Departamento de Computación
Facultad de Informática
UNIVERSIDADE DA CORUÑA

TRABAJO FIN DE GRADO
GRADO EN INGENIERÍA INFORMÁTICA
MENCIÓN EN COMPUTACIÓN

Estudio y diseño de modelos de aprendizaje profundo para generación multimodal de imagen oftalmológica

Estudiante: José Morano Sánchez
Directores: Jorge Novo Buján
José Rouco Maseda

A Coruña, 5 de septiembre de 2019.

A un cuarto sin ascensor

Agradecimientos

Quiero aprovechar estas líneas para dar las gracias a mis dos directores de proyecto, Jorge Novo Buján y José Rouco Maseda. A iniciativa suya empezó este trabajo, y bajo su atenta dirección ha podido avanzar y alcanzar finalmente la forma de esta memoria.

También, a Álvaro Suárez Hervella, en cuya línea de investigación se integra este proyecto. La ayuda prestada en algunos momentos, junto con unas cuantas explicaciones, me han ahorrado muchas preocupaciones y me han resuelto muchas dudas.

Por último, agradecer a Luis Mª Hervella Nieto las opiniones compartidas, que tanto han influido en un camino que aún no ha terminado.

Resumen

En el ámbito oftalmológico, para realizar algunos procesos de diagnóstico, se suele disponer de diferentes modalidades de imagen del fondo de ojo. Generalmente, estas son de dos tipos: *invasivas* —implican la agresión del organismo—, y *no-invasivas* —no implican tal agresión y suelen ser más baratas y sencillas de obtener—. En este trabajo, se estudian y adaptan diferentes modelos de aprendizaje profundo (redes de neuronas convolucionales) para la generación de una modalidad de imagen invasiva, la angiografía, a partir de otra no invasiva, la retinografía, en lo que se conoce como *paradigma de reconstrucción multimodal*. Además, el entrenamiento de estos modelos ha permitido estudiar su utilidad como preentrenamientos de una segunda tarea de carácter finalista, la segmentación de vasos en retinografía, mediante la aplicación de la técnica de *transfer learning*.

La comparación de los resultados de estos modelos para las dos tareas con los de uno de referencia (con un diseño más clásico y sencillo) ha permitido comprobar dos cosas. En primer lugar, que los resultados de estos modelos, aunque se acercan, no alcanzan en muchos casos a los del modelo de referencia. Y en segundo lugar, que la utilización de modelos preentrenados, independientemente de la arquitectura empleada, tiene un impacto positivo en los resultados. Con ellos, las segmentaciones son mejores con un menor tiempo de entrenamiento.

La primera de estas observaciones nos han permitido, primero, cuestionar la relevancia de las variantes arquitecturales y principios de diseño de los modelos nuevos en el dominio escogido; y segundo, confirmar la adecuación de dichos modelos generativos a la tarea de reconstrucción multimodal, permitiendo validar el propio paradigma.

Por otra parte, la segunda de las observaciones nos ha permitido confirmar la utilidad de los preentrenamientos de la tarea de reconstrucción. Cuando se dispone de pocos datos etiquetados, este tipo de preentrenamientos surgen como una buena opción para mitigar esa escasez y mejorar así los resultados.

Abstract

In the ophthalmological field, different modalities of fundus imaging are usually available to perform some diagnosis. These modalities are of two types: invasive —they involve the aggression of the organism—, and non-invasive —they do not imply such aggression and are usually cheaper and easier to obtain—. In this work are studied and adapted different deep learning models (convolutional neural networks) to generate an invasive imaging modality —angiography— from another non-invasive modality —retinography—. This is known as *multimodal reconstruction paradigm*. In addition, the training of these models has allowed us to

study their usefulness as pre-trainined models of a second finalist task: vessel segmentation in retinography. This is achieved through the application of transfer learning technique.

The results obtained by the selected models in both tasks were compared to those of a reference model with a more classic and simple design. This comparison showed two points. First, that the results of the selected models did not reach in many cases those of the reference model. Second, that the use of pre-trained models, regardless of the architecture used, has a positive impact on the results. When transfer learning is applied, segmentation images are better with less training time.

The first point allowed us to question the relevance of the architectural variants and design principles of the selected models in the chosen domain. Moreover, it confirmed the adequacy of generative models to the task of multimodal reconstruction, validating the paradigm itself.

On the other hand, the second point allowed us to confirm the usefulness of pre-trainings from the reconstruction task. This type of pre-training is able to mitigate data scarcity in finalist tasks, and therefore improve the results.

Palabras clave:

- Aprendizaje profundo
- Imagen médica
- Aprendizaje transferido
- Redes de neuronas convolucionales
- Imagen oftalmológica

Keywords:

- Deep learning
- Medical imaging
- Transfer learning
- Convolutional neural networks
- Ophthalmological imaging

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1
Introducción

En este capítulo introductorio se expondrán los problemas principales a abordar en el proyecto, las motivaciones por las que nace y los objetivos que se ha dispuesto conseguir. Finalmente, se detallará la estructura de la memoria y los apartados que la conforman.

1.1 El proyecto, sus motivaciones y sus objetivos

1.1.1 Descripción

En el ámbito oftalmológico es frecuentemente necesario disponer de diferentes modalidades de imagen del fondo de ojo para realizar algunos procesos de diagnóstico. Entre ellas podemos encontrar las denominadas modalidades *invasivas*, que implican la agresión del organismo (e.g. a través de una inyección o una incisión), y las *no-invasivas*, que no implican tal agresión y suelen ser más baratas y sencillas de obtener. La angiografía, que requiere la inyección de un químico fluorescente en el torrente sanguíneo del paciente, es un ejemplo del primer tipo, y la retinografía, que como mucho implica poner unas gotas en el ojo, del segundo. En este proyecto se plantea el estudio, diseño y adaptación de diferentes modelos de aprendizaje profundo para la generación de una modalidad de imagen invasiva, la angiografía, a partir de otra no-invasiva, la retinografía; una tarea que se enmarca dentro del paradigma conocido como *paradigma de reconstrucción multimodal*.

Concretamente, el foco del proyecto se centrará en el estudio de redes de neuronas convolucionales con variantes arquitecturales innovadoras de demostrada utilidad en diversas tareas de referencia en el ámbito del aprendizaje máquina, como la segmentación semántica.

La adecuación de dichas redes o modelos generativos permitirá dos cosas. Primero, validar la idoneidad del paradigma de *reconstrucción multimodal*. Y segundo, estudiar la validez diagnóstica de las imágenes generadas y la potencial utilidad de dichos modelos en procesos de reconocimiento visual de partes representativas del fondo de ojo (como vasos, fóvea o disco óptico, entre otras) e indicios de patologías que puedan estar presentes (edemas, microaneurismas, drusas, etc.). Para esto, incorporaremos además la experimentación con la técnica de *transfer learning* (aprendizaje transferido), consistente en la utilización de modelos preentrenados como punto de partida para el entrenamiento de esos mismos modelos en otras tareas o conjuntos de datos. En nuestro caso, se utilizarán las redes entrenadas para la reconstrucción

multimodal como preentrenamientos de una segunda tarea de carácter finalista: la segmentación de vasos sanguíneos en retinografías. Una tarea, además, de gran interés para el análisis de diversas patologías en las que es clave el estudio del árbol arterio-venoso.

Una vez terminado todo el proceso de experimentación, y en base a los datos experimentales, se valorarán objetivamente los siguientes puntos: a) la adecuación o no de las arquitecturas analizadas y las ideas que implementan al ámbito de imagen médica que nos ocupa: la imagen oftalmológica; y b) la validez de las técnicas y modelos empleados en relación a las tareas objetivo en ese mismo campo.

1.1.2 Motivación

El estudio de nuevos paradigmas y arquitecturas de red, así como de las técnicas de las que hacen uso, es especialmente importante cuando se quiere conocer el camino adecuado a seguir en una línea de investigación. El crecimiento que está experimentando el campo del aprendizaje profundo en los últimos años, con la aparición de gran cantidad de arquitecturas y técnicas de aprendizaje, hace que no sea fácil conocer cuáles de estas técnicas, arquitecturas o elementos arquitectónicos son realmente interesantes en ciertos escenarios y cuáles no.

El análisis de aspectos como estos puede llevar a tomar mejores decisiones de diseño o de elección de las tecnologías a emplear, lo que llevaría en última instancia a unos mejores resultados. Cuando estas decisiones, además, se tienen que tomar en tareas relacionadas con la imagen médica, donde cualquier avance podría tener un impacto directo en la medicina, este conocimiento para una adecuada selección y utilización de las herramientas de las que se dispone multiplica aún más su importancia.

Generalmente, en este ámbito, y en el de imagen en general, los modelos más utilizados son las redes de neuronas convolucionales. El buen funcionamiento que vienen demostrando y mejorando en los últimos años las sitúa, por el momento, como la mejor opción entre el amplio abanico de la inteligencia artificial. En tareas complejas y de referencia como la clasificación de imágenes *naturales*, la segmentación semántica de fotografías o la segmentación de vasos sanguíneos en imágenes de fondo de ojo, este tipo de redes ocupan siempre los primeros puestos a nivel de resultados, y no parece que en el futuro inmediato esto vaya a cambiar.

Demostrar la validez del paradigma de aprendizaje transferido (*transfer learning*) en escenarios con varias modalidades de imagen (retinografías y angiografías) para varias arquitecturas de redes de neuronas convolucionales destacadas, y analizar la relevancia de los elementos constituyentes de cada una de ellas y de los procesos de regularización, inicialización y *data augmentation*, puede reportar grandes beneficios a medio plazo. La identificación de los elementos de mayor potencial, aquellos que más directamente pueden afectar al desempeño de la red, permite hacer una selección adecuada de los mismos y concentrar esfuerzos en su perfeccionamiento, lo que suele venir acompañado de mejoras significativas en los resultados.

1.1.3 Objetivos

El proyecto se divide en dos partes diferenciadas: una correspondiente a todo lo relacionado con la tarea base de reconstrucción multimodal, y otra con todo lo relacionado con la tarea finalista de segmentación de vasos retinianos. La primera comprenderá la experimentación con las diferentes arquitecturas y sus elementos de construcción. La segunda comprenderá la experimentación con las arquitecturas seleccionadas de la parte anterior y con la técnica de aprendizaje transferido, que nos permitirá cuantificar precisamente el potencial de los preentrenamientos en la tarea de reconstrucción.

Los objetivos de la **primera parte** se centran en estudiar, adaptar e implementar los modelos de aprendizaje profundo U-Net [1], FC-DenseNet [2] y ENet [3] para la reconstrucción multimodal de angiografía a partir de retinografía (donde llamaremos a la imagen reconstruida *pseudoangiografía*). La obtención de las redes entrenadas en esta tarea permitirá validar su buen o mal funcionamiento dentro del paradigma, y analizar en cada caso el impacto de ciertos elementos arquitectónicos y metodológicos en la consecución del resultado. La mayor parte de las pruebas a este respecto se concentrarán en la arquitectura FC-DenseNet, considerada, por su constatado buen funcionamiento en tareas de segmentación, la de mayor capacidad y potencial para hacer frente a U-Net, la arquitectura que tomaremos de referencia. A lo largo de todo este proceso, la pretensión es ser capaces de identificar los elementos arquitectónicos o metodológicos que hacen mejorar los resultados de las redes, y por tanto, aquellos en los que sería más interesante profundizar. No obstante, si se diera el caso opuesto, y no se pudiera identificar como especialmente ventajoso alguno o ninguno de los nuevos elementos (al menos integrados en las arquitecturas objeto de estudio), una discusión también sería relevante. El hecho de no detectar una mejora significativa en los resultados podría abrir la pregunta de si estos elementos son realmente convenientes. Además, si los resultados de las redes «nuevas» no mejoran los de U-Net, se podría discutir si las nuevas tendencias a nivel de arquitectura son verdaderamente adecuadas en dominios como este de imagen oftalmológica.

Ya en la **segunda parte**, los objetivos principales son dos. Primero, validar el buen funcionamiento de las arquitecturas seleccionadas (y de las variantes destacadas que de ellas se hayan hecho en la primera fase) en una tarea finalista. Concretamente, la tarea de segmentación de vasos sanguíneos en retinografías. Y segundo, valorar el impacto de la aplicación de la técnica de aprendizaje transferido mediante la utilización de modelos preentrenados en la tarea de reconstrucción. Para cumplir estos objetivos, se experimentará en la tarea de segmentación con los modelos inicializados (1) a través de métodos automáticos y (2) mediante la técnica de aprendizaje transferido. Los resultados en test obtenidos por estos modelos son los que nos permitirán en última instancia validar el buen funcionamiento de las arquitecturas y analizar el impacto del aprendizaje transferido en este contexto.

Como complemento, en esta tarea se tratará también de evaluar la importancia de los

mecanismos de *data augmentation* —aumento artificial de datos—. Para ello, se medirá su impacto en el rendimiento de las redes comparando los resultados obtenidos con diferentes tamaños del conjunto de entrenamiento. De esta forma, se podrá evaluar si para conjuntos más pequeños estos mecanismos son capaces de aportar la suficiente variabilidad.

1.2 Estructura de la memoria

La memoria de este trabajo al completo consta de 8 capítulos, con dos partes bien diferenciadas en la estructura: una relacionada con la tarea base de reconstrucción multimodal y otra con la tarea finalista de segmentación de vasos. Cada una tendrá sus propios resultados y conclusiones, pero ambas harán uso de las mismas tecnologías aplicadas al mismo dominio (por lo que compartirán toda la parte de contextualización) y sus resultados y conclusiones estarán estrechamente relacionados. Así, el resumen organizativo de la memoria es el siguiente:

- **Capítulo 1: Introducción.** Descripción del proyecto, sus motivaciones y objetivos, junto con un resumen de la estructura de la memoria.
- **Capítulo 2: Contextualización.** Descripción del dominio de aplicación, de las tecnologías empleadas, y profundización en los aspectos más relevantes para el trabajo. También, se proporciona una revisión del trabajo más directamente relacionado con las aplicaciones y tecnologías consideradas en el proyecto.
- **Capítulo 3: Planificación.** Detalle del listado de materiales empleados, de la planificación del desarrollo del proyecto, y de la estimación de costes.
- **Capítulo 4: Arquitecturas de red.** Descripción y detalle de las arquitecturas de redes neuronales analizadas en el trabajo para la consecución de las tareas.
- **Capítulo 5: Reconstrucción multimodal de angiografía a partir de retinografía.** Presentación de la metodología y técnicas empleadas para la consecución del objetivo de reconstrucción multimodal, junto con los detalles de experimentación, los resultados obtenidos y las conclusiones.
- **Capítulo 6: Segmentación de vasos en retinografía.** Presentación de la metodología y técnicas empleadas para la consecución del objetivo de segmentación de vasos (*desde cero* y usando aprendizaje transferido a partir de redes de reconstrucción multimodal), junto con sus resultados correspondientes.
- **Capítulo 7: Conclusiones.** Resumen de los principales resultados a nivel global e implicaciones de los mismos en el contexto del problema abordado.
- **Capítulo 8: Trabajo futuro.** Proposición de mejoras y nuevos caminos a seguir a partir del trabajo realizado.

Capítulo 2

Contextualización

Dado que el proyecto se desarrolla en un dominio de aplicación concreto de imagen médica y hace uso de unas tecnologías específicas de aprendizaje máquina, es necesaria una contextualización adecuada que permita al lector conocer el problema y las herramientas a un nivel de detalle razonable. Solo así será posible una verdadera comprensión de los métodos y razonamientos seguidos a lo largo del documento.

Con este objetivo, en este capítulo se presentan las principales características del ojo humano, los tipos de imágenes que típicamente se utilizan para su análisis —y que se emplean en el proyecto— y una revisión del estado del arte de tecnologías de aprendizaje profundo que culmina con una exposición de aquellos trabajos más directamente relacionados con el presente proyecto.

2.1 El ojo humano

De gran importancia en el proceso evolutivo de un sinnúmero de seres vivos, los ojos son posiblemente los órganos sensoriales de mayor trascendencia en nuestro día a día como seres humanos. La complejidad, sensibilidad y frecuente presencia de problemas que presentan hacen de su estudio, y el de sus enfermedades asociadas, una de las principales áreas de la medicina: la **oftalmología**. Si le añadimos a esto la utilidad demostrada que poseen para el diagnóstico de otras enfermedades no específicas del órgano visual como la hipertensión [4] o la diabetes [5], entre otras, tenemos que el interés en su estudio está más que justificado, y que cualquier avance relacionado con él es altamente relevante.

A lo largo de este proyecto, todas las imágenes utilizadas son imágenes oftalmológicas. En ellas, y según su tipo, puede apreciarse, en mayor o menor medida, una parte importante de los elementos que componen el órgano visual humano.

2.1.1 Funcionamiento y anatomía general del ojo humano

Como órgano visual, el ojo es el encargado de detectar y captar la luz. Los seres humanos, al igual que la mayor parte de los mamíferos, disponemos de dos al frente de la cabeza y ligeramente separados (lo que hace posible una percepción tridimensional del mundo [6]) en unas concavidades, las órbitas, que los protegen. Cada uno de ellos posee, en una capa sensible

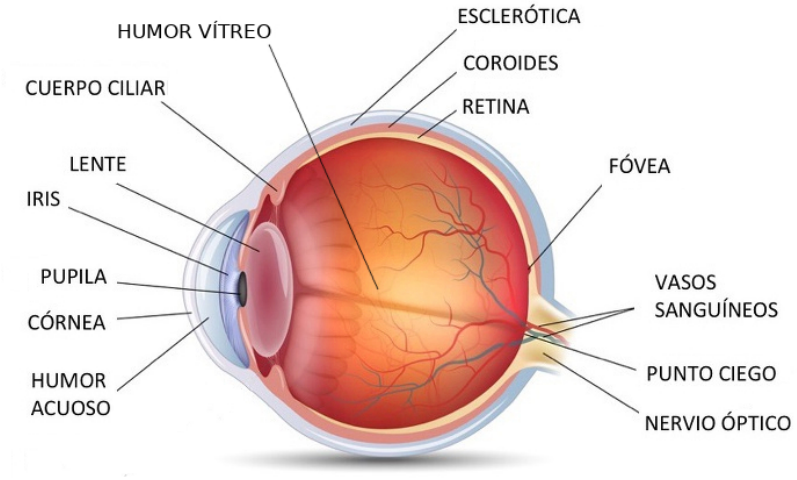


Figura 2.1: Estructura del ojo humano.

situada en la superficie interior del ojo —la **retina**—, unos fotorreceptores que convierten la energía lumínica en las señales eléctricas que serán después transmitidas al cerebro, que integra de toda esa información y consuma el acto de visión [7].

Para hacer ese proceso realidad, es necesario formar una imagen (luz enfocada [6]), centrada en la **mácula** [6] (ver Fig. 2.2), justamente en la capa sensible de la retina, situada en el fondo del ojo (ver Fig. 2.1). Esta luz entra en el globo ocular a través de la **córnea** primero (en la capa exterior[1]) y de la **pupila** después (de cuyo ajuste de diámetro en función de la cantidad de luz se encargan los músculos del **iris** [7]). Seguidamente, esta luz es enfocada por medio de un ajuste de forma de la **lente**[2], de lo que se encarga el músculo ciliar (parte del **cuerpo ciliar**). Una vez esto se consigue, unas células de la retina, los fotorreceptores, se excitan. Las más abundantes, los bastones, detectan la luz tenue y son responsables de la imagen poco detallada que asociamos a la noche. Además, son muy sensibles al movimiento, tienden a concentrarse en los bordes de la retina y su pigmento (rodopsina) se excita con la exposición a la luz verde-azul. Las menos abundantes, los conos, especialmente concentrados en la **fóvea** (situada en el centro de la mácula), nos proporcionan una visión diurna detallada y nos permiten detectar los colores. Hay tres tipos distintos, con formas ligeramente distintas de pigmento cónico (iodopsina), cada una de ellas adecuada para la absorción principal de luz roja, verde o azul [7] (orden según abundancia, de mayor a menor [8]).

Al excitarse, esas células envían señales a las neuronas de las capas superiores de la retina, que a su vez, y respondiendo a patrones de activación determinados, envían señales a las células ganglionares. Los axones de estas células forman el **nervio óptico**, que se conecta

[1] Situada en la parte exterior, al igual que la **esclerótica** (cuya función es protectora).
[2] También llamada cristalino.

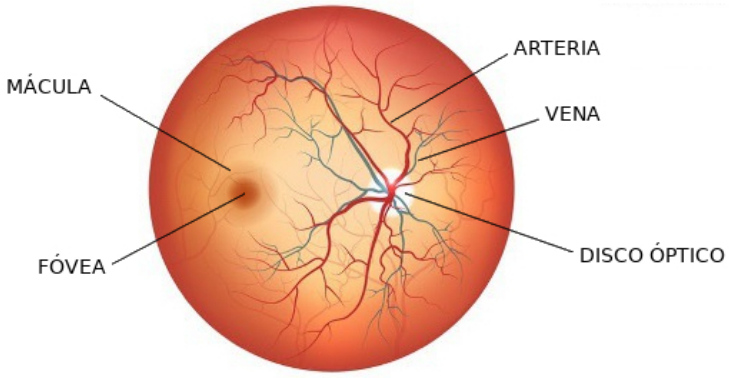


Figura 2.2: Vista del fondo del ojo humano.

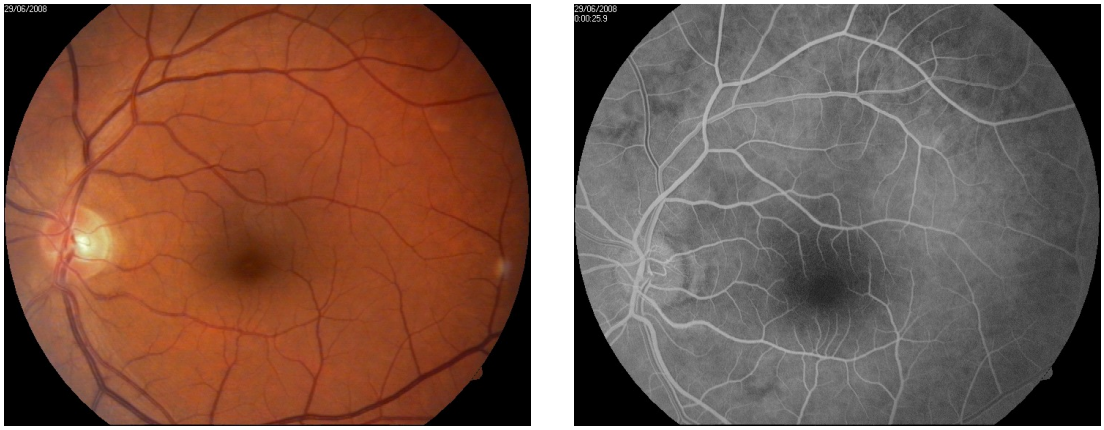
con el cerebro, el responsable final del proceso de integración y procesado de la información. La zona de entrada de ese nervio a la retina se denomina **disco óptico** y representa un punto ciego en el campo visual por la ausencia de conos y bastones [6].

Como elementos también participantes, aunque de forma menos directa, contamos además con el **humor acuoso**, el **humor vítreo**, la **coroides** y el **árbol arterio-venoso** retiniano (ver Figuras 2.1 y 2.2). El humor acuoso mantiene la presión intraocular [9] y el humor vítreo provee soporte físico (manteniene la retina cerca de la coroides) y sirve de almacén y vía de movimiento a los metabolitos [10]. Además, ambos permiten el paso de la luz [7]. Por otro lado, la coroides evita la dispersión lumínica y hace llegar la sangre a la parte exterior de la retina, complementando al árbol arterio-venoso, cuya función es hacer llegar la sangre a la parte interior, donde la circulación por difusión coroidal no es suficiente.

El tiempo de difusión se incrementa con el cuadrado de la distancia, que viene determinada por el propio grosor de la retina. Si el grosor es pequeño, como sucede en la **zona avascular foveal** (FAZ por sus siglas en inglés, ubicada en la fóvea) y en los extremos de la periferia retiniana [8], basta con la circulación por difusión de la coroides. Si el grosor es grande, como en el resto de zonas, el tiempo de difusión hasta llegar a la parte más interna resulta excesivo, por lo que se necesita una estructura de vasos sanguíneos.

2.2 Imagen oftalmológica

En medicina, la imagen es una de las herramientas más útiles y se emplea en casi cualquier ámbito. Diversas modalidades de imagen, como radiografías, tomografías e imágenes de resonancia magnética, entre otras, forman parte del día a día de prácticamente todo el personal sanitario. En el campo de la oftalmología, dos de las más utilizadas son la retinografía y la angiografía. Principalmente, porque la información que revelan es clave a la hora de diagnos-



(a) (b)

Figura 2.3: Par compuesto por una retinografía (a) y una angiografía (b) [*Isfahan dataset*].

ticar un significativo número de enfermedades (como glaucoma o retinopatía diabética, entre muchas otras).

La utilización de una u otra depende del caso de análisis. Ambas ofrecen ventajas e inconvenientes particulares a la hora de identificar ciertas patologías o resaltar ciertas estructuras retinianas. Muchas veces son requeridos ambos tipos para un mismo paciente, y se forman pares retinografía-angiografía para los dos ojos (la Figura 2.3 es un ejemplo de par). El resultado es un conjunto de 4 imágenes con información complementaria para el diagnóstico.

2.2.1 Retinografía

La retinografía es una técnica de imagen médica *no invasiva* que permite obtener fotografías en color del fondo de ojo. La adqusición se hace mediante el uso de un equipamiento específico llamado retinógrafo, que básicamente consiste en un oftalmoscopio equipado con una cámara digital y un *flash*. La única preparación necesaria del paciente es la dilatación de las pupilas con unas gotas (aunque también hay aparatos con los que esto se puede evitar, los conocidos como *no midriáticos*).

Las imágenes obtenidas, las retinografías[3] (ver ejemplo en la Figura 2.3a), ofrecen una buena visibilidad del disco óptico, la fóvea y los vasos sanguíneos principales, pero no tanto de las venas y arterias más pequeñas. Este tipo de fotografías son útiles para el diagnóstico, seguimiento y evaluación de enfermedades como glaucoma, Degeneración Macular Asociada a la Edad (DMAE), y retinopatía diabética, entre otras.

En la Figura 2.3a no es difícil ubicar los distintos elementos mencionados en el Apartado 2.1.1 (mácula, fóvea, disco óptico y vasos sanguíneos) tomando como referencia la Figura 2.2.

[3]Denominaremos de igual modo a la técnica de imagen médica y a las imágenes fruto de su aplicación.

2.2.2 Angiografía con fluoresceína

La angiografía con fluoresceína, angiografía fluoresceínica, o simplemente angiografía, es una técnica de imagen médica que, como la anterior, permite la obtención de imágenes del fondo de ojo. Al contrario que la primera, es *invasiva*, ya que se necesita realizar la inyección intravenosa de un contraste (fluoresceína sódica) al paciente. La principal utilidad del contraste es realzar los vasos sanguíneos u alguna otra estructura (como microaneurismas) en los que es posible la circulación.

El proceso fotográfico al completo sería el siguiente: como antes, se dilatan las pupilas del paciente con unas gotas (aunque los equipos no midriáticos permiten evitarlo); acto seguido, se inyecta el contraste; y en escasos segundos, este llega a los vasos sanguíneos de la retina y se empiezan a tomar las fotografías (en escala de grises para una mejor apreciación de los vasos). Para que las fotografías del fondo de ojo sean adecuadas (ver ejemplo en Figura 2.3b), es necesario un aparato similar a un retinógrafo que disponga tanto de un modo en blanco y negro como de unos filtros apropiados que favorezcan el realce de la fluoresceína sódica.

Este tipo de fotografías se utilizan principalmente en el diagnóstico, seguimiento o evaluación de las patologías con especial interés en las componentes sanguíneas como la retinopatía diabética, las trombosis venosas o la DMAE.

2.3 Aprendizaje máquina

El **aprendizaje máquina** es una rama de las ciencias de la computación [11] que busca la creación de programas informáticos [12] capaces de extraer conocimiento de la experiencia o de grandes (o no tan grandes) volúmenes de datos [13]. Dicho de otro modo, el aprendizaje máquina busca la creación de modelos capaces de encontrar las representaciones adecuadas para un conjunto de datos en el contexto de una tarea determinada [14]. En los últimos tiempos, su uso en aplicaciones de nuestro día a día ha aumentado exponencialmente, y podemos encontrarlo aplicado tanto en algoritmos de recomendación de películas[4] como en programas de análisis de secuencias ADN [11]. El buen funcionamiento que ofrecen en gran cantidad de tareas [15], junto con el abaratamiento de los componentes *hardware* y la creciente accesibilidad de la tecnología, con librerías amigables en los principales lenguajes de programación, ayudan todavía más a extender su utilización a gran cantidad de problemas. Algunos de ellos, del ámbito médico [16, 17, 18, 19].

La medicina, tradicionalmente, se ha guiado por los datos para realizar las labores de diagnóstico. Las mejoras más recientes en prácticamente la totalidad de sus áreas no serían posibles si no se hubiera producido un desarrollo tecnológico importante. La precisión de los

[4] https://movix.ai/about

equipos nuevos, que permite obtener imágenes de gran calidad, junto con la digitalización y el almacenamiento de la información clínica en bases de datos, han propiciado una mejoría significativa en procesos que van desde el diagnóstico a la propia cirugía [20]. El conocimiento adquirido a través de la experiencia personal, del *ejemplo*, en un aprendizaje de carácter heurístico que se nutre precisamente de toda esa información —los datos—, es uno de los mayores pilares de cualquier carrera y trayectoria médica.

Por esta combinación de datos y experiencia, el entorno médico es ideal para la aplicación de algunas técnicas de aprendizaje máquina [20]. En estas técnicas, la aprehensión se basa en la presentación de *ejemplos*, los datos, y el ajuste del sistema en base a ellos remite en cierto modo al concepto de la «experiencia».

2.4 Redes de neuronas artificiales

Entre los muchos métodos de aprendizaje máquina existentes, en este trabajo nos centraremos en las **redes de neuronas artificiales** (ANNs, *Artificial Neural Networks*) y, más concretamente, en su subtipo **redes de neuronas convolucionales** (CNNs, *Convolutional Neural Networks*, las empleadas en el trabajo), ampliamente utilizadas en aplicaciones y problemas relacionados con la imagen médica[5], especialmente en los últimos años [16, 17, 18, 19].

Las ANNs son agrupaciones de *neuronas artificiales*, unas unidades computacionales básicas que, en su forma más simple, hacen lo siguiente: para un vector de entrada dado, primero aplican una suma ponderada de sus valores con unos *pesos* asociados a la propia neurona, y después hacen un corte por umbral tomando el valor de esa suma —lo que se conoce como *función de activación*[6]— [21]. Estas redes pueden tener distintas disposiciones y esquemas de conectividad entre las neuronas, entradas y salidas. Normalmente, estas configuraciones reciben el nombre de *arquitecturas*, y sus principales elementos de construcción son las **capas**: conjuntos usualmente alineados de neuronas con algún rasgo *arquitectónico* distintivo y uniforme para todas ellas [21].

Para más detalles, en el Apéndice A puede verse una introducción a las ANNs, examinando sus principales elementos de construcción y algoritmos de aprendizaje.

2.5 Redes de neuronas convolucionales

Una forma de aplicar las redes de neuronas con imágenes como entrada (en color, escala de grises, etc.) es usar una red neuronal con tantas neuronas de entrada como píxeles tiene

[5] Aunque también son muy numerosas sus aplicaciones fuera de este entorno.
[6] En la bibliografía también puede verse el nombre de *función de transferencia*.

la imagen y tantas capas ocultas antes de la salida como se desee. Sin duda, esta arquitectura podría resolver ciertos problemas en los que las imágenes fueran de pequeño tamaño, e.g. la clasificación de imágenes pequeñas de dígitos escritos a mano[7] [24]. Sin embargo, para imágenes medianas o grandes, como las que acostumbramos a utilizar en nuestro día a día, el número de parámetros de la red sería demasiado grande, llevando a una excesiva carga computacional y a una alta probabilidad de incurrir en sobreajustes (*overfitting*) durante el entrenamiento [25].

Las **redes de neuronas convolucionales** (CNN, *Convolutional Neural Networks*), son capaces de evitar estos problemas y de aprovechar la estructura de ciertas representaciones de datos, como es el caso de las imágenes, cuyos píxeles cercanos mantienen una relación espacial [26]. Además, estas redes son compatibles con la forma de entrenar de las redes clásicas, es decir, mediante el algoritmo de **propagación hacia atrás**.

Con el motivo de arrojar luz al funcionamiento global de este tipo de redes, en el Apéndice B se describen algunos de los conceptos y mecanismos de los que estas hacen uso (apartado 1 y posteriores), como la propia convolución (y las capas convolucionales) y la operación de *pooling*. Además, también se describen el problema de desvanecimiento/explosión de gradientes, al que se enfrentan las redes de gran profundidad (5); algunas de las formas más conocidas de inicializar los parámetros de las redes (6); y uno de los conjuntos de datos más importantes que existen para tareas de clasificación: ImageNet (7).

A continuación, se describirá el esquema general de una red convolucional de clasificación, y se relatará la evolución de las CNNs a lo largo de los últimos años, explicando algunos conceptos innovadores relacionados con estas y que forman parte del estudio realizado en este trabajo, como las *skip connections* o las convoluciones dilatadas.

2.5.1 Red convolucional de clasificación

La Figura 2.4 recoge la arquitectura típica de una CNN para clasificación. Como se pude ver, la organización básica de una arquitectura de este tipo consiste en apilar, sucesivamente, capas convolucionales (seguidas normalmente de una función de activación, e.g. ReLU) seguidas de una capa de *pooling*, y, al final de la red, una parte completamente conectada y *feed-forward* clásica, con una última capa con función de activación *softmax*.

2.6 AlexNet

La arquitectura AlexNet [28] no es, ni mucho menos, la primera red convolucional. Antes ya habían aparecido redes como *Neocognitron* [29] (considerada la verdadera precursora [30])

[7] Como las del *dataset* MNIST, de 28×28 píxeles [22, 23].

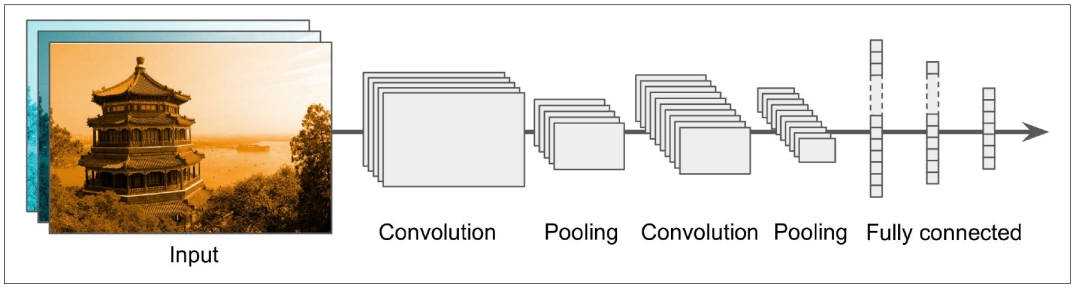


Figura 2.4: Arquitectura típica de una CNN para clasificación [27].

y LeNet [22], una de las más conocidas y base de muchos desarrollos posteriores (entre los que está la propia AlexNet). [27]. Aun así, se considera un hito, y en su momento tuvo una gran repercusión (de hecho, mucha gente la considera el punto de inicio del auge del aprendizaje profundo). Principalmente, esta consideración se debe a la importante victoria del *challenge* ImageNet de 2012, donde superó en 9 puntos porcentuales a la segunda mejor clasificada.

Como es habitual en redes de clasificación (e.g. LeNet), esta arquitectura tiene al final una parte completamente conectada (*fully connected*) con *softmax*. Además, se vale de dos mecanismos aún hoy ampliamente utilizados (como veremos en las arquitecturas descritas en el Capítulo 4): *dropout* y *data augmentation*.

2.6.1 *Dropout*

Dropout [31, 32] es una técnica que busca evitar el sobreajuste y que funciona del siguiente modo: en cada iteración del entrenamiento, cada activación —neurona— (de la capa donde se defina el *dropout*, que no puede ser de salida) tiene probabilidad p de ser deshabilitada, es decir, de no ser tenida en cuenta en esa iteración. Así, se consigue aportar aleatoriedad y variabilidad a las redes, haciendo que sea más difícil caer en situaciones de sobreajuste.

Como detalle, es importante que en test los pesos se ajusten conforme al valor de *dropout* (multiplicando las entradas o dividiendo las salidas de las neuronas por ese valor) para evitar desequilibrios entre el número de activaciones en entrenamiento y test [27].

Spatial Dropout

Este tipo de *dropout* (originalmente denominado como *SpatialDropout*) no se usa en AlexNet, pero resulta interesante por su presencia en el ámbito y por su utilización en una de las arquitecturas empleadas en el trabajo (ENet). Como el anterior, consiste en deshabilitar las activaciones de la red en función de una probabilidad p, pero esta vez, desactivando mapas completos. Si se tiene un conjunto de mapas de características de dimensiones $n_{car.} \times alto \times ancho$, siendo $n_{car.}$ el número de mapas de características, se llevan a cabo $n_{car.}$ intentos de *dropout*

(uno por mapa) y, en el caso de que se produzca, se deshabilitan todas las activaciones del mismo mapa. Hacer esto así viene motivado por la fuerte correlación existente entre las activaciones de distintas posiciones en los mapas de características. En el caso de *dropout* normal, estas posiciones se activarían de forma independiente, dejando algunas de ellas inactivas, lo que puede no ser suficiente para aportar una adecuada variabilidad (independiente) y evitar el sobreentrenamiento (sobreajuste). En el artículo original [33] demuestran que, al menos en algunos escenarios, el uso de *Spatial Dropout* hace que las redes obtengan mejores resultados.

2.6.2 *Data augmentation*

La técnica de *data augmentation* consiste en aumentar el número de ejemplos de entrenamiento de forma artificial, haciendo variaciones y transformaciones plausibles sobre los ejemplos originales del conjunto de datos [27]. De nuevo, como el *dropout*, ayuda a reducir el sobreajuste. En este caso, dando cierta variabilidad al conjunto de entrenamiento y provocando así una mayor tolerancia de la red a cambios en los datos.

Estas variaciones pueden ser de muchas formas, y según su naturaleza (además de cómo sean los ejemplos originales) se potenciará que la red sea más tolerante a unas u otras modificaciones. Algunos ejemplos de operaciones son los cambios de intensidad en las imágenes, los desplazamientos, y las rotaciones. Todas ellas pueden ser aplicadas bien en tiempo real (se hacen dinámicamente durante el entrenamiento) o *a priori*.

2.7 GoogLeNet

GoogLeNet es una red convolucional de clasificación presentada en [34]. Destacó especialmente por su excepcional profundidad (para el momento) y menor número de parámetros en relación a otras redes (como AlexNet [27]). Esto lo logra mediante una arquitectura de múltiples *subredes* [35] o módulos *inception* que permiten integrar una mayor diversidad de escalas de proceso de forma eficiente. En total, la red utiliza módulos *inception* de forma secuencial. Todos ellos, para las capas convolucionales, usan la función de activación ReLU, que también se usa para las demás. Además, también están presentes la regularización por *dropout* y varias partes completamente conectadas. Una de ellas se sitúa al final de la red y las otras dos en lugares intermedios, para calcular el error en ese punto y propagarlo de forma que se mitigue el problema de decaimiento/explosión de gradientes. Todas ellas tienen *softmax* al final como función de activación, al igual que sucedía con AlexNet.

2.7.1 Módulo *inception*

El módulo *inception* es probablemente lo más importante de la arquitectura y lo que más ha trascendido. En la Figura 2.5 se puede ver su estructura. La presencia de filtros de diferentes

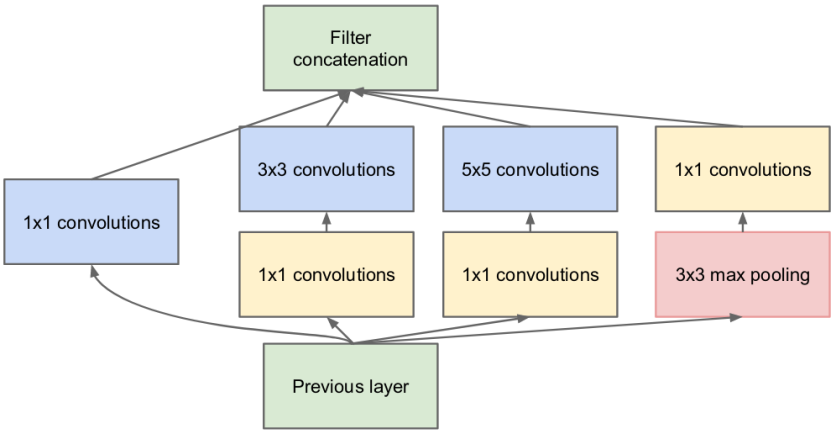


Figura 2.5: Módulo *inception* de GoogLeNet [34].

tamaños permite capturar patrones a diferentes escalas [27], por lo que se suele decir que este módulo es **multiescala**. Todos los caminos usan la misma entrada, pero todos son diferentes, y sus salidas son concatenadas al final para después ser trasladadas al siguiente módulo o capa como una sola.

Otro detalle interesante del módulo es que utiliza convoluciones de 1×1. Como hasta ahora no se habían mencionado, detallaremos aquí qué son exactamente y qué papel juegan dentro de las redes, tomando como ejemplo *inception*.

2.7.2 Convoluciones 1×1

Este tipo de convoluciones toman como entrada una única posición en los mapas de características, es decir, que no pueden capturar patrones bidimensionales usando las posiciones vecinas. No obstante, operan en profundidad, integrando los diferentes mapas de características para cada posición. Si se configuran con menos filtros que la profundidad de la entrada, se consigue **reducir la dimensionalidad** (en profundidad, mayor eficiencia), actuando como una especie de cuello de botella (*bottleneck*), de forma que se aprende una selección de las características de entrada (formalmente es una extracción de características relevantes). Dicho de otro modo, la red aprende la forma más conveniente de reducir la dimensionalidad en profundidad.

2.8 VGGNet

La VGGNet [36], a nivel arquitectónico en cuanto a los elementos constructivos o la organización de las capas, no supuso ningún cambio con respecto a las redes que ya había. Su

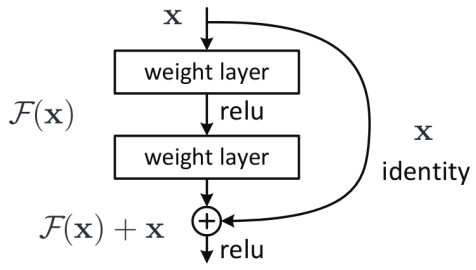


Figura 2.6: Aprendizaje residual: bloque de construcción de ResNet con *skip connection* [37].

relevancia viene del buen funcionamiento demostrado utilizando cascadas de filtros de tamaño 3×3 en lugar de filtros de mayor tamaño. Esta mejora viene motivada por el hecho de que la cascada de filtros abarca el mismo espacio que el filtro de mayor tamaño, pero usando muchos menos parámetros. A su vez, la cascada de filtros admite patrones menos complejos, pero suficientemente apropiados para tareas de análisis de imagen. Estos efectos actúan como elementos regularizadores de la red (menor número de parámetros y por tanto menor complejidad).

2.9 ResNet

Residual Network o ResNet [37] supuso la confirmación de la tendencia de mayor profundidad con menor cantidad de parámetros (inicialmente llegó hasta las 152 capas, pero hubo versiones posteriores con hasta más de 1000 [38]). Uno de los mecanismos clave que incorporó fueron las *skip connections*: conexiones entre partes no sucesivas de la red que permiten saltar varias capas. El objetivo principal de estas conexiones es permitir el paso directo de gradientes hacia las capas de la entrada durante el entrenamiento. Esto evita el problema de los *vanishing gradients*, que se incrementa con la profundidad.

En la Figura 2.6 puede verse el caso concreto de ResNet, donde en las *skip connections* se suma la entrada de una capa a la salida de otra situada más adelante —en dirección a la salida—. Al marco de desarrollo propuesto (*framework*) le dieron el nombre de *deep residual learning*.

2.9.1 *Residual learning*

Siguiendo el artículo original [37] y la Figura 2.6, se puede entender como *residual learning*, o aprendizaje residual, lo siguiente:

Si se considera $\mathcal{H}(\mathbf{x})$ como la función a ajustar por una parte o toda la red (en adelante *sección*), siendo $\mathbf{x}$ la entrada, podemos suponer que la sección, en el caso de que pueda ajustar $\mathcal{H}(\mathbf{x})$, podrá ajustar también la *función residual* $\mathcal{F}(\mathbf{x}) := \mathcal{H}(\mathbf{x}) - \mathbf{x}$ (con entrada y salida de

mismas dimensiones). Esto daría lugar a que $\mathcal{H}(\mathbf{x})$ pasara a ser $\mathcal{F}(\mathbf{x}) + \mathbf{x}$, y en el artículo se demuestra que esta forma tiene ventajas en el proceso de aprendizaje, dado que toma como base la función de identidad (copiar entrada salida) y se centra en encontrar las *perturbaciones* con respecto a ella. Todo ello siempre que la función a ajustar esté más próxima a la identidad que a la *función* 0 (salida a 0).

2.9.2 *Batch Normalization*

Otro aspecto a destacar de ResNet es la utilización del denominado *Batch Normalization* (BN) o *normalización por lote* (presentado en [39]). De forma general, consiste en la normalización de las características de las capas ocultas de la red haciendo que tengan una varianza similar. Este mecanismo atenúa el *vanishing/exploding gradients problem*, o problema de desvanecimiento/explosión de gradientes.

Por otra parte, en el artículo original se aduce una mejora en el comportamiento de las redes por la capacidad de BN de reducir el efecto del *internal covariate shift problem* (descrito a continuación). No obstante, con el tiempo, han aparecido otros artículos que ponen en duda que la ganancia en el rendimiento venga dada por esta cuestión [40].

Internal covariate shift problem

Este problema, presente a veces en las redes profundas, es causado por el cambio en la distribución de activaciones de las redes por la modificación de los parámetros [35]. Se considera realmente un problema cuando los cambios son grandes porque dificultan el entrenamiento de las capas más profundas, al tener estas que estar adaptando sus parámetros (cada vez) a entradas con distribuciones estadísticas diferentes.

Operación de *Batch Normalization*

La idea de BN es la siguiente: añadir una operación en las capas ocultas de la red que ***normalice*** las características, haciendo que tengan una varianza similar y se mantenga relativamente constante su distribución estadística. Para ello, las capas de BN contienen dos parámetros más que se aprenden durante el entrenamiento: β_i y γ_i. La idea es que la salida de ese nodo i donde se aplica BN_i se ajuste en función de los parámetros β_i y γ_i para cada *mini-batch*. Para conseguirlo, es necesario estimar la media y la desviación típica de cada entrada evaluando ambas, para cada entrada, en el *mini-batch* actual.

Esta operación o módulo de BN puede situarse justo después de ser aplicada la función de activación o justo antes (inmediatamente después de calcular los *logits*), lo que parece ser más ventajoso [39].

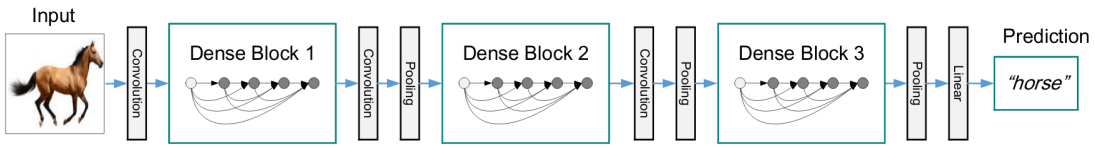


Figura 2.7: Arquitectura DenseNet con 3 bloques densos [41].

2.10 DenseNet

Dense Convolutional Network [41], DenseNet, es una arquitectura de red que utiliza parte de las ideas ya vistas en el módulo *inception* de GoogLeNet y en ResNet. Está formada por bloques densos (*dense blocks*, ver Figura 2.8), que implementan (1) mecanismos similares a las *skip connections* y (2) la integración de patrones multiescala mediante concatenación.

La arquitectura al completo sigue el esquema que se puede ver en la Figura 2.7, con bloques densos unidos por capas (llamadas *de transición*) de convolución y de *pooling*. Las capas de transición reducen la resolución, mientras que los bloques densos la mantienen. Al final, como en las anteriores, hay una parte completamente conectada con *softmax*, para realizar clasificación.

2.10.1 Bloques densos

Por su relación con una de las arquitecturas de este trabajo (FC-DenseNet) entraremos aquí en el detalle de este bloque. Como puede verse en la Figura 2.8, un bloque denso está compuesto por varias capas (H_n con $n \in \{1, ..., 4\}$), cada una con sus mapas de características ($\mathbf{x}_n$ con $n \in \{1, ..., 4\}$; $\mathbf{x}_0$ como entrada). Cada capa está compuesta por una operación de *Batch Normalization*, una ReLU y una convolución de 3×3 (como las vistas en VGG) con k filtros.

A cada una de estas capas entran tanto los mapas de características resultado de la capa inmediatamente anterior como los de todas las precedentes a esta (incluyendo la entrada). Es decir, una concatenación de la salida de todas las capas anteriores y la entrada. Es por esta adición de k mapas de características por cada capa por lo que se llama a este parámetro k *ratio de crecimiento*. Como se puede deducir de la estructura, la resolución de entrada (alto y ancho de los mapas de características) se mantiene en la salida del bloque.

Estos bloques densos permiten tener redes con menos parámetros y más fáciles de entrenar. Principalemente por dos factores. Primero, por el mejorado flujo de información y gradientes que se produce a lo largo de la red [41]. Y segundo, por la reutilización de las características. Además, por las conexiones directas entre los mapas de características entre muchos puntos de la red (lo que permite propagar el error de forma directa) se dispone de una supervisión profunda implícita.

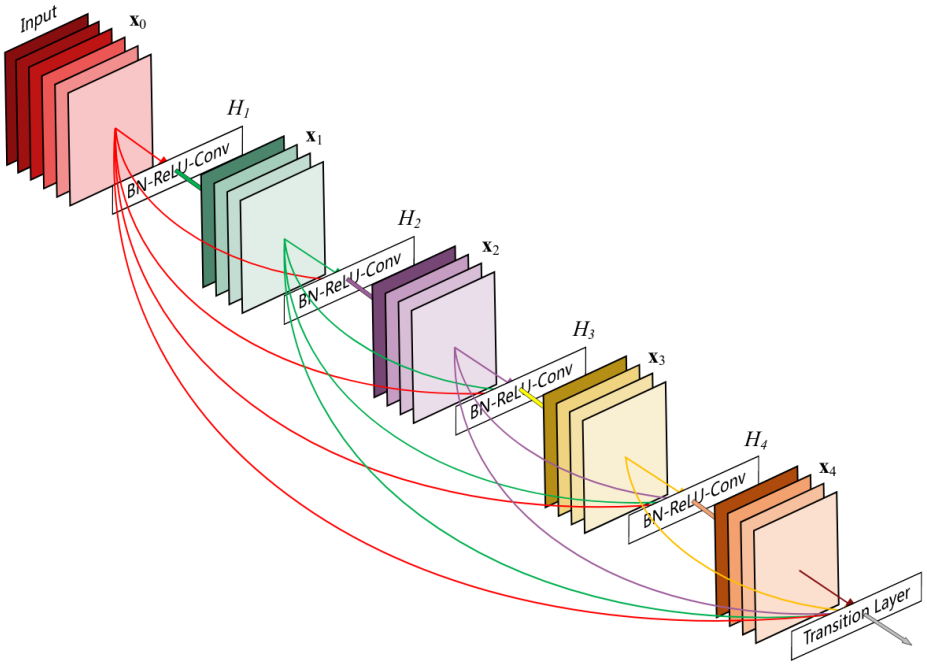


Figura 2.8: Bloque denso de la arquitectura DenseNet con un ratio de crecimiento $k = 4$ [41].

2.11 *Fully Convolutional Network*

La arquitectura *Fully Convolutional Network* [42] fue la que introdujo el concepto de **red completamente convolucional**. Estas redes se caracterizan por construirse únicamente con componentes que operan en regiones locales de la entrada y dependen solamente de coordenadas espaciales relativas (como convolución, *pooling*, y funciones de activación). Esto descarta las partes completamente conectadas habituales en muchas redes neuronales de clasificación y que típicamente se situaban al final, justo antes de la capa de *softmax*. Esta arquitectura resultaba, y aún resulta hoy, útil para cierta variedad de tareas, destacando la clasificación de imágenes, la detección/localización de objetos o la resolución de problemas de correspondencia local [42]. No obstante, para otras muchas, como la segmentación semántica, la utilización de este tipo de redes o bien no es posible de forma directa (sino iterativa y con poco nivel de detalle[8]), o no es eficiente, y en gran parte de los casos las arquitecturas no se pueden entrenar de forma *end-to-end* [42] (desde la entrada en crudo hasta la solución final del problema).

En este sentido, FCN es un remedio para estos problemas. Se puede entrenar *end-to-end*, la aplicación sobre la imagen es directa —no iterativa como en otras aproximaciones— y además

[8]Se hace aquí referencia a aproximaciones que van seleccionando pequeñas partes de la imagen, clasificándolas, de forma que generan un *mapa general* (clasificación por zonas), pero con poco nivel de detalle (zonas generalmente notablemente mayores a un píxel).

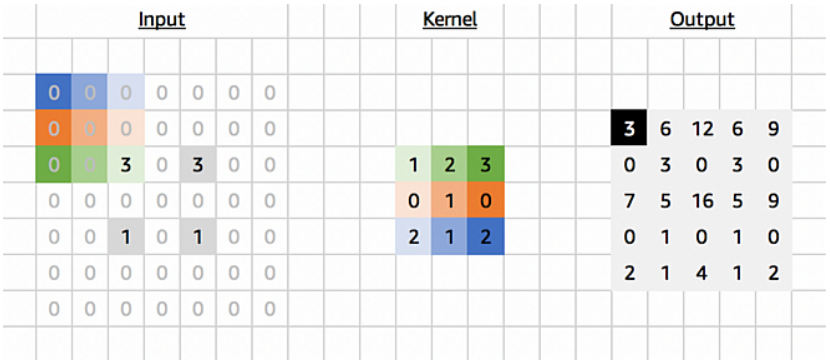


Figura 2.9: Ejemplo de convolución transpuesta en 2D.

permite como entradas imágenes de cualquier tamaño, cosa que no es posible cuando las redes tienen una parte completamente conectada, que posee un número fijo de neuronas de entrada.

La arquitectura La idea clave de la arquitectura es la sustitución de la última parte completamente conectada por una parte completamente convolucional. Esta parte produce una salida de los mismos ancho y alto que la entrada. La forma de conseguirlo es a través de un camino de sobremuestreo, compuesto por por convoluciones 1×1 y convoluciones traspuestas (*transposed convolutions*), que recupera las dimensiones de la entrada en alto y ancho. Para conseguir un nivel de detalle adecuado se utilizan *skip connections*, que combinan la capa de predicción final con otras anteriores. Así, se hace posible la utilización de características extraídas durante el submuestreo en la parte final de sobremuestreo. De esta forma, esta arquitectura permite contrarrestar la pérdida de resolución que provoca el *pooling* (y que dificulta significativamente la recuperación de la posición de los patrones a la salida) y ganar un mayor nivel de detalle a la salida.

2.11.1 *Transposed convolutions*

Las convoluciones traspuestas (*transposed convolutions*) o *upconvolutions*, y a veces mal llamadas deconvoluciones (*deconvolutions*), son un tipo de operación frecuentemente utilizada en redes de neuronas completamente convolucionales para la parte de sobremuestreo (en oposición al *pooling*, que se utiliza para submuestreo e implica pérdida de resolución).

Su funcionamiento se puede ver en la Figura 2.9. Lo que se está haciendo es rotar el *kernel* 180° y aplicarlo sobre las características originales con ciertos *paddings*[9] que permiten obtener la salida en la resolución deseada.

[9] Aquí se muestra de esta forma por simplicidad, la implementación de la operación no se haría de este modo.

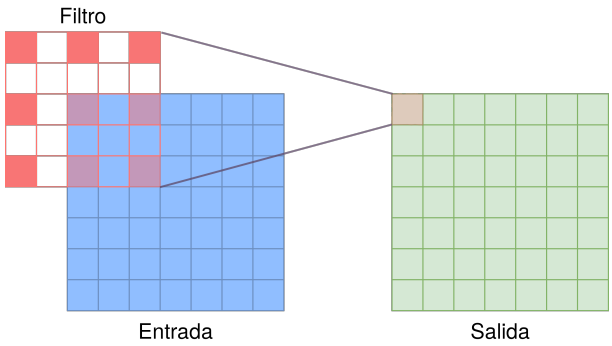


Figura 2.10: Ejemplo de convolución dilatada con dilatación $d = 2$ [45].

2.12 Alternativas a las *transposed convolutions*

Además de las *transposed convolutions*, hay otras formas de recuperar la información perdida de un *pooling*. Entre las principales podemos encontrar las convoluciones dilatadas y las operaciones de *unpooling*.

2.12.1 Convoluciones dilatadas

Las convoluciones dilatadas [43, 44] son un tipo de convolución que soporta la expansión exponencial de los campos receptivos (lo que *ve* el filtro de la entrada en cada paso) sin perder resolución. Esto permite agregar directamente información contextual a múltiples escalas, lo que resulta útil, por ejemplo, para ganar detalle (perfilar mejor los bordes de las regiones segmentadas) en tareas de segmentación semántica [44].

El funcionamiento de este tipo de convoluciones (ver Figura 2.10) es igual al de las convoluciones normales, solo que estas tienen un parámetro de dilatación d que indica la distancia menos uno ($d - 1$, en horizontal y vertical) entre un elemento del *kernel* (un peso) y sus vecinos. A la hora de implementarlo, realmente lo que define este parámetro, más que la distancia de los propios elementos del filtro, es la distancia de aplicación de dichos elementos sobre la imagen original.

2.12.2 *Unpooling*

La técnica u operación de *unpooling* [46] es una operación diseñada para actuar a modo de *inversa* al *pooling*, que, no obstante, no es invertible. Pese a ello, se puede obtener una aproximación lo bastante buena como para resultar útil. Para poder realizar *unpooling* de una operación de *pooling* previa, se guarda la posición del nodo con el valor escogido dentro del área de integración (variable de cambio o *switch*). Al tratar de hacer la operación inversa, se utiliza esta posición almacenada para situar en ella las activaciones de la capa anterior. Esto

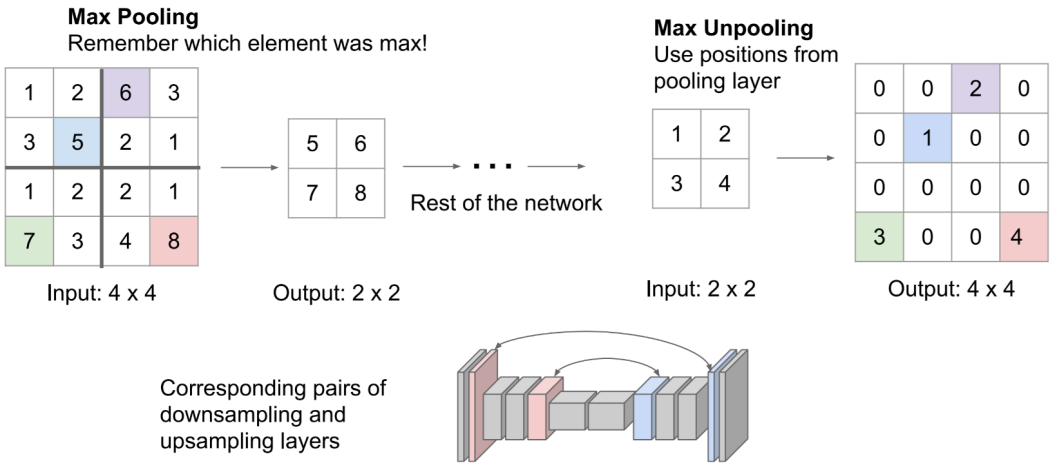


Figura 2.11: Ejemplo de *max-unpooling* [*Nikhil Sardana's lecture notes on FCN* (Stanford)].

permite preservar a *grosso modo* la estructura general de los «estímulos». En la Figura 2.11 puede verse un ejemplo de esta situación para el caso particular de *max-unpooling*.

2.13 Trabajo relacionado

En los apartados anteriores se han presentado algunas de las arquitecturas y técnicas más utilizadas hoy en día en el contexto del *deep learning*. Estas son omnipresentes dentro del grupo de redes que conforman el estado del arte en tareas como la clasificación y la segmentación semántica. Aquí, como se considera esa parte cubierta, hablaremos principalmente de los métodos existentes directamente relacionados con el presente trabajo y sus tecnologías y técnicas. Primero de la parte de reconstrucción y después y brevemente de la parte de segmentación y *transfer learning*, siempre centrándonos en imagen médica.

Paradigma de reconstrucción multimodal

El trabajo más directamente relacionado con este proyecto es el presentado en el MICCAI[10] 2018 por Hervella et al. [47]. En él es donde se propone la tarea de reconstrucción de angiografía a partir de retinografía que aquí realizamos con otras arquitecturas y donde se plantea también la potencial utilidad de la misma para la aplicación de aprendizaje transferido. Este planteamiento viene motivado principalmente por la capacidad de la red de comprender ciertos patrones presentes en la retina, algo que se demuestra en base a las imágenes resultado (*pseudoangiografías*). Ese nivel de comprensión de los modelos entrenados en la tarea de reconstrucción es lo que haría finalmente posible la utilización de dichos modelos como

[10]Medical Image Computing and Computer Assisted Intervention

preentrenamientos en tareas finalistas, como la escogida en nuestro caso de segmentación de vasos. Estos preentrenamientos, presumiblemente, podrían ayudar a mitigar la escasez de datos etiquetados y mejorar los resultados.

La arquitectura U-Net, utilizada en ese trabajo, ofrece un muy buen funcionamiento en gran diversidad de tareas. Sin embargo, su diseño no se ajusta rigurosamente a las nuevas tendencias. Con no demasiadas capas, una estructura muy jerárquica, y un gran número de parámetros, esta arquitectura puede considerarse *clásica* con respecto a algunas de las presentadas en trabajos más recientes. En estos trabajos, las arquitecturas suelen tener muchas más capas y utilizan habitualmente mecanismos como algunos de los vistos hasta ahora para hacer posible su entrenamiento.

De este modo, resulta interesante comparar el funcionamiento de la arquitectura U-Net con algunas arquitecturas más nuevas, que se ciñan más a la tendencia de más profundidad con menos parámetros y que incorporen algunos de los mecanismos que en los últimos años se han descubierto beneficiosos para el entrenamiento de redes de gran profundidad. Las arquitecturas seleccionadas para esta comparación con U-Net han sido FC-DenseNet y ENet. La primera, se ha seleccionado por los buenos resultados obtenidos en algunas tareas de segmentación de referencia y por su integración de ciertas ideas de U-Net y DenseNet. La segunda, por ser una red liviana (con pocos parámetros en comparación con FC-DenseNet y U-Net) y de bastante profundidad, y por incorporar ciertos mecanismos interesantes y muy extendidos hoy en día como las convoluciones dilatadas o el *unpooling*.

Transfer Learning

La inicialización de los pesos de la red con los valores de los parámetros de otro modelo (misma arquitectura) entrenado en una tarea distinta a la objetivo suele reportar beneficios. Esto ocurre principalmente por la gran reusabilidad que presentan, por naturaleza, las imágenes. En muchas tareas complejas, como la clasificación o la segmentación, el entrenamiento en la tarea objetivo partiendo de los pesos del modelo preentrenado (lo que se conoce como *refinamiento*) ha mejorado significativamente los resultados necesitando un menor tiempo de entrenamiento. En [48] y en [49] pueden verse ejemplos de la utilización de esta técnica, llamada *transfer learning*, con modelos preentrenados en un conjunto de datos compuesto por imágenes *naturales* (o *generales*), el primero con DenseNet y el segundo con Inceptionv3, ambos para la clasificación de imágenes del dominio médico. [50] y [51] son más ejemplos.

A pesar de todo, cuando se aplica esta técnica con modelos preentrenados en *datasets* con imágenes muy distintas a las utilizadas en la tarea objetivo, los resultados no son siempre igual de satisfactorios. En tareas de imagen médica, por ejemplo, es mucho más conveniente que los preentrenamientos hayan sido realizados en conjuntos de datos del ámbito médico

que en conjuntos de datos generales [52]. Aunque estos sí aporten en cierta medida [53, 54], es mucho más fácil reutilizar el conocimiento —las características— cuando la naturaleza y la apariencia de las imágenes son parecidas. Por esta razón, otras aproximaciones presentan alternativas a los preentrenamientos clásicos en *datasets* de imágenes *naturales*. Una de las más interesantes es el preentrenamiento con recortes, *parches* cuadrados de las imágenes del conjunto de datos objetivo, y el posterior entrenamiento de esos modelos preentrenados en las imágenes completas. En [55] se puede ver un ejemplo de esto para la segmentación de vasos sanguíneos en retinografía.

Otros trabajos utilizan modelos preentrenados en múltiples modalidades de imagen y dominios (donde unas u otros pueden ser diferentes a los de la tarea final) para mitigar la escasez de datos etiquetados en la tarea que se quiere abordar. Por ejemplo, en [56] se muestra la conveniencia, sobre el entrenamiento desde cero, de utilizar la parte de submuestreo de un modelo preentrenado en un conjunto de datos heterogéneo de imágenes 3D (múltiples dominios y modalidades) en otras tareas de segmentación 3D con otros datos distintos.

Segmentación de vasos

El estado del arte en segmentación en imagen biomédica, y específicamente de vasos sanguíneos en retinografía, lo conforman, principal y mayoritariamente, los trabajos que hacen uso de U-Net [1] y algunas de sus variantes: LadderNet [57], RRU-Net [58] y Deep Residdual U-Net [59]. Todas estas arquitecturas, salvo LadderNet, tienen un camino de submuestreo y otro de sobremuestreo con reutilización de características entre ambos mediante *skip connections*. LadderNet tiene esta estructura duplicada: submuestreo + sobremuestreo + submuestreo + sobremuestreo. Por otra parte, todas ellas reutilizan características dentro de cada uno de los caminos por medio de conexiones residuales.

2.14 El proyecto propuesto

Una vez revisados todos los conceptos necesarios y elementos participantes en las tecnologías y métodos empleados en el presente trabajo, este punto se torna en un lugar idóneo para esclarecer aquello que se pretende evaluar y conseguir dentro del proyecto.

En primer lugar, se valorará cuánto de beneficioso es, o en qué punto deja de serlo, el nuevo paradigma de redes más profundas con menor número de parámetros en el dominio de imagen oftalmológica. Las **arquitecturas, tecnologías, mecanismos y técnicas más nuevas** surgidas del interés en seguir esta guía de diseño parecen reportar beneficios en tareas como la segmentación semántica en imágenes *del mundo* (e.g. calles o escenas diarias), pero su éxito en tareas del ámbito médico no parece, *a priori*, asegurado. Las imágenes de dominio

cerrado, como sucede en imagen médica, tienen un nivel de similitud entre ellas a todos los niveles (siempre que sean fruto de la aplicación de la misma técnica de imagen) que no es comparable al de las que componen *datasets generales* como ImageNet. En el ámbito médico, los pequeños detalles son altamente relevantes, y la estructura de las imágenes, conocida. Por esta razón, se evaluará aquí la bondad de unas arquitecturas de red que implementan algunas de las nuevas técnicas descritas y se sitúan más cerca del nuevo paradigma (FC-DenseNet y ENet) y se compararán con la de una arquitectura de referencia con un diseño más simple y clásico (U-Net) en las tareas seleccionadas de reconstrucción y segmentación. Así, se podrá ver si las primeras son capaces de mejorar los resultados de la segunda o si sucede al contrario.

En segundo lugar, se situará otro foco de interés sobre el ***data augmentation***. Como en imagen médica no es común disponer de gran cantidad de datos etiquetados, un mecanismo de ayuda para paliar los efectos de esa escasez puede ser un *augmentation* adecuado. En qué medida puede afectar a los resultados y compensar la falta de ejemplos reales de entrenamiento en el dominio propuesto es uno de los temas objeto de estudio. Para evaluarlo, compararemos los resultados de las redes para diferentes cantidades de imágenes en el conjunto de entrenamiento en presencia de técnicas de *augmentation*, para ver si este es capaz de hacer que para conjuntos pequeños los resultados no se vean muy afectados.

Por último, **en tercer lugar**, se valorará el impacto de la técnica de **aprendizaje transferido**. Por la escasez de datos etiquetados, el empleo de esta aproximación suele resultar útil en muchos dominios. Los ejemplos de redes que han utilizado como punto de partida en entrenamiento modelos preentrenados en ImageNet son muchos. En imagen médica, no obstante, la diferencia existente entre las imágenes del dominio médico y las imágenes *generales* limita la utilización de estos modelos. La reusabilidad de las características extraídas de imágenes *generales*, aunque existente [53, 54], no ofrece unos resultados tan notables cuando se trata de llevarlas a tipos de imagen como los descritos al comienzo de este capítulo [52]. Por ello, inspirándonos en [47], aprovecharemos la disponibilidad de diferentes modos de imagen de fondo de ojo y obtendremos nuestros propios modelos preentrenados de las arquitecturas de interés en una tarea distinta a la objetivo. La primera, tarea *base* o *de preentrenamiento*, será la reconstrucción multimodal, y la segunda, *objetivo* o *finalista*, la segmentación de vasos. Esto nos permitirá evaluar el impacto general del aprendizaje transferido en cada arquitectura, comparando los resultados con y sin preentrenamiento para los diferentes modelos. Además, se evaluará así la validez del paradigma propuesto en [47], donde se augura un buen desempeño de los modelos entrenados para tareas de reconstrucción multimodal como preentrenamientos para otras tareas objetivo distintas (sobre todo en aquellas en las que haya escasez de datos etiquetados). Este buen desempeño se debería principalmente a la reusabilidad del conocimiento adquirido durante la primera tarea.

Capítulo 3

Planificación

En este capítulo se exponen la planificación previa para el desarrollo del proyecto y los recursos y estimaciones de costes materiales y de recursos humanos.

3.1 Modelo de desarrollo

El modelo de desarrollo escogido ha sido el iterativo incremental (Figura 3.1). Se considera este modelo adecuado en el contexto de investigación en el que se enmarca este proyecto porque permite conocer en todo momento el progreso real del mismo, con una revisión continua de los avances realizados y de los puntos de mejora. Además, en cada una de las iteraciones se dispondrá de resultados *software* utilizables que abordarán los principales aspectos del proyecto (de forma creciente) satisfaciendo determinados requisitos y cuestiones relevantes. Esto permitirá evaluar la adecuación o no de los mismos con el objetivo de valorar posibles cambios en futuras iteraciones.

Por todo esto, se considera que el modelo escogido, en primer lugar, se adapta de forma natural al carácter incremental y de exploración de los procesos de experimentación del presente proyecto, y en segundo, que permite un desarrollo adecuado a las necesidades del proceso de investigación, donde cobra gran importancia la evaluación de los avances.

3.2 Planificación previa

Antes de empezar cualquier proyecto, se necesita hacer previamente una planificación de las tareas a realizar y las fases en las que organizarlas. A continuación, se muestran las principales.

Estudio y análisis del dominio Esta fase tiene por objetivo adquirir los conocimientos necesarios del dominio de la oftalmología y de las tecnologías empleadas en el proyecto: las redes de neuronas artificiales y, más concretamente, las convolucionales, junto con muchas de las técnicas, operaciones y algoritmos de los que hacen uso. También, tiene por fin estudiar con detalle todos los trabajos más directamente relacionados con el proyecto. Para ambas partes se recabará información a través de publicaciones de carácter científico.

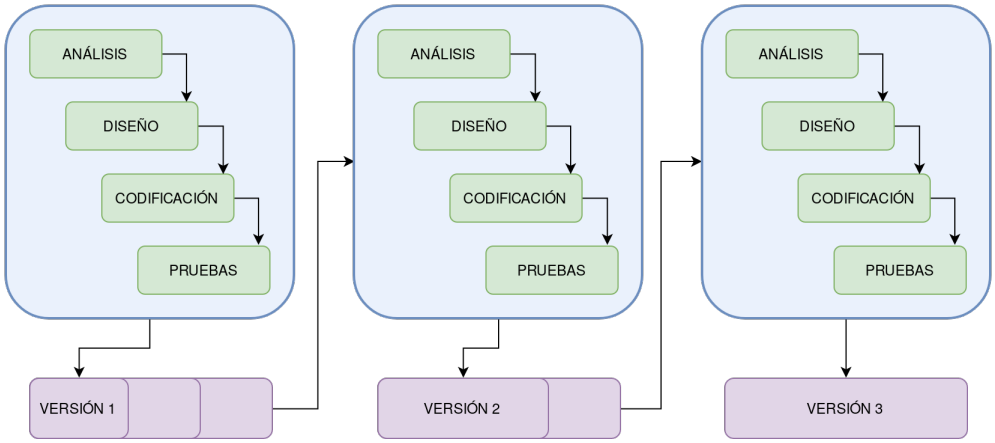


Figura 3.1: Esquema básico del modelo de desarrollo iterativo incremental.

Reconstrucción multimodal Esta fase comprende todo lo relacionado con la primera tarea abordada, la reconstrucción multimodal. En ella se preparán los conjuntos de datos, se planteará la metodología a utilizar, se implementará, y se realizará un estudio de los resultados obtenidos, comprobando la calidad y validez de las implementaciones, de forma que se pueda valorar si se han alcanzado los objetivos fijados para la tarea. Esta fase, además, estará directamente relacionada con la siguiente por la utilización de algunos de los modelos como preentrenamientos para la segmentación.

Segmentación Esta fase comprende todo lo relacionado con la segunda tarea abordada, la segmentación de vasos. En ella se preparán los conjuntos de datos, se planteará la metodología a utilizar, se implementará, y se realizará un estudio de los resultados obtenidos, comprobando la calidad y validez de las implementaciones, de forma que se pueda valorar si se han alcanzado los objetivos fijados en relación a la tarea de reconstrucción.

Realización de la memoria Esta fase comprende todo lo relacionado con la realización de la memoria. En ella se organizarán los resultados y se redactará la memoria, donde se recogerán todo el trabajo realizado, los resultados obtenidos, un minucioso comentario de los mismos, y todas las cuestiones y posibles líneas a seguir que fueron surgiendo durante el desarrollo del proyecto.

En la Figura 3.2 se muestran las estimaciones de tiempos para las diferentes fases y tareas. Estas estimaciones se han realizado en base a proyectos similares de investigación realizados a lo largo de años pasados. Por otra parte, las fases dedicadas a las dos tareas mencionadas se han planteado en iteraciones incrementales, de forma que en cada una de estas iteraciones se alcance alguno de los objetivos parciales establecidos para la tarea.

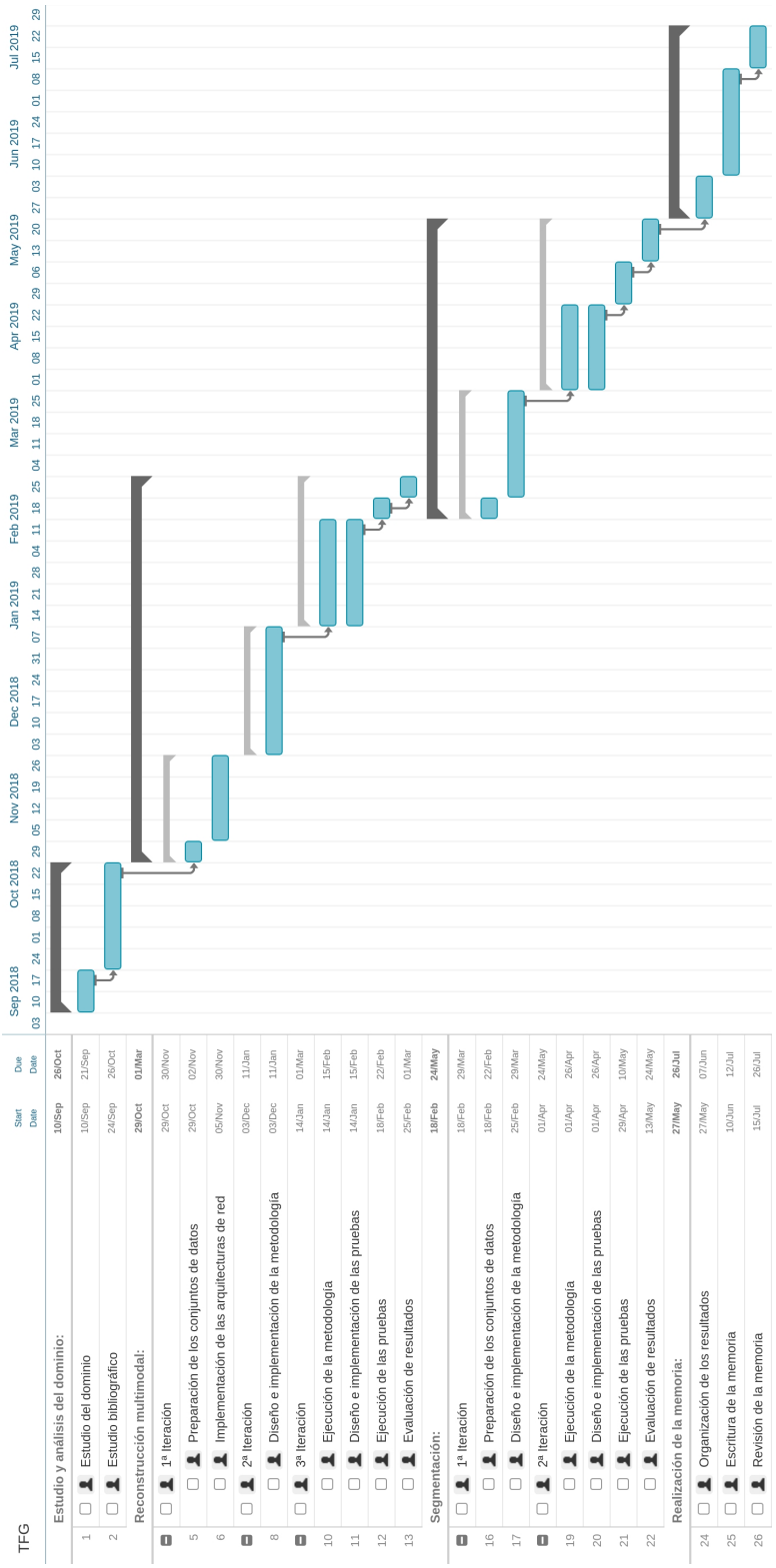


Figura 3.2: Diagrama de Gantt con la línea base del proyecto.

3.3 Recursos

En el transcurso del proyecto se han requerido distintos tipos de recursos materiales y humanos[1]. En los primeros podemos distinguir dos tipos: *hardware* y *software*.

A continuación detallaremos cada uno de los recursos empleados de los diferentes tipos y describiremos brevemente su papel dentro del proyecto.

3.3.1 Recursos materiales

Dentro de los recursos materiales tenemos los recursos *hardware* y *software* que se listan y detallan, junto con su función, en los siguientes apartados.

Recursos *hardware*

- Ordenador portátil Lenovo Legion Y530-15ICH
  - Procesador: Intel® Core™ i5-8300H @ 2.30GHz; 4 *cores*, 8 *threads*
  - Memoria RAM: 16GB
  - Sistema operativo: Pop!_OS 18.04 LTS
  - Tarjeta gráfica: Nvidia GeForce GTX 1050 (4GB)
  - Almacenamiento: SSD 240GB
- Servidor del grupo VARPA
  - Procesador: Intel® Xeon® CPU E5-2650 v4 @ 2.20GHz; 48 *cores*
  - Memoria RAM: 128GB
  - Tarjeta gráfica: Nvidia Tesla K80 (24GB)

El ordenador portátil Lenovo Legion Y530-15ICH ha sido utilizado para todo el trabajo (implementaciones, redacción de la memoria, almacenamiento de resultados, etc.) salvo para el entrenamiento de las redes de neuronas artificiales. Para esto último se ha utilizado el servidor del grupo VARPA, mucho más adecuado para este fin, como se puede ver por sus especificaciones.

[1] Junto a estos dos tipos se podrían contar también los recursos bibliográficos, que ya aparecen detallados en el apartado de referencias.

Recursos *software*

- Visual Studio Code 1.36.0
- PyTorch 0.4.0

EL programa Visual Studio Code se ha empleado para hacer todas las implementaciones relacionadas con el trabajo y para la escritura de la memoria. La librería PyTorch de aprendizaje máquina se ha utilizado para todo lo directamente relacionado con las redes de neuronas: implementación de las arquitecturas, procesos de entrenamiento, test, etc.

Por otra parte, el lenguaje de programación empleado para todo el código es Python versión 3.6. Otras librerías fueron utilizadas también para el proyecto, pero con un papel secundario. Entre ellas se podrían destacar Matplotlib o scikit-image.

3.3.2 Recursos humanos

En el apartado de recursos humanos se asumen cuatro roles principales: jefe de proyecto, analista, diseñador y programador.

El primero de los roles corresponderá a los directores de proyecto, que se encargarán de la supervisión del trabajo del alumno, mientras que a este último corresponderán los otros tres roles, por ser el Trabajo Fin de Grado (TFG) un proyecto de carácter individual.

El trabajo de los primeros viene representado por reuniones semanales de aproximadamente dos horas en las que se resolverán las dudas del alumno y se guiará el desarrollo del proyecto. El trabajo del alumno se realizará a través de una dedicación a tiempo parcial a lo largo de todo el curso.

3.4 Estimación de costes

3.4.1 Costes estimados de los materiales

Como se indica a continuación, los costes materiales del presente proyecto son nulos.

Costes estimados del *hardware*

Los costes del *hardware* los podemos considerar nulos, ya que todos elementos empleados estaban disponibles antes del comienzo del proyecto[2].

[2]Se podría introducir aquí el coste del mantenimiento del servidor, pero se ha decidido obviar por no ser exclusivo de este proyecto.

	Número	Salario	horas × hombre	Inversión
Director de proyecto	2	30€/h	176	5280€
Analista-Diseñador -Programador	1	20€/h	880	17600€
			Total:	22880€

Cuadro 3.1: Estimación de coste de los recursos humanos.

Costes estimados del *software*

Todo el software utilizado (tanto librerías como programas) es de código abierto y gratuito, por lo que en este apartado los costes también se consideran nulos.

3.4.2 Costes estimados de los recursos humanos

Como se ha apuntado antes, el alumno tendrá los roles de analista, diseñador, y programador y los directores del proyecto el de jefes de proyecto.

Con una dedicación parcial de 4 horas al día, suponiendo 5 días laborables por semana a lo largo de las 44 semanas del curso, al alumno le corresponde una estimación total de 880 horas × hombre. A los 2 directores de proyecto, asignándoles un total de 2 horas semanales en forma de reuniones con el alumno, les corresponderá un total de 176 horas × hombre repartidas entre los dos.

Teniendo todo esto en cuenta, y suponiendo un salario de 30€/h para los jefes de proyecto y de 20€/hora para el analista-diseñador-programador, los costes de recursos humanos estimados (que en este caso coinciden con los **costes totales estimados** por ser los costes materiales 0) son los que se pueden ver en el Cuadro 3.1.

Capítulo 4

Arquitecturas de red

En este trabajo, durante el estudio, se tomará como arquitectura *de referencia* a la arquitectura U-Net [1], ampliamente utilizada en dominios de imagen como el que nos ocupa en este trabajo[1], y como arquitecturas *candidatas* (a comparar con U-Net) FCDN [2] y ENet [3]. El diseño de U-Net es el más clásico de las tres. FCDN y ENet se han seleccionado por implementar diferentes técnicas y operaciones innovadoras de las vistas en el Capítulo 2. Además, también motivaron su selección, en el caso de FCDN, los buenos resultados obtenidos en tareas de segmentación y su integración de ciertas ideas de U-Net y DenseNet; y en el de ENet, poder evaluar una arquitectura con algunos mecanismos distintos a los de las anteriores y con un diseño más compacto, de menor capacidad, que las otras dos.

Todas estas arquitecturas fueron presentadas entre el año 2015 y 2017 (U-Net: 2015, ENet: 2016, y FCDN: 2017) como redes de segmentación, y todas contienen algunas de las ideas arquitectónicas descritas en el Capítulo 2.

A continuación se detallan las características de cada una de ellas y algunas de sus semejanzas y diferencias. En el Apéndice C se muestra un resumen comparativo entre todas para facilitar una visión de conjunto.

4.1 U-Net

La arquitectura **U-Net** fue presentada el año 2015 por Olaf Ronneberger et al. en [1]. Es una arquitectura completamente convolucional, con caminos de contracción y expansión simétricos y conectados por *skip connections*, que permite efectuar localizaciones precisas a partir de pocas imágenes de entrenamiento. En el mismo año que se presentó alcanzó los primeros puestos en el *Grand Challenge for Computer-Automated Detection of Caries in Bitewing Radiography* y el *Cell Tracking Challenge*, ambos del ISBI[2] 2015 [60].

Desde su publicación en el MICCAI[3] 2015 ha tenido un gran impacto en el ámbito del aprendizaje profundo orientado a imagen, y particularmente en el relacionado con imagen biomédica [58, 61, 62, 63, 64, 65].

[1] Entre los que también se encuentra el trabajo más directamente relacionado con el nuestro y descrito en el Capítulo 2.

[2] International Symposium on Biomedical Imaging.

[3] International Conference on Medical Image Computing and Computer Assisted Intervention

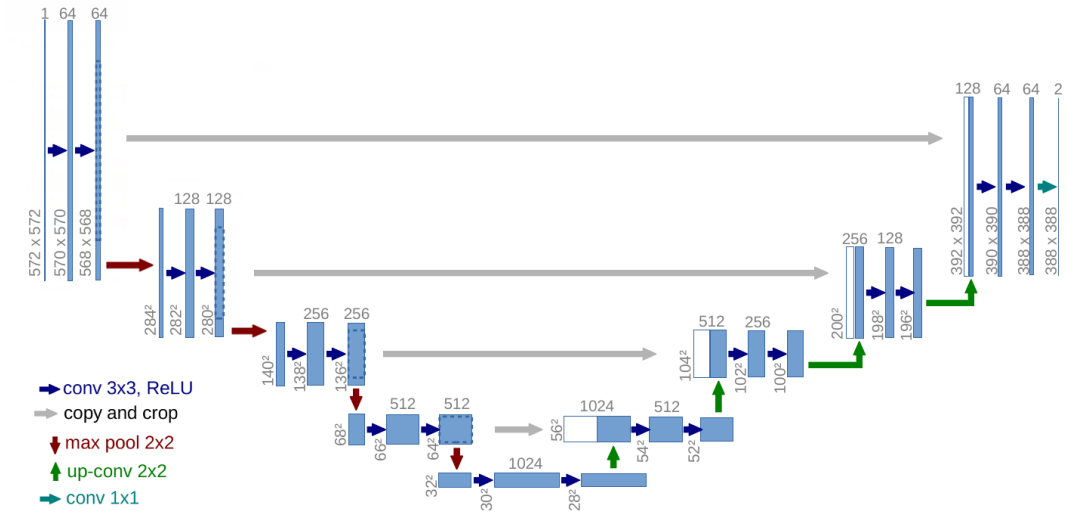


Figura 4.1: Diagrama de la arquitectura U-Net.

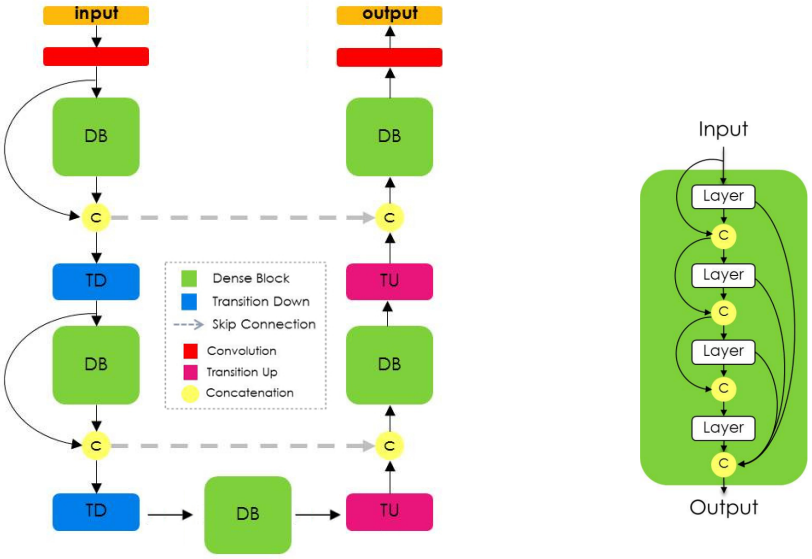
4.1.1 Características arquitectónicas

U-Net es una arquitectura de red neuronal completamente convolucional originalmente presentada para tareas de segmentación en imagen biomédica [1]. Es una extensión de la arquitectura expuesta en [42] (FCN) y previamente comentada en la Sección 2.11. La arquitectura se compone de dos partes (rutas, o caminos), una de contracción (de submuestreo, *downsampling*, o codificación) y otra de expansión (de sobremuestreo, *upsampling*, o decodificación), que se pueden ver en la Figura 4.1 a izquierda y derecha respectivamente.

La parte de contracción está formada por varios bloques de submuestreo. En estos bloques se utilizan cascadas de convoluciones de 3×3 tipo VGGNet con función de activación ReLU, seguidos por una operación de *max-pooling* de 2×2 con paso (*stride*) 2. En cada etapa del submuestreo se duplican el número de canales de características (filtros), mientras que se va reduciendo la dimensionalidad ($\times 2$ a lo alto y ancho) debido al *pooling*.

La parte de expansión está formada por bloques de sobremuestreo que van recuperando paulatinamente la resolución original de la entrada. En cada bloque, en primer lugar, se hace un sobremuestreo del mapa de características usando una convolución traspuestas de 2×2 seguida por una **concatenación** con el mapa de características del nivel correspondiente en la parte de contracción. En la Figura 4.1 las flechas grises representan las *skip connections* que permiten conectar los bloques de ambas partes. El bloque finaliza con una cascada de convoluciones 3×3. El número de características de cada bloque se reduce a la mitad con respecto al anterior.

Las *skip connections*, tal y como han sido pensadas dentro de esta arquitectura, resultan de gran utilidad, pues permiten *compartir* las características del camino de submuestreo con el



(a) (b)

Figura 4.2: (a) Diagrama de la arquitectura FCDN. (b) Diagrama de un bloque denso de 4 Capas [2].

de sobremuestreo. Esto es útil para recuperar detalles en la imagen que pueden ser relevantes a la hora de alcanzar el resultado final —de ahí, en parte, la precisión de las segmentaciones—.

En la capa final se utiliza una convolución de 1×1 para asignar cada vector de características con 64 componentes al número de clases correspondiente (se pasa de 64 canales a solo 2 en este caso). Esta reducción de dimensionalidad es similar a la utilizada en los bloques *inception*.

4.2 Fully Convolutional DenseNet

La arquitectura Fully Convolutional DenseNet (FC-DenseNet o **FCDN**) fue presentada en el *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops* (CVPRW) por Simon Jégou et al. en [2]. Es una red completamente convolucional orientada a problemas de segmentación que está basada en DenseNet [41] e influida por U-Net. Cuenta con la peculiaridad de que en el artículo original[4] se definen 3 modelos distintos de la arquitectura con distinto número de capas. Se evaluó en tareas de segmentación semántica con los conjuntos de datos CamVid [66] y Gatech [67], que contienen imágenes de entornos urbanos. Los resultados obtenidos con el modelo con mayor número de capas superaron en algunos casos los que en aquel momento conformaban el estado del arte.

[4]También en la implementación de los autores disponible en `https://github.com/SimJeg/FC-DenseNet`.

Capa
Batch Normalization
ReLU
Convolución 3×3
Dropout $p = 0.2$

Transición hacia abajo (TD)
Batch Normalization
ReLU
Convolución 1×1
Dropout $p = 0.2$
Max-pooling 2×2

Transición hacia arriba (TU)
Convolución transpuesta 3×3 de paso 2

Cuadro 4.1: Bloques de construcción de la arquitectura FCDN [2].

4.2.1 Características arquitectónicas

Como en el caso de U-Net, es una arquitectura de red completamente convolucional, con un camino de codificación y otro de decodificación (ver Figura 4.2a partes izquierda y derecha respectivamente) y con *skip connections* entre ambas partes. Más allá de esto, las arquitecturas son notablemente diferentes en bastantes aspectos.

Aquí detallaremos solamente el modelo de 103 capas, FCDN103, por ser *a priori* el de mayor potencia computacional y el de mayor relevancia en el artículo. Los otros dos siguen los mismos principios y los cambios no son especialmente significativos, aunque se hará una breve mención a ellos.

La arquitectura FCDN se construye a base de bloques, como puede verse en la Figura 4.2a. Hay bloques de tres tipos: **bloques densos** (DB), de **transición hacia arriba** (TU) y de **transición hacia abajo** (TD). Además, podría considerarse que hay un cuarto subbloque utilizado para formar los bloques densos, que como se puede ver en la Figura 4.2b, se denomina simplemente **Capa** (*Layer*) (lo nombraremos con la letra inicial mayúscula por claridad).

En el Cuadro 4.1 se muestra la arquitectura detallada de los bloques Capa, TD y TU. En la Figura 4.2b, por otro lado, se muestra un esquema de un DB de 4 Capas, con las concatenaciones que se producen a lo largo del bloque y entre las diferentes Capas.

Detalle de los bloques Las **Capas** de los **bloques densos** se componen de una operación de BN seguida por una ReLU, una convolución 3×3 y *dropout* con probabilidad $p = 0.2$. El *ratio de crecimiento* (ver definición en Sección 2.10.1) de la capa está fijado a $k = 16$. Estos mapas son los que después sirven de entrada a la siguiente capa. Los bloques **TD** están compuestos por una operación de BN seguida de una ReLU, una convolución de 1×1, *dropout* de probabilidad $p = 0.2$ y *max-pooling* sin solapamiento de 2×2. Por su parte, los bloques **TU** están compuestos de una convolución 3×3 con paso 2 para compensar la operación de *pooling*.

Arquitectura
Entrada, $m = 3$
Convolución 3×3, $m = 48$
DB (4 capas) + TD, $m = 112$
DB (5 capas) + TD, $m = 192$
DB (7 capas) + TD, $m = 304$
DB (10 capas) + TD, $m = 464$
DB (12 capas) + TD, $m = 656$
DB (15 capas), $m = 896$
TU + DB (12 capas), $m = 1088$
TU + DB (10 capas), $m = 816$
TU + DB (7 capas), $m = 578$
TU + DB (5 capas), $m = 384$
TU + DB (4 capas), $m = 256$
Convolución 1×1, $m = c$
Softmax

Cuadro 4.2: Detalles del modelo de 103 capas de la arquitectura FCDN (FCDN103), siendo m el número de mapas de características al final de un bloque y c el número de clases [2].

Detalle de la arquitectura al completo

En el caso del modelo de 103 capas convolucionales, FCDN103, se tiene la organización de bloques y capas que se muestra en el Cuadro 4.2.

A una entrada que consta de 3 canales, R G y B, le suceden una convolución (1 capa) y un camino de submuestreo compuesto por 5 bloques densos a los que les sigue un TD (en total, 38+5 capas). La salida de cada bloque denso se concatena con la entrada del bloque (ver flechas negras en Figura 4.2b) para formar los mapas de características que servirán de entrada para el siguiente bloque. El camino de submuestreo finaliza en el cuello de botella de la red, formado por un bloque denso de 15 capas (parte de abajo de la Figura 4.2a).

En el camino de sobremuestreo, se tienen 5 bloques densos precedidos de un TU (38+5 capas). Finalmente, se tienen una convolución de 1×1 (1 capa, que sumada a las anteriores hace un total de 103) y una *softmax* para obtener la distribución por clases para cada píxel.

Las entradas de los bloques densos del camino de sobremuestreo son fruto de la concatenación de la salida de las TU y las *skip connections* del camino de submuestreo —que vienen del punto «simétrico» de la arquitectura—. En el camino de sobremuestreo se evita concatenar las entradas y las salidas de cada bloque denso para evitar que el incremento de la resolución espacial de los mapas de características conduzca a una explosión demasiado grande del número de características. Esto permite tener un total de «solo» 256 mapas de características justo antes de la última convolución.

Modelos de menor tamaño. Los modelos de la arquitectura de menor tamaño, de 56 y 67 capas (FCDN56 y FCDN67, respectivamente), son muy similares al descrito en el apartado anterior: las únicas diferencias son los ratios de crecimiento (k) y el número de Capas de los bloques densos. En el caso de la primera, $k = 12$ con 4 Capas por DB, y en la segunda $k = 16$ con 5 Capas por DB.

4.2.2 Discusión

La arquitectura FCDN aplica algunas de las ideas de U-Net y DenseNet. El resultado es una red con un gran desempeño con aproximadamente 3 veces menos parámetros que U-Net, en su modelo con más parámetros, y con más del doble de profundidad en cualquiera de sus modelos. Esto es posible gracias a la utilización de los mencionados bloques densos.

La ganancia en profundidad se podría considerar, a priori, una ventaja con respecto a U-Net, pues se ha visto que la acumulación de capas suele permitir a las redes crear representaciones internas más complejas, generalizando mejor con un menor número de parámetros [34, 36, 38].

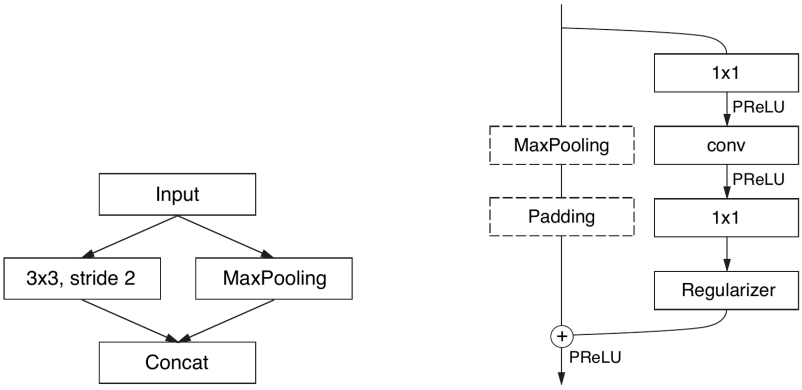
Por otro lado, los bloques densos aportan ciertas ventajas, como la eficiencia a nivel paramétrico, la reutilización de características y la supervisión profunda implícita, como se explicó en la Sección 2.10 dedicada a DenseNet. La reutilización de características también está presente en U-Net a través de las *skip connections*, pero en los bloques densos se hace extensivo a las capas de las que estos se componen, por lo que se puede considerar que en este punto FCDN tiene un potencial mayor.

4.3 Efficient Neural Network

La arquitectura Efficient Neural Network (**ENet**) fue presentada en el año 2016 en [3] por Adam Paszke et al. Al igual que FCDN, se propuso para tareas de segmentación semántica, pero con especial énfasis en el bajo tiempo de procesamiento y en el tamaño reducido de la red. Su evaluación se realizó haciendo segmentación semántica en imágenes de entornos urbanos pertenecientes a los conjuntos de datos CamVid [66], Cityscapes [68] y SUN [69]. Las comparaciones con otras redes o metodologías se hicieron tanto a nivel de resultados de segmentación como de eficiencia, siendo especialmente competitiva en este último apartado.

4.3.1 Características arquitectónicas

ENet es una arquitectura completamente convolucional, con una parte de submuestreo y otra de sobremuestreo (ver Tabla 4.3). A nivel global, la arquitectura se organiza en diferentes *etapas*. En el Cuadro 4.3 pueden verse representadas por las separaciones de líneas horizontales. Las 3 primeras, sin contar la inicial, conforman la parte de submuestreo, y las números 4 y



(a) (b)

Figura 4.3: (a) Bloque inicial de ENet. (b) Módulo *bottleneck* de ENet. [3]

5, la de sobremuestreo. Cada *etapa* está compuesta por uno o varios bloques de construcción como los que se pueden ver en la Figura 4.3. En todos ellos se introduce una operación de *Batch Normalization* entre cada convolución y su consiguiente no-linealidad (PReLU).

Al principio de la arquitectura se sitúa el módulo inicial, que se puede ver en la Figura 4.3a. Este módulo produce una reducción de la resolución de la entrada (y con ello una reducción del número potencial de características) a través de la concatenación del resultado de una convolución 3×3 de *paso 2* y un *max pooling* 2×2.

El resto de la red está compuesta, salvo justo al final, por módulos *bottleneck* como el que se puede ver en la Figura 4.3b. Este módulo, con el mismo nombre y muy similar al presente en ResNet, se divide siempre en dos caminos con la misma entrada, y las salidas de los mismos son siempre sumadas elemento a elemento. Dependiendo del tipo de módulo, si es de submuestreo o de sobremuestreo, estos dos caminos varían sus contenidos.

El primer camino se muestra a la izquierda en la Figura 4.3b. Si el *bottleneck* es de submuestreo, este camino se compone de una operación de *max-pooling* (para reducir la resolución de la imagen a la mitad) y de un *padding* (para igualar la resolución con el otro camino y poder hacer la suma después). Si es de sobremuestreo, en lugar de utilizarse *max-pooling* se utiliza *max-unpooling*, y el *padding* se sustituye por *convoluciones espaciales* (convoluciones 2D que actúan de forma separada en cada uno de los canales[5]) sin *bias*.

El segundo camino (el de la derecha en la Figura 4.3), se compone de 3 convoluciones (2 de ellas 1×1) entre las que siempre hay una operación de BN y una PReLU. Si el módulo es de submuestreo, la primera convolución de 1×1 se sustituye por una de 2×2 de paso 2 en las dos dimensiones, lo que reduce la resolución de la imagen a la mitad. La convolución `conv` que se puede ver en la figura utiliza filtros 3×3 y puede ser una convolución normal, una dilatada o

[5] https://keras.io/layers/convolutional/

Nombre	Tipo	Tamaño de salida
initial		$16 \times 256 \times 256$
bottleneck1.0	downsampling	$64 \times 128 \times 128$
$4\times$bottleneck1.x		$64 \times 128 \times 128$
bottleneck2.0	downsampling	$128 \times 64 \times 64$
bottleneck2.1		$128 \times 64 \times 64$
bottleneck2.2	dilated 2	$128 \times 64 \times 64$
bottleneck2.3	asymmetric 5	$128 \times 64 \times 64$
bottleneck2.4	dilated 4	$128 \times 64 \times 64$
bottleneck2.5		$128 \times 64 \times 64$
bottleneck2.6	dilated 8	$128 \times 64 \times 64$
bottleneck2.7	asymmetric 5	$128 \times 64 \times 64$
bottleneck2.8	dilated 16	$128 \times 64 \times 64$
Repetir sección 2, sin bottleneck2.0		
bottleneck4.0	upsampling	$64 \times 128 \times 128$
bottleneck4.1		$64 \times 128 \times 128$
bottleneck4.2		$64 \times 128 \times 128$
bottleneck5.0	upsampling	$16 \times 256 \times 256$
bottleneck5.1		$16 \times 256 \times 256$
fullconv		$C \times 512 \times 512$

Cuadro 4.3: Arquitectura ENet, con tamaños de salida para una entrada de 512×512.

una traspuesta. A veces es sustituída por una convolución asimétrica: convoluciones sucesivas de 5×1 y 1×5. La regularización se hace por medio de un *SpatialDropout* de probabilidad $p = 0.01$ antes del *bottleneck2.0* y de $p = 0.1$ después.

Al final de la red, para dar lugar a una salida con C canales, uno por clase, se sitúa una convolución traspuesta con C filtros.

Hay que destacar que ENet **no** utiliza *skip connections* entre las partes de submuestreo y sobremuestreo, por lo que en este sentido difiere de FCDN y U-Net.

4.3.2 Discusión

La arquitectura ENet, como se adelantó, ha sido seleccionada principalmente por su formato compacto, con «pocos» parámetros. Además, algunos de los elementos que la componen, como las convoluciones dilatadas y el *unpooling*, tienen también interés a nivel de diseño. La evaluación de su comportamiento, observando si estos mecanismos innovadores son capaces de paliar su relativamente pequeña capacidad, es uno de los puntos de interés principales.

Por otra parte, su rendimiento en las dos tareas seleccionadas nos podría reportar claves relevantes a la hora de valorar la adecuación de redes más livianas, de menor capacidad, a tales tareas o a tareas similares.

Capítulo 5

Reconstrucción multimodal de angiografía a partir de retinografía

En este capítulo se hace una descripción en detalle de todo lo relacionado con el primero de los experimentos abordados: la **reconstrucción multimodal de angiografía a partir de retinografía**. Además, se presentan y discuten los resultados obtenidos y se exponen las conclusiones que de ellos se han podido extraer.

5.1 Reconstrucción multimodal

Tal y como se puede ver en la Figura 5.1, el experimento, en global, consiste en obtener distintas redes de neuronas capaces de construir una imagen con el aspecto y las características de una angiografía con fluoresceína —*pseudoangiografía*— a partir de una retinografía: la aquí llamada *reconstrucción multimodal*. Para ello, las redes de neuronas son entrenadas de forma supervisada en un conjunto de datos adecuado.

En concreto, lo que se hace es entrenar los diferentes modelos en la tarea de reconstrucción para un conjunto de datos creado para este fin, y después, evaluar (tiempo de inferencia, test) estos modelos en otros dos conjuntos diferentes al primero. Las reconstrucciones resultado generadas por las redes a partir de las imágenes de estos dos conjuntos son sobre las que finalmente se aplican dos tipos de evaluación: *cualitativa*, que mide visual y subjetivamente la calidad de tales imágenes, y *cuantitativa*, que mide esta calidad de forma aproximada, objetiva y conforme a un criterio dado. La primera de las evaluaciones dará lugar a observaciones relevantes, y la segunda, a números objetivos.

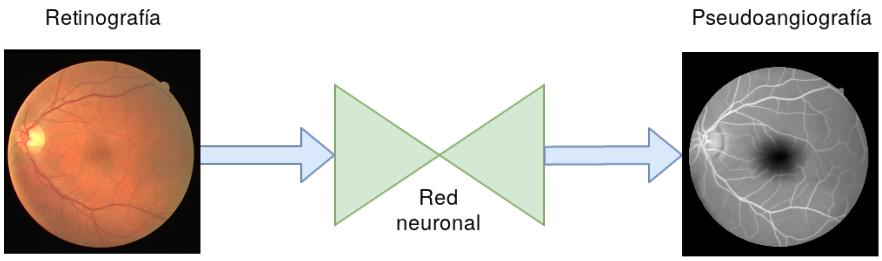


Figura 5.1: Esquema de la tarea de reconstrucción multimodal.

Las arquitecturas de red utilizadas en el experimento son las descritas en el capítulo 4: **U-Net**, **FC-DenseNet** (los 3 modelos) y **ENet**. A todas ellas se les suprimió la operación final de *softmax*, más pensada para tareas de clasificación que de segmentación.

5.2 Experimentación extendida con FC-DenseNet

En esta parte del trabajo se han hecho algunas pruebas adicionales sobre la arquitectura **FC-DenseNet56**[1] (FCDN56). La realización de estas pruebas tiene dos objetivos. Primero, obtener una red que pueda competir con la red de referencia, U-Net. Segundo, hacer una pequeña evaluación de lo ventajoso o no de la presencia de algunos de los elementos de construcción de la arquitectura valorando el impacto de los mismos en los resultados. En concreto, de *Batch Normalization* (BN), *Dropout* y *bias* de la convolución final.

Inicializaciones Las inicializaciones serán también aquí objeto de evaluación. Para ello, se medirá el impacto en los resultados de aplicar dos métodos de inicialización distintos: métodos de He y de LeCun. Esto permitirá detectar los beneficios de uno sobre otro, en el caso de que existan. Para evitar la explosión combinatoria, se optó por no incluir este factor en las pruebas relacionadas con los elementos arquitectónicos. En lugar de ello, se hizo una prueba por cada versión de FCDN (modelos de 56, 67 y 103 capas) para la variante con mejores resultados obtenida de las pruebas con BN, *dropout* y *bias* (convolución final).

Variantes arquitectónicas Para evaluar el impacto de estos elementos en el resultado final de reconstrucción se han creado las variantes de la arquitectura FCDN56 (FC-DenseNet en su modelo de 56 capas) que se pueden ver en el Cuadro 5.1 (se hablará más adelante de los modelos para las inicializaciones). En él los círculos negros reflejan la presencia del elemento, mientras que un espacio en blanco representa su ausencia. En el caso de la inicialización, se especifica el nombre de cuál se está utilizando. Por otro lado, los nombres de las variantes se construyen con cuatro caracteres del modo que se detalla a continuación. El primer carácter indica si el método de inicialización es el de He o el de LeCun por medio de las letras H y L, respectivamente. Los otros tres caracteres indican la presencia o no de Dropout (letra D), Batch Normalization (N) y *bias* en la convolución final (B), en este orden. Cuando sí están presentes, su letra asociada también lo está, pero cuando no lo están, en lugar de la letra correspondiente se utiliza, por claridad, el símbolo «+».

La variante de este modelo menor de 56 capas que mejores resultados obtenga será la que determine los cambios a realizar en los modelos más profundos de 67 y 103 capas.

[1] Por la explosión de posibilidades y su coste, solo se han hecho pruebas de este tipo para esta arquitectura.

	Características			
Variante	**Inicialización**	**Dropout**	**Batch Norm.**	**Bias Conv. Final**
LDN+	LeCun	●	●	
LDNB	LeCun	●	●	●
L+NB	LeCun		●	●
L++B	LeCun			●
L+++	LeCun			
LD+B	LeCun	●		●
LD++	LeCun	●		
L+N+	LeCun		●	

Cuadro 5.1: Variantes arquitectónicas de la arquitectura FCDN56.

Cuando haya variantes de FCDN con las mismas modificaciones pero con distinta profundidad, a los nombres de cada arquitectura modificada se les añadirán un punto y el número de capas. Por ejemplo, para referirnos a la variante de la primera fila del Cuadro 5.1 utilizaríamos el nombre «**LDN+.56**». A veces, por claridad, también se podrá añadir la arquitectura a modo de prefijo (e.g. «**FCDN.LDN+.56**»).

5.3 Conjuntos de datos

5.3.1 Entrenamiento y validación

Para que el aprendizaje sea posible, como se ha mencionado, es necesario un conjunto de datos adecuado, es decir, un conjunto de datos que contenga pares retinografía-angiografía que se puedan comparar directamente punto a punto. Para ello, es necesario que los pares de esos conjuntos estén alineados, de forma que haya en cada par una correspondencia punto a punto entre los elementos presentes en las imágenes, especialmente los vasos sanguíneos.

Así, la red puede aprender cómo se representa exactamente cada elemento presente en las retinografías en su correspondiente angiografía.

Para disponer de los pares retinografía-angiografía se ha recurrido al conjunto de datos *Isfahan MISP* [70]. Este conjunto está formado por un total de 59 pares de imágenes: 29 sin indicios de patología alguna y 30 con signos característicos de retinopatía diabética. La resolución de todas ellas es de 720×576 píxeles.

Las imágenes de Isfahan de las que se dispuso fueron alineadas por medio del método evaluado en este mismo conjunto y descrito en [71]. Este método combina una aproximación basada en características, utilizando marcas (*landmarks*) específicas del dominio, con una aproximación basada en intensidad que hace uso de una métrica de similitud adaptada al dominio. En la Figura 5.2 puede verse un ejemplo de una retinografía de Isfahan antes y

(a) (b) (c) (d)

Figura 5.2: (a) Angiografía sin alinear. (b) Retinografía asociada a la angiografía *a*. (c) Angiografía alineada. (d) Máscara del FOV de la angiografía alineada.

después de realizarse el alineamiento.

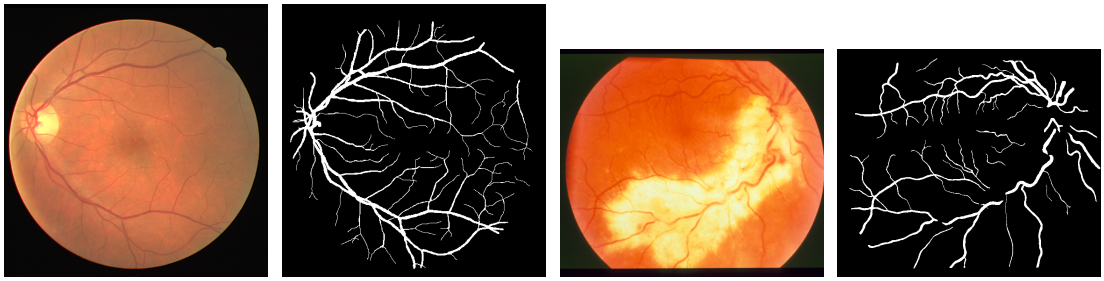
Estas imágenes alineadas son utilizadas tanto para el conjunto de entrenamiento como para el de validación. Para el de entrenamiento, se han escogido al azar 44 imágenes, y se han dejado el resto (15) para el conjunto de validación (proporción de 3 a 1).

5.3.2 Test

Para evaluar el funcionamiento de las redes (es decir, para el test) se utilizan otros dos conjuntos de datos diferentes al de entrenamiento: DRIVE (*Digital Retinal Images for Vessel Extraction*) [72] y STARE (*STructured Analysis of the Retina*) [73].

DRIVE consta de un total de 40 retinografías. 33 sin indicios de patología, y 7 con signos característicos de los estadios iniciales de retinopatía diabética. Estas imágenes, en el reparto original, se dividen en dos conjuntos, uno de entrenamiento y otro de test [72]: *DRIVE-train* y *DRIVE-test*. Todas ellas están acompañadas de una máscara del *Field Of View* (FOV) —en todas circular— que marca la región de interés de la imagen, es decir, los píxeles de la imagen que se corresponden realmente con el fondo de ojo (por ejemplo, en la Figura 5.3a, el FOV es la parte circular de la imagen que no es de color negro). Además, estas imágenes se acompañan de máscaras de segmentación de vasos sanguíneos (imágenes binarias vaso-fondo —blanco, negro, respectivamente—) creadas por expertos (*groundtruth*). Si el conjunto es de test (*DRIVE-test*), hay dos máscaras por imagen (creadas por dos expertos distintos), y si es de entrenamiento (*DRIVE-train*), una creada por uno solo de los expertos. En la Figura 5.3 se puede ver un ejemplo de una retinografía con su segmentación de vasos sanguíneos asociada. La resolución tanto de las máscaras como de las imágenes es de 768×584. La profundidad de color de las retinografías es de 8 *bits*.

STARE se compone de 397 retinografías entre las que hay tanto imágenes con signos de patologías o lesiones como imágenes «sanas». En este caso, la FOV no es completamente circular y la resolución de las imágenes es de 700×605. De entre todas, hay un total de 20 que disponen de una segmentación de vasos manual de dos expertos: Adam Hoover y Valentina



(a) (b) (c) (d)

Figura 5.3: (a) Retinografía de DRIVE. (b) Máscara de vasos de la retinografía *a*. (c) Retinografía de STARE. (d) Máscara de vasos de la retinografía *c*.

Kouznetsova. Las primeras son las más simples, con menos vasos segmentados, mientras que las segundas son muy exhaustivas, y el número de vasos que en ellas aparecen es bastante mayor. Las imágenes que disponen de estas segmentaciones son las que se utilizan como conjunto de evaluación complementario a DRIVE, y en nuestro caso utilizaremos las segmentaciones de Adam Hoover (AH). En la Figura 5.3 se muestra un ejemplo de una retinografía de este conjunto con su máscara de segmentación de vasos sanguíneos asociada.

Se ha decidido seleccionar solamente imágenes con máscaras de segmentación de vasos asociadas porque así lo exige, como se verá, uno de los tipos de evaluación propuestos.

5.4 Detalles de entrenamiento de las redes

La metodología de entrenamiento escogida es común a todas las redes que se han seleccionado. Esto se ha hecho así para poder comparar adecuadamente el comportamiento de todas ellas en la tarea de reconstrucción.

Para calcular el *error de reconstrucción*, inspirándonos en los resultados de [47], utilizamos la **función de error SSIM** (ver Apéndice A, apartado 11.2, para más detalles). Como esta es una medida de similitud, se utiliza «menos SSIM», $-SSIM$. El cálculo solo tiene en cuenta la región de interés, el *Field Of View*. Utilizando SSIM de esta forma, a la hora de hacer los cálculos solo se tienen en cuenta aquellos píxeles que estén dentro del FOV. Se puede saber cuáles de ellos pertenecen a esta región gracias a la máscara del FOV que acompaña a las imágenes de fondo de ojo.

Para optimizar las redes durante el entrenamiento se ha utilizado el **optimizador estocástico Adam**. En todos los casos el tamaño del *mini-batch* es de una sola imagen. La detención del entrenamiento se produce cuando se agota una paciencia de 100 épocas sin mejorar el error en el conjunto de validación.

Para paliar lo máximo posible la escasez de imágenes disponibles para el entrenamiento, se utiliza un *augmentation* consistente en (1) transformaciones afines aleatorias de las imágenes originales, (2) ligeras variaciones de color e intensidad y (3) inversiones aleatorias en los ejes x e y. Este *augmentation* se hace además en tiempo real, con ciertas probabilidades definidas para cada tipo de operación. Todas estas transformaciones son muy comunes y pueden encontrarse en muchos desarrollos relacionados con redes convolucionales.

Para la inicialización de los parámetros de las redes se han utilizado el **método de He** y la inicialización o método de **LeCun**. U-Net utiliza el primero, ENet el segundo y para FC-DenseNet se harán pruebas con los dos para evaluar su impacto en el resultado final, como se ha indicado en la Sección 5.2.

5.5 Métodos de evaluación

Las pseudoangiografías generadas **durante el test** por las diferentes arquitecturas (y sus variantes en el caso de FCDN) son objeto de dos tipos de evaluación: cualitativa y cuantitativa. La evaluación cualitativa se centra en analizar, visualmente, la calidad de las pseudoangiografías. La evaluación cuantitativa se centra en valorar, numéricamente, la calidad de las reconstrucciones de forma automática y en base a un criterio dado, que se verá a continuación.

Ambos tipos de evaluación se realizan sobre las pseudoangiografías obtenidas por las diferentes redes durante el test, en el que se usan como imágenes de partida todas las retinografías de DRIVE y las retinografías de STARE que disponen de segmentación de vasos manual.

5.5.1 Evaluación cualitativa

La evaluación cualitativa se centra en analizar, visualmente, la calidad de las imágenes de salida de las redes, las pseudoangiografías. Para ello es necesario fijarse tanto en la propia pseudoangiografía que se quiere analizar como en la retinografía a partir de la cual esta pseudoangiografía se ha construido. El foco de la evaluación se centra, por un lado, en valorar si hay una correspondencia adecuada entre los elementos y estructuras de las retinografías y los de las pseudoangiografías, y por otro, en valorar si la apariencia de las pseudoangiografías es correcta, es decir, si se muestran en ella los elementos presentes en las retinografías tal y como lo harían en hipotéticas angiografías pareja.

Así, para poder analizar adecuadamente estas dos cuestiones, hay que tener en cuenta: el nivel de ruido, el contraste (especialmente entre los vasos sanguíneos y el fondo), la continuidad de los vasos, la suavidad de los bordes de los vasos, la textura del fondo, el color del fondo, la correspondencia de las estructuras, la coherencia de la representación de las estructuras, y el nivel de visibilidad de los vasos sanguíneos que en la retinografía de partida se aprecian con dificultad, bien por su tamaño, bien por su bajo contraste con el fondo.

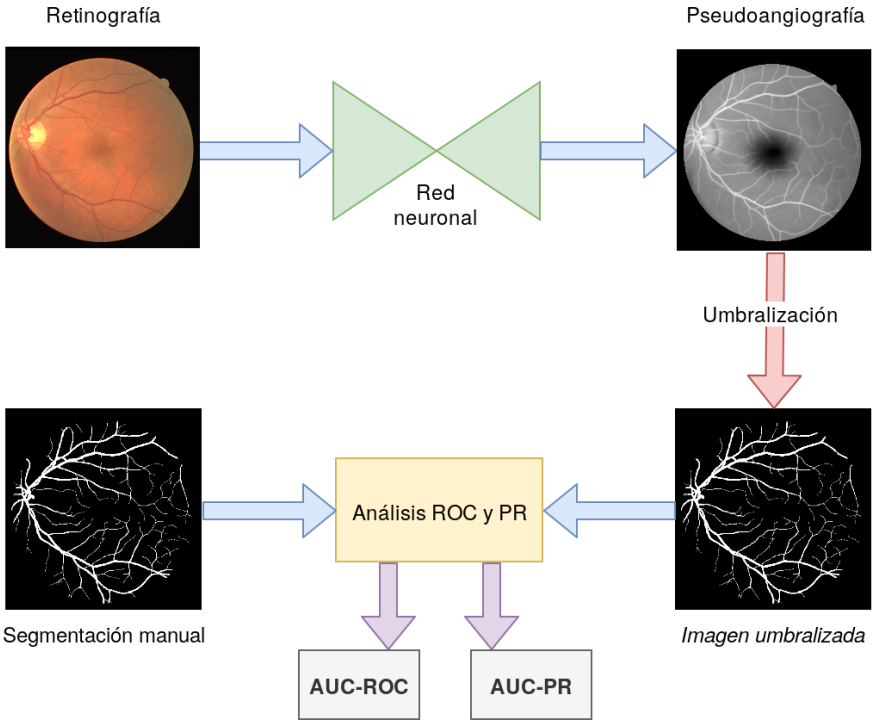


Figura 5.4: Esquema del método de evaluación cuantitativa.

5.5.2 Evaluación cuantitativa

Para obtener resultados cuantitativos del comportamiento de las diferentes arquitecturas en la tarea de reconstrucción multimodal se ha utilizado el método de evaluación mostrado en la Figura 5.4 y propuesto en [47]. A continuación se sigue el razonamiento en que se basa y se hace una descripción de cómo se utiliza exactamente para la presente tarea.

Las retinografías de los conjuntos de evaluación seleccionados cuentan con máscaras de segmentación de vasos sanguíneos creadas manualmente por expertos. Una de las formas más básicas de conseguir una máscara de este tipo de forma automática es por medio de una umbralización directa y global de una imagen de fondo de ojo. Dado que la angiografía realza los vasos, es posible aplicar una umbralización así obteniendo unos resultados aproximados. Como la retinografía no hace ese realce, el resultado de aplicar este mismo método sería mucho menos acertado, es decir, mucho menos parecido a la segmentación manual del experto. De esta forma, los valores de *Area Under Curve* (AUC) de los análisis *Precision-Recall* (PR) y *Receiver Operating Characteristic* (ROC) (se puede ver el detalle de estos análisis en el Apéndice D) serían significativamente mayores para la segmentación generada a partir de la angiografía que para la generada a partir de la retinografía. Si las pseudoangiografías son adecuadas, este aspecto debería mantenerse, y los valores de AUC-ROC y AUC-PR fruto de aplicar una umbralización a una pseudoangiografía deberían ser significativamente mayores a los obteni-

dos para la retinografía correspondiente. De esta forma, los valores de AUC-ROC y AUC-PR estarían midiendo aproximadamente el nivel de calidad de las pseudoangiografías. Es decir, actuarían como métricas aproximadas de calidad de las mismas.

En nuestro caso, nos interesa evaluar cómo de bien están haciendo las reconstrucciones las arquitecturas escogidas y modificadas. Para ello, obtendremos primero las pseudoangiografías a partir de las retinografías de los conjuntos DRIVE y STARE (las que dispongan de segmentación de vasos manual). Después, realizaremos para cada conjunto un solo análisis ROC y un solo análisis PR para todas las imágenes del conjunto. Finalmente, se calculará el valor de AUC para cada análisis. Así, se tendrán en total cuatro valores. Uno de AUC-ROC y otro de AUC-PR para STARE y uno de AUC-ROC y otro de AUC-PR para DRIVE. Estos serán los cuatro valores utilizados como métricas de la calidad (aproximada) de las reconstrucciones de cada arquitectura.

5.5.3 Tipos de evaluación y su ponderación

De los dos tipos de evaluación, en esta tarea se considera más relevante la evaluación cualitativa. A pesar de que tiene la limitación importante de ser subjetiva, en este caso resulta más fiable que la cuantitativa a la hora de valorar la calidad real de las pseudoangiografías. Esto es así por el sesgo presente en el método de evaluación cuantitativa. Si bien este método es útil para evaluar a grandes rasgos la bondad de una arquitectura en la tarea de reconstrucción, no ofrece resultados precisos, pues descuida algunos detalles relevantes de las pseudoangiografías. Al compararse las imágenes resultado únicamente con el *groundtruth* de la segmentación de vasos, en la valoración no se tienen lo suficientemente en cuenta aspectos como la suavidad de los vasos, su continuidad, la textura y el nivel de gris del fondo, o la forma en que se representan en las pseudoangiografías algunas estructuras. Si las redes consiguen construir pseudoangiografías parecidas a las máscaras de segmentación de vasos, los resultados cuantitativos son buenos. Esto sucede cuando se representan los vasos principales con un grosor correcto y un alto contraste con el fondo (por lo que se ven favorecidas las imágenes con fondos oscuros). Los vasos pequeños, al ser pocos píxeles, no se tienen mucho en cuenta. Por otra parte, la calidad de la reconstrucción de elementos importantes como la fóvea o el disco óptico tampoco se valoran, pues en las máscaras de segmentación de vasos no aparecen.

La evaluación cualitativa, a pesar de que no cuantifica estos aspectos de una forma totalmente objetiva, permite tenerlos todos en cuenta. Por medio de la observación de las pseudoangiografías, se pueden hacer valoraciones lo suficientemente acertadas acerca de la bondad de cada arquitectura, o variante, para la tarea de reconstrucción. Siempre, eso sí, que esta observación sea minuciosa y ponga el foco sobre los detalles a los que antes se ha hecho referencia.

			DRIVE-all		STARE-AH	
	Variante	Épocas \ AUC(%)	ROC	PR	ROC	PR
FCDN56	**LDN+**	275	77.98	44.20	73.49	43.87
	LDNB	170	77.91	48.50	72.84	48.33
	L+NB	104	73.86	38.71	68.65	36.67
	L++B	194	85.53	67.28	83.95	58.19
	L+++	197	87.34	71.63	85.60	66.18
	LD+B	328	86.98	71.13	83.81	62.20
	LD++	159	82.55	62.50	77.56	44.26
	L+N+	101	75.61	44.17	71.82	38.86
	H+++	414	85.52	66.92	86.27	63.31
FCDN67	**L+++**	249	86.66	66.61	86.25	60.45
	H+++	242	84.40	65.76	83.33	60.61
FCD103	**L+++**	206	84.70	65.58	83.85	60.54
	H+++	316	83.36	61.77	84.62	60.41
U-Net	-	311	87.07	69.22	88.87	68.19
E-Net	Con BN	238	79.12	54.61	76.03	46.40
	Sin BN	420	82.96	64.43	81.09	58.81

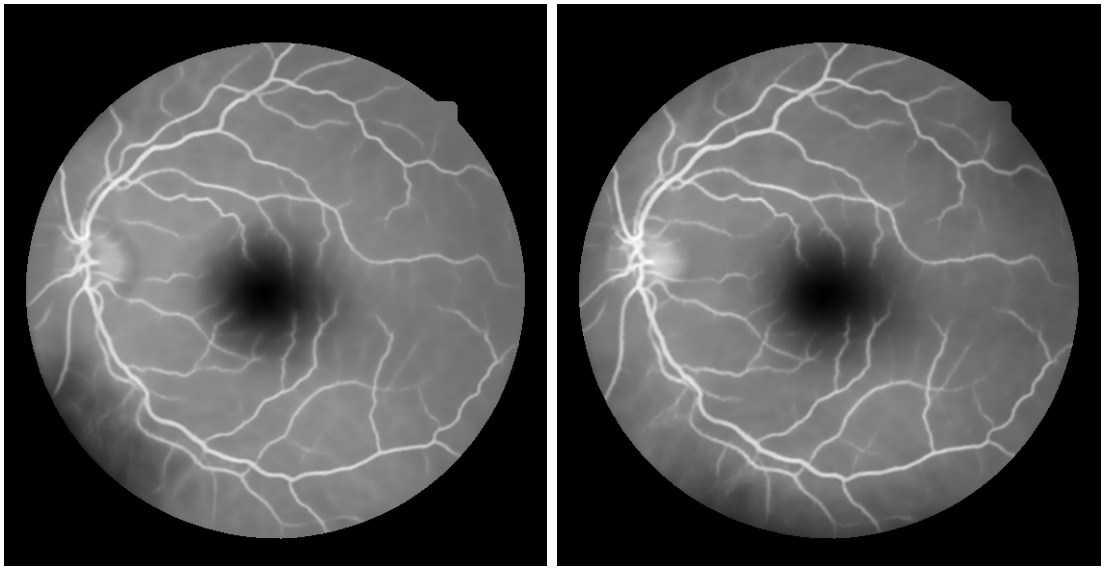
Cuadro 5.2: Resultados de la evaluación cuantitativa para la tarea de reconstrucción multimodal.

5.6 Resultados y discusión

En total, se han entrenado 16 redes distintas a lo largo de aproximadamente 4000 épocas. Entre ellas, hay 7 variantes de la arquitectura FCDN de 56 capas y 1 variante de la arquitectura ENet. Además, también hay una variante de FCDN con 67 capas y otra con 103. Hacen 16 las redes con las arquitecturas sin modificar para cada modelo (U-Net, FCDN y ENet), y las variantes de FCDN (en sus tres modelos) que han sido inicializadas con el método de He en lugar de con el de LeCun. Todas estas redes han sido evaluadas sobre los conjuntos de datos DRIVE y STARE de forma separada. Son los resultados de estas evaluaciones los que aquí se exponen y discuten en los diferentes apartados.

Resultados cuantitativos

En el Cuadro 5.2 se exponen los resultados fruto de la evaluación cuantitativa. En ellos nos apoyaremos a la hora de discutir algunas cuestiones concretas acerca de las diferentes arquitecturas o variantes a lo largo de toda esta sección.



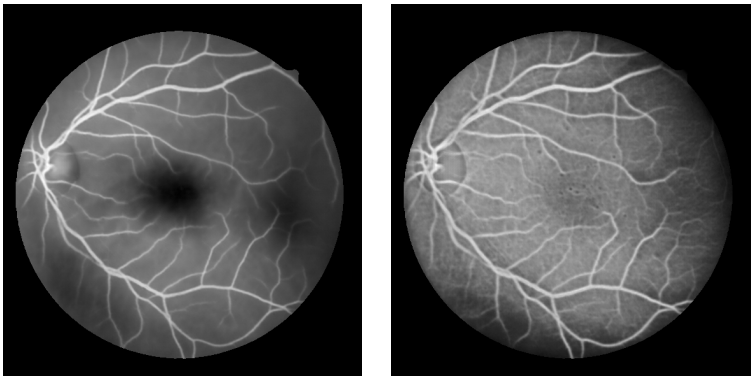
(a) (b)

Figura 5.5: Pseudoangiografías generadas por las variantes **L+++** (sin *bias* en la convolución final) (a) y **L++B** (con *bias* en la convolución final) (b).

5.6.1 FC-DenseNet56 y sus variantes

Para evaluar el impacto de la presencia o no de ***bias*** en la convolución final se han analizado los resultados de las siguientes variantes (pares iguales salvo por el *bias* de la convolución final): **LDN+** y **LDNB**; **L+++** y **L++B**; **LD+B** y **LD++**; **L+NB** y **L+N+**. En el Cuadro 5.2 pueden apreciarse tanto casos en los que la versión sin *bias* supera a la versión con él (e.g. **L+++** y **L++B**) como en los que sucede al revés (**LD+B** y **LD++**). Además, tal y como puede verse para las redes **L+++** y **L++B** en las pseudoangiografías de la Figura 5.5, si se comparan las imágenes generadas por las redes sin y con *bias* (Figuras 5.5a y 5.5b, respectivamente) no se encuentran diferencias relevantes. La réplica que se hace de U-Net en este sentido (en U-Net no hay *bias* en la convolución final), no parece afectar en absoluto (al menos con esta arquitectura y en este problema).

Para analizar el impacto de la operación de ***Batch Normalization*** se han comparado los resultados de las siguientes variantes: **L++B** y **L+NB**; **L+++** y **L+N+**; **LD+B** y **LDNB**; **LD++** y **LDN+**. Como se puede ver en el Cuadro 5.2, esta operación tiene un claro impacto negativo en los resultados cuantitativos. Cuando está presente, estos caen de forma significativa. Si se miran las pseudoangiografías de estas variantes, como la que se muestra en la Figura 5.6b para **L+N+**, se puede ver cómo en algunas de ellas (en alrededor del 40%) aparecen defectos importantes que rebajan mucho su nivel de calidad. Esta especie de ruido en el fondo se da en todos los casos analizados que incorporan BN (**L+NB**, **L+N+**, **LDNB**, y **LDN+**). Su efecto es



(a) (b)

Figura 5.6: Pseudoangiografías de **L+N+** con BN en *modo entrenamiento* (a) y en *modo test* (b) durante el test.

similar al que se produciría por una excesiva sensibilidad a pequeñas estructuras del fondo. Esto todo es lo que explica los resultados del Cuadro 5.2.

Para analizar mejor este fenómeno, y ver así si los defectos se debían al modo de funcionamiento de BN durante el test al utilizar un *mini-batch* de una única imagen (BN casi siempre se utiliza con tamaños de *mini-batch* más grandes)[2], se hizo una prueba a mayores. Esta prueba consistió en establecer que la operación de BN durante el test se comportara igual que durante el entrenamiento. Este cambio del *modo test* al *modo entrenamiento* durante el test hace que la red, en tiempo de inferencia, no utilice los parámetros de ajuste de BN aprendidos durante todo el proceso de optimización (que suelen obtenerse a partir de ponderaciones de los de cada *mini-batch*). En su lugar, utiliza los parámetros de ajuste hallados para cada imagen[3].

La Figura 5.6 muestra un ejemplo de pseudoangiografías obtenidas por la misma variante evaluada utilizando BN en *modo entrenamiento* (Fig. 5.6a) y en *modo test* (Fig. 5.6b). Como se puede observar, los resultados obtenidos a raíz de esta prueba son significativamente distintos. Las imágenes obtenidas en test utilizando BN en *modo entrenamiento* superaron con creces la calidad de las imágenes obtenidas de la otra forma.

La gráfica de la Figura 5.7 muestra los resultados cuantitativos obtenidos por las diferentes alternativas evaluadas con BN en *modo entrenamiento* y en *modo test*. Claramente, se observa que los resultados son mucho mejores cuando esta operación, durante el test, se utiliza en *modo entrenamiento*, lo que es coherente con la apariencia de las pseudoangiografías.

[2]En la implementación utilizada de BN (*PyTorch 0.4.0*), las medias y varianzas se hallan para cada canal, en lo que a veces se conoce como *Spatial Batch Normalization*. Cuando solo una imagen es utilizada a la vez, únicamente las medias y varianzas de sus canales son tenidas en cuenta a lo largo del proceso, de modo que se hacen una especie de normalizaciones individuales y por canal, de las que se lleva un registro en forma de media y varianza móviles.

[3]Cuando se cambia el modo durante el test a *modo entrenamiento*, en lugar de utilizarse las medias y varianzas para la normalización obtenidas de los datos históricos del entrenamiento, se van calculando de nuevo las estadísticas en cada caso sin tener en cuenta las históricas.

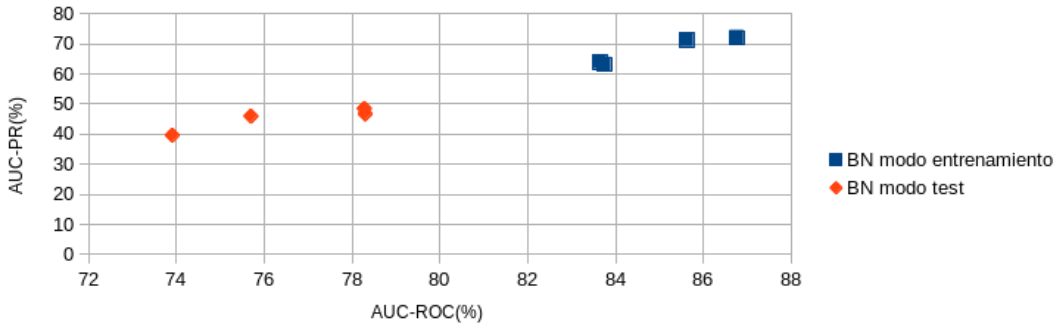
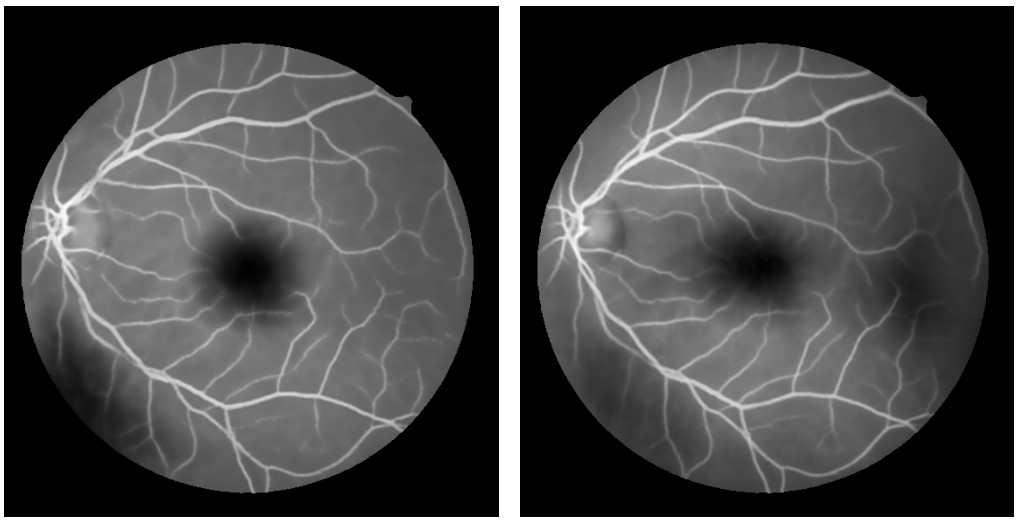


Figura 5.7: Gráfica de los valores AUC-ROC contra AUC-PR según el modo en test de BN.

Aun con esta modificación, los resultados de algunas de las redes sin BN mejoran a los de aquellas con él, como se puede ver si se comparan las pseudoangiografías de la Figura 5.8a (de **L+++**) y de la Figura 5.8b (de **L+N+**) (en la parte derecha se puede ver que **L+N+** representa algunos vasos menos). Esto, unido al comportamiento descrito, es lo que motiva finalmente la consideración de la operación de *Batch Normalization* como innecesaria para la arquitectura FCDN al menos **con tamaño de *mini-batch* igual a 1 imagen**.

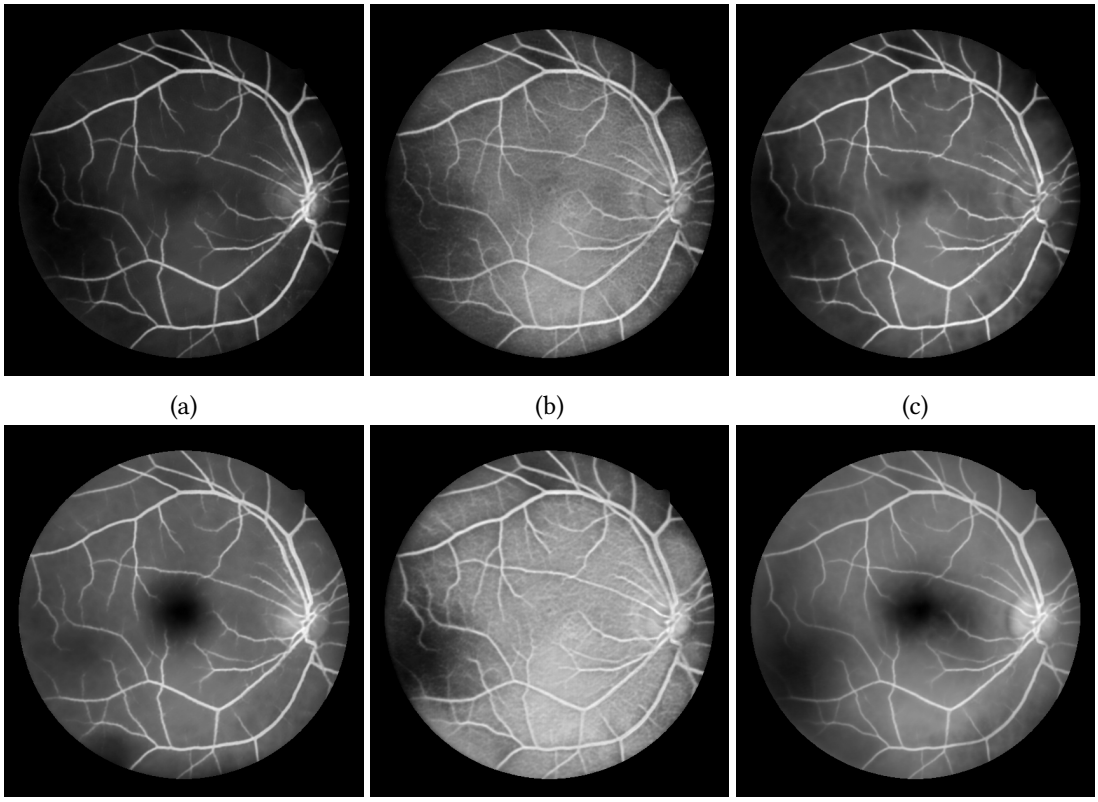


(a) (b)

Figura 5.8: Pseudoangiografías generadas por las variantes **L+++** (a) y **L+N+** (igual que **L+++** pero con BN) (b).

Para analizar el impacto de la operación de ***Dropout*** se han comparado los resultados de las siguientes variantes: **L++B** y **LD+B**; **L+++** y **LD++**; **L+NB** y **LDNB**; **L+N+** y **LDN+**.

Tal y como se puede ver en las pseudoangiografías de las Figuras 5.9a (**LD+B**) y 5.9d (**L++B**), cuando el *dropout* se incluye en las redes, las imágenes que estas obtienen, en las 4 pruebas que se han hecho sin BN (2 comparativas: **L+++** y **LD++**; y **L++B** y **LD+B**), son más



(a) (b) (c)

(d) (e) (f)

Figura 5.9: Pseudoangiografías generadas por **LD+B** (a), **L++B** (d), **LDNB** (b) y **L+NB** (e) evaluadas con BN en *modo test*, y **LDNB** (c) y **L+NB** (f) evaluadas con BN en *modo entrenamiento*.

oscuras, casi con la apariencia de una máscara de segmentación de vasos, y en absoluto con apariencia de angiografía. En bastantes de ellas ni siquiera se ven más vasos difíciles, o con un mayor realce, que en las alternativas sin *dropout*. Los resultados cuantitativos, pese a ello, como se advierte en el Cuadro 5.2, son en ocasiones mejores porque el contraste de los vasos con el fondo sí es mayor, y esto se ve favorecido por el método de evaluación cuantitativa escogido —para este método, las pseudoangiografías perfectas, las que darían lugar a mejores resultados cuantitativos, serían imágenes binarias exactamente iguales a la segmentación manual de vasos—.

Estos resultados se pueden explicar por lo siguiente. En esta tarea, y con estas redes, no se han apreciado en ningún caso situaciones de sobreajuste, con o sin *dropout*. Si se tiene en cuenta que la utilización de *dropout* está motivada por los sobreajustes que se producen a veces cuando se entrenan ANNs como las aquí descritas, esta operación no parece necesaria. La inclusión de un factor aleatorio y ruidoso en el aprendizaje por parte de este método puede ser de gran ayuda si se dan esos sobreajustes, pero cuando no es así, supone un escollo más para el entrenamiento de las redes que perfectamente puede estar limitando su potencial.

Para los casos en los que se usa BN (y se usa en *modo test* durante la evaluación), como se ve en los ejemplos de las Figuras 5.9b y 5.9e, este fenómeno diferenciador no se aprecia, y hasta parece que el *dropout* mitiga los efectos perjudiciales que produce BN, reduciendo el resalte de las manchas blancas. Sin embargo, cuando durante la evaluación se establece el modo de funcionamiento de BN a *modo entrenamiento*, el efecto es el mismo que para los casos en los que no se usa BN, como se puede ver en los ejemplos de las Figuras 5.9c y 5.9f.

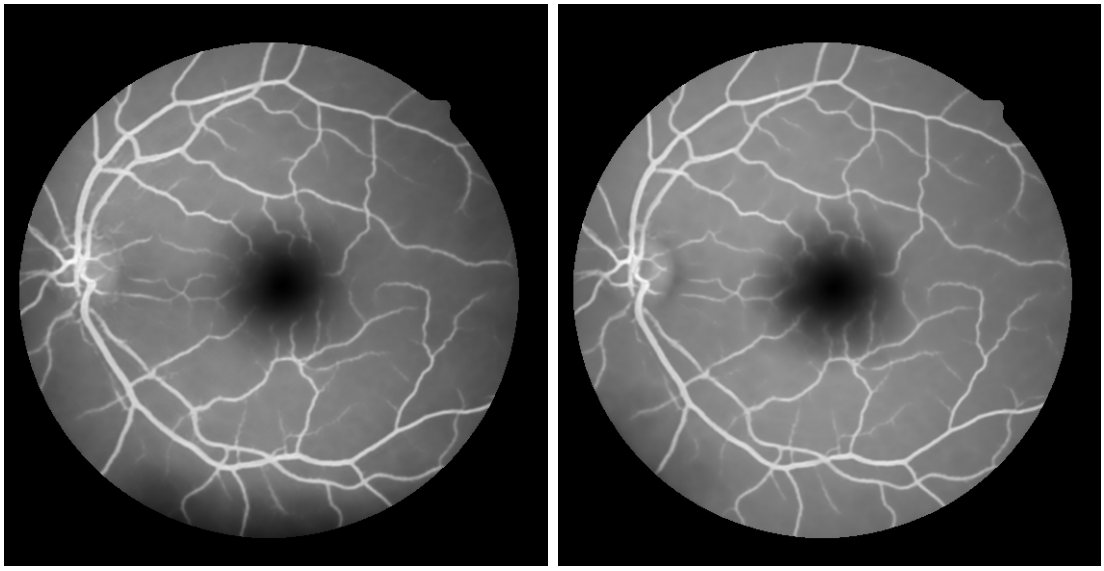
Mejor variante de FCDN56

En el apartado anterior se ha discutido, en base a los resultados cualitativos y cuantitativos en la tarea de reconstrucción, la conveniencia o no de ciertos elementos arquitectónicos. Concretamente, de *dropout*, *bias* de la convolución final, y *Batch Normalization*. Las conclusiones en relación a estos elementos fueron que *dropout* tiene un impacto negativo en los resultados, que BN no parece aportar nada, y que la utilización de *bias* en la convolución final no es en absoluto determinante. Así, a priori, las variantes que deberían ser más convenientes son **L+++** y **L++B**. Para reducir la explosión combinatoria de futuras pruebas, se ha decidido utilizar solo una de las dos para la experimentación posterior. La selección se ha realizado en base a pequeños detalles presentes en las imágenes generadas por ambas variantes.

Es importante tener aquí en cuenta que las diferencias son casuales, fruto de cierta aleatoriedad introducida en los entrenamientos a través del *augmentation*, el orden de presentación de las imágenes, y otros factores. En ningún caso se quiere afirmar que la combinación de elementos de una arquitectura es mejor que otra. Simplemente, se trata de ver, en base a los resultados, cuál de las redes, por ese factor aleatorio, es capaz de generar las mejores pseudoangiografías durante el test.

En la Figura 5.10 se muestran las pseudoangiografías generadas por las alternativas **L++B** (Fig. 5.10a) y **L+++** (Fig. 5.10b). Como se vio al principio del apartado 5.6.1, las pseudoangiografías de estas dos alternativas son muy similares. No obstante, en algunas de ellas, como en las de la mencionada figura, se pueden percibir algunas diferencias. En ciertas pseudoangiografías de la variante **L+++** aparecen, aunque pocos, más vasos resaltados que en las de la variante **L++B**. En las Figuras 5.10a (**L++B**) y 5.10b (**L+++**) se puede observar este detalle en las partes derecha e inferior. Los resultados cuantitativos (Cuadro 5.2) son coherentes con esta observación, y los correspondientes a la variante **L+++** son más altos. Por otro lado, los vasos de la alternativa **L+++** se puede ver que son, aunque muy ligeramente, más suaves que en las pseudoangiografías de la variante **L++B**. Por estos motivos, finalmente se ha optado por seleccionar a la variante **L+++** como la «mejor».

Esta combinación descrita (sin *dropout*, ni BN, ni *bias* en la convolución final) es por tanto la que se ha llevado después a los modelos más profundos de 67 y 103 capas. Modelos con los que se han evaluado las diferencias entre las incializaciones de He y LeCun en los resultados.



(a) (b)

Figura 5.10: Pseudoangiografías generadas por las variantes **L++B** (a) y **L+++** (b).

5.6.2 Pruebas con la inicialización de FCDN

Para evaluar si en esta tarea la **inicialización** de He es más conveniente que la de LeCun, o viceversa, se han entrenado redes inicializadas con uno y otro método. Después, se han analizado los resultados obtenidos por las variantes de los tipos **L+++** (inicializada con LeCun) y **H+++** (inicializada con He) para los 3 modelos de FCDN de distinta profundidad.

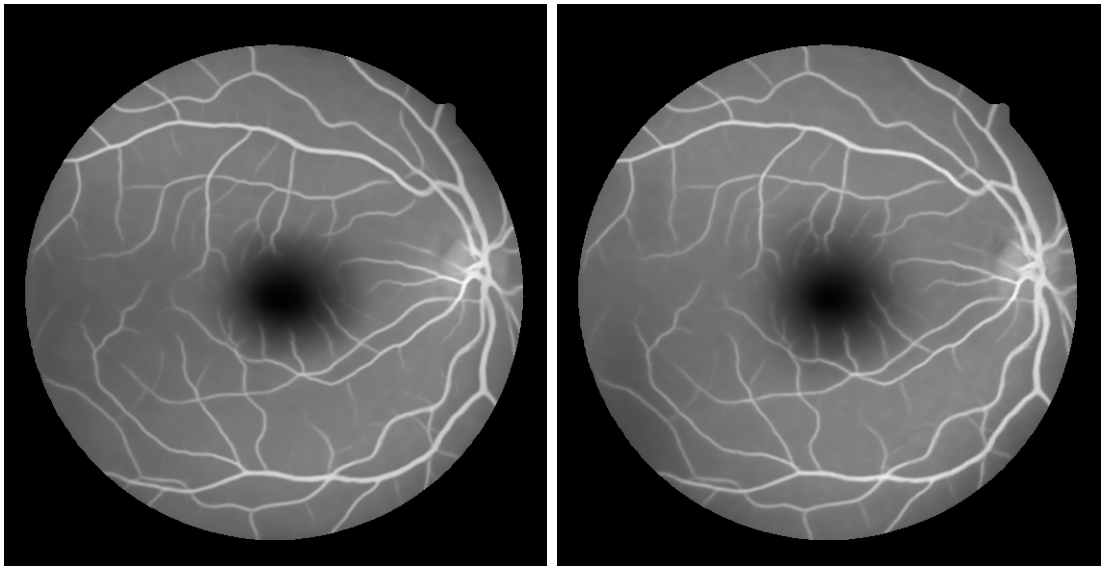
En la Figura 5.11 se muestran ejemplos de pseudoangiografías obtenidas por redes inicializadas con LeCun (**L+++.67**[4], Fig. 5.11a) y con He (**H+++.67**, Fig. 5.11b). Tal y como se puede ver, entre las pseudoangiografías generadas por las redes con una u otra inicialización no hay diferencias significativas. Esto puede deberse a dos motivos relacionados: la similitud de los métodos de inicialización y la convergencia de las redes. Al partir de unos valores iniciales parecidos, y teniendo en cuenta que ambas redes alcanzan un estado de convergencia durante el entrenamiento, es normal que estas acaben llegando al final a soluciones/configuraciones parecidas. Así, no se puede afirmar que una sea mejor que otra.

En general, la conclusión es que tanto con He como con LeCun las redes son capaces de llegar a resultados satisfactorios y estadísticamente equivalentes.

5.6.3 FCDN y sus modelos más profundos

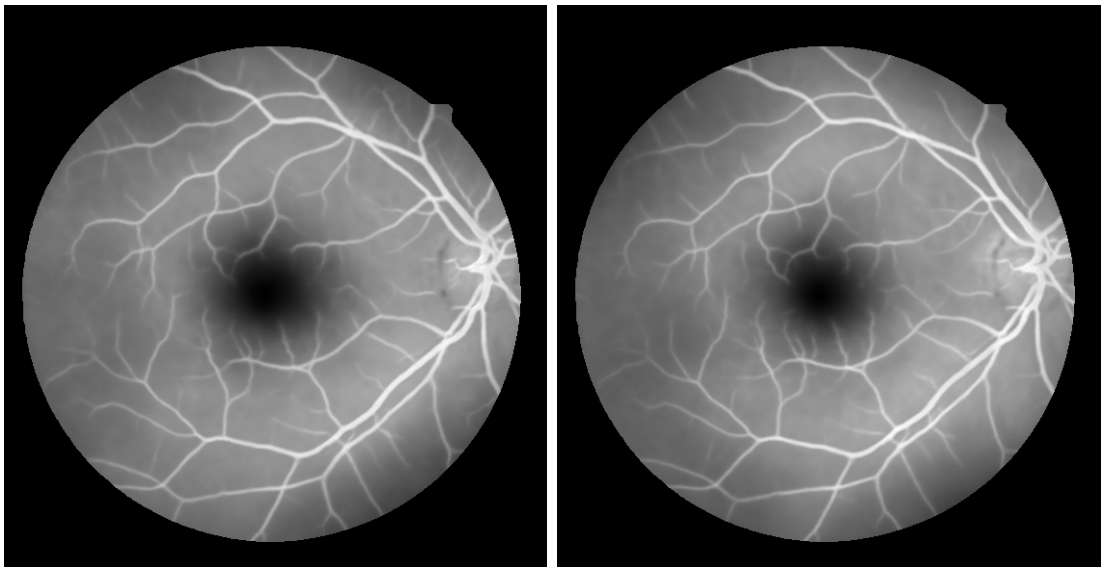
La Figura 5.12 muestra pseudoangiografías generadas por los modelos del tipo **H+++** de 56 (Fig. 5.12a) y 103 capas (Fig. 5.12b). En ella se puede comprobar fácilmente que las imágenes

[4] A partir de ahora, como se utilizarán redes de distintas profundidades, se indicará a cuál de ellas nos referimos.



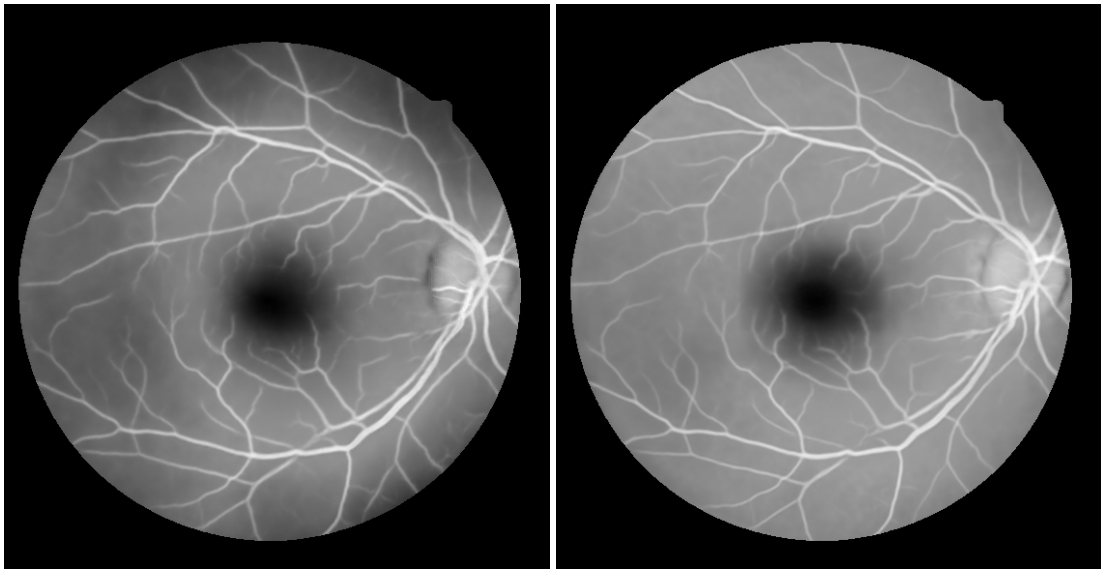
(a) (b)

Figura 5.11: Pseudoangiografías obtenidas por las variantes **L+++.67** (inicializada con LeCun) (a) y **H+++.67** (inicializada con He) (b).



(a) (b)

Figura 5.12: Pseudoangiografías obtenidas por las variantes **H+++.56** (56 capas) (a) y **H+++.103** (103 capas) (b).



(a) (b)

Figura 5.13: Pseudoangiografías de **H+++.56** (a) y U-Net (b).

generadas por los modelos de diferentes profundidades prácticamente no se diferencian entre sí. El hecho de añadir capas no parece que sea directamente beneficioso para la reconstrucción.

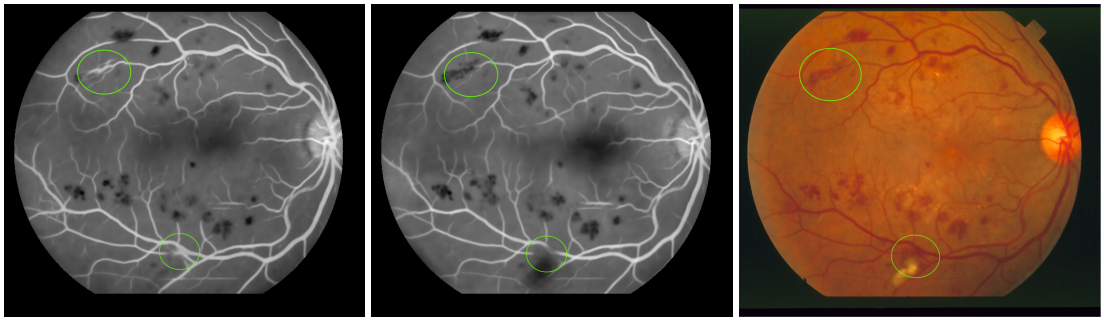
Los resultados cuantitativos parecen transmitir también esta idea. Si se observan en el Cuadro 5.2 los valores de AUC-PR y AUC-ROC para los diferentes modelos del tipo **L+++** o **H+++** en las tres profundidades, se puede ver que estos valores son relativamente similares tanto en DRIVE como en STARE.

Esta gran similitud, a nivel de resultados, de los diferentes modelos, hace que no sea posible evaluar aquí el impacto real de la profundidad en la tarea de reconstrucción para la arquitectura FCDN. Por ello, se considerán sus pseudoangiografías como equivalentes, como sucedía con las distintas inicializaciones, y a la hora de hacer comparativas con otras arquitecturas se podrá seleccionar uno u otro modelo con una u otra inicialización de forma arbitraria.

5.6.4 FCDN y U-Net: calidad de las pseudoangiografías

En la Figura 5.13 se muestran las pseudoangiografías generadas por **H+++.56** (5.13a) y U-Net (5.13b). Como puede apreciarse, ambas son muy similares, pero hay ciertas diferencias interesantes.

A nivel global, en las pseudoangiografías de **H+++.56** los vasos sanguíneos tienden a verse mejor (por ser más claros y tener el fondo un tono más oscuro) en comparación con U-Net. Además, parece que esta red, cuando sí detecta los vasos, les otorga un mayor grosor con unos bordes ligeramente menos difuminados de lo que lo hace U-Net. Por otra parte, U-Net tiene



(a) (b) (c)

Figura 5.14: Pseudoangiografías de **H+++.56** (a) y U-Net (b) a partir de la retinografía nº 139 de STARE (c). Los círculos verdes destacan la reconstrucción de algunas hemorragias.

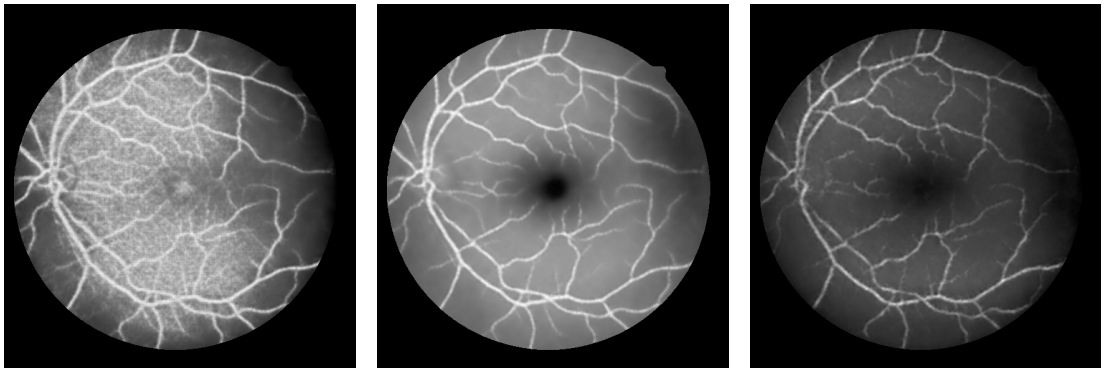
un fondo más uniforme a lo largo de toda la región de interés. Además, es capaz de mantener la continuidad en las zonas de los vasos cercanas al disco óptico con más frecuencia que **H+++.56**, y prácticamente nunca se dan casos de zonas oscurecidas, como ocurre en **H+++.56** (ver bordes de la Fig. 5.13a). A parte de esto, los cambios son mínimos, y en detalles como la representación de vasos difíciles o la suavidad de los bordes de los vasos ambas están bastante a la par.

FCDN y U-Net: capacidad de generalización

En este apartado se comparará la capacidad de generalización de FCDN y U-Net. Para ello, se analizará la respuesta de las redes a imágenes de un conjunto de datos distinto y mucho más complicado que DRIVE e Isfahan: el conjunto **STARE**.

En este conjunto de datos, como ya se advirtió, las retinografías pertenecen a pacientes con diversas patologías y lesiones, en muchos casos de bastante gravedad. Esto hace que obtener buenas reconstrucciones sea más complicado. Además, hay que tener en cuenta que en el conjunto Isfahan no hay imágenes con signos de muchas de estas lesiones y patologías, de forma que algunos de los elementos que se aprecian en las imágenes de STARE *son nuevos* para la red, lo que hace que la dificultad sea aún mayor.

Si se observan las pseudoangiografías generadas por **H+++.56** y U-Net (Figura 5.14) a partir de la retinografía de STARE con signos de retinopatía diabética que se muestra en la Figura 5.14c, se puede ver muy claramente una diferencia entre ambas. La forma de reconstruir algunas hemorragias (los círculos verdes rodean algunos ejemplos) es prácticamente opuesta. En la pseudoangiografía de U-Net, a pesar de que el color de las hemorragias es muy similar al de los vasos sanguíneos, estas no aparecen identificadas como tal. En lugar de estar representadas en un color claro, lo están en un color oscuro, más coherente y habitual en las angiografías reales. **H+++.56** hace justo lo contrario, y elementos como esos los interpreta muchas veces



(a) (b) (c)

Figura 5.15: Pseudoangiografías obtenidas por ENet con BN en *modo test* (a) y en *modo entrenamiento* (b), y sin BN (c) durante el test.

y equivocadamente como vasos sanguíneos, resaltándolos con un color claro. Cuando no es así, y realmente no les da esa consideración de vaso y los «pinta» de color oscuro, todavía no se representan de una forma tan precisa como U-Net, y sus bordes suelen estar difuminados.

Esta robustez de U-Net frente a «elementos extraños» (como hemorragias), junto a una elevada coherencia semántica, son indicativos de un mejor aprendizaje de la estructura del árbol arterio-venoso y de la forma de los vasos que lo componen, es decir, de una mayor capacidad de generalización.

Los resultados cuantitativos, que se pueden ver en el Cuadro 5.2, también reflejan este comportamiento. Así como los mejores resultados en DRIVE son los obtenidos por **L+++.56**, en STARE son los obtenidos por U-Net (aunque la diferencia no sea grande).

Es en base a todo esto que se considera aquí a U-Net como la mejor arquitectura evaluada en este sentido. Su diseño más sencillo y jerárquico, y su mayor número de parámetros, aparentemente, facilitan una percepción global de la imagen, una visión de contexto amplio que permite una coherencia mayor a alternativas como la de FCDN.

5.6.5 ENet y *Batch Normalization*

En la Figura 5.15 se muestran ejemplos de pseudoangiografías generadas por ENet con y sin BN. La Fig. 5.15a muestra una imagen de test cuando BN se utiliza en *modo test*. La Fig. 5.15b, cuando se utiliza en *modo entrenamiento*. Por último, la Fig. 5.15c muestra una pseudoangiografía generada por la misma arquitectura pero sin BN. Tal y como se puede ver, cuando no se utiliza la operación de *Batch Normalization* en *modo entrenamiento* durante la evaluación, esta tiene el mismo efecto en ENet que en FCDN. Las mismas manchas blancas, esa especie de hipersensibilidad a leves estructuras del fondo que se producían para FCDN, se producen para ENet de la misma forma. De igual modo, al cambiar el funcionamiento de

BN durante el test al mismo que durante el entrenamiento, estos problemas desaparecen y las imágenes resultado son mucho mejores.

La novedad aquí con respecto a FCDN es que la incorporación de la operación de BN sí compensa. Si se comparan las Figuras 5.15b y 5.15c, generadas por variantes con y sin BN[5], respectivamente, se puede observar sin dificultad que la correspondiente a la variante con BN (Figura 5.15b) tiene mucha más calidad. La cantidad de vasos definidos, y su visibilidad, son mayores, y la apariencia en general de la pseudoangiografía es más realista. Una de las razones de que esto sea así puede ser la dificultad de ENet, por lo general, para alcanzar un estado de convergencia (en comparación con FCDN). Como BN acelera el entrenamiento, cuando está presente, ENet es capaz de encontrar una solución más rápidamente, en menos épocas, y más adecuada. Es decir, la operación de BN está facilitando a la red alcanzar un estado de convergencia y obtener una configuración adecuada en menos tiempo.

A causa de estos resultados[6], se optó finalmente por considerar a la variante con BN como la más adecuada y será la que se use para la tarea de segmentación.

5.6.6 El papel de ENet

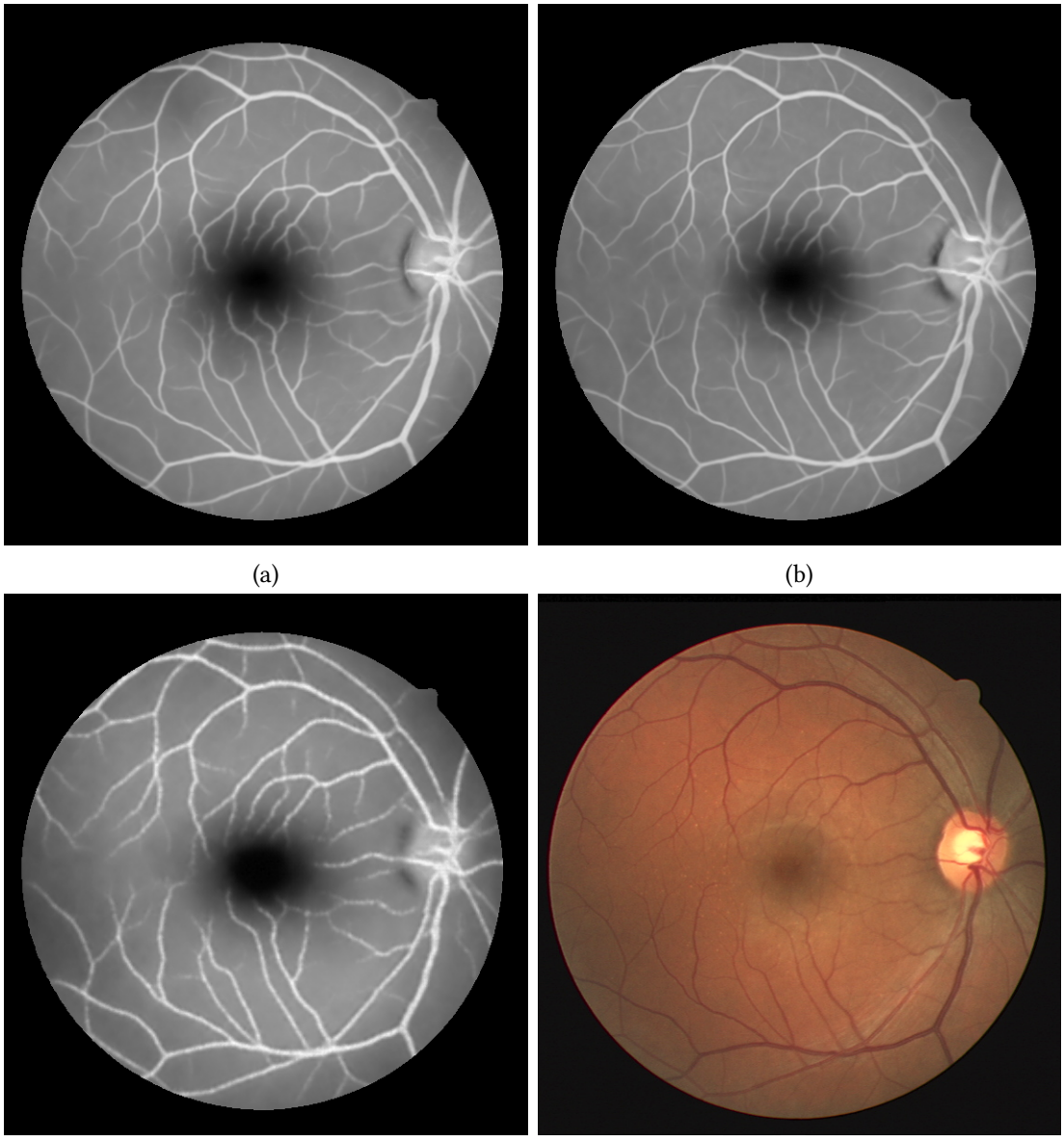
Con respecto a ENet, era interesante ver la magnitud de la diferencia entre sus reconstrucciones y las de U-Net y FCDN. Principalmente, por ser ENet una red de mucha menor capacidad, con muchos menos parámetros que las otras dos, y por ver cuánto podrían compensar esa carencia los mecanismos que incorpora en relación a las otras (como las convoluciones dilatadas o el *unpooling*).

En la Figura 5.16 se muestran la retinografía de partida y las pseudoangiografías generadas por las arquitecturas **FCDN.H+++.56**[7] (Fig. 5.16a), U-Net (Fig. 5.16b), y ENet (Fig. 5.16c) a partir de la retinografía nº 36 de DRIVE (Fig. 5.16d). Tal y como se puede ver, ENet es capaz de realizar las reconstrucciones de una forma simple más o menos satisfactoria, siendo capaz de representar, en su mejor versión, las principales estructuras de la retina. En las pseudoangiografías, los vasos principales, la fóvea y el disco óptico son fácilmente identificables, y su representación es bastante acertada. No obstante, el nivel de detalle es bastante bajo, y en comparación con las pseudoangiografías de FCDN y U-Net, las de ENet resultan pobres. En ellas, son constantes la omisión de pequeñas estructuras y la falta de suavidad, lo que provoca algunos defectos en la continuidad de los vasos, en la representación de los vasos pequeños, y en la reconstrucción de algunos elementos importantes como el disco óptico.

[5] Nótese aquí que se hace referencia a la utilización de BN en *modo entrenamiento* durante la evaluación.

[6] En un caso parecido al de *dropout* para FCDN, se han obviado aquí los resultados de la evaluación cuantitativa a la hora de tomar una decisión por favorecer a la alternativa con pseudoangiografías de mayor contraste. En este caso, la alternativa sin BN.

[7] Como se ha dicho, en FCDN no hay una «mejor» opción como tal, por lo que se ha seleccionado una entre las mejores de forma arbitraria.



(a) (b)

(c) (d)

Figura 5.16: Pseudoangiografías de **L+++.56** (a), U-Net (b) y ENet (c) a partir de la retinografía nº 36 de DRIVE (d).

Con respecto a lo que se afirma en el artículo original, que la red tiene un buen equilibrio entre eficacia y eficiencia, se puede decir que al menos en gran parte es cierto. A pesar de que los resultados no son en absoluto tan buenos como los de FCDN y U-Net, la red es mucho más pequeña y la velocidad de procesamiento de las imágenes, mucho mayor (del orden de 15-20 veces más rápido). Además, el entrenamiento es menos exigente en recursos *hardware* que para las otras dos arquitecturas, y su ejecución es posible en equipos mucho más modestos en memoria y potencia.

5.7 Conclusiones

En este grupo de experimentos centrados en la tarea de reconstrucción multimodal se ha conseguido obtener, con aprendizaje supervisado en un conjunto de datos adecuado, distintos tipos de redes de neuronas (con diferentes arquitecturas) capaces de construir pseudoangiografías razonablemente buenas a partir de simples retinografías.

Algunas de las pruebas que se han hecho a nivel de arquitectura para los modelos FC-DenseNet y ENet dejaron en evidencia la difícil tarea de valorar objetivamente el impacto de ciertos elementos arquitectónicos en el resultado final. La evaluación cualitativa de las imágenes resultado no dio lugar a argumentos concluyentes a favor de la inclusión o exclusión de ciertos elementos; y la evaluación cuantitativa, en este caso, todavía menos, pese a que sí se mostró coherente respecto al análisis de calidad de las pseudoangiografías. De todo ello se pudo desprender que en estos dominios no parece que sean verdaderamente relevantes esos elementos, y que su presencia o no en las arquitecturas no resulta un aspecto clave.

Por otro lado, las distintas comparaciones entre las arquitecturas —U-Net, FC-DenseNet y ENet— han permitido ver claramente matices diferenciadores en las imágenes resultado. Para U-Net y FC-DenseNet los resultados en DRIVE han sido muy similares, salvo pequeños detalles, y no ha sido hasta la prueba en STARE, un conjunto de datos mucho más complicado, cuando se advirtieron una mayor capacidad de generalización, robustez y coherencia semántica por parte de U-Net. Por otra parte, las pruebas con ENet han permitido confirmar que con arquitecturas livianas también se pueden obtener resultados decentes (aunque a gran distancia de los de las otras dos redes) en los que se vean reflejados los principales elementos estructurales del fondo de ojo, como la fóvea, el disco óptico y los vasos de mayor tamaño o más visibles; lo que refleja un cierto nivel de comprensión de la estructura retiniana por parte de la red.

Todas estas pruebas han sido, además, la fuente proveedora de los modelos entrenados que se utilizan, por medio de la aplicación de la técnica de aprendizaje transferido, en la tarea de segmentación de vasos, abordada en el siguiente capítulo.

Capítulo 6

Segmentación de vasos en retinografía

En este capítulo se hace una descripción en detalle de todo lo relacionado con el segundo de los experimentos abordados: **la segmentación de vasos en retinografía**. En este orden, se exponen: los detalles de experimentación, la metodología de entrenamiento, los resultados de las evaluaciones y su discusión (entrelazada con los anteriores). Las conclusiones que de los resultados se han podido extraer se integran en el Capítulo 7: Conclusiones.

6.1 Descripción del experimento

Tal y como se puede ver de forma esquemática en la Figura 6.1, el experimento, desde una perspectiva global, consiste en obtener distintas redes de neuronas capaces de construir, a partir de retinografías, imágenes en escala de grises (idealmente, binarias) que nos revelen las partes de la imagen original que se corresponden con vasos sanguíneos. Dicho de otro modo, redes que nos den un mapa o máscara de segmentación de los vasos sanguíneos que aparecen en la imagen del fondo de ojo. Para hacer esto posible, todas las redes de neuronas son entrenadas de forma supervisada en un conjunto de datos adecuado.

En concreto, lo que se hace es entrenar los diferentes modelos en la tarea de segmentación para un conjunto de datos dado, y después, evaluar (tiempo de inferencia, test) estos modelos en otros dos conjuntos diferentes al primero. Para ello, se utilizan tanto redes entrenadas a partir de un modelo preentrenado en reconstrucción como redes entrenadas a partir de los valores otorgados por un método de inicialización automático. Las segmentaciones resultado

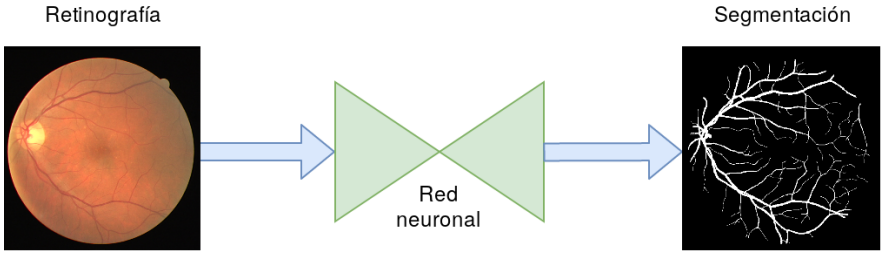


Figura 6.1: Esquema de la tarea de segmentación.

generadas por las redes en los dos conjuntos de test son sobre las que finalmente se aplican dos tipos de evaluación: cualitativa, que mide visual y subjetivamente la calidad de tales imágenes, y cuantitativa, que mide esta calidad de forma objetiva y conforme a un criterio dado. La primera de las evaluaciones da lugar a observaciones relevantes, y la segunda, a números objetivos.

Arquitecturas de red

De ENet, se ha escogido la versión con BN, por ser la de mejores resultados en reconstrucción. Por coherencia con esa tarea se ha utilizado siempre BN en *modo entrenamiento* durante los tests se segmentación.

Por otra parte, de FCDN se han seleccionado, también por ser las de mejores resultados, las variantes sin *Batch Normalization*, sin *dropout* y sin *bias* en la convolución final. Además, por considerar los resultados con ambas inicializaciones (He y LeCun) equivalentes, y para evitar la explosión combinatoria de pruebas, se ha seleccionado, de forma arbitraria, solo una de ellas: el método de He. Por último, los 3 modelos para las 3 profundidades se han mantenido para poder comprobar si en esta tarea sucede lo mismo que en la anterior en relación a la profundidad. Es decir, si para la tarea de segmentación los modelos de más capas (67 y 103) no aportan nada con respecto al de menos (56).

Sumando U-Net, esto da lugar a 5 modelos: **FCDN.H+++.[56, 67, 103]**, **U-Net** y **ENet con BN**. Por ser la profundidad el único elemento de cambio en los modelos de FCDN normalmente nos referiremos a ellos como FCDN56, FCDN67 y FCDN103, sobrentendiendo que todos ellos utilizan inicialización de He y no tienen ni *dropout*, ni BN ni *bias* en la convolución final.

Detalles del experimento

Como novedad respecto al experimento anterior descrito en el capítulo 5, en este se utilizan **modelos preentrenados**. Concretamente, se emplean aquellos obtenidos para la tarea de reconstrucción multimodal (los escogidos en el apartado anterior). Esto se hace con el objetivo de evaluar cuánto de beneficioso es aplicar esta técnica en comparación con el entrenamiento *desde cero* para las diferentes arquitecturas. Para hacer esta comparación, se hacen para cada arquitectura dos entrenamientos distintos. En uno de ellos, los valores iniciales de los parámetros de la red son iguales a aquellos del modelo preentrenado. En el otro, estos valores se corresponden con los obtenidos por medio de una función de inicialización.

Interés del aprendizaje transferido en esta tarea

Cuando se habló en su momento del estado del arte ya se nombraron algunos de los trabajos que emplean con éxito la técnica de aprendizaje transferido para obtener sus modelos

finales. Se vio cómo algunos de ellos utilizan redes preentrenadas en ImageNet, o en general en algún conjunto compuesto por *imágenes naturales* o *generales*, y otros en conjuntos con imágenes cercanas o pertenecientes al dominio de aplicación concreto. La segunda alternativa suele ser la más ventajosa.

En nuestro caso, se estima que el trabajo hecho de aprendizaje de la estructura vascular durante la tarea de reconstrucción puede ser beneficioso para la tarea de segmentación de vasos. Esto es así porque en la tarea de reconstrucción gran parte del objetivo es resaltar el árbol arterio-venoso. Este beneficio, provocado por la reutilización del conocimiento, puede llevar a las redes aquí utilizadas a un entrenamiento más rápido y estable y a unos mejores resultados, tanto a nivel cualitativo como cuantitativo. La asignación de un mayor nivel de claridad a las venas y arterias con respecto a un fondo más oscuro en las pseudoangiografías está, al fin y al cabo, bastante relacionada con la construcción de una imagen en apariencia binaria[1] en la que todo, salvo el árbol arterio-venoso (de color blanco), sea negro.

6.1.1 Conjuntos de datos

En este experimento, para los **conjuntos de datos de entrenamiento y validación**, se hace uso de la parte de entrenamiento de DRIVE, DRIVE-train, ya descrito en el apartado 5.3.2 del Capítulo 5.

Por otra parte, para la **evaluación** (test) del comportamiento de las redes en la tarea de segmentación se utiliza el subconjunto de DRIVE DRIVE-test y el subconjunto de STARE formado por las imágenes que disponen de segmentación manual. Ambos también descritos en el apartado 5.3.2.

Particiones de los conjuntos de datos

Para cada modelo se hacen ahora pruebas con diferentes tamaños del conjunto de entrenamiento. Las imágenes utilizadas son las correspondientes a DRIVE-train, que hacen un total de 20. Los casos del experimento, cuatro, se corresponden al número de imágenes a utilizar: 1, 5, 10 y 15. En todos ellos y para todas las redes se usan los mismos conjuntos. Cada conjunto contiene todas las imágenes de los menores a él (sin repetición). Es decir, el conjunto formado por 1 imagen es subconjunto del de 5, este del de 10, y el de 10 del de 15. El número restante de las 20 imágenes (19, 15, 10 y 5, respectivamente) son asignadas al conjunto de validación.

La división de las 20 imágenes de DIRVE-train en 15 y 5 se hizo de forma aleatoria, igual que la elección del orden dentro de cada uno.

[1] En realidad habrá más valores que 2 en las imágenes de salida de las redes, pero esa es la idea básica.

6.2 Detalles de entrenamiento de las redes

La metodología de entrenamiento escogida es común a todas las redes que se han seleccionado. Esto se ha hecho así para poder comparar adecuadamente el comportamiento de todas ellas entre sí en la tarea de segmentación.

Para la **inicialización** de los parámetros de las redes, como se ha apuntado, tenemos dos variantes: una de aprendizaje transferido, que coge los valores iniciales de los parámetros del correspondiente modelo preentrenado, y otra de *inicialización automática*, que parte de los valores obtenidos por el método de He (U-Net y FCDN) o el de LeCun (ENet).

Para calcular el ***error de segmentación*** en cada caso, dado que se dispone de la imagen resultado y la máscara de segmentación manual de vasos (donde hay dos clases: vaso y fondo), se utiliza *Binary Cross-Entropy* (ver Apéndice A, apartado 11.1, para más detalles), una función ampliamente utilizada en este tipo de problemas.

Para el entrenamiento de las redes se ha utilizado el **optimizador estocástico Adam**. El tamaño del *mini-batch* es siempre de una sola imagen. El entrenamiento se detiene cuando se agota una paciencia de 130 *ciclos* sin mejorar el error en el conjunto de de validación. En cada *ciclo* las imágenes son presentadas a la red n veces, pudiendo haber repeticiones. Esto es útil para utilizar como punto de comparación el número de imágenes presentadas o *vistas* (con repetición), en lugar de las épocas. Así, cuando se hacen pruebas con conjuntos de datos de entrenamiento de diferente tamaño se puede tomar como referencia ese valor. En nuestro caso, n es siempre igual a 30, por lo que 130×30 será la paciencia de entrenamiento en número de imágenes presentadas a la red (o *vistas*).

Como en el Capítulo 5, para paliar lo máximo posible la escasez de imágenes disponibles para el entrenamiento, se utiliza un ***data augmentation*** consistente en (1) transformaciones afines aleatorias de las imágenes originales, (2) ligeras variaciones de color e intensidad y (3) inversiones aleatorias en los ejes x e y.

6.3 Métodos de evaluación

Las segmentaciones generadas **durante el test** por los diferentes modelos seleccionados son objeto de dos tipos de evaluación: cualitativa y cuantitativa. La evaluación cualitativa se centra en analizar, visualmente, la calidad de las segmentaciones. La evaluación cuantitativa, en valorar, numéricamente y de forma automática, esta calidad.

Ambos tipos de evaluación se realizan sobre las pseudoangiografías obtenidas por las diferentes redes durante el test, en el que se usan como imágenes de partida todas las retinografías de DRIVE-test y las retinografías de STARE que disponen de segmentación de vasos manual.

6.3.1 Evaluación cualitativa

La evaluación cualitativa se centra en analizar, visualmente, la calidad de las imágenes de salida de las redes, las segmentaciones de vasos. Para ello es necesario fijarse tanto en la propia segmentación que se quiere analizar como en la retinografía a partir de la cual esta segmentación se ha obtenido. Además, en algunos casos, también es útil observar las segmentaciones manuales.

El foco de la evaluación se sitúa sobre la continuidad y la suavidad de los vasos segmentados, en el número de vasos segmentados (especialmente vasos *difíciles*), y en la coherencia que guardan las segmentaciones con respecto a las retinografías de partida y las segmentaciones manuales (e.g. si realmente existe un vaso que se ha segmentado).

6.3.2 Evaluación cuantitativa

La evaluación cuantitativa, para esta tarea, se hace exactamente igual a como se describe en el apartado 5.5.2 para la tarea anterior: mediante los análisis ROC y PR. En este caso, este tipo de evaluación es mucho más adecuado. Ahora, la salida de las redes son segmentaciones, por lo que al hacer los análisis ROC y PR, las imágenes sobre las que se aplican los umbrales son exactamente de la misma naturaleza que aquellas con las que se comparan —las segmentaciones creadas por los expertos (las segmentaciones de AH en el caso de STARE)—. Es decir, imágenes binarias[2] en las que los vasos se representan con el color blanco y lo demás con el color negro. Así, los valores de AUC que se obtienen a partir de ellos nos sirven como métricas objetivas y estándar[3] de la calidad de las segmentaciones. Con ellas, se pueden hacer comparaciones acertadas y objetivas entre los resultados de unos y otros modelos. Es por este motivo por lo que la evaluación cuantitativa es ahora mucho más relevante.

A pesar de todo, el método tiene ciertas limitaciones. En los análisis ROC y PR, todos los puntos ponderan igual. Es decir, un píxel bien segmentado vale lo mismo independientemente de dónde se sitúe (e.g. un píxel de una zona interior de un gran vaso y uno del borde de un vaso pequeño). Esto hace que los valores fruto de esta evaluación no contengan información relevante acerca de algunos detalles, como la continuidad y suavidad de los vasos, o la presencia de vasos difíciles. Estos aspectos no se cuantifican, y si se quieren tener en cuenta todavía es necesario recurrir a la observación de las imágenes resultado, es decir, a la evaluación cualitativa de las segmentaciones.

[2] En el caso de la red esto es muy parecido, pero no exacto, pues la salida de la red no está compuesta por solo dos valores límite (por ejemplo 0 y 1), sino por valores próximos a ellos.

[3] Es muy habitual encontrar estas métricas en trabajos relacionados con tareas de segmentación, por lo que se pueden utilizar los valores obtenidos no solo para comparaciones dentro del trabajo, sino para comparaciones con otros métodos ajenos.

6.4 Resultados y discusión

En total, se han entrenado 40 redes distintas, con una cantidad global de imágenes presentadas (suma del número de imágenes *vistas* por cada red en entrenamiento[4]) ligeramente superior a 600000. Para cada modelo de los seleccionados (5 en total), se han obtenido 8 redes distintas: redes con y sin preentrenamiento para 4 conjuntos de entrenamiento de diferente tamaño (1, 5, 10 y 15). Todas estas redes han sido evaluadas, de forma separada, sobre los conjuntos de datos STARE y DRIVE-test. Son los resultados de estas evaluaciones los que aquí se exponen y discuten a lo largo de los diferentes apartados.

Tablas de resultados

En el Apéndice E se muestran los Cuadros E.1 y E.2, donde se exponen los resultados obtenidos de la evaluación cuantitativa de las redes en los conjuntos de datos DRIVE-test y STARE, respectivamente. Además, se añade el Cuadro E.3, que contiene el número de imágenes presentadas a la red (o *vistas* por la red) hasta la obtención de la mejor configuración de pesos durante el entrenamiento.

Como no es fácil la percepción de ciertos patrones en los resultados tan solo viendo estos cuadros, se ha decidido expresar los valores que contienen en forma de gráficas. Así, los resultados de la evaluación cuantitativa de las segmentaciones estarán siempre representados gráficamente, y en esas representaciones será en las que se apoye la discusión.

Cabe aquí mencionar que, en general, se utilizarán en las gráficas los valores de AUC-PR, por ser más adecuados cuando hay un desbalanceo entre las clases a favor de la *clase negativa* (clase *no-vaso* o *fondo* en nuestro caso). No obstante, las representaciones con AUC-ROC son muy similares, y las conclusiones extraídas a través de los valores de AUC-PR también se podrían haber extraído a través de los de AUC-ROC.

FCDN y sus modelos

En la Figura 6.2 se muestran el número de imágenes *vistas* (presentadas a la red durante el entrenamiento) contra los valores de AUC-PR (fruto de la evaluación cuantitativa) para los 3 modelos de FCDN de diferente profundidad. La inicialización de estos modelos se ha realizado a través de la técnica de aprendizaje transferido o mediante el método de He (en la gráfica, los modelos cuyo nombre termina por «_nop»). Además, el tamaño de los puntos indica el tamaño (número de imágenes) del conjunto de entrenamiento —1, 5, 10 o 15—.

Estos resultados muestran cómo en los casos sin preentrenamientos las segmentaciones son mejores cuanto mayor es la profundidad del modelo (que en este caso conlleva también

[4]Se entiende que son las mismas imágenes, repetidas, del conjunto de entrenamiento.

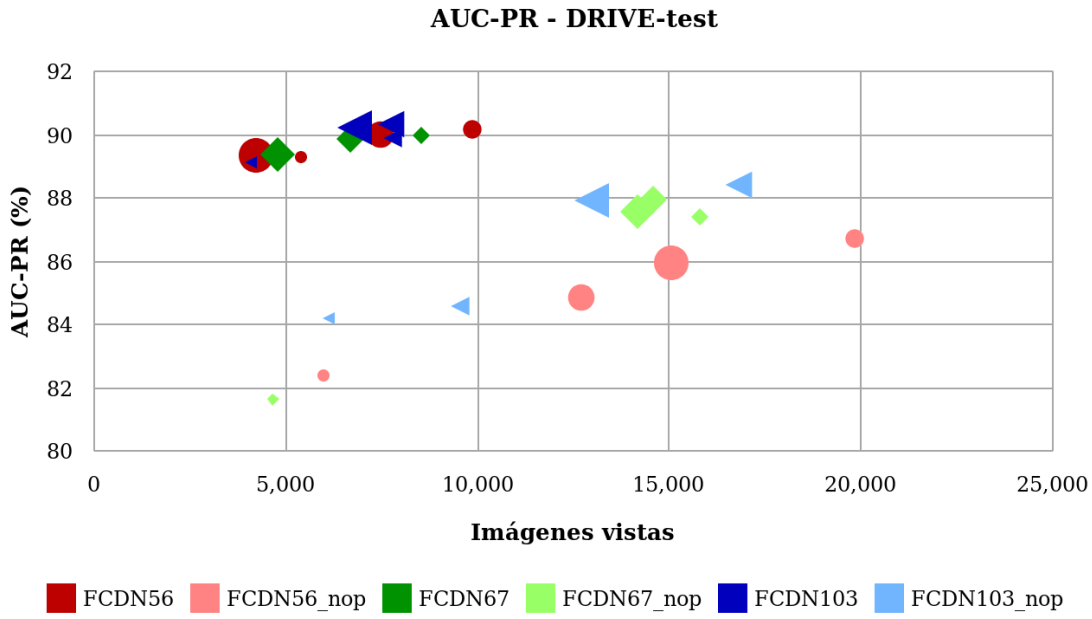


Figura 6.2: Número de imágenes vistas contra AUC-PR(%) en DRIVE-test para los tres modelos de la arquitectura FCDN. El tamaño de los puntos indica el volumen del conjunto de entrenamiento (1, 5, 10 o 15). La parte _nop de los nombres indica que la red no dispuso de preentrenamiento.

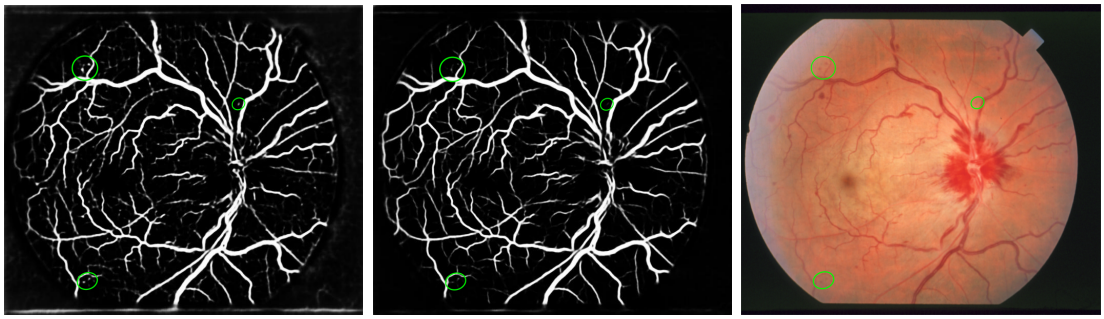
más parámetros).

Resulta lógico que esto suceda así, y que sean los modelos más *grandes* los que obtengan mejores resultados, pues su capacidad, al menos teóricamente, también es mayor.

En los modelos en los que se aplicó la técnica de aprendizaje transferido estas diferencias se reducen, como se ve en la gráfica, y los resultados son más estables y más robustos a frente a esos cambios arquitecturales. Es decir, con presencia de preentrenamiento, el impacto de la profundidad y el número de parámetros del modelo se mitiga. No obstante, no se elimina, y los resultados del modelo de 103 capas destacan ligeramente sobre los resultados de los modelos de 56 y 67 capas, que están bastante a la par.

La Figura 6.3 muestra un ejemplo de segmentaciones para los modelos de FCDN de 56 y 103 capas (con preentrenamiento y 15 imágenes en el conjunto de entrenamiento) a partir de la retinografía nº 5 de STARE. En ella puede verse la mejoría anticipada por los resultados cuantitativos para la red más profunda. La segmentación correspondiente a la arquitectura de 103 capas (Fig. 6.3b) preserva mejor la continuidad de los vasos. Además, la suavidad y definición de los mismos, y la presencia de vasos difíciles, también son mayores.

Por otro lado, el modelo más profundo también demuestra una mayor coherencia semántica. Como se observa en la Figura 6.3, el modelo de 103 capas tiende a confundirse menos con



(a) (b) (c)

Figura 6.3: Segmentaciones de FCDN56 (a) y FCDN103 (b) con preentrenamiento y 15 imágenes en el conjunto de entrenamiento a partir de la retinografía nº 5 de STARE (c).

los *elementos extraños* de las retinografías de STARE. Es decir, menos veces y de forma mucho menos notable les da la consideración de vasos a ciertas estructuras de carácter patológico, como las hemorragias (de las que se señala algún ejemplo con círculos verdes). Así, se alcanza una mayor calidad de segmentación.

Aprendizaje transferido

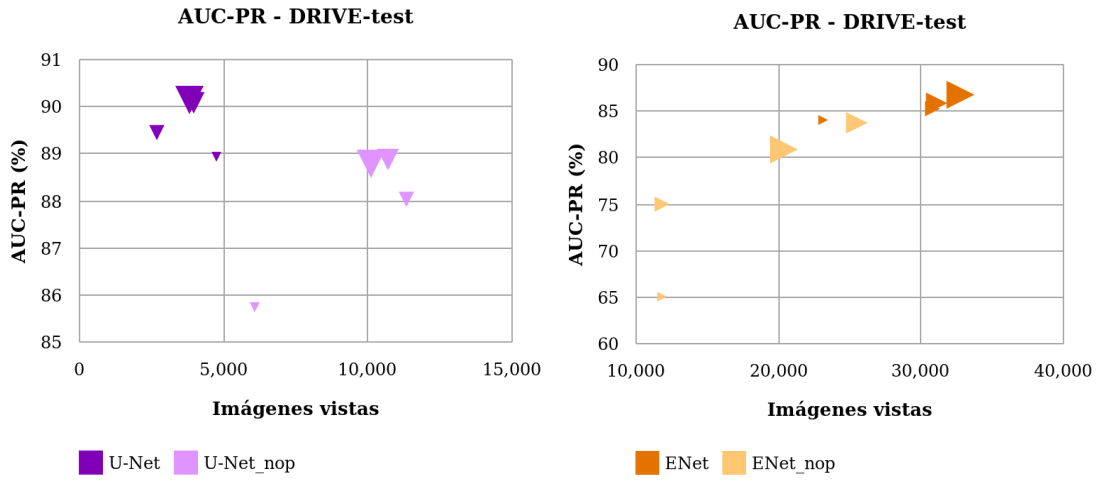
Los resultados con respecto al aprendizaje transferido son los más destacadso de la parte de segmentación, y unos de los más importantes del trabajo.

En las Figuras 6.2, 6.4a y 6.4b se muestran los valores de AUC-PR contra el número de imágenes *vistas* para los 3 modelos de FCDN, UNet, y ENet, respectivamente, tanto con preentrenamiento como con inicialización automática (redes cuyo nombre termina por «_nop»). De nuevo, el tamaño de los puntos indica el tamaño del conjunto de entrenamiento —1, 5, 10 o 15—.

En todas estas gráficas se pueden ver dos cuestiones relevantes. Primero, que independientemente de la arquitectura empleada, los valores de AUC-PR de las redes que utilizan preentrenamientos en la tarea de reconstrucción son mucho más altos que para las inicializadas de forma automática. Segundo, que en la gran mayoría de casos en los que se dispone de preentrenamiento, el entrenamiento se completa antes. O dicho de otro modo, que las redes inicializadas mediante métodos automáticos necesitan *ver* muchas más imágenes para converger a una buena solución.

El único elemento discordante en este último punto es ENet. Para esta arquitectura, el número de imágenes vistas cuando se parte de un modelo preentrenado es mayor (ver Fig. 6.4b).

En la Figura 6.5 se pueden ver segmentaciones en validación de la arquitectura ENet con y sin preentrenamiento para 21000 imágenes vistas. En ellas se puede apreciar que en los casos



(a) (b)

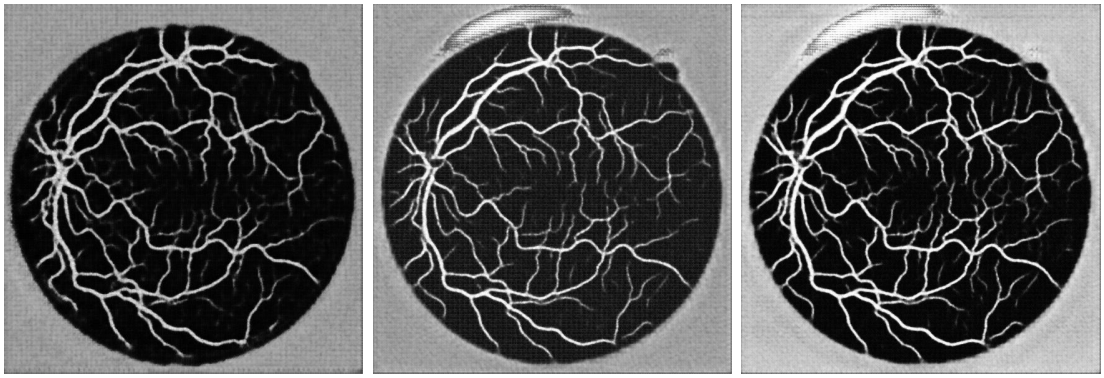
Figura 6.4: Número de imágenes vistas contra AUC-PR(%) en DRIVE-test para las arquitecturas U-Net (a) y ENet (b). El tamaño de los puntos indica el volumen del conjunto de entrenamiento (1, 5, 10 o 15). La parte _nop de los nombres indica que la red no dispuso de un modelo preentrenado.

con preentrenamiento las redes alcanzan soluciones bastante buenas[5] relativamente pronto. No obstante, siguen optimizándose sus parámetros, y las segmentaciones son cada vez de mayor calidad, con menor *ruido* y una mayor definición de los vasos más pequeños (ver Fig. 6.5c para 34500 imágenes *vistas*). Para los casos sin preentrenamiento, esto no sucede, y los entrenamientos paran antes porque no es posible encontrar nuevos caminos para hacer mejores las segmentaciones. Cuando la inicialización es automática, y el tamaño del conjunto de entrenamiento es pequeño (1 y 5 imágenes), esta parada se produce todavía más temprano por la incapacidad de generalizar. En estos casos, las imágenes resultado tienen siempre mucho ruido (ver ejemplo en Figura 6.7a), y la paciencia de entrenamiento se agota por el estancamiento del error de validación. A pesar de esto, esta paciencia no se ha aumentado, pues es muy posible que si se hiciese las redes llegaran a sobreajustar.

Para analizar si estos resultados cuantitativos son coherentes con la calidad de las segmentaciones, estas se han evaluado cualitativamente.

En la Figura 6.6 puede verse un ejemplo de segmentaciones de FCDN103 con y sin preentrenamiento para un conjunto de entrenamiento de 15 imágenes. La retinografía de partida de las mismas es la que se ha mostrado anteriormente en la Figura 6.3c: nº 5, STARE. Como se puede observar, cuando se utilizan preentrenamientos las segmentaciones son mucho más acertadas. Por ejemplo, en la Figura 6.6, la red preentrenada prácticamente no confunde la

[5]Con «soluciones buenas» nos referimos aquí a una configuración de la red (conjunto de valores de los parámetros optimizados) que da lugar a buenas segmentaciones.

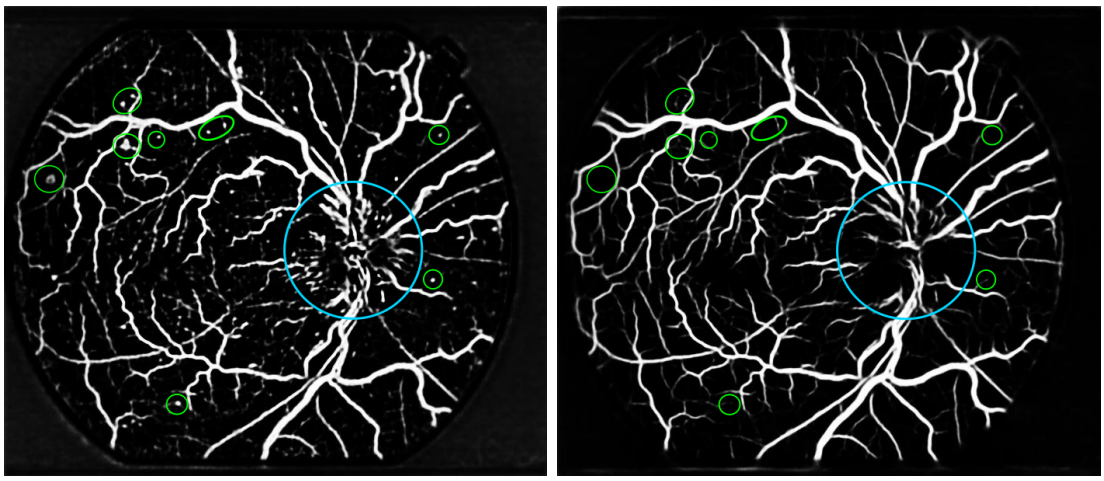


(a) (b) (c)

Figura 6.5: Segmentaciones en validación de la arquitectura ENet, con 21000 imágenes vistas sin preentrenamiento (a) y con él (b), y con 34500 imágenes *vistas* con preentrenamiento (c). Todas ellas generadas a partir de la retinografía nº 39 de DRIVE-train.

lesión del disco óptico (círculo azul) y las hemorragias (círculos verdes) con vasos sanguíneos, mientras que la no preentrenada, sí. Además, los vasos de las imágenes generadas por los modelos con preentrenamiento tienen una suavidad, continuidad y definición, en general, significativamente mayores.

Estas diferencias son comunes a todos los casos, y siempre las redes en las que se aplicó la técnica aprendizaje transferido dan lugar a mejores segmentaciones que sus equivalentes inicializadas de forma automática.



(a) (b)

Figura 6.6: Segmentaciones de FCDN103 con 15 imágenes en el conjunto de entrenamiento sin preentrenamiento (a) y con él (b) a partir de la retinografía nº 5 de STARE (Fig. 6.3c).

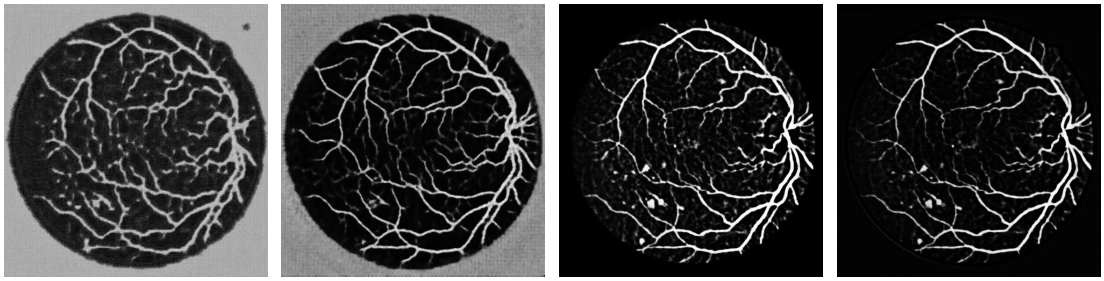
Número de imágenes vistas durante el entrenamiento

Otra cuestión interesante que dejan ver las gráficas anteriores (principalmente las de FCDN [Fig. 6.2] y ENet [Fig. 6.4b]) es la aparente conveniencia de alargar los entrenamientos. En prácticamente todos los casos, los puntos de las alternativas parecen ajustarse a una especie de recta o función logaritmo que asocia a un mayor número de imágenes vistas mayores valores de AUC. El problema aquí es que esta relación puede ser realmente una quimera, y es posible que si se dejara entrenar las redes más allá de lo que lo han hecho, se dieran situaciones de sobreajuste. Especialmente para los casos con pocas imágenes en el conjunto de entrenamiento. Aún así, no se sabe con total certeza, y para averiguarlo habría que hacer pruebas específicas, manipulando los valores de paciencia o definiendo un número fijo de imágenes vistas. De esta forma, podría verse claramente qué sucede en las imágenes resultado y en las curvas de aprendizaje. No se han hecho este tipo de pruebas en este trabajo por quedar fuera de los objetivos principales.

Número de imágenes en el conjunto de entrenamiento

En las gráficas anteriores puede verse que en algunos casos las redes entrenadas con 5 y 10 imágenes superan a las redes entrenadas con 15. Por otro lado, puede verse también que aquellas entrenadas con una sola imagen son frecuentemente las que peores resultados obtienen (sobre todo en ENet). En el primer caso, parece que las operaciones de *data augmentation* son capaces de compensar ampliamente la pérdida de variabilidad del conjunto de entrada. En el segundo, aunque sin duda está ayudando, las transformaciones de *augmentation* no parecen suficientes para que estas redes puedan competir tan claramente con el resto. Esto es debido, principalmente, a que la materia prima de esas transformaciones es muy limitada —una sola imagen—. Para algunos de estos casos incluso sucede que los entrenamientos son detenidos en menos épocas de lo habitual, mayoritariamente cuando no hay preentrenamiento. El motivo es la gran dificultad de generalizar (como ya se ha explicado en el apartado 6.4) con una cantidad de información tan baja. A pesar de ello, los resultados no son tan malos como cabría esperar, y si se suman a los del primer caso, es posible confirmar lo beneficioso de la técnica y su capacidad de compensar en gran medida la menor cantidad de datos etiquetados.

La validez de estos resultados se pudo comprobar también en las propias segmentaciones. En la Figura 6.7 pueden verse ejemplos para los modelos FCDN56 y ENet sin preentrenamiento con tamaños del conjunto de entrenamiento de 1 y 15 imágenes. En ellas se puede apreciar que la segmentación de la red de FCDN56 entrenada con 15 imágenes, a pesar de ser parecida a la de la entrenada con 1 sola imagen, tiene menos puntos inconexos, mal segmentados. Por otro lado, también se observa que en la comparación análoga para ENet estas diferencias son mucho mayores. La mayor capacidad, en este caso, parece estar ayudando a las redes a obtener unos mejores resultados en condiciones más difíciles, con menos datos etiquetados.



(a) (b) (c) (d)

Figura 6.7: Segmentaciones de FCDN56 y ENet sin preentrenamiento para conjuntos de entrenamiento de 1 imagen (a, c) y 15 (b, d) a partir de la retinografía nº 14 de DRIVE-test.

FCDN, ENet y U-Net

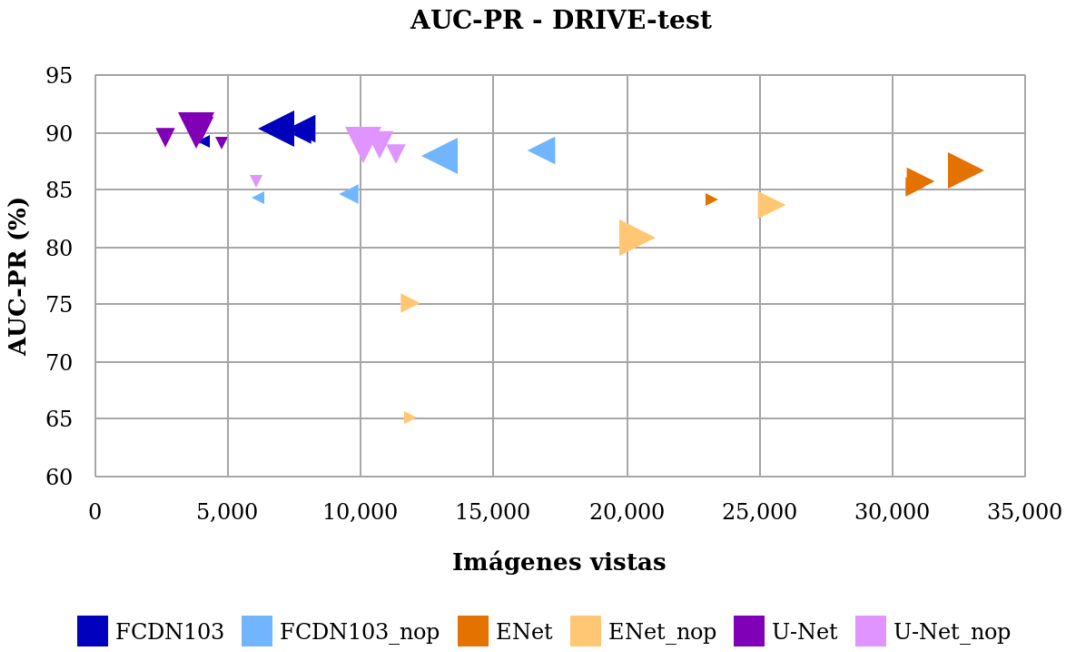
En el apartado anterior se han tratado todos los temas de forma más o menos separada para cada arquitectura. En este, compararemos los resultados de las redes entre sí teniendo en cuenta en cada caso el número de imágenes vistas, el tamaño del conjunto de entrenamiento y los valores de AUC-PR en DRIVE-test y STARE.

Las Figuras 6.8a y 6.8b muestran cómo son los resultados cuantitativos (AUC-PR) de las diferentes arquitecturas (FCDN, U-Net y ENet) para los conjuntos de datos DRIVE-test y STARE, respectivamente. Como se puede observar, las gráficas evidencian que U-Net y FCDN tienen unos resultados muy parecidos en DRIVE-test y a distancia de ENet, mientras que en STARE los mejores resultados son los de U-Net, a poca distancia de los de FCDN y a bastante más de los ENet. Otro detalle destacable es la menor cantidad de imágenes *vistas* de U-Net en comparación con FCDN y ENet.

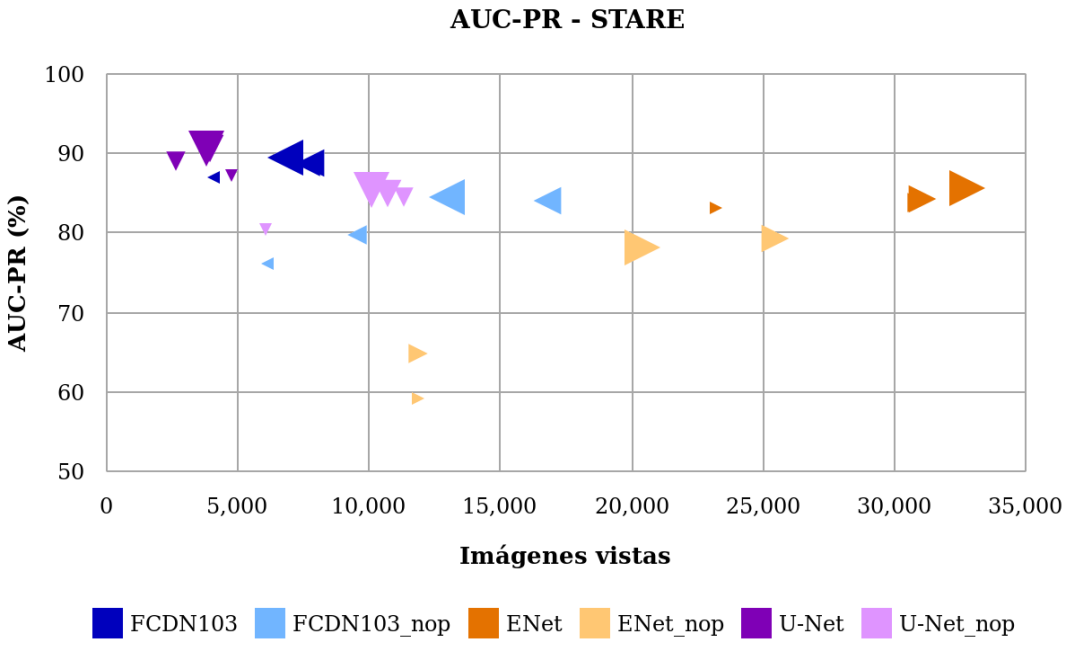
En cuanto a las segmentaciones de las diferentes redes, pueden verse ejemplos para FCDN y U-Net en las Figuras 6.9a y 6.9b, respectivamente. Concretamente, estas segmentaciones pertenecen a las redes FCDN103 y U-Net con preentrenamiento y 15 imágenes de entrenamiento. Tal y como se puede ver en en la figura, salvo algunos detalles de continuidad y suavidad para los vasos más pequeños, los cambios entre las imágenes de salida de FCDN y U-Net, cuando la evaluación es en DRIVE-test, son prácticamente imperceptibles.

Para el caso de STARE, se toma la Figura 6.9 como ejemplo. En ella se muestran segmentaciones generadas a partir de la imagen nº 139 de STARE (Fig. 5.14c) por FCDN103, U-Net y ENet en sus versiones con preentrenamiento y 15 imágenes de entrenamiento. En estas imágenes, aunque no sean muy grandes, ya se ven algunas diferencias más. Las redes con la arquitectura FCDN[6], tal y como se veía en la parte de reconstrucción, donde tendían a resaltar como vasos las zonas de hemorragias, hacen ahora algo muy similar con las segmentaciones, como heredando en cierta forma ese comportamiento. Allí donde U-Net deja un espacio en

[6]Para ENet, en este sentido, sucede igual que para FCDN.



(a)



(b)

Figura 6.8: Número de imágenes vistas contra AUC-PR(%) para las arquitecturas U-Net, ENet y FCDN en DRIVE-test (a) y en STARE (b). El tamaño de los puntos indica el volumen del conjunto de entrenamiento (1, 5, 10 o 15). La parte _nop de los nombres indica que la red no dispuso de un modelo preentrenado.

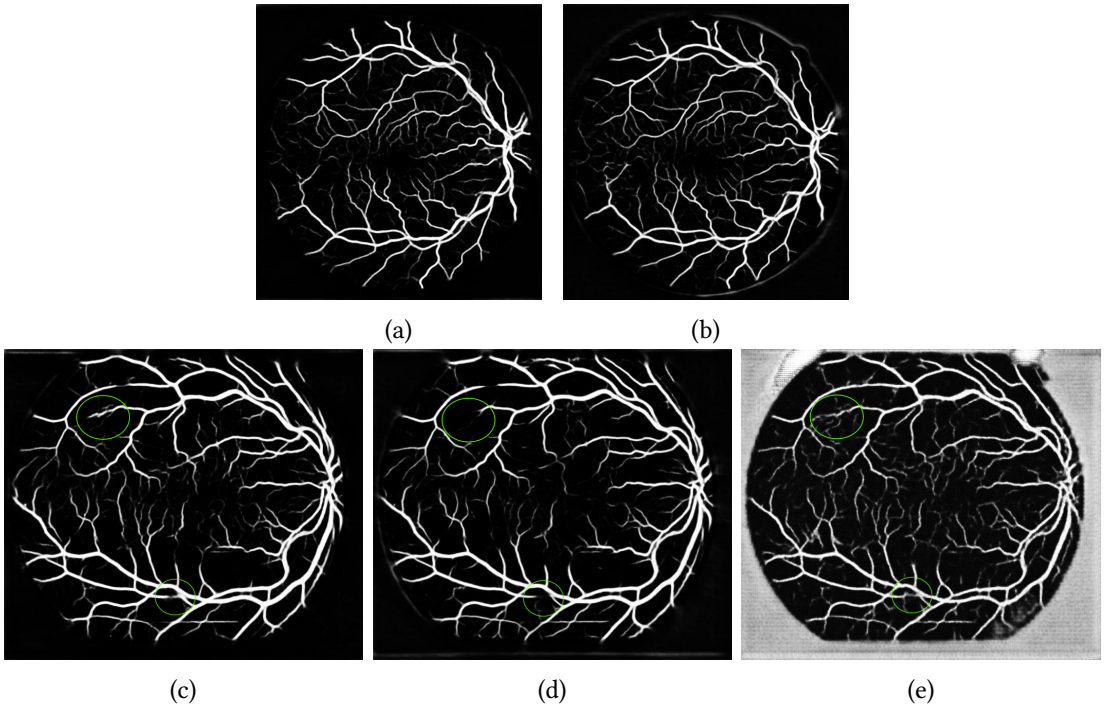


Figura 6.9: Segmentaciones de FCDN103 (a) U-Net (b) y ENet (c) con 15 imágenes en el y con preentrenamiento a partir de la retinografía nº 139 de STARE (Fig. 5.14c).

negro, FCDN dibuja en ocasiones una sección de vaso que une las partes del vaso original a ambos lados de la hemorragia. No sabríamos decir si esta forma de hacer las segmentaciones es la más adecuada o no, ni si un caso es mejor que el otro. Básicamente, porque entre las propias segmentaciones manuales hay cierta discrepancia. En las máscaras de STARE creadas por Adam Hoover (AH), esos huecos existen y son tal cual aparecen en las segmentaciones de U-Net, pero en las de Valentina Kouznetsova, más exhaustivas y con más vasos representados, se les da continuidad, independientemente de la existencia de las hemorragias, de forma parecida a como lo hace FCDN. Al utilizar en la evaluación las segmentaciones de AH como referencia, los resultado cuantitativos de U-Net se han visto favorecidos.

Al comparar las segmentaciones de las distintas arquitecturas, como en el ejemplo de la Figura 6.9, se advierte que las pertenecientes a U-Net y a la mejor versión de FCDN (FCDN103) son muy similares. Si se obvia el detalle mencionado para ciertos elementos de carácter patológico, se tiene que las diferencias entre ellas son muy sutiles. Por otro lado, las segmentaciones de ENet, como era previsible, son en comparación mucho menos precisas. A pesar de que la red con esta arquitectura es capaz de hacer segmentaciones bastante buenas y bastante acertadas, estas tienen una menor suavidad y la continuidad de los vasos se preserva peor que en las de las otras dos arquitecturas. Además, la *sensación visual* (pues realmente tienen el mismo nº de píxeles) es que las segmentaciones de ENet tienen una «menor resolución».

Capítulo 7

Conclusiones

En este trabajo se han organizado los experimentos en dos partes, la 1ª centrada en la tarea de reconstrucción multimodal y la 2ª en la tarea de segmentación de vasos. En ambas se ha conseguido obtener, con aprendizaje supervisado en conjuntos de datos adecuados, redes con distintas arquitecturas capaces de realizar una u otra tarea de forma satisfactoria. En el primero, estas redes han sido capaces de construir pseudoangiografías razonablemente buenas; en el segundo, segmentaciones de vasos detalladas. En ambos casos las imágenes de partida han sido únicamente retinografías.

En la **1ª parte**, algunas de las pruebas que se han hecho a nivel de arquitectura para los modelos FCDN y ENet evidenciaron la difícil tarea de valorar objetivamente el impacto de ciertos elementos arquitectónicos en el resultado final. La evaluación cualitativa de las pseudoangiografías no dio lugar a argumentos concluyentes a favor de la inclusión o exclusión de ciertos elementos; y la evaluación cuantitativa, en este caso, todavía menos, aunque se mostró coherente con la evaluación cualitativa de las pseudoangiografías. De todo ello se pudo concluir que en estos dominios de aplicación restringidos hay ciertos elementos que no son relevantes en los resultados finales, como el *bias* de la convolución final, y que su impacto es también muchas veces dependiente de la arquitectura, como se vio en el caso de *Batch Normalization*, donde en FCDN no aportaba y en ENet sí.

Por otro lado, las distintas comparaciones entre las arquitecturas U-Net, FCDN y ENet han permitido ver claramente matices diferenciadores en las imágenes generadas. En DRIVE, las de U-Net y FCDN han resultado ser muy similares, salvo pequeños detalles, y no ha sido hasta la prueba en STARE, un conjunto de datos mucho más complicado, cuando se advirtieron una ligera mayor capacidad de generalización, robustez y coherencia semántica por parte de U-Net. Por otra parte, las pruebas con ENet han permitido confirmar que con arquitecturas livianas también se pueden obtener resultados decentes (aunque a distancia del de las otras dos redes) en los que se vean reflejados los principales elementos estructurales del fondo de ojo, como la fóvea, el disco óptico y los vasos de mayor tamaño y de tamaño medio. Esto refleja un cierto nivel de comprensión de la estructura retiniana por parte de la red.

En esta parte del trabajo se abre además la pregunta, por los en general mejores resultados de U-Net, de si es realmente conveniente la construcción de redes cada vez más profundas, con los mecanismos que estas incorporan para hacer posible la convergencia, o si es mejor orientar las arquitecturas hacia modelos menos profundos y más jerárquicos, con gran número

de parámetros pero con diseños más clásicos, como es el ejemplo de U-Net.

Además, las pruebas de esta primera parte dispusieron los modelos entrenados utilizados, por medio de la técnica de *transfer learning*, en la siguiente tarea a abordar, la segmentación de vasos.

Para la **2ª tarea**, más compleja que la primera, las pruebas realizadas con los diferentes modelos de la arquitectura FCDN nos han permitido comprobar que en escenarios como este aumentar la profundidad de los modelos afecta positivamente en gran parte de los casos, y que las segmentaciones por ellos conseguidas suelen tener unas características más adecuadas. A mayores, la utilización de varios tamaños para los conjuntos de entrenamiento y validación posibilitó la evaluación del impacto de la reducción de variabilidad en los ejemplos de entrenamiento cuando se están utilizando mecanismos de *data augmentation*. A partir de estos resultados fue posible confirmar lo beneficioso de tales técnicas y lo relevante de la variabilidad por ellas aportadas a la hora de optimizar las redes durante el proceso de entrenamiento.

Finalmente, como punto más importante, se demostró también la conveniencia de utilizar modelos preentrenados en la tarea de reconstrucción multimodal a la hora de abordar tareas como la de segmentación de vasos. Cuando se escogen como punto de partida de los entrenamientos, las redes obtienen mejores resultados con un menor tiempo de entrenamiento (habitualmente) que sus equivalentes con inicializaciones como He o LeCun.

En nuestro caso concreto, el camino hecho en la comprensión de la estructura de la retina, especialmente del árbol arterio-venoso, durante la primera de las tareas, facilita enormemente a las redes el entrenamiento, y permite centrar los esfuerzos en el refinamiento de los detalles. También se ha comprobado con los diferentes modelos en qué grado mitiga el uso de preentrenamientos las diferencias arquitectónicas, y cómo en estos casos se alcanza una estabilidad a nivel de resultados que no se puede apreciar para los modelos entrenados *desde cero*.

La utilización de varios modos de imagen para entrenamiento surge así como una buena alternativa cuando no se dispone de gran cantidad de datos etiquetados. La aplicación de técnicas como el aprendizaje transferido permite el aprovechamiento de lo aprendido en tareas como la reconstrucción multimodal, donde no se usan imágenes con máscaras de segmentación manual asociadas, para aplicaciones finalistas distintas, como la segmentación de vasos en retinografía, en la que sí son necesarias esas máscaras. Así, se consigue obtener con un número menor de imágenes de segmentación manuales elaboradas por expertos mejores imágenes resultado que las que se conseguirían con redes entrenadas desde cero. Dicho de otro modo, la utilización de preentrenamientos en tareas de reconstrucción multimodal es capaz de mitigar la escasez de datos etiquetados para tareas finalistas, aprovechando el conocimiento adquirido durante las primeras de forma que se mejoran los resultados de las segundas, acortando además los tiempos de entrenamiento necesarios.

Capítulo 8
Trabajo futuro

A partir de este trabajo hay muchos caminos que se pueden seguir y que podrían resultar interesantes. Los principales o considerados como más importantes son los que se detallan a continuación en las siguientes secciones.

8.1 Funciones de error

Durante el trabajo, se advirtió que una de las limitaciones que tienen funciones de error como BCE para la tarea de segmentación es que todos los puntos de la imagen ponderan igual. Es decir, que no hay ajustes, por ejemplo, en función de la dificultad de la segmentación en cada zona (e.g. para vasos pequeños), de la cantidad de vasos sí segmentados, o del desequilibrio entre el número de píxeles de fondo y de vaso. Con BCE, el punto de vista es estrictamente global, y vale lo mismo un punto bien clasificado en el interior de un vaso principal que en el borde de un vaso pequeño.

Si se pudiera diseñar una función de error que no ponderara todos los puntos por igual, y que tuviera en cuenta propiedades más elaboradas y complejas como las antes mencionadas, creemos que se podrían obtener, muy probablemente, mejores imágenes de segmentación, con una mayor sensibilidad a los detalles.

En esta línea, en los últimos tiempos han surgido artículos que tratan de solucionar este problema. Funciones de error alternativas a las más clásicas, como *FocalLoss* [74], utilizan ponderaciones en función de la *dificultad* de la segmentación de cada píxel, gestionando el desbalanceo de clases. Así, se consigue obtener segmentaciones más precisas en problemas en los que ese desbalanceo favorece ampliamante a la clase *negativa* o *fondo* sobre la clase *objeto* (lo que se segmenta).

Las ideas que implementa esta función de error, entre otros ejemplos, podrían ser un buen punto de partida, y su aplicación y evaluación en la tarea de segmentación, un buen primer paso.

8.2 Arquitecturas y sus elementos

Otro aspecto interesante a explorar en mayor medida es el impacto de determinadas estructuras u operaciones dentro de las arquitecturas. Por ejemplo, en el caso de FC-DenseNet,

se podrían hacer modificaciones como eliminar las *skip connections* y ver así cuánto de importantes son para el resultado final. También se podría reducir el número de capas aumentando el número de parámetros, utilizar convoluciones dilatadas en lugar de *poolings*, etc. En definitiva, pruebas a nivel de arquitectura que nos permitan confirmar o descartar de forma más o menos concluyente algunas de las hipótesis que fueron surgiendo durante el trabajo.

Dentro de esto, una de las dudas que surgen la consideramos de especial interés: la idoneidad del nuevo paradigma de más capas con menor número de parámetros en este tipo de dominios. Este cuestionamiento es fruto de los resultados obtenidos en las dos tareas, tal y como se expuso en los apartados correspondientes en su momento. U-Net, *a priori* la de diseño más sencillo, con menor número de capas y mayor número de parámetros, es muchas veces la que obtiene mejores resultados. Esto invita a pensar que, o bien ciertos elementos arquitectónicos *novedosos* no aportan realmente demasiado a nivel de resultados, o que no son las arquitecturas escogidas las más aptas para el problema (relacionado en parte con lo anterior). O también, que en ciertos casos, como el nuestro, compensa no tener demasiadas capas pero sí muchos parámetros.

Sea como fuere, en este trabajo no se ha podido comprobar si de verdad es alguno de ellos o ninguno y hay que descartar los tres. Hacer una evaluación rigurosa y en detalle permitiría quizás vislumbrar el camino que convendría seguir, tanto a nivel de paradigma como a nivel de elementos arquitectónicos, y se podría averiguar si es alguna de esas tres opciones.

Un análisis como este que fuera concluyente tendría una gran relevancia, y podría condicionar en muchos casos la selección de una u otra arquitectura, o unos determinados elementos arquitectónicos, a la hora de abordar problemas con CNNs en dominios cerrados.

8.3 Aprendizaje multitarea

En este trabajo se han expuesto dos tareas diferentes pero relacionadas: la reconstrucción multimodal y la segmentación de vasos. Con ambas se pudo demostrar, por un lado, que la técnica de aprendizaje transferido puede ser satisfactoriamente aplicada en dominios de aplicación donde se disponga de varios modos de imagen; y por otro, que la utilización de modelos preentrenados en tareas de reconstrucción permite obtener redes con mejores resultados en un menor tiempo de entrenamiento que sus equivalentes sin aprendizaje transferido.

Por su relación con aprendizaje transferido, y dadas las ventajas descritas de esta última, es por ello que se considera el aprendizaje multitarea [75] como un camino natural a tomar. Como paradigma de aprendizaje, la idea en la que se basa es que hay muchos casos en los optimizar una red para resolver dos o más tareas (relacionadas) a un mismo tiempo puede ser beneficioso para los resultados de todas ellas, gracias a la frecuente transversalidad que presenta el conocimiento. De alguna forma, lo que se busca es que las tareas se complementen,

y que unas y otras suplan las posibles carencias que pudieran tener. De este modo, desde el punto de vista de ANNs, en aprendizaje multitarea se consigue obtener una sola red y un solo conjunto de parámetros o características[1] capaz de resolver dos o más tareas a un tiempo, y en ocasiones, mejor de lo que lo podría hacer en cada una si se abordaran por separado.

En nuestro caso, sería interesante evaluar el comportamiento de la red con aprendizaje multitarea para las tareas de reconstrucción multimodal y segmentación de vasos; y ver si de este modo se supera de nuevo a las redes con aprendizaje *monotarea* e incluso con aprendizaje transferido.

8.4 Técnicas de *data augmentation*

Visto el buen funcionamiento del *data augmentation* en el trabajo, que se comprobó en el segundo de los experimentos, consideramos la exploración de otros métodos de *augmentation* (como las transformaciones elásticas [76]), o la incorporación de nuevas operaciones a aquellos de los que ya se disponen, una vía potencialmente provechosa.

Si las simples transformaciones escogidas en el trabajo han sido capaces de paliar en gran medida la menor cantidad de imágenes durante el entrenamiento, aportando la variabilidad necesaria para la obtención de un modelo capaz de generalizar, otros métodos más elaborados es muy posible que pudieran ayudar a obtener redes más robustas, con una mayor capacidad de generalización y una mejor respuesta a imágenes con características distintas a las utilizadas durante el entrenamiento, tal y como se probó en nuestro caso con el conjunto STARE.

También sería interesante, en esta línea, hacer una valoración más precisa del impacto de las técnicas de *data augmentation* en los resultados finales, realizando entrenamientos distintos (con repeticiones) en los que unos dispusieran de este tipo de operaciones y otros que no.

Por último, como en el trabajo se han utilizado varias operaciones simultáneamente, quizás sería interesante también hacer entrenamientos con unas u otras combinaciones, de forma que se pudiera comprobar si todas las operaciones están realmente aportando y cuánto lo hacen si así es. De esta forma, se podría valorar qué operaciones son más convenientes.

8.5 Patología

En imagen médica, una de las aplicaciones más interesantes de las CNNs es el apoyo al diagnóstico. Tareas como la localización, segmentación o clasificación en imágenes del tipo de las vistas en este trabajo pueden suponer una gran ayuda a los médicos y personal sanitario, y

[1] Hay múltiples formas de llevar a la práctica el concepto de aprendizaje multitarea.

junto con otras tecnologías como herramientas de visualización avanzadas se puede conseguir que los diagnósticos sean más sencillos de obtener y más acertados.

En imagen oftalmológica, esta línea de investigación tiene bastante fuerza, y todos los años surgen nuevas aproximaciones con mejores resultados que las anteriores. La clasificación de imágenes en sanas y patológicas, o la detección de indicios de patología para enfermedades como retinopatía diabética o degeneración macular asociada a la edad (DMAE) son algunas de las tareas que algunos trabajos con redes de neuronas convolucionales han abordado desde hace tiempo (e.g. [77, 78] —retinopatía diabética— y [79, 80, 81] —DMAE—), y en algunos casos con bastante éxito.

En esta línea, sería interesante llevar esta metodología hacia aplicaciones finalistas relacionadas con patología, y ver cómo de exitosa puede ser la aplicación de algunas de las ideas aquí evaluadas, y confirmadas, como el aprendizaje transferido a partir de una tarea de reconstrucción de imagen multimodal, para la solución de problemas reales y altamente complejos como los que se han mencionado (entre otros muchos).

Esto también permitiría evaluar hasta qué punto las arquitecturas empleadas en el trabajo, sobre todo FC-DenseNet y U-Net, son capaces de abordar esos problemas, y si sigue siendo U-Net la más adecuada.

Apéndices

Apéndice A

Redes de neuronas artificiales

1 Neurona artificial

Una **neurona artificial**, en su forma más simple (ver figura A.1), es un modelo computacional que toma como entrada un vector de valores y les aplica una suma ponderada con unos *pesos* asociados a la propia neurona, seguida de un corte por umbral [21]. Ese corte representa la decisión de transmisión de la neurona y recibe el nombre de *función de activación*[1] [21].

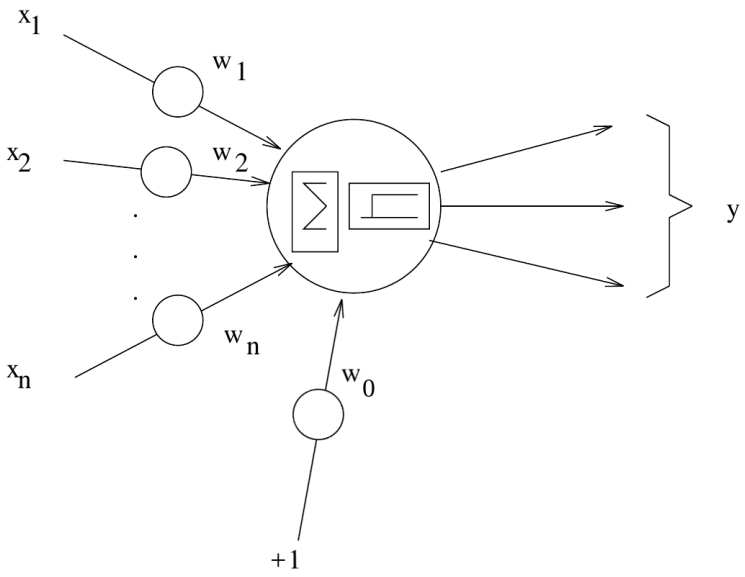


Figura A.1: Modelo básico de neurona artificial.

Matemáticamente se define la salida de una neurona artificial y como

$$y = F\left[\sum_{i=1}^{n} x_i w_i + w_0\right] , \tag{A.1}$$

donde n denota el número de conexiones de entrada, $\overrightarrow{x} = (+1, x_1, x_2, \ldots, x_n)$ el vector de entrada, $\overrightarrow{w} = (w_0, w_1, w_2, \ldots, w_n)$ el vector de pesos, y F la función de activación. El

[1] En la bibliografía también puede verse el nombre de *función de transferencia*.

resultado antes de aplicar F se conoce como *logit*.

La entrada que se puede ver en la parte inferior de la Figura A.1 asociada al peso w_0 y con valor $+1$, se llama *bias*. Su función es la de capturar la parte invariante de las predicciones. Es decir, es una forma de regular cómo de fácil es que la neurona tenga un 1 a la salida [24]. Esto es útil, por ejemplo, en casos donde las entradas están centradas en una media distinta de 0 [35]. En definitiva, lo que permite el *bias* es desplazar la función de activación, de forma que se produzca un mejor ajuste de los datos.

2 Funciones de activación

Hay muchas funciones de activación distintas, cada una con sus ventajas, inconvenientes y aplicaciones adecuadas. Aquí nos centraremos solamente en aquellas relacionadas con las arquitecturas de red utilizadas en este trabajo: sigmoide, ReLU, PReLU y softmax. Se puede ver la representación de todas ellas en la Figura A.2.

Todas ellas aportan *no linealidad* a la función de la neurona artificial, lo que permite extraer patrones más complejos de los datos. Este aspecto es todavía más relevante cuando las neuronas se combinan en múltiples capas, pues ayuda a la creación de composiciones de funciones diferentes más potentes que las diferencia de las redes de una sola capa[2] [35].

Sigmoide

Esta función de activación (ver Figura A.2a) recibe su nombre de su forma. Formalmente, se define como

$$S(x) = \frac{1}{1+e^{x}} \,. \tag{A.2}$$

Como puede verse, sus valores de salida oscilarán siempre entre 0 y 1. Esto es útil cuando se quiere trabajar con probabilidades. Además, permite trabajar con valores *outliers* sin problemas (los reduce pero no los elimina [27]). Por otra parte, tiene la desventaja de que *satura* en los límites, por lo que el gradiente es prácticamente 0 para valores cuyo valor absoluto es grande. Esto provoca que los pesos de algunas neuronas se actualicen muy lentamente durante procesos de aprendizaje con gradiente descendente [35].

ReLU

La función de activación *Restricted Linear Unit* (ReLU) [82] (ver Figura A.2b) es hoy en día una de las más utilizadas en aprendizaje profundo con redes de neuronas convolucionales (se

[2] Si las redes de dos o más capas solo usaran funciones lineales no tendrían mayor potencia que aquellas de una sola capa.

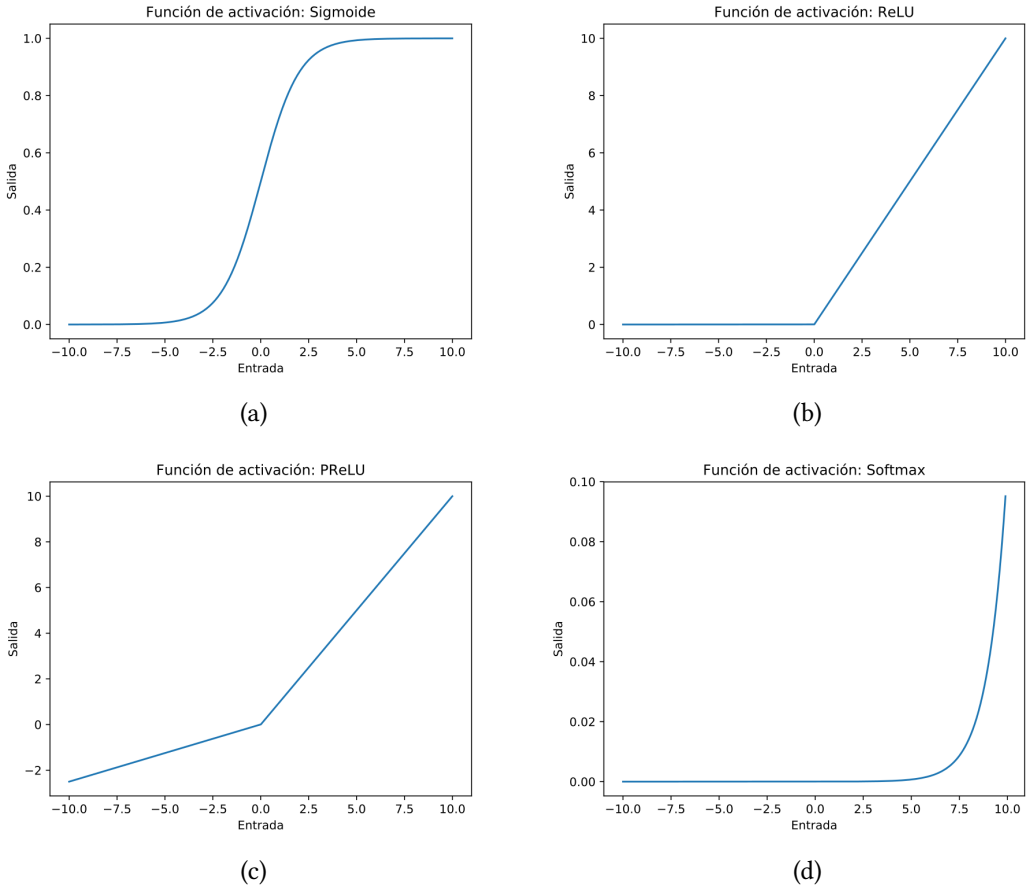


Figura A.2: Ejemplos de funciones de activación. (a) Sigmoide. (b) ReLU. (c) PReLU. (d) Softmax.

explicará en qué consisten más adelante). Matemáticamente, se define así:

$$S(x) = max(0, x) \ . \tag{A.3}$$

Esta función de activación tiene varias características que la hacen ser elegida frente a alternativas como sigmoide. Principalmente, estas derivan de la preservación del gradiente constante para valores altos, al contrario que sigmoide. Entre las principales se pueden contar las siguientes: (1) la rapidez en el entrenamiento, pues las redes alcanzan una solución —convergen— mucho antes [28]; y (2) porque el comportamiento de redes grandes entrenadas en conjuntos de datos de elevado tamaño mejora de forma significativa, en parte por lo anterior [28] y en parte porque, en cierto modo, ReLU es prácticamente un modelo lineal (se compone de dos partes lineales), y este tipo de modelos tienen ciertas propiedades que hacen que generalicen bien.

PReLU

PReLU es una modificación de ReLU que se define del siguiente modo [83]:

$$S(x) = max(0, x) + \omega \cdot min(0, x) \ . \tag{A.4}$$

Como puede verse en la expresión anterior y en la Figura A.2c, la función consta de dos partes lineales, $y = \omega \cdot x$ en $(-\infty, 0]$ e $y = x$ en $(0, \infty)$. El valor de ω es *entrenable* junto con el resto de pesos de la neurona durante el entrenamiento.

Comparada con ReLU, PReLU tiene todas sus ventajas en relación a sigmoide y, a mayores, se comporta mejor en redes de neuronas de alta profundidad, obteniendo mejores resultados, especialmente cuando los conjuntos de datos son grandes. Esto se debe a la ReLU tiene una zona plana sin gradiente ni salida, una zona de «neurona muerta» que en PReLU no hay.

Softmax

Softmax [84] es una función de activación (ver Figura A.2d) de la forma [35]:

$$S\left(y_i'\right) = \frac{e^{v_i}}{\sum_{j=1}^{C} e^{v_j}} \ , \qquad \forall i \in \{1, \ldots, C\} \ , \tag{A.5}$$

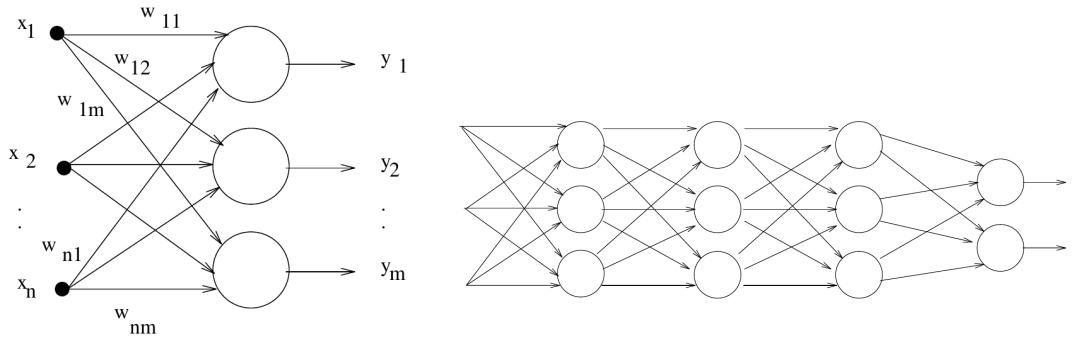
donde C es el número de clases entre las cuales clasificar y v_i la entrada i a la función de activación. La idea general es que dadas C entradas relaes $v_i \in \{1, \ldots, C\}$, esta función devuelve la probabilidad asociada de pertenencia a cada clase (valor entre 0 y 1 y la suma de todas las probabilidades igual a 1 [85]), destacando un poco más aquella que más por encima está del resto —si esta diferencia es significativa— (actúa como una especie de función max pero diferenciable) [86]. Esta función, como le pasaba a la sigmoide, también puede saturar: a 1 cuando una entrada es mucho mayor a las demás y a 0 cuando una entrada no es la de valor máximo y este es mucho mayor [26].

3 Arquitecturas de red neuronal

A nivel arquitectónico se podrían establecer dos grandes grupos: el de las redes formadas por una única capa (redes monocapa), y el de las redes formadas por al menos dos (redes multicapa). Generalmente, las más complejas, y por tanto de mayor potencia computacional, son las del segundo grupo.

Las **redes monocapa**, como bien se ha dicho, son aquellas que solo tienen una capa cuya salida coincide con la salida de la red, tal y como puede verse en el esquema general de la Figura A.3a. Las entradas no se cuentan estrictamente como capas [3], ya que se limitan a distribuir el

[3] Aunque serán llamadas de la forma *capa de entrada* por su extendido uso en la bibliografía.



(a) (b)

Figura A.3: (a) Esquema de una red monocapa [21]. (b) Ejemplo de red multicapa [21].

vector de entrada [21].

La salida $\overrightarrow{y}$ de una red monocapa se define como

$$\overrightarrow{y} = F\left(W\overrightarrow{x}\right) \ . \tag{A.6}$$

donde W denota la matriz $m \times n$ con los pesos de las neuronas de la capa, $\overrightarrow{x}$ el vector de entrada e $\overrightarrow{y}$ el vector de salida. El número de filas m de la matriz de pesos se corresponde con el número de neuronas de la capa y el número de columnas n con el rango del vector de entrada.

Por otro lado, las **redes multicapa** agrupan las neuronas en capas sucesivas interconectadas. Esto aumenta la capacidad de la red siempre que la función de activación de alguna de las capas sea no lineal[4] [21]. Por esta razón, la inmensa mayoría de las redes que hoy en día se utilizan son de este estilo. En la Figura A.3b se muestra un ejemplo de este tipo de red. A las capas intermedias se las denomina *capas ocultas*, mientras que la última recibe el nombre de *capa de salida*. La *capa de entrada* está formada por el vector de entrada de la primera capa, por lo que no tiene pesos asociados. En la Figura A.3b se muestra, por ejemplo, una red con 1 capa de entrada, 3 capas ocultas y 1 capa de salida.

4 Tipos de redes según su conectividad

Las arquitecturas de redes más comunes son *feed-forward* y *completamente conectadas*. La red de la Figura A.3b es un ejemplo.

Feed-forward hace referencia a que las redes solamente poseen conexiones «hacia adelante» (dirigidas hacia la capa de salida), y no «hacia atrás» (dirigidas hacia la capa de entrada)

[4]De no ser así, serían equivalentes a las redes monocapa.

[24].

Sin embargo, existen también arquitecturas de red con alguna conexión de ese segundo tipo, «hacia atrás», de forma que tienen algún *bucle*. En estas redes se establecen habitualmente distintos tiempos y momentos de activación para las neuronas, lo que permite simular más de las que realmente hay [24].

Por otro lado, el término *completamente conectadas* (*fully-connected*) hace referencia a que las neuronas de cada capa están conectadas con cada una de las neuronas de las capas adyacentes. De nuevo, la Figura A.3b es un ejemplo de este tipo de redes.

Como antes, no todas las redes son de esta forma, y existen ejemplos, como las redes de neuronas convolucionales, de las que hablaremos más en detalle posteriormente, que utilizan otro tipo de conectividad distinta y basada en conexiones locales.

5 Aprendizaje supervisado

El aprendizaje, en las ANNs, consiste en modificar los pesos de las conexiones de la red (principalmente, pues hay otros *parámetros* [86]) de forma que se obtenga la salida deseada para cualquier entrada dada [21]. Esto se consigue presentando ejemplos de entrada a la red y ajustando los pesos en función de una *regla de aprendizaje* [21, 24, 85].

En el tipo de aprendizaje denominado **aprendizaje supervisado** se cuenta con unos *pares de entrenamiento* compuestos por una entrada y una *salida deseada* [21, 86]. El ajuste de parámetros se hace en función de la diferencia (*valor de error*) entre la salida de la red y la salida deseada [21, 86] que se obtiene por medio de una función llamada *función de pérdida* o *de error* (*loss*) [86]. Hay varias formas de hacer el ajuste en función de esos valores de error, y en este trabajo se hará uso de uno de los más utilizados: el optimizador estocástico Adam, una extensión del algoritmo clásico de gradiente descendente estocástico (SGD) que utiliza ratios de aprendizaje individuales adaptativos calculados a partir de estimaciones sobre los gradientes.

Otros tipos de aprendizaje Además del supervisado, hay otros tipos de aprendizaje, y los principales se conocen como **aprendizaje no supervisado** y **aprendizaje por refuerzo**. En el primero no se dispone de las salidas deseadas, de modo que lo que se busca es encontrar ciertos patrones o regularidades en las entradas que, generalmente[5] [13], permitan formar grupos o clases [86]. En el segundo se intentan aprender *políticas* buenas —secuencias correctas de *acciones* que lleven a la solución— [86], por lo que se debe ser capaz de proporcionar a la red una evaluación global de cómo de bien lo está haciendo desde la última vez que se evaluó [21].

[5] Pues hay otros tipos.

Aproximación clásica de ingeniería de características

En la aproximación clásica, a las redes neuronales se les pasa como entradas los valores de una serie de *características* extraídos de los datos en bruto —los ejemplos disponibles para una tarea dada—. Una característica es todo atributo que se considere útil para la consecución de la tarea, o dicho de otro modo, toda propiedad o cualidad de los ejemplos cuyo valor sea *representativo* y facilite a la red la obtención de resultados correctos. Se entiende como valor representativo un valor que distinga de una forma adecuada a los ejemplos entre sí (bien de forma individual o en grupos). Por ejemplo, si se están clasificando fotografías de limones y limas, una característica adecuada podría ser el color medio.

De esta manera, antes de realizar el proceso de entrenamiento propiamente dicho, hay un proceso manual de extracción de características por parte de los diseñadores conocido como *ingeniería de características*, que tiene por fin la definición de características buenas para unos ejemplos y una tarea dados.

Así, el proceso completo de obtener una solución para un ejemplo dado consistiría en dos fases: una de obtención de los valores de las características para el ejemplo dado y otra de obtención de la salida, con esos valores como entrada, por parte de la red.

6 Aprendizaje profundo

El *aprendizaje profundo* (*deep learning*) es una rama del aprendizaje máquina que pone especial énfasis en organizar en capas sucesivas el aprendizaje de las representaciones de los datos [14]. Esto es, en una organización jerárquica —en varios niveles— del «conocimiento». En redes de neuronas artificiales (los modelos más utilizados para su implementación) esto se traduce en organizar el «conocimiento» en un mayor número de capas.

Para esta tecnología, esta idea supuso un importante cambio de paradigma. Hasta ese momento, la tendencia era utilizar redes de neuronas con pocas capas pero con muchas neuronas por capa. Ahora, por el contrario, se busca cada vez más profundidad (más capas) con menos neuronas por capa. Estos cambios hacen mejorar el rendimiento y el comportamiento de las redes en problemas más complejos [34, 36, 38]. Desde un punto de vista teórico, no obstante, hay que resaltar que un mayor número de capas no otorga a las redes una mayor capacidad de representación que aquellas con solo 2 y muchas más neuronas.

Por otra parte, los grandes avances de la computación en GPU (*Graphics Processing Unit*) favorecieron enormemente la aplicación de este nuevo planteamiento, acelerando el aprendizaje de redes mucho mayores.

Otro cambio importante fue el cambio de enfoque con respecto a la forma clásica de hacer las cosas: como se ha dicho antes, lo más frecuente era proporcionar a las redes entradas «elaboradas», fruto de un proceso de ingeniería de características, que facilitaran a la red las

decisiones. Ahora, ese proceso se elimina, y a las redes se les presentan los datos en crudo, de forma que son ellas las encargadas de llevar a cabo esa de extracción de características relevantes. Este nuevo enfoque permite ahora un tipo de aprendizaje denominado *end-to-end*, en el que todo el proceso de aprendizaje se puede considerar como único y global —directamente de los datos a las soluciones— sin etapas ni pasos intermedios.

Esta nueva forma de hacer las cosas, con las nuevas redes de alta capacidad, tuvo dos beneficios importantes. Por un lado, simplificó el proceso, permitiendo eliminar la laboriosa extracción de las características *a mano*. Y por otro, mejoró los resultados, al aprovechar la capacidad de las redes para obtener por sí solas representaciones internas adecuadas a partir de los ejemplos de entrada en crudo.

En este contexto, las redes de neuronas convolucionales son las que mejor se han adaptado, y se han impuesto sobre las redes clásicas, completamente conectadas, con una mejor generalización y una mayor facilidad de entrenamiento [87].

7 Gradiente descendente

Una de las formas clásicas y más utilizadas para *optimizar* los pesos de las redes minimizando la función de error para los ejemplos de un conjunto de datos es el método del gradiente descendente. Básicamente, consiste en, partiendo de unos pesos iniciales, hacer ajustes iterativos siguiendo la dirección de gradiente, mediante la siguiente regla de actualización:

$$\overrightarrow{w}_{k+1} = \overrightarrow{w}_k - \alpha_k \nabla J\left(\overrightarrow{w}_k\right) \ , \tag{A.7}$$

donde J es la función de error, $\nabla J\left(\overrightarrow{w}_k\right)$ el gradiente de J para el vector de pesos $\overrightarrow{w}_k$ en el paso k y α el *ratio de aprendizaje* (*learning rate*), que marca la «relevancia» de la actualización[6] [35].

La actualización de los pesos se puede hacer después de obtener la salida, y por lo tanto el error, para un ejemplo de entrada (*gradiente descendente estocástico*[7], *stochastic gradient descent*, SGD), para un subconjunto del total (*gradiente descendente por lotes*, *minibatch gradient descent*) o para todos los ejemplos del conjunto de entrenamiento (*gradiente descendente*, GD). Independientemente de cómo se actualicen los pesos, cada vez que se han presentado a la red todos los ejemplos de entrenamiento al menos una vez, se dice que se ha completado una *época* (*epoch*).

[6] A mayor ratio de aprendizaje más afecta la actualización a los pesos actuales.

[7] La etiqueta «estocástico» viene dada por la selección aleatoria que se hace de los ejemplos utilizados para hacer la evaluación de los gradientes.

8 Método de propagación hacia atrás (*backpropagation*)

En el apartado anterior se ha visto cómo actualizar los pesos de las redes (optimizándolos), pero no cómo calcular el gradiente de la función de error necesario para ello. En una red monocapa es sencillo, pues se dispone de la salida deseada para esa capa, y por lo tanto del error, pero cuando hay capas ocultas, como en las redes multicapa, este error no está disponible. El método de propagación hacia atrás es precisamente el que se encarga de cubrir ese hueco, permitiendo computar todos los gradientes necesarios para la actualización de los pesos para todas las capas, incluidas las intermedias, donde no se tiene la salida deseada [21].

Resumen del algoritmo

1. Seleccionar el par (entrada-salida deseada) de entrenamiento y presentarlo a la red.
2. Obtener la salida de la red.
3. Calcular el error con respecto a la salida deseada.
4. Propagar el error hacia atrás (en sentido contrario al del paso 2) a través de los pesos, ajustándolos de forma que se minimice el error que tendría la misma entrada aplicada de nuevo [21].

8.1 Detalle del algoritmo básico [25]

Para poder calcular el error en las capas ocultas se necesita conocer el error de la capa siguiente y propagarlo hacia atrás. El objetivo es minimizar la función de error, donde las variables son todos los pesos de la red, pues son los que determinan la salida para una entrada dada. La estrategia sería la siguiente: mediante programación dinámica, se calculan las derivadas del error para una capa utilizando las calculadas para las activaciones de la capa siguiente (en dirección a la capa de salida). Al conocer estas derivadas, no resulta demasiado difícil obtener las correspondientes a los pesos.

Asumiendo que la función de error es el error cuadrático medio, que la función de activación de todas las neuronas es la función sigmoide y que estas no tienen *bias*, podemos calcular las derivadas de la función de error E en la capa de salida como

$$E = \frac{1}{2} \sum_{j \in \ salida} (\widehat{y}_j - y_j)^2 \Rightarrow \frac{\partial E}{\partial y_j} = -\left(\widehat{y}_j - y_j\right) \ , \tag{A.8}$$

donde y_j e $\widehat{y}_j$ son la salida y la salida deseada, respectivamente, de la neurona j.

Para obtener las derivadas del error para la capa anterior, en la que se ubica la neurona i (ver Figura A.4), deberíamos antes saber cómo afecta la salida de las neuronas de la capa i a los *logits* (salidas antes de la aplicación de la función de activación) de las neuronas de la

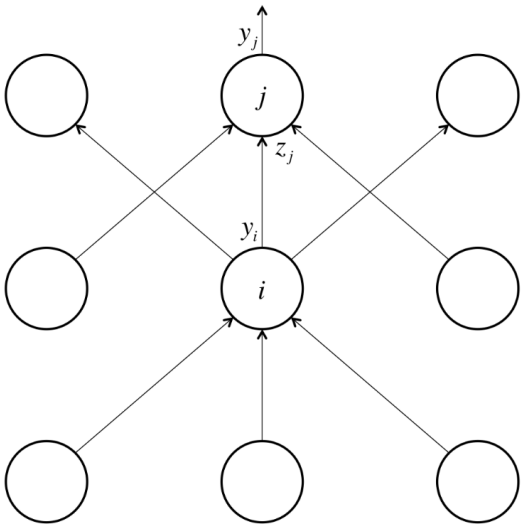


Figura A.4: Diagrama de referencia para la explicación del método de propagación hacia atrás [25].

capa j. Teniendo en cuenta que la derivada parcial del *logit* con respecto a la salida de la capa anterior (entrada en esta) es simplemente el peso de la conexión entre las neuronas i y j w_{ij}

$$\frac{\partial z_j}{\partial y_i} = \frac{\partial}{\partial y_i} \sum_i w_{ij} y_i = w_{ij} \ , \tag{A.9}$$

tenemos que lo anterior se puede obtener de la forma

$$\frac{\partial E}{\partial y_i} = \sum_j \frac{\partial E}{\partial z_j} \frac{\partial z_j}{\partial y_i} = \sum_j w_{ij} \frac{\partial E}{\partial z_j} \tag{A.10}$$

Además, podemos deducir:

$$\frac{\partial E}{\partial z_j} = \frac{\partial E}{\partial y_j} \frac{\partial y_j}{\partial z_j} = y_j \left(1 - y_j\right) \frac{\partial E}{\partial y_j} \ , \tag{A.11}$$

puesto que:

$$y = \frac{1}{1 + e^{-z}} \Rightarrow \frac{dy}{dz} = \frac{e^{-z}}{\left(1 + e^{-z}\right)^2} = \cdots = y\left(1 - y\right) \ . \tag{A.12}$$

Si juntamos las ecuaciones A.10 y A.11, podemos expresar al fin las derivadas del error en la capa i en términos de las derivadas del error en la capa j:

$$\frac{\partial E}{\partial y_i} = \sum_j w_{ij} y_j \left(1 - y_j\right) \frac{\partial E}{\partial y_j} \ . \tag{A.13}$$

Una vez que se han obtenido las derivadas parciales de la función de error con respecto

a las activaciones de las neuronas, ya se puede saber cómo cambia el error en función de los pesos, lo que nos dice, al fin y al cabo, cómo han de ser ajustados para minimizar el error. Esto nos daría cómo modificar los pesos después de cada ejemplo de entrada:

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial z_j}{w_{ij}} \frac{\partial E}{\partial z_j} = y_i y_j \left(1 - y_j\right) \frac{\partial E}{\partial y_j} \, . \tag{A.14}$$

La fórmula final de modificación de los pesos una vez vistos todos los ejemplos de entrada del conjunto de datos sería la siguiente:

$$\Delta w_{ij} = - \sum_{k \in \, dataset} \alpha \, y_i^{(k)} y_j^{(k)} \left(1 - y_j^{(k)}\right) \frac{\partial E^{(k)}}{\partial y_j^{(k)}} \, , \tag{A.15}$$

donde $E^{(k)}$ es la función de error para el ejemplo k e $y_i^{(k)}$ e $y_j^{(k)}$ las salidas de las neuronas i y j para el ejemplo k.

9 Optimizador estocástico *Adam*

Adam es un algoritmo presentado en [88] y enfocado a la optimización de objetivos estocásticos con espacios de parámetros de alta dimensionalidad (e.g. función de error a minimizar cuyos parámetros son los pesos) que solo requiere gradientes de primer orden para cumplir su función.

Algoritmo 1: *Adam*

Requiere: α: tamaño de paso (*stepsize*)
Requiere: $\beta_1, \beta_2 \in [0, 1)$: ratio de decaimiento exponencial para las estimaciones de momento
Requiere: $f(\theta)$: función estocástica objetivo con parámetros θ
Requiere: $f\theta_0$: vector de parámetros inicial
$m_0 \leftarrow 0$ (Inicializa el 1er vector de momento)
$v_0 \leftarrow 0$ (Inicializa el 2º vector de momento)
$t \leftarrow 0$ (Inicializa el paso de tiempo)
mientras θ_t *no converja* **hacer**
  $t \leftarrow t + 1$
  $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Obtiene los gradientes para el paso de tiempo t)
  $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$
  $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$
  $\widehat{m}_t \leftarrow m_t / \left(1 - \beta_1^t\right)$
  $\widehat{v}_t \leftarrow v_t / \left(1 - \beta_2^t\right)$
  $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \widehat{m}_t / \left(\sqrt{\widehat{v}_t} + \epsilon\right)$ (Actualiza los parámetros)
fin
devuelve θ_t (Parámetros resultado)

Algoritmo

En el algoritmo 1 se puede ver el pseudocódigo del algoritmo *Adam*. El objetivo final es el de minimizar el valor esperado de $f(\theta)$, una función estocástica escalar (función objetivo), $\mathbb{E}[f(\theta)]$, con respecto a sus parámetros, θ.

El valor esperado es el valor medio que resultaría si se repitiera el mismo experimento un elevado número de veces. Es decir, lo que se trata es de minimizar la función objetivo no para un caso concreto, sino para todos.

El gradiente, según la notación seguida, es $g_t \leftarrow \nabla_\theta f_t(\theta)$, y se refiere a las derivadas parciales de f_t con respecto a sus parámetros θ evaluadas en el instante de tiempo t.

El algoritmo actualiza las medias móviles exponenciales del gradiente (m_t) y del gradiente al cuadrado (v_t) mediante los hiperparámetros β_1 y β_2, que controlan el ratio de decaimientode las medias. Estas dos medias son estimaciones del 1er momento (media), y del 2º momento (varianza no centrada: varianza a la que no se le resta la media [89]). Como ambas se inicializan a vectores 0, están en cierto modo condicionadas en favor de ese número, por lo que se hace un ajuste que corrige ese sesgo, dando lugar a las estimaciones $\widehat{m}_t$ y $\widehat{v}_t$.

El término ϵ se añade al denominador en la actualización de parámetros para mejorar la

estabilidad numérica del método [90] (evitando divisiones por cero [91]). Por otra parte, g_t^2 representa el cuadrado elemento a elemento de g_t, $g_t \odot g_t$, mientras que β_n^t tiene su significado habitual: β_n elevado a t.

Hay que tener en cuenta que el algoritmo 1 no es la versión más eficiente posible, pero se muestra así por claridad.

10 Criterio de parada de los algoritmos de gradiente descendente

Puede haber diferentes alternativas que sirvan de criterio de parada de los algoritmos de gradiente descendente. Algunas de las más utilizadas son el entrenamiento durante un número de épocas fijo, o hasta que el error, evaluado en un *conjunto de validación*, no mejore durante un determinado número de épocas (paciencia).

En la primera, se dejan de actualizar los pesos dando por terminado el entrenamiento cuando todos los datos del conjunto de datos han sido presentados n veces (n épocas). En la segunda, hay que definir un conjunto de datos para validación diferente del de entrenamiento. Cada cierto número de épocas (típicamente, aunque se pueden definir los intervalos en base a otros aspectos, e.g. número de imágenes presentadas), se hace un test del comportamiento de la red sobre ese conjunto de validación, obteniendo un valor de error. Este valor se guarda (error de validación) en una lista. Cuando han pasado n épocas (paciencia) en las que no se mejora el error de validación, se para el entrenamiento. Finalmente, nos quedamos con la red cuyo valor de error en validación sea el más bajo de todos los almacenados.

Como se ha dicho, estas son dos de las alternativas más empleadas, pero se pueden definir otros criterios de parada diferentes y, dependiendo del problema y de su naturaleza, pueden interesar unas u otras opciones.

11 Funciones de error (*loss*)

Se hizo antes referencia al papel de las funciones de error en el proceso de entrenamiento de las redes neuronales. Aquí se presentan las funciones de error empleadas en los experimentos de este trabajo: *binary cross-entropy* (BCE), y *Structural Similarity* (SSIM) *index*.

11.1 *Binary Cross-Entropy*

El *Binary Cross-Entropy* (BCE) es un caso particular de *Cross-Entropy* con solo dos clases. Se utiliza tradicionalmente en problemas de clasificación, donde se suele asumir que hay una cierta distribución de probabilidad óptima de la clase correcta para cada ejemplo de entrada.

Matemáticamente, se puede definir la función de *loss* de BCE $BCELoss$ para la salida y_i y salida deseada $\widehat{y}_i$ de una clase i como

$$BCELoss\left(y, \widehat{y}\right) = -\left[\widehat{y}\, log\left(y\right) + \left(1 - \widehat{y}\right)\, log\left(1 - y\right)\right] \,, \tag{A.16}$$

donde C es el número total de clases, en este caso 2 ($C = 2$).

Además, a veces es posible definir un peso concreto para cada elemento de un lote de tamaño N modificando la función de la forma

$$\begin{aligned} BCELoss\left(y, \widehat{y}\right) &= L = \left\{l_1, l_2, \ldots, l_N\right\}^\top \,, \\ l_n &= -w_n \left[\widehat{y}\, log\left(y\right) + \left(1 - \widehat{y}\right)\, log\left(1 - y\right)\right] \,, \end{aligned} \tag{A.17}$$

11.2 *Structural Similarity Index*

Structural Similarity Index (SSIM) fue presentado en [92] y es un método creado específicamente para medir la calidad *percibida* de una imagen. Normalmente, se utiliza como métrica para la evaluación de super-resolución y síntesis o reconstrucción de imágenes. Para poder aplicarlo son necesarias una imagen objetivo (considerada de calidad perfecta) y una imagen a evaluar, de la que se obtendrá la medida de calidad [92]. Globalmente, está dividido en tres fases comparativas basadas en luminancia, contraste y estructura. Habitualmente se utilizan ventanas o parches (*patches*, regiones normalmente rectangulares) de las imágenes, no las imágenes enteras.

Asumiendo señales discretas, se estima la **luminancia** de una imagen como la media de la intensidad, y el **contraste** como la desviación típica (raíz cuadrada de la varianza). Con estas estimaciones, SSIM se define como

$$SSIM(\mathbf{x}, \mathbf{y}) = \frac{\left(2\mu_{\mathbf{x}}\mu_{\mathbf{y}} + C_1\right) + \left(2\sigma_{\mathbf{xy}} + C_2\right)}{\left(\mu_{\mathbf{x}}^2 + \mu_{\mathbf{y}}^2 + C_1\right) + \left(\sigma_{\mathbf{x}}^2 + \sigma_{\mathbf{y}}^2 + C_2\right)} \,, \tag{A.18}$$

donde $\mu_{\mathbf{x}}$ y $\mu_{\mathbf{y}}$ son las medias locales de las regiones $\mathbf{x}$ e $\mathbf{y}$ de dos imágenes, $\sigma_{\mathbf{x}}$ y $\sigma_{\mathbf{y}}$ sus desviaciones típicas locales y $\sigma_{\mathbf{xy}}$ la covarianza local entre $\mathbf{x}$ e $\mathbf{y}$. Esta métrica no suele utilizarse de forma global, sino localmente en varias regiones o ventanas de las imágenes, de forma que se obtiene un valor de similaridad por cada uno de ellos. La combinación de esos valores se puede hacer mediante de varias formas; algunas de ellas son calcular la media o hacer la suma. Como SSIM es una métrica de similaridad, para utilizarlo como una función de error a minimizar se utiliza $-SSIM$.

12 Algunos problemas típicos del entrenamiento de ANNs

El entrenamiento de las redes de neuronas, como se ha visto, es un proceso que depende de muchos factores. El conjunto de datos, el criterio de parada o las funciones de error son solo algunos de los más determinantes.

Cuando alguno de ellos no es adecuado, surgen problemas que pueden ser detectados a través de la evaluación de la red a lo largo del entrenamiento. Por ello, se utilizan las curvas de aprendizaje, que representan gráficamente los valores de error en validación y en entrenamiento a lo largo de las épocas.

Aquí, aunque brevemente, haremos mención a algunos de los problemas más clásicos y cómo detectarlos a través de esas gráficas. En concreto hablaremos del sobreajuste (*overfitting*) y el subajuste o infraajuste (*underfitting*).

12.1 Sobreajuste

Se dice que una red «sobreajusta», y por tanto que tiene un problema de sobreajuste, cuando no es capaz de generalizar [27]. Esto es, cuando es capaz de encontrar buenas soluciones para los datos de entrenamiento pero esas soluciones no funcionan bien para los datos no vistos como los de los conjuntos de test y validación. Esto se traduce en grandes diferencias entre los errores en entrenamiento y en test o validación [35]. En las curvas de aprendizaje de la Figura A.5 se puede apreciar que se empieza a producir sobreajuste a partir de la época 150, aproximadamente. En estos casos, la red está «memorizando» los ejemplos de entrenamiento, no siendo capaz de encontrar los patrones relevantes en ellos y que sean extrapolables a otros ejemplos distintos.

Normalmente[8], esto sucede por uno o varios de los siguientes motivos: (1) la red escogida es demasiado compleja —tiene más *capacidad* de la necesaria para el problema— [26]; (2) se entrena durante demasiado tiempo —la red de forma natural tiende a ir reduciendo el error en en conjunto de entrenamiento, por lo que puede que aprenda a memorizar para reducirlo—; (3) cuando hay mucho ruido en los ejemplos de entrenamiento —información innecesaria para la consecución de la solución— [27]; y (4) cuando los ejemplos de los conjuntos no están bien escogidos. Por ejemplo: si en una tarea de clasificación, durante el entrenamiento, se utilizan muchas más imágenes de ciertas clases que de las demás, la red clasificadora tenderá a asignarle a las imágenes de test las etiquetas de esas clases sobrerrepresentadas.

Por las razones antes descritas, es importante definir un criterio de parada adecuado que asegure que la red no entrene de más, definir unos conjuntos de entrenamiento, validación y test con ejemplos adecuados, en los que haya la variabilidad suficiente en cada uno, y escoger

[8] No se contemplan aquí todos los casos, solo algunos de los más comunes.

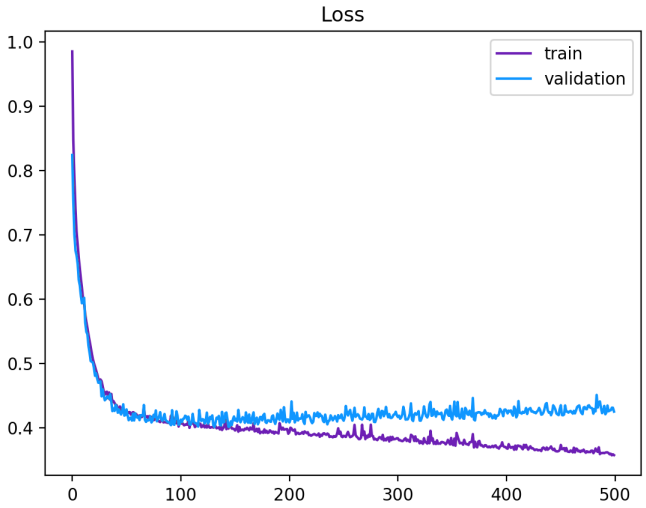


Figura A.5: Ejemplo de sobreajuste.

una arquitectura de red con una capacidad acorde a la naturaleza y complejidad del problema a abordar.

12.2 Subajuste

El subajuste, como se puede suponer, es el caso opuesto al del sobreajuste. Ocurre cuando la red no es capaz de obtener un bajo error en el conjunto de entrenamiento. Normalmente, se asocia esta situación a una capacidad de la red demasiado baja con respecto al problema a abordar. Otros factores que pueden conducir a que esto ocurra pueden ser una variablidad demasiado grande en el conjunto de entrenamiento —lo que hace difícil encontrar patrones que se puedan aprender— o la abundancia de ruido —mucha información sin interés, que evita el aprendizaje impidiendo a la red centrarse en lo verdaderamente relevante—.

Apéndice B

Redes de neuronas convolucionales

1 Operación de convolución

La operación de convolución, en su forma más general, es una operación sobre dos funciones continuas con un argumento de valor real. En nuestro caso, donde las imágenes son señales digitales bidimensionales y discretas, utilizaremos la definición discretizada para señales 2D

$$(I * K)(i,j) = \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} K\left[k,l\right] \cdot I\left[i-k, j-l\right], \tag{B.1}$$

donde I denota la imagen de entrada, K el *kernel* o filtro con el que se opera, de tamaño $M \times N$, e $(I * K)$ el resultado de la operación, donde $*$ representa el operador de convolución.

En la Figura B.1 se muestra un ejemplo de convolución con un *kernel* de 2×2. Intuitivamente, la convolución es el resultado de realizar el producto escalar del *kernel* desplazado e invertido con capa parche local de la imagen. Nótese que en la Ecuación B.1 el *kernel* se invierte con respecto a la imagen debido a la resta en la indexación. Esto se hace para que la operación sea conmutativa, en contraposición a la operación de correlación cruzada, que es equivalente pero sin inversión y no conmutativa.

El resultado de la operación de convolución puede interpretarse como un mapa con una fuerza de correspondencias (*matching*) entre el patrón representado en el *kernel* y cada posición de la imagen.

2 Capas convolucionales

Una capa convolucional está formada por un conjunto determinado de *kernels* o filtros, cuyos coeficientes serán ajustados durante el entrenamiento. El resultado de la convolución de los datos de la capa anterior con cada uno de estos filtros da lugar a un mapa de características 2D. Cada uno de estos mapas (salida de cada neurona) forma un canal del tensor[1] de salida de la capa. Por lo tanto, para conectar capas convolucionales entre sí, o usar imágenes de entrada

[1] Los tensores son generalizaciones de escalares, vectores y matrices para un número arbitrario de índices [93, 94].

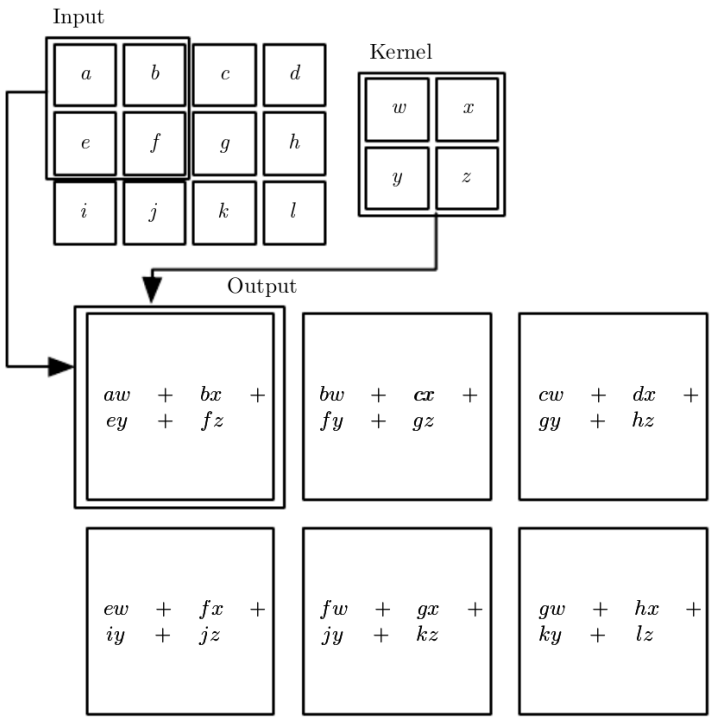


Figura B.1: Ejemplo de convolución sin inversión del *kernel* [26].

multicanal (como puede ser una imagen a color, RGB), es necesario utilizar convoluciones multicanal. Estas convoluciones se pueden definir como

$$h_{ijp}^{(q+1)} = \sum_{r=-a}^{a} \sum_{s=-b}^{b} \sum_{k=0}^{d_q-1} w_{rsk}^{(p,q)} h_{i+r,j+s,k}^{(q)} \qquad a = \frac{F_q - 1}{2} \ , \ \ b = \frac{G_q - 1}{2}$$
$$\forall i \in \{a, \ldots, L_q - a\} \ \ , \ \ \forall j \in \{b, \ldots, B_q - b\} \ \ , \ \ \forall p \in \{0, \ldots, d_{q+1} - 1\} \tag{B.2}$$

donde $h_{ijk}^{(q)}$ denota los $L_q \times B_q \times d_q$ elementos del tensor de entrada de la capa q; $h_{ijp}^{(q+1)}$ denota los $(L_q - F_q) \times (B_q - G_q) \times d_{q+1}$ elementos del tensor de salida; y $w_{rsk}^{(p,q)}$ denota los $F_q \times G_q \times d_q$ elementos del tensor de pesos asociados al filtro p en la capa q.

En este caso, los filtros no solo realizan una combinación lineal de los píxeles adyacentes, sino también de todos los valores para los canales de entrada en cada punto. Los filtros utilizados suelen ser no muy grandes, cuadrados, y de tamaño impar para que haya un elemento central. Además siempre deben tener la misma profundidad que el número de canales de salida de la capa anterior [35]. El número de características de los mapas viene determinado por el número de filtros utilizados (determinan la profundidad del mapa) y por la dimensión espacial de la entrada (determinan el alto y ancho del mapa), tal y como se puede ver en la

Figura B.2. De esta manera se conoce de antemano el número de canales de entrada y salida de cada capa, pero su dimensión dependerá de la dimensión espacial de la entrada. Esto permite aplicar estas capas a imágenes de entrada de tamaño arbitrario.

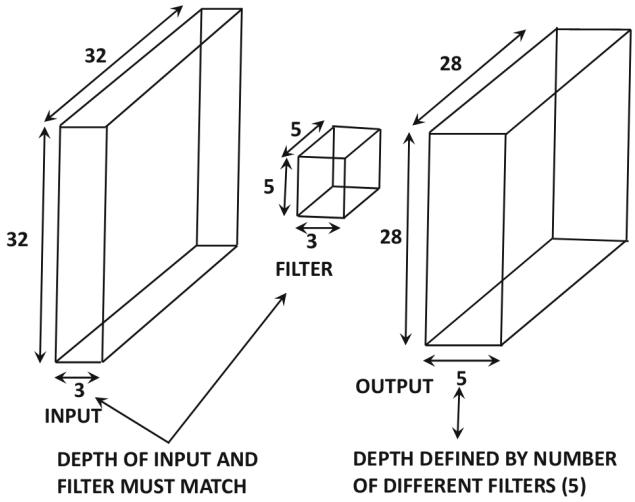


Figura B.2: Ejemplo de entrada, filtro y salida en una convolución [35].

Nótese que el resultado de la convolución definido en la Ecuación B.2 es de menor tamaño a lo alto y a lo ancho que la entrada. Esto se debe a que el filtro no se puede aplicar en las zonas de bordes. Para solucionarlo, es común añadir píxeles (*padding*) en los bordes y así mantener la *huella espacial* [35] (mismos ancho y alto) después de aplicar la convolución.

Por otro lado, es común que la convolución se aplique a todos los lugares de la imagen. Sin embargo, en ocasiones es conveniente aplicarla en posiciones alternas de la imagen, dejando un cierto «espacio» (*stride*) entre los resultados adyacentes. Esto se conoce como *strided convolution* o convolución *de paso*. Si el tamaño del paso es 2 (convolución **de paso 2**), entonces se deja un espacio de 1 píxel entre cada resultado de convolución; si es de 3, se dejan 2; y así sucesivamente. Esto nos permite reducir la huella espacial de la entrada.

3 Ventajas de las redes de neuronas convolucionales

Algo que todavía no se ha mencionado es qué representan, dentro de la lógica vista hasta ahora de redes neuronales, los filtros y los mapas de características. Los filtros son los pesos que se ajustarán durante el entrenamiento, y los mapas de características, los *logits*, es decir, los valores resultado de las neuronas antes de aplicar las funciones de activación. Así, cada una de las posiciones de los mapas de características representa la salida de una neurona. La Figura B.3 ilustra bien estos papeles dentro de la red.

Por otro lado, el funcionamiento de las funciones de activación no difiere del visto para

las redes clásicas. La función se aplica sobre cada valor de una entrada dada de $L_q \times B_q \times d_q$ y devuelve una salida del mismo tamaño compuesta por los valores resultado en cada caso (correspondencia 1 a 1).

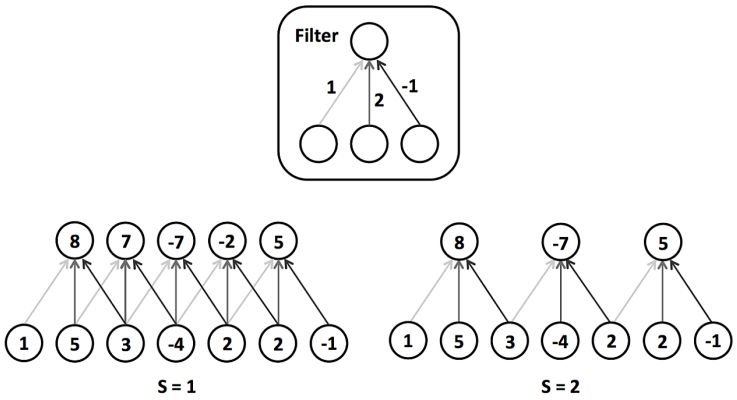


Figura B.3: Ejemplo de convolución. Aquí las neuronas serían las características y los pesos serían el propio filtro, que se iría desplazando sobre la capa [25].

Hay tres aspectos clave que le confieren a las redes de neuronas convolucionales importantes ventajas en algunos dominios de aplicación con respecto a otros sistemas de aprendizaje máquina: conectividad local, parámetros compartidos y representaciones equivariantes a la traslación.

Conectividad local Mientras que las redes vistas hasta ahora son completamente conectadas, al interactuar todas las neuronas de una capa con las de la siguiente, en las CNNs las interacciones tienen un carácter local. Esto es debido a que el resultado de cada punto de la imagen solo depende de las posiciones de entrada en la vecindad definida por el kernel, que es de menor tamaño espacial que la entrada (ver ejemplo en Figura B.3).

Parámetros compartidos Los pesos de los filtros afectan a la salida de múltiples neuronas, pues cada una de las posiciones de cada mapa de características se considera una salida neuronal.

Así, a la hora del aprendizaje, la red no aprende los parámetros para cada lugar concreto, sino un conjunto de ellos válido para todos los lugares a la vez, lo que reduce considerablemente el número de parámetros libres [26].

Representaciones equivariantes a la traslación Debido a la particularidad de la conectividad local y a la compartición de parámetros, la convolución presenta una propiedad llamada equivarianza a la traslación [26]. Esto es, que cuando se hace una traslación de los píxeles de

la entrada en cualquier dirección, en el resultado de aplicar sobre ella la convolución se podrá apreciar también esta traslación (si se compara con el resultado cuando la traslación no existe) [35].

Estos aspectos, a parte de influir en la eficiencia del modelo, tienen utilidad para restringir las posibles soluciones en aplicaciones concretas. Por ejemplo, si se da el caso de que ciertos detalles (e.g. bordes horizontales, un objeto) son relevantes en una imagen, es probable que lo sean independientemente de dónde aparezcan. Por otro lado, si este objeto se mueve, es útil que su representación también se mueva en la salida [26]. Por último, pese a que puede que una imagen tenga millones de píxeles , la conectividad local permite restringir las soluciones encontradas a aquellas que tienen en cuenta únicamente relaciones entre píxeles cercanos.

4 *Pooling*

La operación de *pooling*, o de votación local, permite simplificar un mapa de activaciones dado. Para ello, se integran los valores de cada región local mediante el cálculo del valor máximo, mínimo, media, etc. que represente al conjunto de activaciones en la región.

Cuando la operación de $pooling$ se hace mediante la selección del valor máximo, se conoce como *max-pooling*, siendo esta la modalidad más habitualmente utilizada en redes convolucionales. Al igual que con las capas convolucionales, las capas de *pooling* pueden utilizar *strides*, siendo común que el paso sea igual al tamaño del *pooling*.

La principal utilidad del *pooling*, además de la potencial adición de no linearidad, es la incorporación de cierta invarianza a pequeñas traslaciones. Esto es interesante ya que habitualmente es relevante detectar la presencia de ciertas características en una región, pero no tanto su posición concreta [26].

5 Problemas de desvanecimiento y explosión de gradientes

Durante el entrenamiento con métodos de gradiente descendente es común encontrarse con problemas de desvanecimiento[2] y explosión de gradientes [3]. Estos problemas son precisamente los que motivaron el desarrollo de algunos de los métodos aquí descritos, como Glorot y He [27]. El **desvanecimiento de gradientes** consiste en la casi nula modificación de los pesos en las *capas bajas* (aquellas cercanas a la capa de entrada) debido a que la reducción progresiva de los gradientes en la propagación hacia atrás del error termina por producir valores de modificación de pesos extremadamente pequeños en esas capas. Por otro lado, la **explosión de gradientes** provoca valores inestablemente altos de modificación de pesos en las

[2] A veces también denominado de «decaimiento».
[3] En inglés, *vanishing/exploding gradients problem*.

capas bajas de la red debido al aumento progresivo de los gradientes en la retropropagación del error.

En ambos casos, las redes no serían capaces, durante el entrenamiento, de obtener una configuración de parámetros satisfactoria. En el primer caso, porque la no modificación de los pesos en las capas afectadas evitaría la convergencia **a una solución buena**. En el segundo, porque las modificaciones son demasiado grandes, provocando oscilaciones que impiden la convergencia de la red.

6 Inicializaciones y aprendizaje transferido

En la primera iteración del entrenamiento de una red de neuronas, los pesos —que en CNNs definen los **filtros**— todavía no se han sometido a ningún ajuste. Para los algoritmos de optimización basados en gradiente descendente el punto de comienzo (los pesos iniciales) resulta clave, pues condicionará el camino tomado durante la minimización de la función de error para los ejemplos del entrenamiento. Así, se deduce que los valores iniciales de los pesos, aquellos que se le asignan antes de empezar el entrenamiento, son claves para la obtención de una red que satisfaga los requerimientos de la tarea para la que se entrena.

Para obtener estos valores iniciales, dado que por número y por nivel de abstracción es absolutamente imposible definirlos a mano de forma adecuada, es común en la actualidad seguir dos estrategias diferentes: usando métodos de inicialización automática (normalmente aleatoria) adaptados a la arquitectura de la red —conexiones, tipo de capas, etc.—; y cogiendo los peso de un modelo ya entrenado (*preentrenado*) para una tarea afín.

6.1 Métodos de inicialización automáticos

Como se puede suponer, hay muchas opciones diferentes; cada una adecuada para unos u otros escenarios y con sus propias ventajas e inconvenientes. A pesar de todo, hay algunas consideradas clásicas: inicialización constante y aleatoria directa; y otras estrategias más avanzadas que a lo largo de los últimos años han demostrado una mayor adecuación y suelen estar entre los métodos más utilizados: Glorot y He.

Con la **inicialización constante**, todos los pesos de la red son inicializados a un valor constante C, típicamente 0 o 1. En la actualidad prácticamente no se utiliza, al menos como inicialización integral de todos los pesos de la red.

Una alternativa mucho más común es la **aleatoria directa**, que consiste en dar valor a los pesos de forma aleatoria y de acuerdo a una distribución determinada.

Una de las distribuciones más habituales es la distribución uniforme entre unos valores mínimo y máximo, de forma que todos los valores situados en ese rango tienen la misma probabilidad de ser escogidos.

Otra también frecuente es la distribución normal con media $\mu = 0$ y desviación típica $\sigma = 1$.

Es frecuente ver estas dos incializaciones, constante y aleatoria directa, aplicadas dentro de estrategias de inicialización más complejas que hacen uso de heurísticas relacionadas con la información arquitectónica de las redes.

Inicialización de Glorot

La inicialización de Glorot, también conocida como «Xavier» [95][4], inicializa los pesos de forma aleatoria, pero usando parámetros calculados de antemano a partir del tamaño de las capas. Por defecto se usa la distribución uniforme $\mathcal{U}$, aunque es también posible usar distribución normal, $\mathcal{N}$. Matemáticamente, se define así:

$$W_{j+1} \sim \mathcal{U}\left[-\frac{\sqrt{6}}{\sqrt{n_j + n_{j+1}}}, \frac{\sqrt{6}}{\sqrt{n_j + n_{j+1}}}\right] , \tag{B.3}$$

donde n_j denota el tamaño de la capa j. Es decir, que n_j y $n_j + 1$ denotan el número de entradas y salidas, respectivamente, de la capa a inicializar. Esta inicialización parte de la idea de que para que una señal fluya de forma adecuada a lo largo de la red la varianza de las entradas de una capa debe ser igual a la de sus salidas; con los gradientes, en las dos direcciones (pasos hacia adelante y hacia atrás) debería suceder lo mismo.

Esta inicialización ha reportado mejores resultados de entrenamiento que las aproximaciones clásicas, logrando evitar en la mayoría de casos el problema de desvanecimiento o explosión de gradientes. Por ello, se ha convertido en una de las más utilizadas en arquitecturas de red con funciones de activación del tipo tangente hiperbólica [95].

Inicialización de He

La inicialización de He [83], a la que veces se llama inicialización de «Kaiming»[5], es una inicialización orientada a redes profundas con funciones de activación de tipo ReLU o semejantes, como PReLU. He inicializa los pesos de acuerdo a una distribución normal de la forma

$$W_{j+1} \sim \mathcal{N}\left(0, \sqrt{\frac{2}{n_j}}\right) , \tag{B.4}$$

donde n_j es el número de neuronas de la capa anterior, esto es, el número de entradas de la capa a inicializar. Además, los *bias* se inicializan a 0 ($\mathbf{b} = 0$). También hay una versión que usa una distribución uniforme definida igual que la vista para Glorot pero sin sumar el número de neuronas n_{j+1}.

[4] Por ser Xavier Glorot su autor principal.
[5] Por ser Kaiming He su autor principal.

Esta inicialización ha demostrado ser mejor que la de Glorot cuando se tienen redes de notable profundidad y que utilizan funciones de activación ReLUs y/o variantes [83]. No obstante, la mayor parte de las veces, cuando la función de activación utilizada es la de tangente hiperbólica, Glorot sigue siendo más adecuada.

Inicialización de LeCun

La inicialización de LeCun [96] está orientada a redes con la función de activación sigmoide. Matemáticamente, se describe así:

$$W_{j+1} \sim \mathcal{U}\left[-\frac{1}{\sqrt{n_j}}, \frac{1}{\sqrt{n_j}}\right] , \tag{B.5}$$

donde n_j es el número de neuronas de la capa anterior, esto es, el número de entradas de la capa a inicializar.

6.2 Modelos preentrenados: aprendizaje transferido

El aprendizaje transferido (*transfer learning*) es una técnica de aprendizaje máquina que consiste en la utilización de modelos de red preentrenados como puntos de partida para el entrenamiento de esos mismos modelos en otras tareas o conjuntos de datos. Es decir, primero se inicializan los parámetros del modelo usando los parámetros de un modelo entrenado previamente en una tarea o conjunto de datos afín a la tarea objetivo. Y después se *refina* este modelo preentrenado sobre el conjunto de datos objetivo. De esta forma, se busca aliviar la escasez de datos etiquetados para la tarea objetivo aprovechándose de la alta *reusabilidad* de las características extraídas para un conjunto de datos diferente pero relacionado [35].

Uno de los casos más frecuentes es el de utilizar modelos preentrenados en el conjunto de datos *ImageNet* [97], que consta de más de un millón de imágenes pertenecientes a 1000 clases distintas. Las características usadas en estos modelos se han demostrado de utilidad para gran cantidad de aplicaciones, y la inicialización usando el modelo preentrenado supone una ventaja frente a la inicialización aleatoria.

7 ImageNet

El ImageNet Large Scale Visual Recognition Challenge (ILSVRC), frecuentemente abreviado como Imagenet Challenge, es un *challenge* altamente relevante en el mundo de la visión artificial. La tarea principal[6] del mismo consiste en la clasificación y localización de objetos en imágenes entre un total de 1000 categorías (uno solo por imagen). Los conjuntos de datos son

[6]Hay otras también relevantes, pero nos centraremos aquí en la de mayor impacto.

creados a partir de la base de datos ImageNet [97]. El conjunto de entrenamiento se compone de un total de 1.2 millones de imágenes, el de validación de 50000 y el de test de 150000. Las mil categorías incluyen gran cantidad de objetos del mundo natural y también objetos artificiales, y entre ellas se pueden encontrar, por ejemplo, animales, vehículos u objetos del día a día.

El gran reto que supone esta tarea ha terminado por convertir a este *challenge* en uno de los más relevantes y de mayor interés en el campo de la visión por ordenador. Cada año, decenas de nuevas aproximaciones son presentadas, y algunas de ellas trascienden más allá de los resultados y llegan a ser empleadas íntegra o parcialmente en otros dominios, como es el caso de la mayor parte de las que veremos más adelante como AlexNet o ResNet.

Apéndice C

Arquitecturas: esquema general

En el Capítulo 4 dedicado a las arquitecturas FCDN, ENet y U-Net se ha facilitado el detalle de todas ellas partiendo de los artículos originales donde fueron presentadas en cada caso. En este apartado mostramos un resumen/esquema general en forma de tabla que permite ver rápidamente los elementos principales de cada una de ellas y las diferencias y similitudes que estas tienen entre sí. Además, se aportan aquí otros datos no mencionados antes que serán relevantes más adelante para la discusión de los resultados (e.g. el número de parámetros) y un breve comentario apuntando lo que se considera más relevante.

El Cuadro C.1 muestra cómo la arquitectura U-Net es, con diferencia, la que posee un mayor número de parámetros, seguida por FCDN y, ya a bastante más distancia, ENet. Estas dos últimas se ajustan mejor a la tendencia actual de menos parámetros usando más capas. Así, a pesar de que U-Net tiene muchos más parámetros, no tiene ni la mitad de capas convolucionales de las que tiene la menos profunda de las alternativas (FCDN56).

Por otro lado, ENet es la que mayor *diversidad* de operaciones incluye, mientras que U-Net, la que menos. A nivel de diseño, U-Net es la más sencilla, lo que no quiere decir que sea peor. Con este diseño sencillo, altamente jerárquico y con gran cantidad de parámetros, es probable que el contexto gane importancia, lo que suele ser beneficioso.

Otro detalle también interesante es la ausencia en ENet de *skip connections* entre las partes de submuestreo y sobremuestreo que sí hay en U-Net y FCDN. Esto puede dificultar a la red la obtención de una salida detallada, pues no se aprovechan, como sí hacen U-Net y FCDN, las características extraídas en el camino de submuestreo.

En relación al *Batch Normalization*, U-Net es la única que no lo menciona en el artículo original. En el caso del *dropout* sí lo menciona, pero muy brevemente sin dar detalle alguno. ENet, por su parte, utiliza *SpatialDropout* y es la única de todas que utiliza PReLU como función de activación.

	U-Net	FCDN56	FCDN67	FCDN103	ENet
Nº de capas convolucionales	23	56	67	103	89
Nº de parámetros	31 031 745	1 353 696	3 420 672	9 216 448	355 398
Tipos de convoluciones	normales traspuestas	normales traspuestas strided	normales traspuestas strided	normales traspuestas strided	normales dilatadas traspuestas asimétricas strided
Tipos de pooling	max	max	max	max	max max-unpooling
Batch Normalization	No	Sí	Sí	Sí	Sí
Dropout	No*	Sí	Sí	Sí	Sí (SpatialDropout)
Funciones de activación	ReLU	ReLU	ReLU	ReLU	PReLU
Skip connections (encoder-decoder)	Sí	Sí	Sí	Sí	No
Relación con otras arquitecturas	FCN, VGG	DenseNet U-Net	DenseNet U-Net	DenseNet U-Net	ResNet Inception

Cuadro C.1: Esquema general comparativo de las características arquitectónicas de U-Net, FCDN y ENet, con las características descritas en los artículos originales.

Apéndice D

Análisis ROC y PR

La evaluación cuantitativa a la que se ha hecho referencia evalúa las imágenes resultado con respecto a las segmentaciones de vasos creadas a mano por los expertos. Esta evaluación se hace a través de los análisis *Receiver Operator Characteristic* (ROC) y *Precision-Recall* (PR). Tanto el primero como el segundo utilizan umbrales variables para producir múltiples imágenes binarias de segmentación. Todos los resultados individuales obtenidos, que en nuestro caso serán además a lo largo de todo el conjunto de datos en el que se haga la evaluación, se añaden a las curvas ROC y PR.

Como se puede ver en el ejemplo de la Figura D.1, la curva ROC muestra el Ratio de Falsos Positivos (RFP) contra el Ratio de Verdaderos Positivos (RVP), mientras que la curva PR muestra la Precisión (P) contra el *Recall* (R). Para poder construir las curvas, todos estos valores necesitan obtenerse para cada umbral.

El RFP es el ratio de píxeles «no-vaso» que se han clasificado incorrectamente como vaso:

$$RFP = \frac{FalsosPositivos}{FalsosPositivos + VerdaderosNegativos} \tag{D.1}$$

El RVP es el ratio de píxeles «vaso» que se han clasificado correctamente como tal:

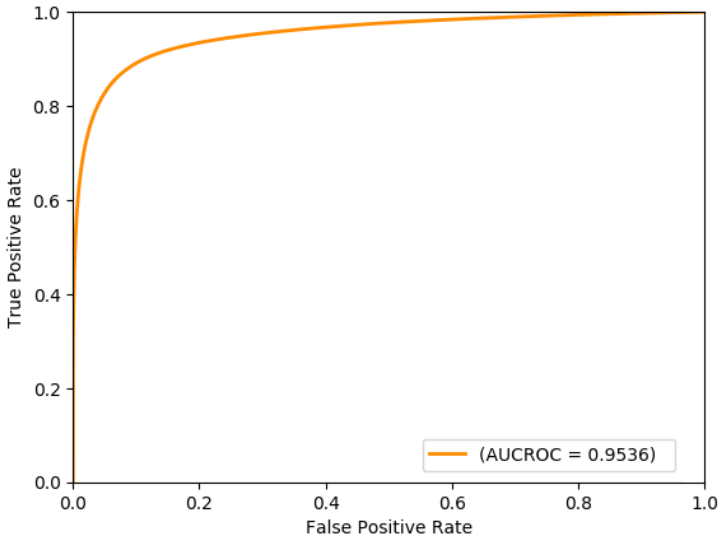
$$RVP = \frac{VerdaderosPositivos}{VerdaderosPositivos + FalsosNegativos} \tag{D.2}$$

La Precisión es el ratio de píxeles correctamente clasificados como «vaso»:

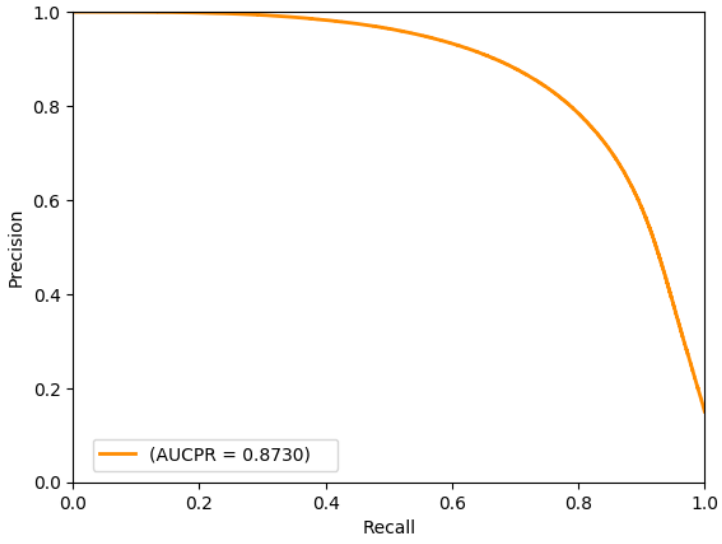
$$\textit{Precisión} = \frac{VerdaderosPositivos}{VerdaderosPositivos + FalsosPositivos} \tag{D.3}$$

El *Recall* es lo mismo que el RFP.

La información de las curvas ROC y PR se pueden integrar en el valor de AUC (*Area Under Curve*), con máximos y mínimos de 0 y 1, respectivamente, siendo el 1 el mejor resultado posible; es decir, aquel que coincide con la segmentación manual.



(a)



(b)

Figura D.1: Ejemplos de curvas ROC (a) y PR (b).

Apéndice E

Tablas de resultados cuantitativos en la tarea de segmentación de vasos

		DRIVE-test							
		1		5		10		15	
Arq.	Pre. \AUC(%)	ROC	PR	ROC	PR	ROC	PR	ROC	PR
FCDN56	Sí	97.30	89.28	97.55	90.13	97.47	89.96	97.23	89.29
	No	94.66	82.36	96.26	86.68	95.62	84.84	95.97	85.93
FCDN67	Sí	97.21	89.17	97.44	89.96	97.40	89.86	97.20	89.39
	No	93.95	81.63	96.57	87.41	96.73	87.95	96.55	87.57
FCDN103	Sí	97.14	89.11	97.37	89.84	97.50	90.29	97.47	90.17
	No	95.32	84.16	95.58	84.53	96.88	88.39	96.67	87.87
ENet	Sí	94.12	84.04	94.63	85.23	95.01	85.71	95.62	86.62
	No	87.95	65.01	92.16	74.98	95.00	83.67	94.05	80.75
U-Net	Sí	97.03	88.92	97.19	89.44	97.43	90.07	97.46	90.14
	No	95.76	85.71	96.60	88.01	96.98	88.86	96.95	88.78

Cuadro E.1: Resultados de la evaluación cuantitativa de la tarea de segmentación de vasos en DRIVE-test para las diferentes arquitecturas.

		STARE-AH							
		1		5		10		15	
Arq.	Pre. \AUC(%)	ROC	PR	ROC	PR	ROC	PR	ROC	PR
FCDN56	Sí	97.43	86.60	97.84	88.36	97.89	88.61	97.93	88.55
	No	93.95	76.55	94.98	79.32	94.81	78.35	94.57	78.20
FCDN67	Sí	96.63	85.12	97.48	87.68	97.66	88.08	97.53	87.56
	No	93.04	76.16	95.73	80.43	96.41	82.77	96.16	81.78
FCDN103	Sí	97.17	86.85	97.68	88.20	97.74	88.64	98.00	89.36
	No	92.97	76.12	95.74	79.53	96.88	83.96	96.91	84.30
ENet	Sí	95.49	83.03	95.79	83.8	95.87	84.1	96.64	85.59
	No	88.85	59.16	92.84	64.75	95.48	79.08	95.19	78.05
U-Net	Sí	97.53	87.19	97.86	88.85	98.27	90.39	98.34	90.51
	No	95.55	80.26	96.77	84.42	96.85	84.76	97.02	85.31

Cuadro E.2: Resultados de la evaluación cuantitativa de la tarea de segmentación de vasos en STARE para las diferentes arquitecturas.

		Imágenes (CdE)			
Arq.	Pre.	1	5	10	15
FCDN56	Sí	5400	9900	7500	4230
	No	6000	19860	12720	15090
FCDN67	Sí	4710	8520	6690	4800
	No	4650	15780	14580	14160
FCDN103	Sí	4110	7800	7770	6810
	No	6120	9570	16830	12990
ENet	Sí	23220	30900	31110	32820
	No	11910	11910	25530	20430
U-Net	Sí	4770	2700	3990	3840
	No	6090	11340	10710	10110

Cuadro E.3: Número de imágenes vistas durante el entrenamiento en la tarea de segmentación para cada una de las redes hasta la obtención de la mejor configuración.

Apéndice F

Glosario de siglas y acrónimos

ADN *Ácido desoxirribonucleico.*

AH *Adam Hoover.*

ANN *Artificial Neural Network*, red de neuronas artificiales.

AUC *Area Under Curve.*

BCE *Binary Cross-Entropy.*

BN *Batch Normalization*, normalización por lote.

CamVid *Cambridge-driving Labeled Video Database.*

CdE *Conjunto de Entrenamiento.*

CNN *Convolutional Neural Network*, red de neuronas convolucionales.

CVPRW *Conference on Computer Vision and Pattern Recognition Workshops.*

DB *Dense Block*, bloque denso.

DenseNet *Dense Network.*

DMAE *Degeneración Macular Asociada a la Edad.*

DRIVE *Digital Retinal Images for Vessel Extraction.*

ENet *Efficient Neural Network.*

Gatech *Georgia Institute of Technology.*

GD *Gradient Descent*, gradiente descendente.

FAZ *Foveal avascular zone*, zona avascular foveal.

FCN *Fully Convolutional Network.*

FC-DenseNet, FCDN *Fully Convolutional DenseNet.*

FOV *Field Of View.*

ISBI *International Symposium on Biomedical Imaging.*

MICCAI *Medical Image Computing and Computer Assisted Intervention.*

PR *Precision-Recall.*

PReLU *Parametric Rectified Linear Unit.*

ReLU *Rectified Linear Unit.*

RFP *Ratio de Falsos Positivos.*

ROC *Receiver Operating Characteristic.*

RVP *Ratio de Verdaderos Positivos.*

SGD *Stochastic Gradient Descent*, gradiente descendente estocástico.

SSIM *Structural Similarity.*

STARE *Structured Analysis of the Retina.*

TFG *Trabajo Fin de Grado.*

TD *Transition Down block*

TU *Transition Up block*

VARPA *Visión Artificial y Reconocimiento de Patrones.*

VGGNet *Visual Geometry Group Network.*

Apéndice G

Glosario de términos

Angiografía Técnica de imagen médica invasiva que permite obtener fotografías en blanco y negro del fondo de ojo con un especial resalte de los vasos sanguíneos.

Aprendizaje máquina Rama de las ciencias de la computación que busca la creación de programas informáticos capaces de extraer conocimiento de la experiencia o de ciertos volúmenes de datos.

Arquitectura (de red neuronal) Configuración determinada de la disposición y la conectividad entre neuronas, entradas y salidas en una red de neuronas artificiales.

Bastones Células fotorreceptoras de la retina responsables de la detección de luz tenue.

Batch Normalization Operación utilizada en redes de neuronas que normaliza las características de las capas ocultas de la red haciendo que tengan una varianza similar.

Campo receptivo Región de la entrada a la convolución que «cubre» o «ve» el filtro en cada paso.

Capa (en redes de neuronas) Conjunto usualmente alineado de neuronas con algún rasgo arquitectónico distintivo y uniforme para todas ellas.

Conos Células fotorreceptoras de la retina (especialmente concentrados en la fóvea) responsables de la percepción del color.

Convolución Operación sobre dos funciones continuas con un argumento de valor real en la cual están basadas las redes de neuronas convolucionales.

Convolución dilatada Tipo de convolución que soporta la expansión exponencial de los campos receptivos.

Convolución traspuesta Operación utilizada en redes de neuronas completamente convolucionales para hacer sobremuestreo.

Data augmentation Técnica aplicada en aprendizaje máquina consistente en aumentar el número de ejemplos de entrenamiento de forma artificial mediante variaciones y transformaciones plausibles sobre los ejemplos originales del conjunto de datos.

Disco óptico Nombre que recibe la zona de entrada del nervio óptico a la retina.

Dropout Técnica que se aplica en redes de neuronas y que busca evitar el sobreajuste mediante la desactivación de ciertas neuronas con una probabilidad p a lo largo del entrenamiento.

Fóvea Zona de la retina, situada en el centro de la mácula, con una especial concentración de conos (destinados principalmente a la percepción del color).

Gradiente Generalización de la función derivada a funciones de más de una variable (funciones multivariable).

Mácula Zona de la retina, situada aproximadamente en el centro de la parte posterior de la misma, donde se enfoca la luz. Es la parte más sensible de la retina, con mayor concentración de conos y bastones.

Neurona artificial Unidad computacional básica que realiza una suma ponderada de sus valores de entrada y aplica a esa suma un corte por umbral.

Oftalmología Área de la medicina centrada en el estudio las enfermedades del ojo y de su tratamiento.

Pooling Operación utilizada en redes de neuronas convolucionales para hacer sobremuestreo y que permite simplificar un mapa de activaciones integrando los valores de una región en un solo valor representativo.

Red de neuronas artificiales Agrupación de neuronas artificiales con un determinado esquema de conectividad.

Red de neuronas convolucional Tipo de red de neuronas artificiales basada en las ideas de conectividad local y compartición de parámetros, con la convolución como operación central, capaces de aprovechar la estructura de ciertas representaciones de datos como las imágenes.

Regularización, Técnicas de regularización Nombre que reciben algunas técnicas empleadas en redes de neuronas convolucionales (entre otros) que suelen tener por fin evitar el sobreajuste. Este tipo de técnicas busca o bien reducir la complejidad del problema (e.g. simplificando el modelo) o bien aumentar el número de datos disponibles para hallar su solución. Un ejemplo del primer tipo es *dropout*, que mediante la desactivación de neuronas de forma dinámica durante el entrenamiento crea en cierto modo redes más sencillas (menos parámetros para los que hallar un valor); y un ejemplo del segundo es *data augmentation*, que hace crecer el número de datos disponibles para la solución del problema.

Retina Superficie sensible situada en la superficie interior del ojo y formada principalmente por bastones, conos y neuronas.

Retinografía Técnica de imagen médica no invasiva que permite obtener fotografías en color del fondo de ojo.

Skip connection Conexión entre partes no sucesivas de una red de neuronas artificiales que permite saltar varias capas.

Sobreajuste Situación que se da en redes de neuronas artificiales (entre otros) por diversos motivos y que tiene como síntoma principal la incapacidad de las redes de generalizar.

Subajuste Situación que se da en redes de neuronas artificiales (entre otros) cuando estas no son capaces de obtener un bajo error durante el entrenamiento.

Unpooling Operación utilizada en redes de neuronas convolucionales para sobremuestreo y que actúa *a modo* de inversa al *pooling*.

Bibliografía

[1] O. Ronneberger, P.Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, ser. LNCS, vol. 9351. Springer, 2015, pp. 234–241, (available on arXiv:1505.04597 [cs.CV]). [Online]. Available: http://lmb.informatik.uni-freiburg.de/Publications/2015/RFB15a

[2] S. Jégou, M. Drozdzal, D. Vazquez, A. Romero, and Y. Bengio, "The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation," in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, July 2017, pp. 1175–1183.

[3] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "Enet: A deep neural network architecture for real-time semantic segmentation," *ArXiv*, vol. abs/1606.02147, 2017.

[4] R. Kawasaki, N. Cheung, J. Jin Wang, R. Klein, B. Ek Klein, M. F. Cotch, A. Richey Sharrett, S. Shea, F. Islam, and T.-Y. Wong, "Retinal vessel diameters and risk of hypertension: The multiethnic study of atherosclerosis," *Journal of hypertension*, vol. 27, pp. 2386–93, 09 2009.

[5] T. Walter, J. . Klein, P. Massin, and A. Erginay, "A contribution of image processing to the diagnosis of diabetic retinopathy-detection of exudates in color fundus images of the human retina," *IEEE Transactions on Medical Imaging*, vol. 21, no. 10, pp. 1236–1243, Oct 2002.

[6] M. C. P. Marín, *Óptica fisiológica. El sistema óptico del ojo y la visión binocular.* Universidad Complutense de Madrid, 2006.

[7] C. Starr, R. Taggart, C. Evers, and L. Starr, *Biology: The Unity and Diversity of Life*, 14th ed. Cengage Learning, 2016.

[8] G. D. Hildebrand and A. R. Fielder, "Anatomy and physiology of the retina," in *Pediatric Retina*, J. Reynolds and S. Olitsky, Eds. Springer-Verlag, 2011, ch. 2.

[9] H. Davson, "Aqueous humour and the intraocular pressure," in *Physiology of the Eye*, 4th ed. Academic Press, 1980, ch. 2.

[10] L. A. Remington, *Clinical Anatomy and Physiology of the Visual System*, 3rd ed. Butterworth-Heinemann, 2012.

[11] A. Burkov, *The Hundred-Page Machine Learning Book*, 1st ed. Andriy Burkov, 2019.

[12] T. M. Mitchell, *Machine Learning*. McGraw-Hill, 1997.

[13] A. C. Müller and S. Guido, *Introduction to Machine Learning with Python*, 1st ed. O'Reilly Media, 2016.

[14] F. Chollet, *Deep Learning with Python*, 1st ed. Greenwich, CT, USA: Manning Publications Co., 2017.

[15] S. Pouyanfar, S. Sadiq, Y. Yan, H. Tian, Y. Tao, M. P. Reyes, M.-L. Shyu, S.-C. Chen, and S. S. Iyengar, "A survey on deep learning: Algorithms, techniques, and applications," *ACM Comput. Surv.*, vol. 51, no. 5, pp. 92:1–92:36, Sep. 2018. [Online]. Available: http://doi.acm.org/10.1145/3234150

[16] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. van der Laak, B. van Ginneken, and C. I. Sánchez, "A survey on deep learning in medical image analysis," *Medical Image Analysis*, vol. 42, pp. 60 – 88, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1361841517301135

[17] L. Lu, Y. Zheng, G. Carneiro, and L. Yang, Eds., *Deep Learning and Convolutional Neural Networks for Medical Image Computing - Precision Medicine, High Performance and Large-Scale Datasets*, ser. Advances in Computer Vision and Pattern Recognition. Springer, 2017. [Online]. Available: https://doi.org/10.1007/978-3-319-42999-1

[18] M. I. Razzak, S. Naz, and A. Zaib, *Deep Learning for Medical Image Processing: Overview, Challenges and the Future*. Cham: Springer International Publishing, 2018, pp. 323–350. [Online]. Available: https://doi.org/10.1007/978-3-319-65981-7_12

[19] S. M. Anwar, M. Majid, A. Qayyum, M. Awais, M. Alnowami, and M. K. Khan, "Medical image analysis using convolutional neural networks: A review," *Journal of Medical Systems*, vol. 42, no. 11, p. 226, Oct 2018. [Online]. Available: https://doi.org/10.1007/s10916-018-1088-1

[20] N. Materials, "Ascent of machine learning in medicine," *Nature Materials*, vol. 18, no. 5, pp. 407–407, 2019. [Online]. Available: https://doi.org/10.1038/s41563-019-0360-1

[21] A. Moreno, E. Armengol, J. Béjar, L. Belanche, U. Cortés, R. Gavaldà, J. M. Gimeno, B. López, M. Martín, and M. Sànchez, *Aprendizaje automático*, 1st ed. Edicions UPC, 1994.

[22] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov 1998.

[23] Y. LeCun, C. Cortes, and C. J. Burges. (2019) The mnist database of handwritten digits. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[24] M. A. Nielsen, *Neural Networks and Deep Learning*. Determination Press, 2015. [Online]. Available: http://neuralnetworksanddeeplearning.com/

[25] N. Buduma and N. Locascio, *Fundamentals of Deep Learning: Designing Next-Generation Machine Intelligence Algorithms*, 1st ed. O'Reilly Media, Inc., 2017.

[26] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[27] A. Géron, *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow*, 2nd ed. O'Reilly Media, 2019.

[28] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS'12. USA: Curran Associates Inc., 2012, pp. 1097–1105. [Online]. Available: http://dl.acm.org/citation.cfm?id=2999134.2999257

[29] K. Fukushima, "Neural network model for a mechanism of pattern recognition unaffected by shift in position - Neocognitron," *Trans. IECE*, vol. J62-A(10), pp. 658–665, 1979.

[30] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85 – 117, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0893608014002135

[31] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *CoRR*, vol. abs/1207.0580, 2012, cite arxiv:1207.0580. [Online]. Available: http://arxiv.org/abs/1207.0580

[32] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014. [Online]. Available: http://dl.acm.org/citation.cfm?id=2627435.2670313

[33] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler, "Efficient object localization using convolutional networks," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 648–656.

[34] C. Szegedy, Wei Liu, Yangqing Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 1–9.

[35] C. C. Aggarwal, *Neural Networks and Deep Learning*. Springer International Publishing, 2018.

[36] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2015.

[37] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 770–778.

[38] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, "Deep networks with stochastic depth," in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 646–661.

[39] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ser. ICML'15. JMLR.org, 2015, pp. 448–456. [Online]. Available: http://dl.acm.org/citation.cfm?id=3045118.3045167

[40] S. Santurkar, D. Tsipras, A. Ilyas, and A. Mądry, "How does batch normalization help optimization?" in *Proceedings of the 32Nd International Conference on Neural Information Processing Systems*, ser. NIPS'18. USA: Curran Associates Inc., 2018, pp. 2488–2498. [Online]. Available: http://dl.acm.org/citation.cfm?id=3327144.3327174

[41] G. Huang, Z. Liu, L. v. d. Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017, pp. 2261–2269.

[42] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 3431–3440.

[43] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected crfs," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. [Online]. Available: http://arxiv.org/abs/1412.7062

[44] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. [Online]. Available: http://arxiv.org/abs/1511.07122

[45] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," *CoRR*, vol. abs/1603.07285, 2016. [Online]. Available: http://arxiv.org/abs/1603.07285

[46] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 818–833.

[47] Á. S. Hervella, J. Rouco, J. Novo, and M. Ortega, "Retinal image understanding emerges from self-supervised multimodal reconstruction," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2018*, A. F. Frangi, J. A. Schnabel, C. Davatzikos, C. Alberola-López, and G. Fichtinger, Eds. Cham: Springer International Publishing, 2018, pp. 321–328.

[48] X. Xu, J. Lin, Y. Tao, and X. Wang, "An improved densenet method based on transfer learning for fundus medical images," in *2018 7th International Conference on Digital Home (ICDH)*, Nov 2018, pp. 137–140.

[49] J. Chang, J. Yu, T. Han, H. Chang, and E. Park, "A method for classifying medical images using transfer learning: A pilot study on histopathology of breast cancer," in *2017 IEEE 19th International Conference on e-Health Networking, Applications and Services (Healthcom)*, Oct 2017, pp. 1–4.

[50] Y. Yu, H. Lin, J. Meng, X. Wei, H. Guo, and Z. Zhao, "Deep transfer learning for modality classification of medical images," *Information*, vol. 8, no. 3, 2017. [Online]. Available: https://www.mdpi.com/2078-2489/8/3/91

[51] S. P. K. Karri, D. Chakraborty, and J. Chatterjee, "Transfer learning based classification of optical coherence tomography images with diabetic macular edema and dry age-related macular degeneration," *Biomedical optics express*, vol. 8, no. 2, pp. 579–592, Jan 2017, 28270969[pmid]. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/28270969

[52] M. Raghu, C. Zhang, J. M. Kleinberg, and S. Bengio, "Transfusion: Understanding transfer learning with applications to medical imaging," *CoRR*, vol. abs/1902.07208, 2019. [Online]. Available: http://arxiv.org/abs/1902.07208

[53] N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway, and J. Liang, "Convolutional neural networks for medical image analysis: Full training or fine tuning?" *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1299–1312, May 2016.

[54] H. Ravishankar, P. Sudhakar, R. Venkataramani, S. Thiruvenkadam, P. Annangi, N. Babu, and V. Vaidya, "Understanding the mechanisms of deep transfer learning for medical images," in *Deep Learning and Data Labeling for Medical Applications*, G. Carneiro, D. Mateus, L. Peter, A. Bradley, J. M. R. S. Tavares, V. Belagiannis, J. P. Papa, J. C. Nascimento, M. Loog, Z. Lu, J. S. Cardoso, and J. Cornebise, Eds. Cham: Springer International Publishing, 2016, pp. 188–196.

[55] T. Birgui Sekou, M. Hidane, J. Olivier, and H. Cardot, "Retinal blood vessel segmentation using a fully convolutional network – transfer learning from patch- to image-level," in *Machine Learning in Medical Imaging*, Y. Shi, H.-I. Suk, and M. Liu, Eds. Cham: Springer International Publishing, 2018, pp. 170–178.

[56] S. Chen, K. Ma, and Y. Zheng, "Med3d: Transfer learning for 3d medical image analysis," *CoRR*, vol. abs/1904.00625, 2019. [Online]. Available: http://arxiv.org/abs/1904.00625

[57] J. Zhuang, "Laddernet: Multi-path networks based on u-net for medical image segmentation," *CoRR*, vol. abs/1810.07810, 2018, withdrawn. [Online]. Available: http://arxiv.org/abs/1810.07810

[58] M. Z. Alom, C. Yakopcic, M. Hasan, T. M. Taha, and V. K. Asari, "Recurrent residual U-Net for medical image segmentation," *Journal of Medical Imaging*, vol. 6, no. 1, pp. 1 – 16 – 16, 2019. [Online]. Available: https://doi.org/10.1117/1.JMI.6.1.014006

[59] Z. Zhang, Q. Liu, and Y. Wang, "Road extraction by deep residual u-net," *IEEE Geoscience and Remote Sensing Letters*, vol. 15, no. 5, pp. 749–753, May 2018.

[60] U. of Freiburg. U-net: Convolutional networks for biomedical image segmentation. [Online]. Available: https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/

[61] Z. Zhou, M. M. Rahman Siddiquee, N. Tajbakhsh, and J. Liang, "Unet++: A nested U-Net architecture for medical image segmentation," in *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, D. Stoyanov, Z. Taylor, G. Carneiro, T. Syeda-Mahmood, A. Martel, L. Maier-Hein, J. M. R. Tavares,

A. Bradley, J. P. Papa, V. Belagiannis, J. C. Nascimento, Z. Lu, S. Conjeti, M. Moradi, H. Greenspan, and A. Madabhushi, Eds. Cham: Springer International Publishing, 2018, pp. 3–11.

[62] H. Zhu, F. Shi, L. Wang, S.-C. Hung, M.-H. Chen, S. Wang, W. Lin, and D. Shen, "Dilated dense U-Net for infant hippocampus subfield segmentation," *Frontiers in neuroinformatics*, vol. 13, pp. 30–30, Apr 2019, 31068797[pmid]. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/31068797

[63] Y. Weng, T. Zhou, Y. Li, and X. Qiu, "NAS-Unet: Neural architecture search for medical image segmentation," *IEEE Access*, vol. 7, pp. 44 247–44 257, 2019.

[64] T. Falk, D. Mai, R. Bensch, Ö. Çiçek, A. Abdulkadir, Y. Marrakchi, A. Böhm, J. Deubner, Z. Jäckel, K. Seiwald, A. Dovzhenko, O. Tietz, C. Dal Bosco, S. Walsh, D. Saltukoglu, T. L. Tay, M. Prinz, K. Palme, M. Simons, I. Diester, T. Brox, and O. Ronneberger, "U-Net: deep learning for cell counting, detection, and morphometry," *Nature Methods*, vol. 16, no. 1, pp. 67–70, 2019. [Online]. Available: https://doi.org/10.1038/s41592-018-0261-2

[65] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3D U-Net: Learning dense volumetric segmentation from sparse annotation," in *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2016 - 19th International Conference, Athens, Greece, October 17-21, 2016, Proceedings, Part II*, 2016, pp. 424–432. [Online]. Available: https://doi.org/10.1007/978-3-319-46723-8_49

[66] G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla, "Segmentation and recognition using structure from motion point clouds," in *ECCV (1)*, 2008, pp. 44–57.

[67] S. H. Raza, M. Grundmann, and I. Essa, "Geometric context from video," *IEEE CVPR*, 2013.

[68] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[69] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, "Sun database: Large-scale scene recognition from abbey to zoo," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 2010, pp. 3485–3492.

[70] M. Kashefpur, R. Kafieh, S. Jorjandi, H. Golmohammadi, Z. Khodabande, M. Abbasi, A. Akbar Fakharzadeh, M. Kashefpoor, and H. Rabbani, "Isfahan misp dataset," *Journal of Medical Signals and Sensors*, vol. 7, pp. 43–8, 12 2016.

[71] Álvaro S. Hervella, J. Rouco, J. Novo, and M. Ortega, "Multimodal registration of retinal images using domain-specific landmarks and vessel enhancement," *Procedia Computer Science*, vol. 126, pp. 97 – 104, 2018, knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 22nd International Conference, KES-2018, Belgrade, Serbia. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050918311876

[72] J. Staal, M. Abramoff, M. Niemeijer, M. Viergever, and B. van Ginneken, "Ridge based vessel segmentation in color images of the retina," *IEEE Transactions on Medical Imaging*, vol. 23, no. 4, pp. 501–509, 2004.

[73] A. D. Hoover, V. Kouznetsova, and M. Goldbaum, "Locating blood vessels in retinal images by piecewise threshold probing of a matched filter response," *IEEE Transactions on Medical Imaging*, vol. 19, no. 3, pp. 203–210, March 2000.

[74] T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2018.

[75] Y. Zhang and Q. Yang, "A survey on multi-task learning," *CoRR*, vol. abs/1707.08114, 2017. [Online]. Available: http://arxiv.org/abs/1707.08114

[76] M. Gabrani and O. J. Tretiak, "Elastic transformations," in *Conference Record of The Thirtieth Asilomar Conference on Signals, Systems and Computers*, vol. 1, Nov 1996, pp. 501–505 vol.1.

[77] T. Meng, C. Wu, T. Jia, Y. Jiang, and Z. Jia, "Recombined convolutional neural network for recognition of macular disorders in sd-oct images," in *2018 37th Chinese Control Conference (CCC)*, July 2018, pp. 9362–9367.

[78] A. Govindaiah, M. A. Hussain, R. T. Smith, and A. Bhuiyan, "Deep convolutional neural network based screening and assessment of age-related macular degeneration from fundus images," in *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, April 2018, pp. 1525–1528.

[79] D. Doshi, A. Shenoy, D. Sidhpura, and P. Gharpure, "Diabetic retinopathy detection using deep convolutional neural networks," in *2016 International Conference on Computing, Analytics and Security Trends (CAST)*, Dec 2016, pp. 261–266.

[80] P. Prentašić and S. Lončarić, "Detection of exudates in fundus photographs using convolutional neural networks," in *2015 9th International Symposium on Image and Signal Processing and Analysis (ISPA)*, Sep. 2015, pp. 188–192.

[81] X. Li, T. Pang, B. Xiong, W. Liu, P. Liang, and T. Wang, "Convolutional neural networks based transfer learning for diabetic retinopathy fundus image classification," in *2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, Oct 2017, pp. 1–11.

[82] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ser. ICML'10. USA: Omnipress, 2010, pp. 807–814. [Online]. Available: http://dl.acm.org/citation.cfm?id=3104322.3104425

[83] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ser. ICCV '15. Washington, DC, USA: IEEE Computer Society, 2015, pp. 1026–1034. [Online]. Available: http://dx.doi.org/10.1109/ICCV.2015.123

[84] J. S. Bridle, "Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition," in *Neurocomputing*, F. F. Soulié and J. Hérault, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1990, pp. 227–236.

[85] M. T. Hagan, H. B. Demuth, M. H. Beale, and O. D. Jesús, *Neural Network Design*, 2nd ed. Martin T. Hagan, 2014.

[86] E. Alpaydın, *Introduction to Machine Learning*, 2nd ed. The MIT Press, 2010.

[87] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436 EP –, May 2015. [Online]. Available: https://doi.org/10.1038/nature14539

[88] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. [Online]. Available: http://arxiv.org/abs/1412.6980

[89] W. Marty, *Portfolio Analytics: An Introduction to Return and Risk Measurement*, 1st ed., ser. Springer Texts in Business and Economics. Springer International Publishing, 2013.

[90] PyTorch. (2019) Pytorch documentation: torch.optim. [Online]. Available: https://pytorch.org/docs/stable/optim.html

[91] MathWorks. (2019) Matlab documentation: Trainingoptionsadam. [Online]. Available: https://es.mathworks.com/help/deeplearning/ref/nnet.cnn.trainingoptionsadam.html

[92] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE TRANSACTIONS ON IMAGE PROCESSING*, vol. 13, no. 4, pp. 600–612, 2004.

[93] T. Rowland and E. W. Weisstein, "Tensor," *MathWorld–A Wolfram Web Resource*, 2019. [Online]. Available: http://mathworld.wolfram.com/Tensor.html

[94] J. Patterson and A. Gibson, *Deep Learning: A Practitioner's Approach*, 1st ed. O'Reilly Media, Inc., 2017.

[95] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, Y. W. Teh and M. Titterington, Eds., vol. 9. Chia Laguna Resort, Sardinia, Italy: PMLR, 13–15 May 2010, pp. 249–256. [Online]. Available: http://proceedings.mlr.press/v9/glorot10a.html

[96] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, *Efficient BackProp*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 9–48. [Online]. Available: https://doi.org/10.1007/978-3-642-35289-8_3

[97] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, June 2009, pp. 248–255.