	0.591490	0.481120
61	0.270490	0.115730	0.106610	0.059050
62	0.530960	0.502260	0.516470	0.507290
63	0.154950	0.116750	0.109270	0.057640
64	0.194090	0.137880	0.209950	0.088890
65	0.575580	0.553370	0.564470	0.500410
66	0.363510	0.436600	0.457060	0.381980
67	0.619840	0.549840	0.613820	0.524930
68	0.601220	0.387630	0.536370	0.344670
69	0.662200	0.590300	0.695030	0.593550
7	0.616790	0.623760	0.701910	0.588790
70	0.542480	0.442580	0.538410	0.445500
71	0.416950	0.212070	0.314530	0.167370
72	0.519110	0.501820	0.481770	0.485910
73	0.617450	0.486450	0.470040	0.383180
74	0.569480	0.323240	0.458010	0.315240
75	0.170550	0.099010	0.084210	0.048100
76	0.413240	0.46009	0.524980	0.386020
77	0.646440	0.60876	0.604340	0.552510
78	0.502070	0.50564	0.482760	0.434140
79	0.224740	0.43026	0.096250	0.111510
8	0.133210	0.154230	0.151240	0.082710
80	0.390390	0.60307	0.285650	0.380110
81	0.340370	0.27278	0.287320	0.184770
82	0.475870	0.44825	0.328350	0.297950

83	0.667250	0.63776	0.677650	0.632330
84	0.356990	0.24907	0.181420	0.143250
85	0	0.00909	0.001810	0
86	0.247950	0.18158	0.113950	0.092590
87	0.615670	0.62079	0.525260	0.553480
88	0.434580	0.3656	0.383200	0.296220
89	0.651070	0.57538	0.507680	0.480730
9	0.229030	0.174470	0.187500	0.097680
90	0.699050	0.644680	0.682550	0.610930
91	0.667760	0.629690	0.635900	0.560440
Dice promedio	0.483109	0.449753	0.426089	0.369898

Tabla 14: Resultado de la evaluación con tamaño de parche 41x41

7.8 Resultado de la evaluación con parches de 45x45

Atlas	Error cuadrático medio	Entropía cruzada binaria	Distancia de Jaccard	Coeficiente de Dice
1	0.21593	0.41587	0.22112	0.29887
10	0,801740	0,142490	0,840310	0,821640
100	0,034790	0,883430	0,083700	0,083290
11	0,766600	0,871210	0,813920	0,845520
12	0,550480	0,641350	0,535570	0,628850
13	0.60132	0.68667	0.53131	0.49578
14	0,163100	0,304590	0.23246	0,295140
15	0,375050	0,506820	0,413460	0,384840
16	0.1655	0,226820	0.1524	0,227750
17	0,310490	0,572460	0,385300	0,428750
18	0.60897	0.69679	0.63551	0.6988
19	0,216850	0.52404	0,278020	0.31909
2	0.65112	0.76878	0.71193	0,745930
20	0.06973	0.15575	0.10696	0.06583
21	0,474210	0,612390	0,501500	0,560130
22	0,646500	0,792880	0,747500	0,769190
24	0.61374	0,719370	0.64059	0,718720
26	0,609200	0,724210	0,735800	0,726710
27	0,474440	0.64676	0,459040	0,590470
28	0,653500	0.73488	0.65585	0,671460
29	0.23527	0.43796	0.40662	0.37064
30	0,645660	0,545500	0,539560	0,501060
31	0.27729	0.35441	0.26788	0.30605

32	0,363480	0,592720	0,605810	0,568770
33	0.43366	0,537620	0,518860	0.54783
34	0.34012	0.4322	0.34059	0.39223
35	0,394770	0.54226	0,372580	0,445900
36	0,336260	0,557360	0,487890	0,622810
37	0.55352	0,616100	0,535640	0,465350
38	0,456690	0,526530	0,428360	0,509430
39	0.07493	0.16597	0.10651	0.16268
4	0,115960	0.29395	0,259030	0.3238
40	0,720450	0,675290	0.71301	0,665500
41	0.65191	0.6312	0.6087	0.58971
43	0,231240	0.39022	0.34321	0,283970
44	0.63924	0.61426	0.62425	0.57898
45	0,469680	0,523170	0.49251	0.47613
46	0,401150	0,420560	0,348640	0,300920
48	0.52129	0.60714	0,503190	0,583720
49	0.01075	0,058700	0.00832	0,016510
5	0,457190	0.65799	0.58239	0.59539
50	0,570070	0,613110	0,597190	0,547090
51	0.60297	0.5716	0.40102	0.33663
52	0.25996	0,303870	0,302920	0,213030
53	0,664160	0,484010	0.51936	0.48631
54	0,606330	0.59163	0,568930	0,493520
55	0,560730	0.63759	0.56089	0,646460
56	0.0767	0.18273	0,083940	0,102840
57	0.50411	0,510400	0,479380	0,573330
58	0,638680	0,688060	0.70507	0.71121

59	0.4971	0,566380	0,478130	0,514200
6	0,484590	0,614880	0.47763	0.58986
60	0,490210	0,598650	0,615650	0,609830
61	0.11338	0.20396	0.14737	0.24866
62	0.57697	0.48287	0.49589	0,438070
63	0.08013	0.21153	0.12932	0,228340
64	0,115060	0,405050	0,211350	0,270540
65	0.58855	0,588990	0,601350	0,590620
66	0,311320	0,454440	0,425760	0.40427
67	0,553410	0,585050	0.62759	0.60273
68	0.45468	0.55122	0.60547	0,656220
69	0.57692	0,640140	0,709510	0,680390
7	0.51846	0,726740	0,685810	0,676060
70	0,415500	0.58649	0,584730	0.62217
71	0.20652	0.38491	0.37649	0.53158
72	0,500830	0,525360	0.44017	0,388870
73	0.51772	0.56612	0.50956	0,578250
74	0.39863	0.48887	0,517220	0.59916
75	0.08862	0,137640	0,102550	0,158500
76	0,323560	0,538230	0,533310	0,501750
77	0,579400	0,659900	0.6617	0,686500
78	0,440100	0,510890	0.53567	0,505690
79	0,362480	0.3444	0,076220	0.06602
8	0.06491	0,160900	0,124000	0,194980
80	0,462500	0,551990	0,172020	0,220700
81	0,217570	0,416230	0,326650	0,418710
82	0,367130	0.47377	0,396210	0.48077

83	0,639550	0,661560	0.66948	0.64842
84	0.23972	0,277300	0,250570	0,383380
85	0	0	0	0
86	0,148400	0,226040	0,146180	0,223330
87	0,648200	0,625100	0,626440	0,586030
88	0,292470	0,536200	0,470110	0.54341
89	0,598470	0,611590	0,601590	0,669810
9	0,045100	0,270270	0,189440	0.08589
90	0,640760	0,684820	0.70893	0,694710
91	0.56946	0.68493	0.67249	0.72951
Dice promedio	0,413171	0,505116	0,447093	0,469224

Tabla 15: Resultado de la evaluación con tamaño de parche 45x45