Universitat Oberta de Catalunya

uoc.edu

Diagnóstico de miopía patológica en imágenes de fondo de ojo mediante aprendizaje profundo

Daniel López Robles
Grado en Ingeniería Informática
Área de Inteligencia Artificial

Joan M. Núñez Do Río
Carles Ventura Royo

19/06/2020

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Diagnóstico de miopía patológica en imágenes de fondo de ojo mediante aprendizaje profundo*
Nombre del autor:	*Daniel López Robles*
Nombre del consultor/a:	*Joan M. Núñez Do Río*
Nombre del PRA:	*Carles Ventura Royo*
Fecha de entrega (mm/aaaa):	06/2020
Titulación:	*Grado en Ingeniería Informática*
Área del Trabajo Final:	*Inteligencia Artificial*
Idioma del trabajo:	*Castellano*
Palabras clave:	*Miopía patológica, imágenes fondo de ojo, aprendizaje profundo*

Resumen del Trabajo (máximo 250 palabras): *Con la finalidad, contexto de aplicación, metodología, resultados i conclusiones del trabajo.*

La miopía es un problema de salud pública global. En su forma más grave, denominada miopía patológica, puede ocasionar secuelas graves, desembocando en una pérdida irreversible de visión. La complejidad del diagnóstico se ve agravada por el creciente envejecimiento de la población. La tasa de personas mayores de 60 años está creciendo el doble que la de profesionales de oftalmología. Por todo ello es fundamental potenciar el uso de nuevas tecnologías para un diagnóstico precoz de la miopía patológica.

La Inteligencia Artificial ofrece herramientas prometedoras para facilitar el diagnóstico clínico a través del aprendizaje profundo, subcampo del aprendizaje automático. Las redes neuronales convolucionales, especializadas en el procesamiento de imágenes, permiten tratar problemas complejos como la clasificación de imagen médica.

En este trabajo se exploran soluciones basadas en redes neuronales convolucionales para la detección de la miopía patológica en imágenes de fondo de ojo, incluyendo el uso de técnicas de transferencia de conocimiento con los modelos VGGNet, ResNet y GoogleNet. Los resultados alcanzados demuestran el potencial del aprendizaje profundo para contribuir en el diagnóstico precoz de la miopía patológica.

Abstract (in English, 250 words or less):

Myopia is a global public health problem. Its most severe form, called pathological myopia, can lead to irreversible loss of vision. The complexity of the diagnosis is aggravated by the emerging issues of population ageing around the world. The rate of people over 60 is growing twice as high as that of ophthalmology professionals. Therefore, it is essential to promote the use of new technologies for an early diagnosis of pathological myopia.

Artificial Intelligence provides promising tools to facilitate clinical diagnosis through Deep Learning. Convolutional Neural Networks, specialized in image processing, allow to address complex problems such as medical image classification.

This paper explores convolutional neural network-based solutions for detecting pathological myopia in fundus images, including the use of transfer learning with the VGGNet, ResNet and GoogleNet models. The results achieved demonstrate the potential of Deep Learning to contribute to the early diagnosis of pathological myopia.

Índice

Lista de figuras

Lista de tablas

1. Introducción

1.1 Contexto y justificación del Trabajo

La miopía es un defecto de visión que provoca la incapacidad de enfocar correctamente objetos lejanos. Está causada por un exceso de curvatura de la córnea, de potencia del cristalino o de distancia entre la parte delantera y la parte posterior del ojo. En la alta miopía o miopía magna esta distancia es mayor de 26,5 milímetros o el defecto de visión superior a las 6 dioptrías [1]. Como subgrupo más específico, la miopía patológica se define como alta miopía asociada a alguna patología posterior debida a la elongación de la distancia entre la parte delantera y la parte posterior del ojo [1]. Se trata de una afección grave, ya que puede provocar pérdida de visión irreversible.

La elevada incidencia de la miopía la convierte en un problema de salud global. Se estima que en 2050 prácticamente la mitad de la población mundial será miope [2]. En España, factores de estilo de vida están incrementando el riesgo de miopía en los niños, aumentando su prevalencia del 17% en 2016 al 20% en 2017 [3]. Además, la alta miopía es la principal causa de la afiliación a la Organización Nacional de Ciegos Españoles (ONCE) [4]. La forma más grave de miopía, la miopía patológica, ya afecta a un 3% de la población mundial [1] y es la cuarta causa más común de ceguera irreversible en los países desarrollados [5].

El diagnóstico temprano de la miopía patológica es importante desde un punto de vista clínico por su gravedad e incidencia. Además, supone un reto debido al creciente envejecimiento de la población. La tasa de población mayor de 60 años está creciendo el doble que la de profesionales de oftalmología [6]. Por todo ello, es crucial emplear avances tecnológicos que permitan automatizar tareas de diagnóstico.

Dentro del área de la Inteligencia Artificial, el aprendizaje computacional (*Machine Learning*) ofrece herramientas para el análisis automatizado de imágenes médicas. Esta tecnología permite desarrollar modelos predictivos capaces de clasificar imágenes en distintas categorías. Como parte del aprendizaje computacional, el aprendizaje profundo (*Deep Learning*) proporciona una herramienta adecuada para trabajar con imágenes: las redes neuronales convolucionales (*Convolutional Neural Network*, CNN). Se trata de un tipo de red neuronal especializada en el procesamiento de topologías cuadriculadas, tales como imágenes [7].

En el presente trabajo se estudiarán los conceptos teóricos relacionados con este problema y se implementarán diversas redes neuronales convolucionales para clasificar imágenes de fondo de ojo con y sin miopía patológica, con el fin

de comparar las distintas estrategias y sus resultados. Este trabajo se desarrollará utilizando el conjunto de datos ofrecido en *PALM: PAthoLogic Myopia Challenge* [8].

1.2 Objetivos del Trabajo

1.2.1 Objetivos generales

- G1. Investigar la disciplina del aprendizaje computacional con especial énfasis en las redes neuronales convolucionales.
- G2. Experimentar distintos métodos y técnicas para obtener modelos de aprendizaje computacional capaces de clasificar imágenes de fondo de ojo con miopía patológica y sin miopía patológica.

1.2.2 Objetivos específicos

- E1. Analizar literatura médica sobre miopía patológica.
- E2. Analizar literatura sobre aprendizaje computacional.
- E3. Analizar literatura sobre aprendizaje profundo.
- E4. Analizar el conjunto de datos.
- E5. Investigar y elegir un entorno de ejecución para desarrollar el código.
- E6. Implementar redes neuronales convolucionales para la clasificación de imágenes de fondo de ojo en dos categorías: con y sin miopía patológica.
- E7. Optimizar los modelos para lograr la combinación de hiperparámetros que mejores resultados ofrezcan.
- E8. Experimentar con la técnica del *transfer learning* para reutilizar modelos preentrenados de arquitecturas estándar de CNN.
- E9. Comparar resultados de los distintos modelos y experimentos.

1.3 Enfoque y método seguido

El proyecto se abordará y desarrollará desde dos perspectivas. Por un lado, la vertiente teórica, para contextualizar el problema e introducir los conceptos necesarios. Por otro lado, un apartado más práctico, en el que se aplicarán los conceptos teóricos.

Desde el punto de vista teórico, se estudiará el contexto del problema:

- Literatura médica sobre miopía patológica.
- *Machine Learning* y *Deep Learning* de manera genérica, con especial hincapié en *Convolutional Neural Networks*.

Desde el punto de vista práctico, el objetivo general que perseguimos es diseñar redes neuronales convolucionales para la clasificación de imágenes de fondo de ojo. Para ello, será necesario:

- Adquirir conocimientos sobre las tecnologías que utilizaremos para desarrollar el código: Python, TensorFlow, Keras.
- Analizar el juego de datos.
- Elaborar un *pipeline* funcional para el tratamiento de los datos y el entrenamiento y validación de los modelos.
- Aplicar técnicas de optimización para lograr la combinación de hiperparámetros que produzcan mejores modelos.

El código se desarrollará en el lenguaje de programación Python, utilizando la librería Keras sobre TensorFlow. Se decide utilizar TensorFlow frente a PyTorch por la mayor implantación en el mercado laboral del primero. Como ejemplo, en el momento de redactar esta memoria, la búsqueda por “pytorch” en el portal de empleo *indeed* ofrecía 29 resultados frente a los 87 del término “tensorflow”.

Para decidir entorno de ejecución se explora la oferta de servicios en nube pública de Microsoft Azure, Google Cloud Platform y Amazon AWS, pero no se localiza ningún servicio cuyo coste sea asumible para el proyecto. Se decide utilizar Google Colab por los siguientes motivos:

Cuadernos de Jupyter en la nube sin instalación ni configuración de ninguna clase.
Ofrece de forma gratuita recursos hardware con potencia que a priori debería ser suficiente para llevar a cabo el proyecto.
Se integra fácilmente con Google Drive, por lo que permitirá alojar el *dataset* en dicho servicio.

1.4 Planificación del Trabajo

La Figura 1 muestra el diagrama de Gannt de la planificación temporal del proyecto. En la parte izquierda se muestra la fecha de inicio y fin de cada tarea. En el gráfico de la derecha, cada columna corresponde a una semana, en gris el fin de semana.

Nombre	Fecha de inicio	Fecha de fin
PEC1. Plan de trabajo	3/03/20	15/03/20
Análisis inicial del trabajo	3/03/20	12/03/20
Elaboración del plan	13/03/20	15/03/20
Entrega PEC1	16/03/20	16/03/20
PEC2. Desarrollo del trabajo . Fase 1	17/03/20	15/05/20
Aprendizaje de TensorFlow y keras	17/03/20	15/05/20
Análisis del juego de datos	17/03/20	24/03/20
Análisis de contexto: literatura médica	23/03/20	29/03/20
Análisis de contexto: ML / DL	28/03/20	5/04/20
Análisis de contexto: DL en oftalmología	4/04/20	12/04/20
Investigar entornos de ejecución	11/04/20	16/04/20
Evaluar evolución del proyecto	17/04/20	19/04/20
Entrega PEC2	20/04/20	20/04/20
PEC3. Desarrollo del trabajo. Fase 2	21/04/20	17/06/20
Revisar y completar apartados teóricos	21/04/20	17/06/20
Conseguir pipeline funcional	21/04/20	19/05/20
Entrenamiento y validación de modelos	27/04/20	1/06/20
Evaluar evolución del proyecto	31/05/20	1/06/20
Entrega PEC3	1/06/20	1/06/20
PEC4. Redacción de la memoria	2/06/20	17/06/20
Redacción de la memoria del TFG	2/06/20	17/06/20
Entrega PEC4	17/06/20	17/06/20
PEC5. Entregas finales	17/06/20	28/06/20
Elaboración de la presentación	17/06/20	21/06/20
Entrega PEC5a	21/06/20	21/06/20
Defensa pública	22/06/20	28/06/20
Fin PEC5b	28/06/20	28/06/20

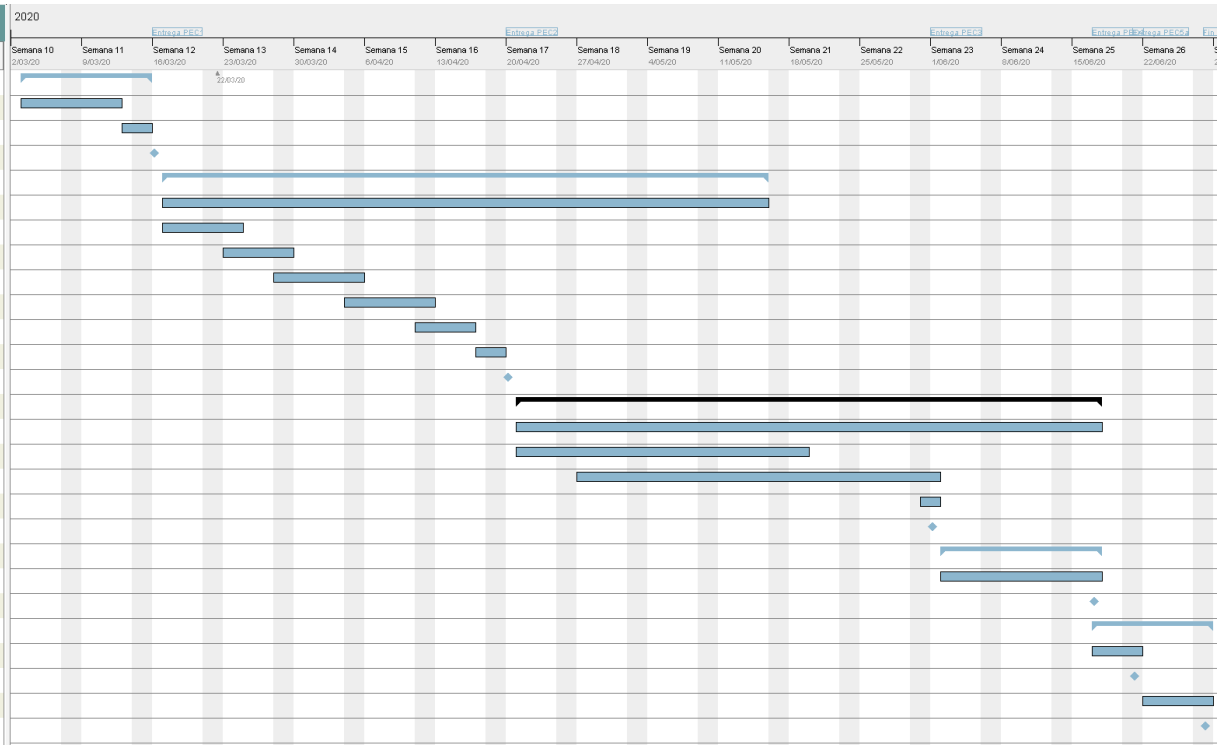

Figura 1 Planificación temporal

1.5 Breve sumario de productos obtenidos

El producto obtenido en este trabajo es la descripción de los distintos modelos desarrollados para clasificar imágenes de fondo de ojo con y sin miopía patológica, los distintos experimentos, las métricas obtenidas y su comparativa.

Por otro lado, la documentación del trabajo constará de la memoria y de la presentación del TFG:

- La memoria es este documento, en el que se hace un estudio del contexto del problema, se exploran los conceptos teóricos involucrados y se ofrecen los resultados y conclusiones de los experimentos prácticos.
- La presentación del TFG es el documento utilizado durante la defensa del proyecto, con la información más relevante del mismo en un formato más breve.

1.6 Breve descripción de los otros capítulos de la memoria

Capítulo 2. Miopía patológica. En el segundo capítulo se exploran los conceptos médicos relacionados con el problema. Se describe el defecto visual, la miopía, de forma genérica, hasta el subgrupo específico objeto de este trabajo, la miopía patológica. Revisamos la literatura médica para explicar los conceptos necesarios para entender este problema y analizar su impacto global y local.

Capítulo 3. Aprendizaje computacional. Introducción general sobre *Machine Learning y Deep Learning*. Se describen las distintas técnicas (métodos con aprendizaje supervisado, aprendizaje por refuerzo y aprendizaje no supervisado), así como los tipos de problema que resuelven (de regresión y de clasificación), haciendo hincapié en aquellas que guardan mayor relación con el presente trabajo. Posteriormente, se exploran los conceptos relacionados con las redes neuronales en general y las redes neuronales convolucionales en particular, incluyendo algunas arquitecturas estándar. Por último, se introducen conceptos generales relacionados con el entrenamiento y la validación de los modelos.

Capítulo 4. Metodología. Se describen las propuestas de solución para el problema de clasificación de imágenes de fondo de ojo en dos clases: con miopía patológica y sin miopía patológica. Se especifica la forma de poner en práctica los conceptos vistos previamente.

Capítulo 5. Experimentos y resultados. Se reflejan los diferentes experimentos, las métricas obtenidas y la comparativa. Se utilizarán diferentes técnicas y distintos modelos para clasificar imágenes de fondo de ojo en dos clases: con y sin miopía patológica.

Anexo 1. Relación de tablas con las métricas obtenidas en los experimentos de *Randomized Search*.

2. Miopía patológica

2.1 Miopía

La miopía es un problema de salud global. Un reciente estudio del *American Academy of Ophthalmology* predice que en 2050 habrá 4.758 personas miopes, lo que supone prácticamente la mitad de la población mundial (Figura 2) [2].

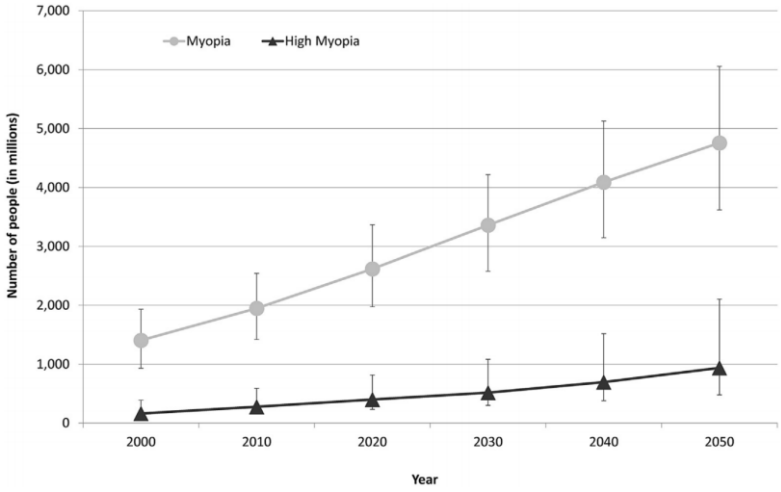

Figura 2 Predicción de la tendencia de la incidencia de la miopía y la alta miopía. HOLDEN et al. *Global Prevalence of Myopia and High Myopia and Temporal Trends from 2000 through 2050* [2]

En España, el riesgo de miopía y alta miopía en los niños se está viendo incrementado por factores del estilo de vida como el uso de dispositivos electrónicos [3]. Además, la alta miopía es la principal causa de la afiliación a la Organización Nacional de Ciegos Españoles (ONCE) [4].

La elevada y creciente incidencia de la miopía y la gravedad de la miopía patológica hacen que sea imprescindible su diagnóstico temprano. La miopía patológica afecta actualmente a cerca de un 3% de la población mundial [1] y es la cuarta causa más común de ceguera irreversible en los países desarrollados [5].

Los rayos de luz que rebotan en los objetos atraviesan la córnea, que es la parte más externa del ojo. Se trata de una membrana curva y transparente que provoca la primera refracción de los rayos de luz.

Detrás de la córnea se encuentra el iris, un diafragma que se abre y se cierra para regular la cantidad de luz que pasa a través de la pupila. En circunstancias de alta intensidad lumínica, el iris se expande, reduciendo así la cantidad de luz que atraviesa la pupila. Lo contrario ocurre en circunstancias de baja intensidad lumínica.

A través de la pupila, la luz llega al cristalino, que es una lente que se encarga de enfocar la imagen. El músculo ciliar se encarga de modificar el cristalino: para

enfocar objetos cercanos, el músculo ciliar se comprime para hacer más pequeño y grueso el cristalino, aumentando así la refracción de la luz; en el caso contrario, el músculo ciliar se relaja para hacer más grande y fino el cristalino, reduciendo la refracción de la luz y permitiendo así enfocar objetos lejanos.

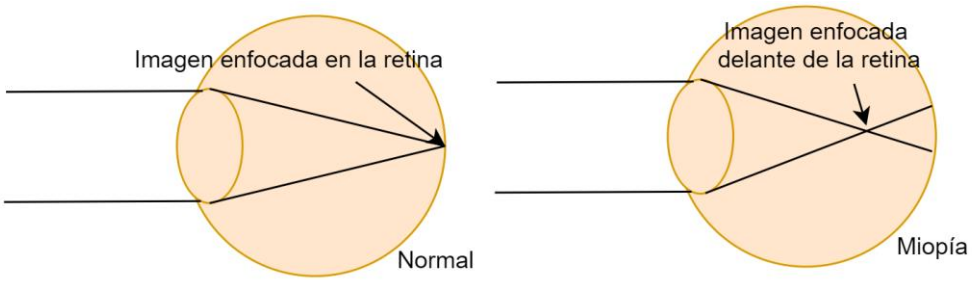

Figura 3 Refracción correcta en la retina y refracción incorrecta producida por la miopía

En el ojo sano, la luz se enfoca en la parte posterior del mismo. Esta zona está recubierta por la retina, un tejido con millones de células sensibles a la luz que la transforman en impulsos eléctricos para enviarlos al cerebro a través del nervio óptico.

En el ojo miope, este proceso de refracción de la luz que culmina con los rayos enfocados en la retina no se produce correctamente para los objetos lejanos. Los rayos de luz no convergen en la retina, sino en un punto focal situado delante de ella, de forma que los objetos lejanos se ven borrosos. En oftalmología, el defecto de refracción del ojo se denomina “ametropía”. La Figura 3 muestra esquemáticamente un ojo sano que enfoca correctamente en la retina y un ojo miope que sufre la ametropía.

2.2 Alta miopía y miopía patológica

Existe un subgrupo de la miopía denominado “alta miopía” o “miopía magna”, que se define por un defecto de visión superior a las 6 dioptrías o una longitud axial mayor de 26,5 milímetros [1]. La dioptría es una unidad de medida que expresa el poder de refracción de una lente. En los defectos de visión, cuantas más dioptrías, más grave es la afección. La longitud axial es la distancia entre la parte delantera y la parte posterior del ojo. Una longitud axial anormal provoca defectos refractivos de la visión como el que estamos estudiando.

La forma más grave de miopía y objeto de este trabajo es la miopía patológica, también llamada miopía degenerativa. Se denomina así a la alta miopía con alguna patología posterior debida a la elongación de la longitud axial. La pérdida de visión relacionada con la miopía patológica puede ser progresiva e irreversible [1]. Las patologías asociadas a la miopía patológica producen cambios en:

- El polo posterior del ojo, formado por la retina, el nervio óptico, la mácula y el humor vítreo. La mácula es una pequeña mancha de color amarillento, ubicada en la retina, que se encarga de la visión central. El humor vítreo es el gel del interior del ojo que mantiene su forma, consiguiendo que la

superficie de la retina sea uniforme para que la recepción de las imágenes sea correcta. Las patologías asociadas a esta parte del ojo incluyen la maculopatía, el fondo teselado, el estafiloma posterior y la degeneración retino-coroidea.

- El disco óptico. Es la zona de la retina que comunica con el nervio óptico. No dispone de receptores de luz, por lo que también recibe el nombre de “punto ciego”. Los problemas del disco óptico asociados a la miopía patológica incluyen la atrofia parapapilar y la inclinación.

3. Aprendizaje computacional

3.1 Introducción

La Inteligencia Artificial es una rama de las ciencias de la computación. No existe una única definición para el término, de forma que una definición más precisa requiere especificar los objetivos que se persiguen [9]. En su libro *Artificial Intelligence: A Modern Approach*, Stuart J. Russell y Peter Norvig proponen una clasificación en base a dos dimensiones: por un lado, si el objetivo que se persigue es un comportamiento determinado o bien una forma de razonar determinada; por otro, la forma de medir la corrección de los programas, denominada "racionalidad" [10]. De esta forma surgen cuatro clasificaciones para Inteligencia Artificial, que son solo algunas de las posibles.

El aprendizaje computacional o aprendizaje automático (*Machine Learning)* es un subcampo de la Inteligencia Artificial. La particularidad de los sistemas de aprendizaje computacional es que aprenden de los datos sin estar explícitamente programados. En la programación tradicional en IA es necesario definir manualmente las reglas que permiten a los programas tomar decisiones. Al contrario, un programa basado en técnicas de *Machine Learning* adquiere su propio conocimiento a partir de los datos en bruto [11].

3.2 Técnicas de aprendizaje

Habitualmente las técnicas de aprendizaje computacional se clasifican en tres grupos: métodos con aprendizaje supervisado, aprendizaje por refuerzo y aprendizaje no supervisado. En el aprendizaje supervisado, en el que se enmarca el problema de clasificación de este trabajo, se dispone del conocimiento completo sobre qué respuesta se tiene que dar en una determinada situación [12]. Dicho de otra forma, el conjunto de entrenamiento del juego de datos que se utilizará para entrenar el sistema contiene el atributo que queremos predecir para nuevos datos.

El aprendizaje supervisado se utiliza para diferentes tipos de problemas, siendo los más habituales los problemas de regresión y los problemas de clasificación:

- Problemas de regresión: cada observación del conjunto de entrenamiento, cada instancia, contiene un atributo de tipo numérico con la solución buscada. Un ejemplo muy común sería el precio de una vivienda a partir de los datos de ubicación, tamaño, número de habitaciones, etc. El problema de regresión lineal múltiple consiste en encontrar el hiperplano que mejor se ajusta a los datos, es decir, que minimiza el error o función de coste.
- Problemas de clasificación: en este caso el atributo objetivo es categórico o binario, en lugar de numérico. Se pretende predecir la clase de cada instancia. Un ejemplo sería predecir la clase de una flor en base a sus características morfológicas, como en caso del conocido conjunto de datos sobre la flor iris [13].

3.3 Redes neuronales artificiales

El aprendizaje profundo (*Deep Learning*) es un subcampo del *Machine Learning* capaz de tratar los problemas más exigentes de la Inteligencia Artificial, que involucran funciones complejas y alta dimensionalidad. Problemas centrales de esta disciplina, como el reconocimiento de voz u objetos o la clasificación de billones de imágenes. Su principal herramienta es la red neuronal artificial.

Las redes neuronales artificiales (ANN, *Artificial Neural Network*) son modelos de *Machine Learning* inspiradas en las redes de neuronas biológicas del cerebro humano. La unidad básica de las ANN es la *Threshold Logic Unit* (TLU) o *Linear Threshold Unit* (LTU). Siguiendo la analogía biológica, una TLU sería asimilable a una neurona. De la misma forma que una neurona recibe señales de entrada a través de las dendritas y envía una señal de salida a través del axón, la TLU toma números como entradas, con un peso asociado a cada una, y devuelve otro número como salida. La TLU calcula la suma ponderada de las entradas:

$$z = w_1 x_1 + w_2 x_2 + \cdots + w_n x_n = X^T W$$

Esto sería la ecuación de un modelo de regresión lineal múltiple sin el término independiente que define en qué punto la recta corta al eje y. En el caso de las TLU, se añade una variable de entrada adicional que siempre toma el valor 1 denominada sesgo o *bias*, $x_0 = 1$, de forma que la suma ponderada queda de esta manera:

$$z = w_0 x_0 + w_1 x_1 + w_2 x_2 + \cdots + w_n x_n = X^T W$$

Por último, la TLU aplica una función de activación (*step function*) a esta suma y devuelve el resultado:

$$h_W(X) = step(z)$$

Un ejemplo de función de activación sería la función Heaviside o función escalón unitario, que devuelve 0 si la entrada es mejor que 0 y 1 si la entrada es mayor o igual que 0.

$$Heaviside(z) = \begin{cases} 0 \; si \; z < 0 \\ 1 \; si \; z \geq 0 \end{cases}$$

La **¡Error! No se encuentra el origen de la referencia.** muestra la representación gráfica de una TLU.

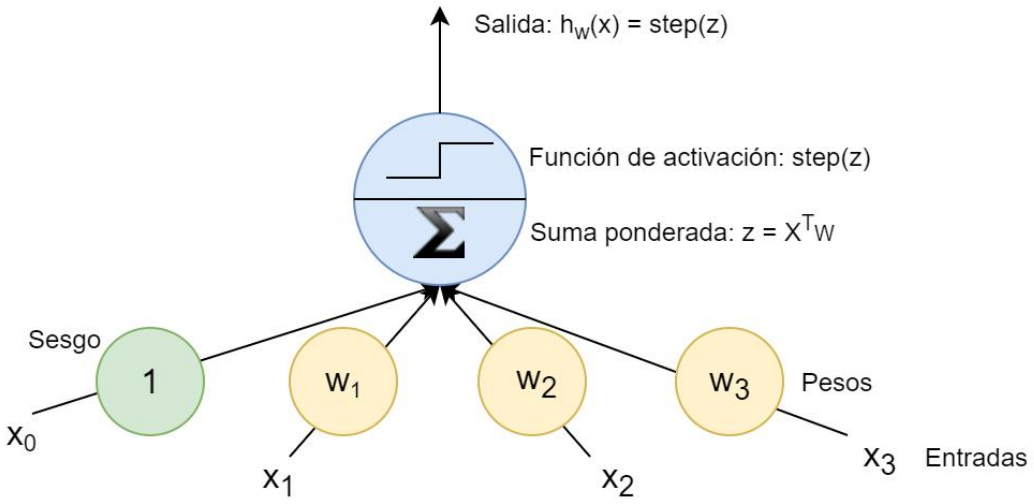

Figura 4 *Threshold Logic Unit*

Se denomina Perceptrón a una arquitectura de redes neuronales artificiales compuesta por una sola capa de TLU, estando conectada cada TLU a todas las entradas. En ocasiones también se llama Perceptrón a una red neuronal de una sola TLU, por lo que el Perceptrón también se puede considerar como una neurona artificial [11]. Arquitecturas más complejas, con varias capas de TLU, se denominan Perceptrón Multicapa (*Multilayer Perceptron*, MLP). Las capas más cercanas a la entrada también se conocen como “capas bajas” de la red, mientras que las más cercanas a la salida son las “capas altas”. Esta profundidad, fruto de añadir capas, es la que da nombre al aprendizaje profundo.

3.3.1 Redes neuronales convolucionales

Las redes neuronales convolucionales (*Convolutional Neural Network*, CNN) son un tipo de red neuronal especializada en el procesamiento de topologías cuadriculadas, tales como imágenes —en definitiva, una cuadrícula de píxeles— [7]. La convolución es la operación matemática que da nombre a estas redes neuronales y que se produce en las capas convolucionales. En el contexto de las CNN, la convolución consiste en aplicar un filtro o *kernel* a la imagen de entrada para extraer una característica. El filtro es una matriz de menor tamaño que la imagen de entrada, que se va desplazando sobre esta, multiplicando las celdas superpuestas y sumando los productos. El resultado obtenido es el mapa de una característica (*feature map*), por ejemplo, bordes o líneas verticales u horizontales.

La **¡Error! No se encuentra el origen de la referencia.**5 muestra un ejemplo de filtro para detectar bordes verticales. La convolución sobre un área de la imagen original con todos los píxeles del mismo valor produce un valor de 0 en la salida. Sin embargo, cuando se aplica el filtro sobre un borde vertical, la salida toma el valor 3.

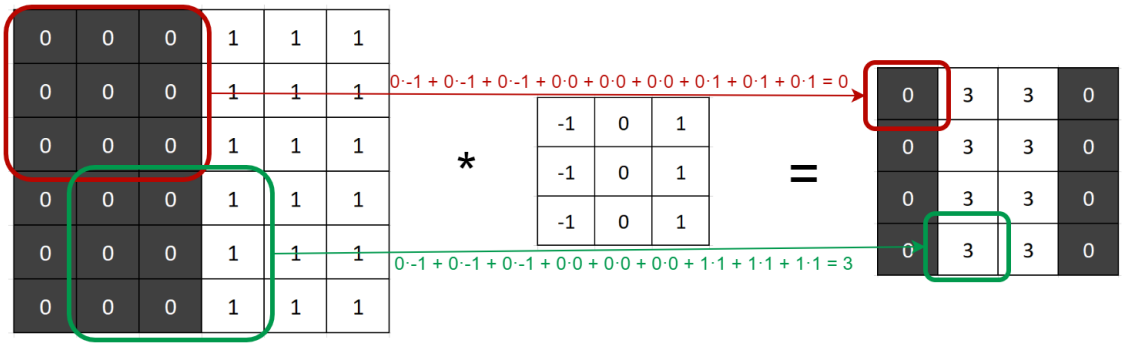

Figura 5 Convolución de un filtro que detecta bordes verticales.

La salida de las capas convolucionales se conecta con la entrada de las *pooling layers*. Son capas de reducción de muestreo, que reducen el tamaño de la imagen, disminuyendo en consecuencia la potencia de cómputo y la cantidad de memoria necesarias para trabajar con ella. En las *pooling layers* también se aplica un *kernel*, pero en este caso no tiene pesos asociados. Solo agrega las entradas aplicando una función de agregación, habitualmente el máximo o la media. Por ejemplo, un *pooling kernel* de 2x2 de una *max pooling layer* que reciba como entradas 2, 7, 4, 3, propagará a la siguiente capa solo el valor más alto, el 7, como muestra la **¡Error! No se encuentra el origen de la referencia.**6.

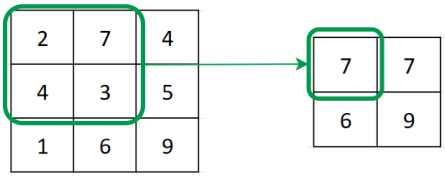

Figura 6 Aplicación de un filtro 2x2 de max pooling.

La cantidad de celdas que se desplazan los filtros sobre la entrada se denomina *stride*. En los ejemplos de **¡Error! No se encuentra el origen de la referencia.**5 y **¡Error! No se encuentra el origen de la referencia.**6 el *stride* es 1. Un valor mayor para este hiperparámetro provoca que la salida producida sea menor, reduciendo la dimensionalidad y, en consecuencia, la complejidad computacional del modelo.

Una arquitectura CNN típica está formada por varios conjuntos de capas convolucionales seguidas de una *pooling layer*. A medida que la red neuronal va procesando la imagen, esta va disminuyendo de tamaño por efecto de las *pooling layers*. Por otro lado, su profundidad va aumentando gracias a los *feature maps* generados en las capas convolucionales. Al final de la red, en las capas más altas, se añade una o varias capas totalmente conectadas, que reciben su nombre porque la salida de todas las neuronas de la capa está conectada a la entrada de todas las neuronas de la siguiente capa. Por último, la capa superior de la red devuelve la predicción para las entradas. Puede estar formada por una neurona, por ejemplo, para devolver el número correspondiente a la clase de la instancia recibida por la entrada; o bien por varias, por ejemplo, una neurona por cada clase del modelo para predecir el porcentaje de probabilidad de que la instancia recibida por la entrada pertenezca a cada clase.

3.3.2 Arquitecturas estándar de CNN

VGGNet: El equipo de Karen Simonyan y Andrew Zisserman, de la Universidad de Oxford, consiguió el primer puesto en localización y el segundo puesto en clasificación en el ILSVRC 2014 (ImageNet Large Scale Visual Recognition Challenge) con esta arquitectura [14]. VGGNet está formada por conjuntos de dos o tres capas convolucionales seguidas de una *pooling layer*. La variante VGG16 se compone de 16 capas convolucionales mientras que la variante VGG19 alcanza las 19 capas convolucionales.

ResNet: Kaiming He et al, de Microsoft Research, obtuvieron el primer puesto en clasificación, detección y localización en el ILSVRC 2015 con esta arquitectura [15]. La variante empleada en el *challenge* tenía 152 capas, pero otras variantes de ResNet tienen 34, 50 y 101 capas. El nombre ResNet proviene de Residual Network. Estas redes se caracterizan porque la entrada de una capa también se añade a la salida de otra capa posterior. Esto se denomina *skip connection* o *shortcut connection* y está representado en la Figura 77. Por un lado, permite entrenar redes neuronales muy profundas evitando el problema del desvanecimiento de gradiente (*vanishing gradient*). Por otro lado, permite acelerar el entrenamiento. La salida de un conjunto de capas con *skip connection* que todavía no se haya entrenado, que tenga los pesos a 0, será igual a la entrada, es decir, modelará la función identidad. Dado que es habitual que la función objetivo sea similar a la función identidad, esta técnica acelera el entrenamiento.

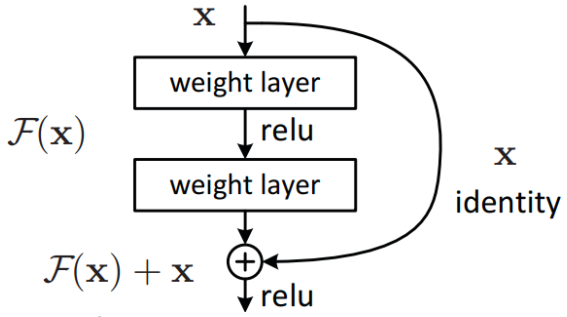

Figura 7 Bloque de construcción de ResNet que ofrecen como ejemplo He et al en su trabajo [15]

GoogleNet: Christian Szegedy et al, de Google Research, ganaron el ILSVRC 2014 con esta arquitectura [16]. Se caracteriza por utilizar subredes llamadas “módulos inception”. La Figura 88 muestra un ejemplo de módulo inception con reducción de dimensionalidad. La señal de entrada se envía a cuatro capas distintas. En el segundo nivel de capas del módulo se usan filtros de distintos tamaños, lo que permite capturar patrones a distintas escalas. Todas las capas usan stride de 1 y padding “same”, lo que provoca que las salidas tengan el mismo tamaño y puedan concatenarse. La reducción de dimensionalidad se produce en las capas con filtros de tamaño 1x1. Obviamente estos filtros no pueden detectar patrones espaciales porque solo convolucionan un pixel, pero sí pueden detectar patrones en profundidad. Devuelven menos feature maps de los que reciben por la entrada, por lo que reducen dimensionalidad, con los beneficios que conlleva: reducción de coste computacional, aceleración del entrenamiento y mejora de la capacidad de generalización de modelo.

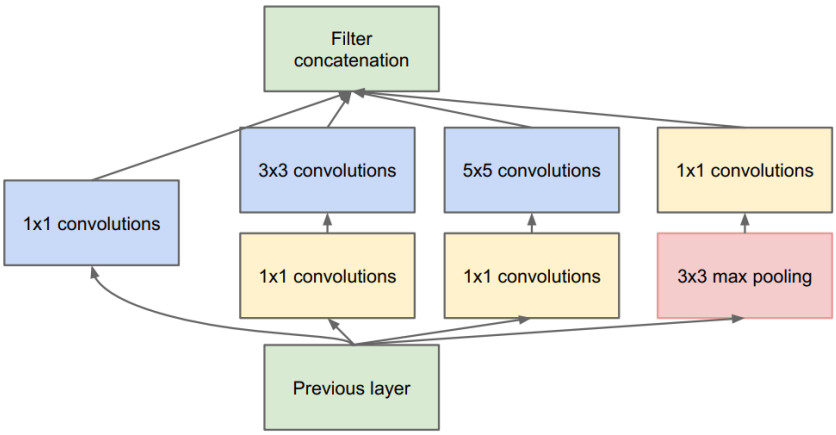

Figura 8 Ejemplo de módulo *inception* con reducción de dimensionalidad de GoogleNet que ofrecen Szegedy et al en su trabajo [16]

3.4 *Data augmentation*

Data augmentation es una técnica para aumentar artificialmente el conjunto de entrenamiento mediante la transformación de las imágenes reales del *dataset*. Aumentar el conjunto de entrenamiento facilita el entrenamiento y reduce el *overfitting*. Es especialmente importante al tratar con imagen médica, dado que no es fácil conseguir muestras con la patología objetivo. Algunas de las transformaciones que pueden aplicarse a las imágenes son:

- *Shearing*: transformación geométrica que inclina la imagen.
- *Zoom*: ampliación de la imagen.
- *Horizontal flip*: volteo horizontal de la imagen.
- *Vertical flip*: volteo vertical de la imagen.
- Modificación del brillo de la imagen.
- Rotación de la imagen.

3.5 Entrenamiento de los modelos

Un modelo es una representación conceptual simplificada de una realidad más compleja. El objetivo del *Machine Learning* es la construcción automatizada de modelos entrenados a partir de datos de un determinado dominio, que permitan predecir información sobre nuevas observaciones no utilizadas durante el entrenamiento.

Entrenar un modelo consiste en adaptar sus parámetros para que se ajuste al conjunto de datos de entrenamiento. Para medir cómo de correcto es ese ajuste se usa una función de coste, función de pérdida o función de error. Algunas fuentes usan estos términos de forma intercambiable, mientras que otras distinguen algunos matices [7]. La función de coste más habitual para modelos de regresión lineal es el error cuadrático medio (RMSE, *Root Mean Square Error*) [11]. Es la media de la diferencia elevada al cuadrado entre el valor real y el valor predicho para cada observación. Elevando al cuadrado se consigue penalizar

más los errores mayores. Con esta función de coste es posible utilizar una ecuación que directamente calcula los parámetros del modelo que la minimizan. Se trata del mínimo error cuadrático medio o *Normal Equation*. El problema es que esta ecuación es computacionalmente ineficiente, con una complejidad de entre $O(n^{2.4})$ y $O(n^3)$. Además, solo sirve para regresión lineal.

El descenso del gradiente (*Gradient Descent*) es un algoritmo de optimización de función de coste de propósito más general. Se basa en calcular las derivadas parciales de cada parámetro del modelo para hallar el gradiente de la función de coste. El gradiente es un vector que indica hacia dónde asciende la función. Para minimizar la función de coste será necesario modificar los parámetros del modelo para avanzar en sentido contrario del gradiente, descendiendo. Este proceso se repite iterativamente hasta converger en un mínimo de la función. Es importante la cantidad de distancia recorrida en cada iteración, llamada ratio de aprendizaje (*learning rate*). Un valor demasiado bajo para el *learning rate* puede provocar que se requieran demasiadas iteraciones para hallar un mínimo. Por el contrario, un *learning rate* demasiado grande puede causar que el algoritmo nunca converja, que no logre hallar un mínimo.

En las redes neuronales es mucho más complejo determinar cómo un cambio en un parámetro afecta a la función de coste. Modificar los pesos de una neurona puede afectar al resultado final por cualquiera de las conexiones de esa neurona con las capas sucesivas. Además, los pesos de las conexiones entre las siguientes capas afectarán a su vez al resultado final. Para solucionar esto, se utiliza la retropropagación de errores (*backpropagation*). Se parte de una combinación de parámetros que devuelve una salida errónea y se propaga la señal de error hacia atrás, pero a cada neurona solo se le envía una fracción de la señal de error en función de cuánto se haya implicado en generar el resultado final. De esta manera se determina cuánto hay que modificar cada parámetro para el cálculo del gradiente.

3.6 Validación de los modelos

En el aprendizaje supervisado, los modelos de *Machine Learning* se elaboran utilizando tres conjuntos de datos diferentes, en el mejor de los casos. Se entrenan sobre un conjunto de datos —conjunto de entrenamiento—, se evalúan sobre un conjunto distinto para orientar la optimización —conjunto de validación— y una vez terminados se prueban sobre otro conjunto que no se ha visto durante el entrenamiento —conjunto de test—. Esto es importante para evitar el *overfitting*.

Si solo se dispone de un conjunto de entrenamiento, es habitual reservar una parte para usar como conjunto de validación. El riesgo de esta práctica es seleccionar un conjunto de validación que no sea representativo del *dataset*. Para evitarlo, es más conveniente usar validación cruzada (*cross validation)*.

3.6.1 Validación cruzada

La validación cruzada *(cross validation)* consiste en dividir el conjunto de entrenamiento en k particiones, habitualmente llamadas *folds*. Se realizan k entrenamientos del modelo, reservando en cada ocasión uno de los *folds* para utilizar como conjunto de validación. De esta forma los resultados obtenidos son más robustos frente a la distribución de los datos que si simplemente se seleccionase un conjunto estático de los datos como conjunto de validación.

La Figura 99 muestra un ejemplo gráfico con 5-*fold*. El conjunto de entrenamiento se divide en cinco particiones, lo que produce cinco divisiones (*split*). En cada una de ellas, una partición distinta se utiliza como conjunto de validación y el resto de los datos conforman el conjunto de entrenamiento.

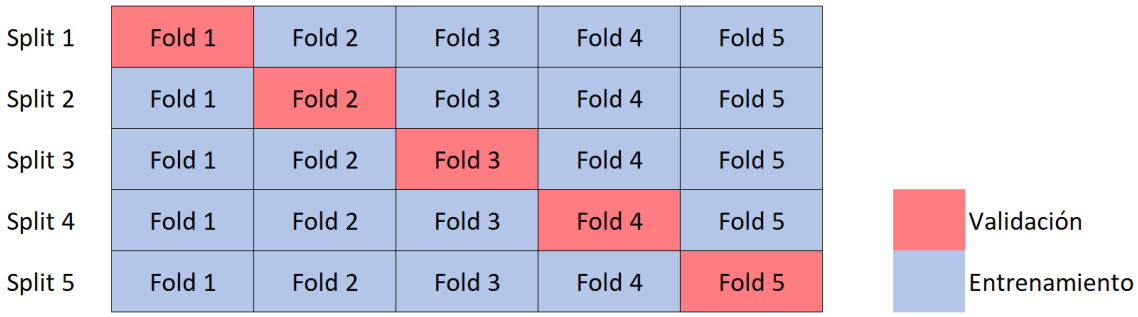

Figura 9 *Cross validation*

La validación cruzada no elimina el riesgo de generar conjuntos de entrenamiento y validación poco representativos del total de los datos y es importante asegurase de que los *folds* conservan la distribución del conjunto de datos original. En primer lugar, si las observaciones están ordenadas por clase y las particiones se realizan sin mezclar (*shuffle*), probablemente se crearán conjuntos de validación que solo contengan instancias de una clase, conjuntos de entrenamiento que no tengan ninguna instancia de alguna clase y escenarios similares. Por otro lado, mezclar las muestras no es suficiente en el caso de juegos de datos con clases desbalanceadas (*class imbalance*), es decir, en los que la presencia de observaciones de alguna(s) clase(s) en el *dataset* es muy distinta de la de otra(s). Este problema es habitual en imagen médica, porque obviamente es mucho más fácil obtener imágenes de pacientes sanos. Una solución es la validación cruzada estratificada (*stratified cross validation*).

En la *stratified cross validation*, los *folds* se crean estratificados, manteniendo el porcentaje de observaciones de cada clase del total del conjunto de datos. La Figura 1010 muestra una representación visual de esta forma de generar 4-*fold*. A partir de la fila “Clase” se puede ver que, si las particiones se hicieran sin estratificar, la primera tendría solo observaciones de la clase 0, las siguientes solo de clase 1 y las últimas solo de clase 2. Incluso con *folds* generados mezclando las muestras, es probable que se produjesen particiones con mucha presencia de la clase 2, dado que más de la mitad de las observaciones del *dataset* pertenecen a ella. Esto arrojaría métricas de validación engañosas. Sin embargo, con la estratificación, cada conjunto de validación mantiene la proporción de observaciones de cada clase.

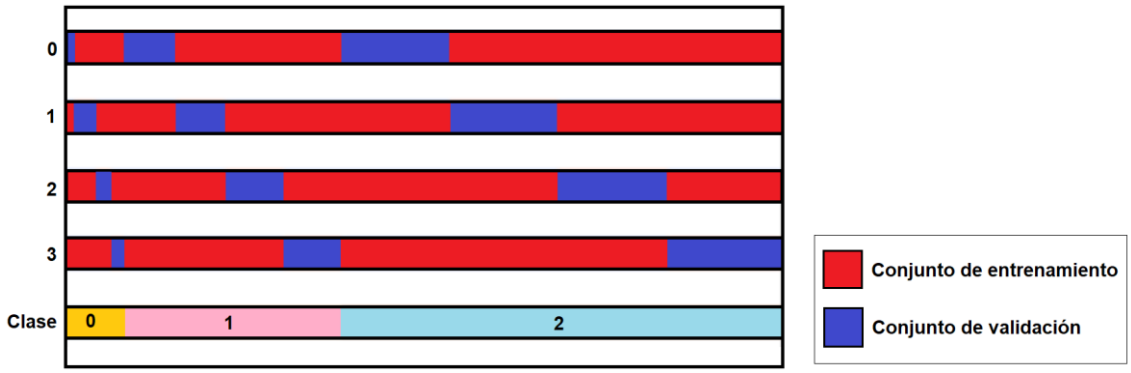

Figura 10 *Stratified cross validation*

3.6.2 Métricas de rendimiento

Las métricas de validación se utilizan para explorar y cuantificar la capacidad de los modelos para predecir la información objetivo (por ejemplo, a qué clase pertenece una imagen) a partir de nuevos datos de entrada (por ejemplo, imágenes de fondo de ojo). Están relacionadas con los conceptos de verdadero y falso positivo y negativo:

- Verdadero positivo (*TP*, *True Positive*): muestra de la clase objetivo etiquetada correctamente como tal.
- Falso positivo (FP, *False Positive*): muestra etiquetada como clase objetivo cuando en realidad no lo es.
- Verdadero negativo (TN, *True Negative*): muestra que es no es de la clase objetivo y correctamente no se ha etiquetado como tal.
- Falso negativo (FN, *False Negative*): muestra que pertenece a la clase objetivo y sin embargo no se ha etiquetado como tal.

Las métricas que se obtienen a partir de distintos cálculos con los conceptos anteriores son:

- *Accuracy*: porcentaje de imágenes clasificadas correctamente.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

- *Precision*: número de positivos correctos entre el total de positivos predichos.

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

- *Recall*: número de positivos correctos entre el total de positivos reales.

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

- ROC (*Receiver Operating Characteristic*) y ROC AUC (*Area Under the Curve*): la curva ROC representa la tasa de verdaderos positivos (TPR, *True Positive Rate*) frente a la tasa de falsos positivos (FPR, *False Positive Rate*). TPR es otra forma de llamar al *recall*. FPR es la razón entre los falsos positivos y el total de observaciones de la clase negativa.

$$FPR = \frac{FP}{FP + TN}$$

- El área bajo la curva ROC (ROC AUC) es una forma más cómoda de comparar dos clasificadores. Un clasificador perfecto tiene un ROC AUC de 1, mientras que un clasificador aleatorio tiene un ROC AUC de 0.5 y su curva ROC discurre por la diagonal de la gráfica.

Los TP, FP, TN y FN habitualmente se representan mediante matrices de confusión. La Tabla 1 muestra una matriz de confusión para un clasificador binario. En un eje se sitúan las clases reales y en el otro las predichas. En la intersección de la misma clase en fila y columna se reflejan los positivos, las muestras clasificadas correctamente, mientras que en el resto de los campos se reflejan los negativos. En el ejemplo, hay 10 de muestras de cada clase. 9 de las 10 muestras de clase 1 se han etiquetado correctamente (TP), mientras que la muestra restante se ha etiquetado erróneamente como clase 0 (FN). 8 de las 10 muestras de clase 0 se han etiquetado correctamente (TN), mientras que las otras 2 se han etiqueta como clase 1 (FP).

Tabla 1 Matriz de confusión para clasificación binaria

		Clase predichas	
		Clase 0	**Clase 1**
Clases reales	**Clase 0**	8 (TN)	2 (FP)
	Clase 1	1 (FN)	9 (TP)

3.7 *Transfer learning*

La técnica del *transfer learning* consiste en reutilizar capas preentrenadas de redes neuronales existentes que desempeñen tareas similares a las que se persiguen. Esto acelera el entrenamiento de la red y además requiere menos datos de entrenamiento [11].

Es habitual que la capa de salida de la red deba sustituirse, porque probablemente no devuelva el número de datos necesarios. Esto ocurre, por ejemplo, en caso de que la cantidad de clases distintas a predecir en un problema de clasificación sea diferente. A partir de ahí, la cantidad de capas a reutilizar depende de lo similar que sea la tarea que desempeña la red neuronal original y la que se quiere implementar. Las capas más bajas de la red modelan características más básicas, por lo que es más probable que sean útiles para un uso general. Sin embargo, las capas más altas modelan características más complejas, que variarán según el dominio del problema. Por ejemplo, una red neuronal preentrenada con un *dataset* de razas de perros detectará bien patrones como la forma de las orejas o el tamaño de las patas, pero probablemente no ofrecerá un buen rendimiento para imagen médica.

De la misma manera que se puede variar el número de capas a reutilizar del modelo original, también se puede modificar cuáles de las capas reutilizadas se reentrenan. Mantener fijos los pesos de una capa, sin alterarlos durante el entrenamiento, se denomina “congelar” la capa (*freeze*). Para determinar cuántas capas congelar, es relevante la cantidad de datos de que se disponga.

Cuanto mayor sea el conjunto de entrenamiento, más capas será posible descongelar para reentrenar.

3.8 Optimización de los modelos

La optimización de los modelos de aprendizaje computacional requiere configurar muchos hiperparámetros. La búsqueda manual de la combinación que ofrezca los mejores resultados es ineficiente y puede ser inasumible en cuanto al coste temporal que supondría. Para realizar esta tarea de forma automatizada se emplean técnicas de optimización como *Randomized Search* y *Grid Search*.

Randomized Search genera un número de combinaciones aleatorias de hiperparámetros, seleccionando un valor aleatorio de entre los establecidos para cada hiperparámetro en cada iteración. Es útil cuando el espacio de búsqueda es amplio, de forma que una búsqueda exhaustiva requeriría demasiado tiempo.

Grid Search es una técnica similar a *Randomized Search*, pero en este caso se exploran todas las posibles combinaciones de hiperparámetros de forma exhaustiva, no solo un subconjunto aleatorio. Por este motivo, está más indicada para espacios de búsqueda más pequeños.

3.9 Deep Learning para diagnóstico de miopía patológica

Otros trabajos publicados sobre el *dataset* de PALM permiten obtener una visión del estado del arte en la aplicación de *Deep Learning* a la clasificación, segmentación y localización de imagen médica de oftalmología.

El algoritmo de Xie et al. consiguió el primer puesto en la tarea de clasificación en PALM [17]. Se basa en *transfer learning*, utilizando ResNet50 preentrenado con los pesos de ImageNet. Aplica *data augmentation* sobre el conjunto de entrenamiento, añadiendo ruido Gaussiano y rotación aleatoria de 30 grados. Con esta aproximación, se consiguió un valor de AUC de 0.998. El mismo equipo logró también el primer puesto en la tarea de localización de la fóvea, utilizando de nuevo *transfer learning*, pero con la arquitectura VGG19. Para la tarea de segmentación del disco óptico, Xie et al. utilizaron nuevamente *transfer learning*, con una arquitectura combinada de ResNet34 y U-Net, logrando el segundo puesto en el *challenge*.

Freire et al [18]. proponen una aproximación diferente para la tarea de clasificación, utilizando otros *datasets* para aumentar el conjunto de entrenamiento. Añaden las 749 imágenes del *dataset* RIGA y las 800 imágenes de los conjuntos de entrenamiento y validación de REFUGE. Estos *datasets* no ofrecen etiquetas para la miopía patológica, por lo que los autores asumen que todas las imágenes pertenecen a ojos sanos. Para el conjunto de entrenamiento proporcionado por PALM, simplifican las clases “normal” y “hight myopia” en una única “non-pm”. Al igual que Xie et al., utilizan *transfer learning*, aunque en este caso con la arquitectura Xception y sin *data augmentation*.

Realizan un entrenamiento corto, de solo 4 épocas. Con esta aproximación, reportan un valor de AUC de 0.9957 sobre el conjunto de test.

4. Metodología

4.1 Conjunto de datos

Se utilizará el conjunto de datos ofrecido en *PALM: PAthoLogic Myopia Challenge* [8]. PALM fue un evento parte del *IEEE International Symposium on Biomedical Imaging* (ISBI) de 2019 [19]. El *dataset* está formado por imágenes de fondo de ojo en color tomadas con una Zeiss Visucam 500. Contiene 400 imágenes etiquetadas con la clase, como la que puede verse en la Figura 11:

- Miopía patológica: 213 imágenes (53,25%).
- Alta miopía: 26 imágenes (6,5%).
- Vista normal: 161 imágenes (40,25%).

Figura 11 Muestra del conjunto de datos [8]. Imagen de fondo de ojo y anotación de atrofia, delimitada por círculos verdes.

Dado el escaso tamaño del conjunto de entrenamiento, se unificarán las clases de alta miopía y vista normal, para trabajar solo con 2 en lugar 3. Será, por lo tanto, un problema de clasificación binario.

4.2 Preprocesamiento de imágenes: normalización y *data augmentation*

Se normalizará el valor de cada pixel entre 0 y 1. Asimismo, dado el reducido tamaño del conjunto de entrenamiento, se utilizará *data augmentation* para aumentar la cantidad de imágenes con las que entrenar los modelos. Se usará las siguientes transformaciones:

- *Shearing*: de forma aleatoria dentro de un rango (0.2).
- *Zoom*: de forma aleatoria dentro de un rango (0.2).
- *Horizontal flip*.

4.3 Modelos

En primer lugar, se plantea una arquitectura de red neuronal convolucional clásica, de entre uno y tres conjuntos de capa convolucional junto a capa de reducción de muestreo. Es un modelo que se centra en los fundamentos de las CNN, si bien arquitecturas estándar como VGG son, en definitiva, variaciones con más capas de esta aproximación. El fin de este modelo sencillo es optimizarlo mediante *Randomized Search y Grid Search*, específicamente:

- Tamaño de imágenes de entrada: 16x16, 32x32, 64x64, 128x128, 256x256 píxeles.
- Número de conjuntos de capas convolucional+*MaxPooling*: 1, 2, 3.
- Número y tamaño de los filtros:
    - Filtros en la primera capa: 16, 32, 64.
    - Tamaño de filtros de la primera capa: 7x7, 5x5.
    - Filtros del resto de capas: 16, 32, 64.
    - Tamaño de filtros del resto de capas: 3x3, 2x2.
    - Tamaño MaxPool de la primera capa: 3x3, 2x2.
- Función de optimización: SGD, Adam.
- *Learning date*: 0,001, 0,01, 0,1, 0,3.
- *Momentum*, solo para el caso de SGD: 0,0, 0,4, 0,8.
- Tasa de *dropout*: 0,3, 0,5, 0,7.
- Épocas: 15, 30, 50.
- Tamaño de *batch*: 2, 4, 8, 16, 32.

Dado que se trata de un problema de clasificación binaria, para la última capa se opta por sigmoid como función de activación y salida de una dimensión, es decir, en un array de una dimensión se devolverá la predicción de clase para cada muestra. De la misma forma, se utilizará binary_crossentropy como función de pérdida.

Por otro lado, se utilizarán modelos estándar para experimentar con *transfer learning*:

- VGG16.
- VGG19.
- ResNet50.
- InceptionV3 (GoogleNet).

Inicialmente se congelarán todas las capas preentrenadas con los pesos de ImageNet y se entrenará solo la capa de salida. En principio se planteaba utilizar la última capa para la clasificación de las imágenes de la misma forma que en los experimentos de optimización, de una dimensión de salida con función de activación sigmoid. Sin embargo, de esta forma el entrenamiento no funcionaba correctamente. Para los experimentos de *transfer learning* se utilizará una capa de clasificación con 2 dimensiones de salida y función de activación softmax. Es decir, la salida será una matriz de 2 dimensiones, con forma *número de muestras a predecir* x *número de clases*. Para cada muestra se devuelve la probabilidad

de que pertenezca a cada clase. El resto de hiperparámetros se configurarán en función de los resultados obtenidos en los experimentos de optimización. No se repiten los experimentos de *Randomized Search y Grid Search* para cada uno de los experimentos de *transfer learning* porque se prioriza explorar varias arquitecturas diferentes, y de lo contrario los experimentos resultarían excesivamente extensos.

Los experimentos de *transfer learning* requieren un preprocesamiento adicional de las imágenes de entrada para adaptarlas a las necesidades de cada arquitectura. Por ejemplo, VGG16 requiere convertir RGB (Red Green Blue) a BGR (Blue Green Red) y normalizar cada canal respecto al *dataset* ImageNet. La Figura 12 muestra el aspecto original de una imagen del conjunto de datos y el resultado de preprocesarla para VGG16.

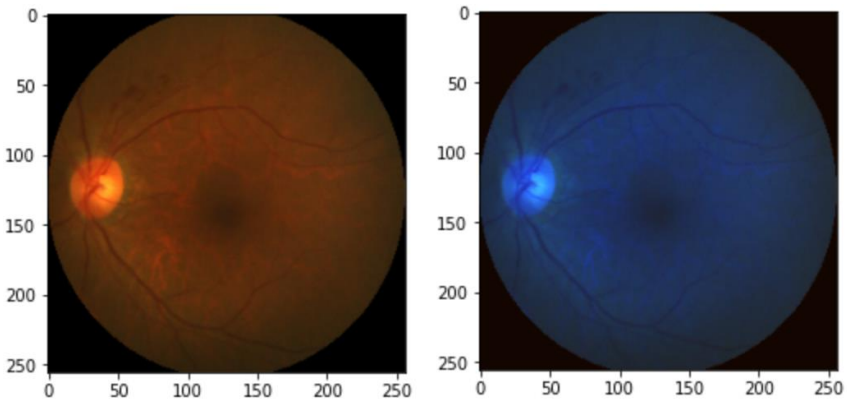

Figura 12 Imagen de entrada antes y después de ser preprocesada para VGG16

4.4 Validación de los modelos: *stratified cross validation*

El conjunto de datos está limitado a las 400 imágenes del conjunto de entrenamiento. Se utilizará validación cruzada estratificada (*stratified cross validation)* durante el entrenamiento de los modelos.

Se itera sobre los índices de entrenamiento y validación para cada *fold*. En cada iteración se crea un generador de imágenes con *data augmentation* para entrenamiento, y otro generador de imágenes sin *data augmentation* para validación. Se construye el modelo, se compila, se entrena utilizando el generador de entrenamiento y se valida utilizando el generador de validación. Es importante que se cree un modelo nuevo en cada iteración, porque de lo contrario en cada *fold* se entrenaría siempre el mismo modelo, de forma que se produciría *overfitting*.

4.5 Optimización de los modelos

La búsqueda de los hiperparámetros de la red neuronal que ofrezcan los mejores resultados se realizará mediante técnicas de optimización. Estas técnicas facilitan que se puedan realizar experimentos con distintos valores de hiperparámetros de forma automatizada y, por lo tanto, mucho más eficiente y rápida que una búsqueda manual.

En primer lugar, se explorará un conjunto amplio de posibles hiperparámetros mediante *randomized search*. Una vez acotados los valores de los hiperparámetros que optimizan los modelos, se realizará una búsqueda exhaustiva entre un conjunto de valores más reducido, mediante *grid search*. Las clases empleadas para implementar estas técnicas llevan a cabo *cross validation* automáticamente, por lo que los pipelines para estos procesos de optimización serán una versión simplificada del pipeline completo, sin necesidad de programar la lógica de iteración sobre los *folds*.

Por último, se probarán las mejores combinaciones de hiperparámetros en el *pipeline* completo definido anteriormente.

5. Experimentos y resultados

5.1 Experimentos de optimización

5.1.1 Randomized search

Este experimento consiste en aplicar la técnica *de* optimización *Randomized Search* sobre la arquitectura de red neuronal convolucional clásica descrita en el apartado 4.3 Modelos. El objetivo es explorar de forma automatizada un espacio de búsqueda amplio de distintos valores para multitud de hiperparámetros de la red neuronal, para encontrar los que arrojen mejores resultados de forma que se puedan explorar de forma exhaustiva.

Se ofrecen los resultados obtenidos en las Tablas 9-13 del Anexo 1. Dado que la validación se hace mediante *cross validation* con 5 *folds*, la precisión es la media de los 5 experimentos. Cada tabla corresponde a un tamaño de imagen de entrada distinto.

Discusión

Explorar todas las combinaciones de estos hiperparámetros de forma exhaustiva supondría entrenar 777.600 modelos, lo que conllevaría una cantidad de tiempo inasumible. De ahí la utilidad de aplicar *Randomized Search* para acotar valores de hiperparámetros más interesantes. Una vez localizadas estas combinaciones de hiperparámetros más óptimas, se podrá realizar una búsqueda exhaustiva con *Grid Search* sobre ese espacio de búsqueda ya reducido.

A la vista de los resultados:

- Es más conveniente utilizar tamaños de entrada más grandes, de 128x128 ó 256x256 píxels.
- Un solo conjunto de capa convolucional + reducción de muestreo es demasiado simple para ofrecer buenos resultados.
- Se obtienen mejores métricas con la función de optimización Adam y un *learning rate* bajo, de 0.001.
- Se obtienen mejores resultados con los valores altos de épocas y tamaño de batch.

5.1.2 *Grid search*

Este experimento consiste en utilizar *Grid Search* para explorar un número de combinaciones lo más acotado posible según las conclusiones del experimento con *Randomized Search*:

- Tamaño de imágenes de entrada: 128x128 ó 256x256.
- Número de conjuntos de capas convolucional+*MaxPooling*: 2 ó 3.

- Número y tamaño de los filtros: 16 ó 32 filtros por capa convolucional. Primera capa de 5x5 y el resto de 2x2. Filtro de *max pool* de la primera capa de 3x3 y el resto 2x2.
- Función de optimización: Adam.
- *Learning rate*: 0,001.
- Tasa de *dropout*: 0,5.
- Épocas: 30 ó 50.
- Tamaño de *batch*: 32 ó 16.

Por su tamaño más contenido, los resultados para las 32 combinaciones por cada tamaño de imagen de entrada se ofrecen en este mismo apartado, en las Tablas 2 y 3.

Tabla 2 *Accuracy (SD)* de mejor a peor de cada combinación de hiperparámetros generada mediante *Grid Search* para imágenes de entrada con tamaño 128x128 píxeles.

Accuracy	Batch	Epochs	Layers	Filtros L1	Filtros n
0.940 (0.017)	32	50	2	16	16
0.940 (0.009)	16	50	2	16	32
0.938 (0.025)	32	50	3	32	32
0.938 (0.014)	16	30	3	32	16
0.935 (0.035)	32	30	3	32	32
0.935 (0.029)	16	30	2	16	16
0.933 (0.031)	32	30	2	32	16
0.933 (0.027)	16	30	3	16	32
0.930 (0.026)	32	50	3	16	16
0.930 (0.020)	32	50	2	16	32
0.930 (0.017)	16	30	2	16	32
0.930 (0.006)	32	30	3	32	16
0.928 (0.051)	16	50	3	16	32
0.928 (0.024)	16	50	2	32	16
0.928 (0.017)	32	50	2	32	16
0.925 (0.032)	32	50	3	32	16
0.925 (0.018)	32	30	2	16	16
0.923 (0.017)	16	30	3	16	16
0.923 (0.017)	32	30	2	32	32
0.920 (0.024)	32	30	3	16	32
0.920 (0.023)	32	30	3	16	16
0.920 (0.010)	16	30	3	32	32
0.918 (0.026)	32	50	2	32	32
0.918 (0.010)	32	30	2	16	32
0.915 (0.009)	16	30	2	32	16
0.913 (0.042)	32	50	3	16	32
0.913 (0.034)	16	50	3	16	16
0.910 (0.050)	16	50	2	16	16
0.903 (0.033)	16	50	3	32	16
0.900 (0.040)	16	50	3	32	32
0.900 (0.031)	16	50	2	32	32
0.900 (0.017)	16	30	2	32	32

Tabla 3 *Accuracy (SD)* de mejor a peor de cada combinación de hiperparámetros generada mediante *Grid Search* para imágenes de entrada con tamaño 256x256 píxeles

Accuracy	Batch	Epochs	Layers	Filtros L1	Filtros n
0.953 (0.027)	32	50	3	32	16
0.943 (0.034)	32	30	3	16	32
0.940 (0.023)	32	30	2	16	32
0.938 (0.022)	16	50	3	16	32
0.935 (0.034)	16	50	2	32	32
0.935 (0.032)	32	50	3	32	32
0.935 (0.009)	32	50	3	16	32
0.930 (0.023)	32	50	2	32	32
0.930 (0.020)	32	30	3	32	32
0.928 (0.024)	32	30	2	32	32
0.928 (0.023)	16	30	3	32	16
0.928 (0.020)	16	30	3	16	16
0.928 (0.018)	16	50	3	16	16
0.928 (0.012)	16	30	2	16	32
0.925 (0.020)	32	30	3	32	16
0.925 (0.008)	16	30	3	32	32
0.923 (0.028)	32	50	2	16	16
0.923 (0.025)	16	30	3	16	32
0.923 (0.023)	32	50	3	16	16
0.920 (0.042)	16	50	3	32	16
0.920 (0.030)	16	50	2	16	32
0.920 (0.017)	32	50	2	16	32
0.920 (0.006)	32	30	2	32	16
0.918 (0.023)	16	50	2	32	16
0.915 (0.022)	32	50	2	32	16
0.913 (0.014)	32	30	2	16	16
0.910 (0.020)	16	50	3	32	32
0.910 (0.018)	16	30	2	16	16
0.910 (0.015)	32	30	3	16	16
0.908 (0.020)	16	50	2	16	16
0.905 (0.032)	16	30	2	32	16
0.883 (0.069)	16	30	2	32	32

Discusión

A la vista de estos resultados se puede concluir que la parametrización obtenida mediante *Randomized Search* es buena, porque 31 de las 32 combinaciones ofrecen una *accuracy* media superior al 90%.

Se construye el modelo se acuerdo a la combinación de hiperparámetros óptima según la investigación con *Randomized Search* y *Grid Search* de los apartados 6.1.1 y 6.1.2:

- Tamaño de imágenes de entrada: 256x256.

- Número de conjuntos de capas convolucional+*MaxPooling*: 3.
- Número y tamaño de los filtros: primera capa convolucional de 32 filtros de 5x5. El resto de capas, 16 filtros de 2x2. Filtro de *max pool* de la primera capa de 3x3 y el resto 2x2.
- Función de optimización: Adam.
- *Learning date*: 0,001.
- Tasa de *dropout*: 0,5.
- Épocas: 50.
- Tamaño de *batch*: 32.

En la Tabla 44 se muestran las medias y desviaciones estándar para los conjuntos de validación.

Tabla 4 Medias y desviación estándar de la CNN optimizada mediante *Randomized Search* y *Grid Search*.

	Accuracy	**Precision**	**Recall**	**AUC**
Media (SD)	0.92 (0.029)	0.944 (0.04)	0.906 (0.045)	0.923 (0.026)

La Figura 13 muestra las matrices de confusión de cada *fold*.

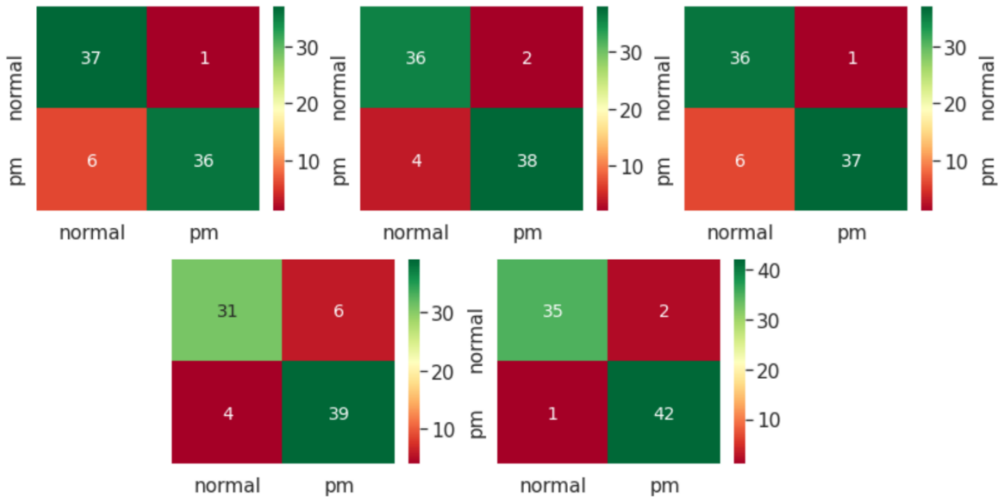

Figura 13 Matrices de confusión para la CNN clásica optimizada mediante *Randomized Search y Grid Seach.*

Una vez optimizado el modelo y los hiperparámetros mediante *Randomized Search* y *Grid Search* para conseguir los resultados expuestos, se exploraron otras técnicas con intención de investigar posibles mejoras.

- Entrenamientos con un número de épocas mayor.
- *Learning rate* dinámico, decrementando su valor en 0.2 cuando no se produjese una mejora de la métrica *val_loss* en las últimas 5 épocas.

Sin embargo, se descartó su uso porque no reflejaron ninguna mejora significativa en los resultados obtenidos.

5.2 Experimentos de *Transfer learning*

5.2.1 VGG16

En la Tabla 55 se ofrecen las medias y desviaciones estándar para los conjuntos de validación.

Tabla 5 Medias y desviación estándar de la CNN VGG16 pre-entrenada con los pesos de imagenet

	Accuracy	**Precision**	**Recall**	**AUC**
Media (SD)	0.938 (0.034)	0.940 (0.031)	0.941 (0.032)	0.982 (0.015)

La Figura 14 muestra las matrices de confusión de cada *fold*.

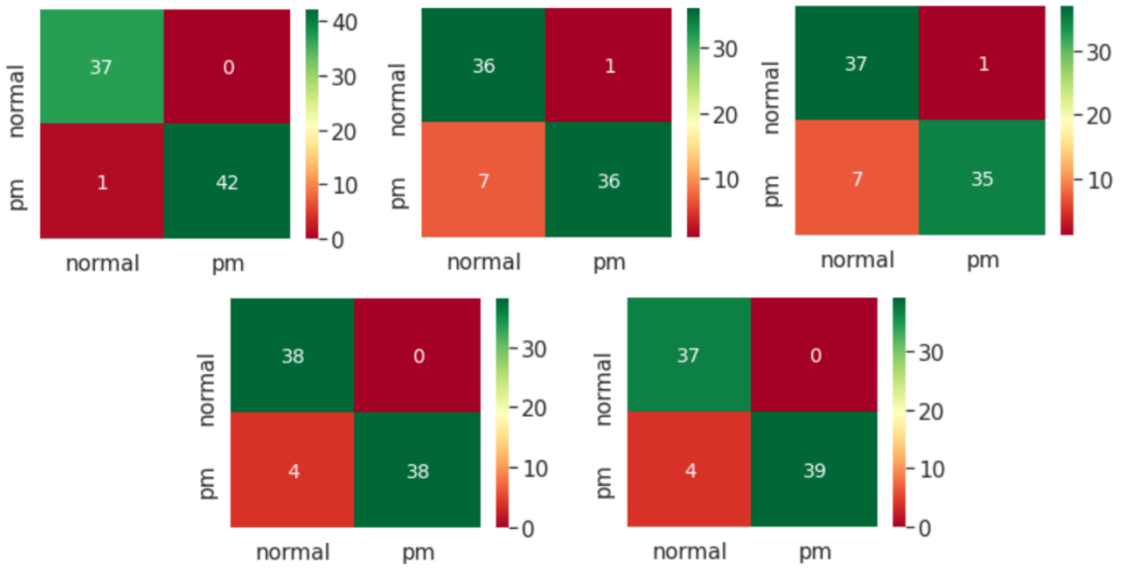

Figura 14 Matrices de confusiones para VGG16

La Figura 15 muestra la curva ROC para un valor de AUC de 0.997, muy cercana a la que ofrecería un clasificador perfecto.

Figura 15 Curva ROC de uno de los folds de VGG16 (AUC 0.997)

Se prueba a descongelar el último bloque de VGG16, pero los resultados empeoran.

5.2.2 VGG19

Mismo experimento que con VGG16, pero con la versión de 19 capas.

En la Tabla 66 se ofrecen las medias y desviaciones estándar para los conjuntos de validación.

Tabla 6 Medias y desviación estándar de la CNN VGG16 pre-entrenada con los pesos de imagenet

	Accuracy	**Precision**	**Recall**	**AUC**
Media (SD)	0.850 (0.040)	0.861 (0.032)	0.855 (0.037)	0.911 (0.035)

La Figura 16 muestra las matrices de confusión de cada *fold.*

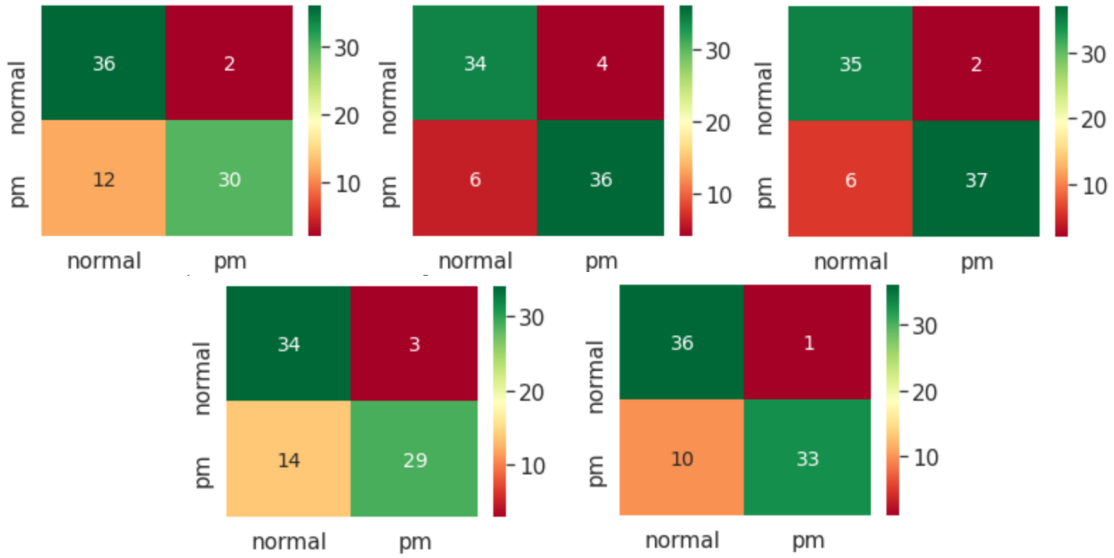

Figura 16 Matrices de confusión para VGG19

La Figura 17 muestra la curva ROC para el mejor *fold*, con un valor de AUC de 0.971.

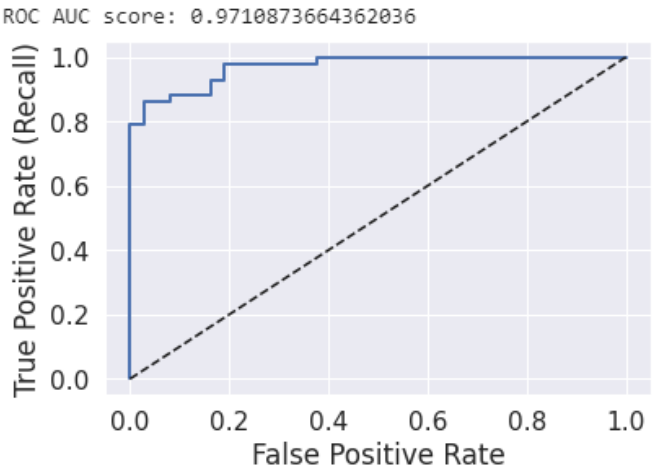

Figura 17 Curva ROC de uno de los *folds* de VGG19 (AUC 0.971)

5.2.3 ResNet50

Entrenar un modelo de ResNet50 según lo previsto en la metodología, de la misma forma que los modelos de VGG16 y VGG19 de los apartados anteriores, produce que todas las observaciones sean etiquetadas como clase normal siempre. No se detecta ninguna imagen como miopía patológica. La Figura 18 muestra un ejemplo de matriz de confusión para este resultado.

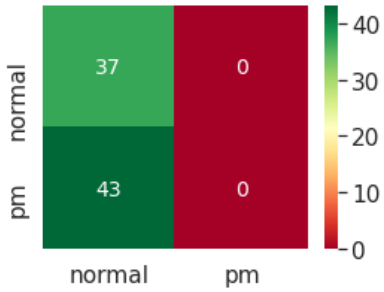

Figura 18 Matriz de confusión para ResNet50

De la misma forma, la curva de entrenamiento (Figura 19) muestra que la red neuronal no está aprendiendo.

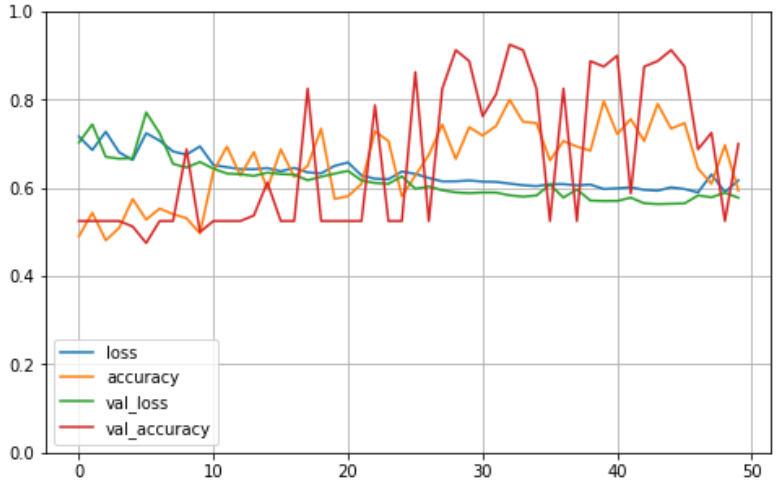

Figura 19 Curva de aprendizaje de uno de los *folds* de ResNet50

Se cambia la función de activación de la última capa. Se prueba a congelar distintos bloques de capas. Se prueban distintas funciones de optimización y valores de *learning rate*. Sin embargo, no se logra hacer funcionar este modelo adecuadamente para el conjunto de datos.

5.2.4 InceptionV3 (GoogleNet)

Se obtienen los resultados de la Tabla 77.

	Accuracy	**Precision**	**Recall**	**AUC**
Media (SD)	0.547 (0.047)	0.570 (0.042)	0.549 (0.038)	0.613 (0.064)

Tabla 7 Medias y desviación estándar de las métricas para InceptionV3

La Figura 20 muestra la curva ROC de uno de los *folds*, muy cercana a la que mostraría un clasificador aleatorio.

Figura 20 Curva ROC para uno de los *folds* de InceptionV3

5.3 Discusión

La Tabla 8 muestra la comparativa de métricas de los modelos obtenidos mediante los experimentos de los apartados 5.1 y 5.2.

Tabla 8 Comparativa de métricas obtenidas por los distintos modelos

	Accuracy	**Precision**	**Recall**	**AUC**
RS y GS	0.920 (0.029)	0.944 (0.04)	0.906 (0.045)	0.923 (0.026)
VGG16	0.938 (0.034)	0.940 (0.031)	0.941 (0.032)	0.982 (0.015)
VGG19	0.850 (0.040)	0.861 (0.032)	0.855 (0.037)	0.911 (0.035)
ResNet50	**-**	**-**	**-**	**-**
InceptionV3	0.547 (0.047)	0.570 (0.042)	0.549 (0.038)	0.613 (0.064)

En primer lugar, se ha podido comprobar la eficacia de las técnicas de *Randomized Search* y *Grid Search* para optimizar modelos de aprendizaje computacional. Gracias a *Randomized Search* se ha podido hacer un cribado inicial de hiperparámetros, que de forma manual habría supuesto realizar 777.600 entrenamientos diferentes. De esta primera exploración sobre un espacio de búsqueda amplio se han podido extraer conclusiones para realizar una búsqueda exhaustiva de un conjunto de valores de hiperparámetros mucho más reducido. Esta segunda búsqueda de 64 combinaciones mediante *Grid Search* ha permitido encontrar una configuración de hiperparámetros que logra unas métricas de 0.920 (0.029) de *accuracy* y 0.923 (0.026) de ROC AUC de media sobre los distintos conjuntos de validación generados mediante *stratified cross validation*.

Por otro lado, se ha experimentado con *transfer learning*, utilizando diferentes modelos preentrenados. Utilizando VGG16 se han mejorado los resultados obtenidos por el modelo optimizado con *Randomized Search* y *Grid Search*, consiguiendo una *accuracy* de 0.938 (0.034) y una ROC AUC de 0.982 (0.015) de una forma rápida y sencilla, demostrando el potencial de esta técnica.

Los mejores resultados se han obtenido, por tanto, con los modelos más sencillos: la arquitectura sencilla de tres bloques de capa convolucional y *MaxPool* y VGG16 preentrenado con los pesos de ImageNet y congelando todas las capas excepto la última.

Con los modelos más complejos no se han logrado resultados satisfactorios. Especialmente en el caso de ResNet50, que no se ha conseguido realizar un entrenamiento correcto de la red neuronal.

6. Conclusiones

En este trabajo se ha investigado la miopía patológica. Se ha constatado el impacto y la gravedad de la enfermedad, su creciente importancia y cómo el aprendizaje profundo puede contribuir a la necesaria detección precoz de la patología. Para ello, se han llevado a cabo experimentos de implementación de redes neuronales convolucionales que han permitido analizar distintas técnicas de construcción, entrenamiento, validación y optimización de modelos de aprendizaje automático. Gracias a la amplitud de los experimentos abordados ha sido posible comprobar cómo a través de las técnicas de optimización como *Randomized Search* y *Grid Search* es posible hallar de forma eficiente los valores óptimos para los hiperparámetros de una red neuronal. De esta forma se ha logrado implementar un clasificador de imágenes de fondo de ojo con y sin miopía patológica obteniendo métricas de 0.920 (0.029) de *accuracy* y 0.923 (0.026) de ROC AUC. Asimismo, se ha podido emplear la técnica del *transfer learning* para reutilizar modelos preentrenados de distintas arquitecturas estándar de CNN. Utilizando VGG16 se han mejorado los resultados anteriores, consiguiendo una *accuracy* de 0.938 (0.034) y una ROC AUC de 0.982 (0.015).

Mediante el estudio de los conceptos teóricos relacionados con el problema, se han ampliado los conocimientos iniciados en el itinerario de Computación del Grado en Ingeniería Informática. En la asignatura de “Aprendizaje computacional” se introducen conceptos de *Machine Learning* que se repasan en este trabajo, pero apenas se estudian las redes neuronales. En la asignatura de “Minería de datos” hay un módulo dedicado a las redes neuronales, pero no tiene una especial relevancia en el transcurso del curso (por ejemplo, no se trabajan desde el apartado práctico en ninguna de las PEC). En este trabajo se afianzan estos conocimientos y se profundiza en el aprendizaje de las redes neuronales en general y de las redes neuronales convolucionales en particular. También se ha logrado el objetivo de implementar redes neuronales convolucionales capaces de clasificar imágenes de fondo de ojo con y sin miopía patológica. Se han explorado diversas técnicas de *Machine Learning* a través de múltiples experimentos.

Durante el transcurso del proyecto se ha adaptado la planificación temporal en varias ocasiones. Ha sido especialmente necesario solapar tareas que se habían planificado como secuenciales. La amplitud y profundidad del tema estudiado ha requerido paralelizar tareas de las distintas vertientes del trabajo. De la misma forma, ha sido necesario lidiar con la carencia de conjunto de validación y conjunto de test independientes. Inicialmente PALM ofrecía un *dataset* con los 3 conjuntos de datos, de 400 imágenes cada uno. Sin embargo, en el momento de la realización de este trabajo solo se ofrecía el conjunto de entrenamiento. A pesar de los esfuerzos por contactar con los proveedores del *dataset* e incluso con investigadores que han realizado otros trabajos con los mismos datos, no ha sido posible obtener esta información faltante. Esta aparente desventaja ha servido para enriquecer los experimentos con *stratified cross validation*, que probablemente no se habría utilizado de contar con ese conjunto de validación adicional.

En este trabajo se han llevado a cabo diversos experimentos basados en una de las tareas que se proponían en PALM: la clasificación de imágenes para identificar las que presentan miopía patológica. Como interesantes líneas de trabajo de futuro se podrían abordar otras tareas sobre el mismo conjunto de datos, como detección y segmentación de lesiones de retina o localización de la fóvea.

Dentro de las tareas que sí se han abordado, también hay líneas de trabajo adicionales que podrían explorarse. Sería interesante ampliar la investigación de *transfer learning* con ResNet50, dado que el primer puesto en el *challenge* se consiguió con esta metodología [17], así como investigar otras arquitecturas. También se podría incrementar el conjunto de entrenamiento utilizando otros *datasets* similares, como proponen Freire et al. en su trabajo sobre el mismo conjunto de datos de PALM [18].

7. Glosario

- **Alta miopía**: miopía de más de 6 dioptrías o con distancia anteroposterior del ojo superior a 26,5 milímetros.

- **CNN**: Convolutional Neural Network, red neuronal convolucional. Red neuronal especializada en

- ***Cross validation*** (validación cruzada): técnica de validación que consiste en particionar los datos en *k folds*, de forma que se realizan *k* entrenamientos del modelo, utilizándose en cada uno de ellos un *fold* diferente como conjunto de validación.

- ***Data augmentation***: técnica para aumentar artificialmente el conjunto de entrenamiento mediante la transformación de las imágenes reales.

- ***Dataset***: conjunto de datos con el que se entrena, valida y prueba un sistema de aprendizaje computacional.

- ***Deep Learning*** (aprendizaje profundo): subcampo del *Machine Learning* capaz de tratar los problemas más complejos de la Inteligencia Artificial mediante redes neuronales artificiales.

- ***Fine-tuning***: técnicas de optimización de modelos de aprendizaje computacional mediante la búsqueda automatizada de valores óptimos de hiperparámetros.

- ***Feature map***: salida generada por un kernel aplicado a una imagen de entrada, que resalta una característica o patrón de la imagen de entrada.

- ***Fold***: cada una de las particiones de datos producida con *cross validation*.

- ***Kernel***: filtro que se aplica a las imágenes de entrada en las capas convolucionales de las CNN para extraer patrones, o en las capas de reducción de muestreo para simplificar la complejidad computacional.

- ***Machine Learning*** (aprendizaje computacional o aprendizaje automático): subcampo de la Inteligencia Artificial dedicado al diseño de sistemas que aprenden de los datos sin estar explícitamente programados.

- **Miopía**: defecto de visión causado por un exceso de curvatura de la córnea, de potencia del cristalino o de distancia anteroposterior del ojo, que provoca la incapacidad de enfocar correctamente objetos lejanos.

- **Miopía patológica**: alta miopía con alguna patología posterior debida a la elongación de la distancia anteroposterior del ojo.

- **Red neuronal artificial**: modelo de *Deep Learning* inspirado en las redes neuronales del cerebro humano, capaz de tratar problemas de Inteligencia Artificial de gran complejidad.

- ***Strafified cross validation*** (validación cruzada estratificada): variación de *cross validation* en la que los *folds* se generan manteniendo la distribución de los datos del conjunto original.

- ***Stride***: cantidad de celdas que se desplazan los filtros o kernels sobre la entrada en las capas convolucionales o de reducción de muestreo de las CNN.

8. Bibliografía

[1] RYAN, Stephen, WILKINSON, Charles, SCHACHAT, Andrew, HINTON, David y WIEDEMANN, Peter. *Retina*. ISBN 9780323401975. China: Elsevier, 2018.

[2] HOLDEN et al. *Global Prevalence of Myopia and High Myopia and Temporal Trends from 2000 through 2050*. San Francisco, California: American Academy of Ophthalmology, 2016.

[3] MRUGACZ, Malgorzata, ÁLVAREZ-PEREGRINA, Cristina C., SÁNCHEZ-TENA, Miguel Ángel M. A., MARTÍNEZ-PÉREZ, Clara C. y VILLA-COLLAR, Cesar C. *Prevalence and Risk Factors of Myopia in Spain*. Hindawi, 2019.

[4] RUIZ ROMERO, José María, ARIAS BARQUET, Luis, GÓMEZ-ULLA, Francisco, SUÁREZ DE FIGUEROA, Marta, GARCÍA ARUMÍ, José, NADAL, Jeroni, FERNÁNDEZ-VEGA SANZ, Álvaro, MONTERO, Javier A, ARMADÁ, Félix, JURGENS, Ignasi y GARCÍA LAYANA, Alfredo. *Manejo de las Complicaciones Retinianas en la Alta Miopía. Guías de Práctica Clínica de la SERV.* ISBN 978-84-608-5908-6. Sociedad Española de Retina y Vítreo, 2014.

[5] RUIZ-MEDRANO, Jorge, MONTERO Javier, A, FLORES-MORENO, Ignacio, ARIAS, Luis, GARCÍA-LAYANA, Alfredo y RUIZ-MORENO José M. *Myopic maculopathy: Current status and proposal for a new classification and grading system (ATN)*. Prog Retin Eye Res. 2019;69:80-115. doi:10.1016/j.preteyeres.2018.10.005

[6] RESNIKOFF, Serge, FELCH, William, GAUTHIER, Tina-Marie, SPIVEY, Bruce. *The number of ophthalmologists in practice and training worldwide: A growing gap despite more than 200,000 practitioners*. The British journal of ophthalmology. 96. 783-7. 10.1136/bjophthalmol-2011-301378. 2012.

[7] GOODFELLOW, Ian, BENGIO, Yoshua y COURVILLE, Aaron. *Deep Learning*. ISBN 978-0262035613. Massachusetts: MIT Press, 2016.

[8] FU, Huazhu, LI, Fei , ORLANDO, José Ignacio, BOGUNOVIC , Hrvoje, SUN, Xu, LIAO, Jingan, XU, Yanwu, ZHANG, Shaochong, ZHANG, Xiulan, *PALM: PAthoLogic Myopia Challenge*, IEEE Dataport, 2019. [en línea] [fecha de consulta: 13/03/2020] Disponible en: http://dx.doi.org/10.21227/55pk-8z03

[9] TORRA I REVENTÓS, Vicenç. *Qué es la inteligencia artificial*. Barcelona: UOC (s/f). Disponible en: http://cvapp.uoc.edu/autors/MostraPDFMaterialAction.do?id=00163094

[10] J. RUSELL, Stuart y NORVIG, Peter. *Artificial Intelligence: A Modern Approach*. Englewood Cliffs, New Jersey: Prentice-Hall Inc, 1995.

[11] GÉRON, Aurélien. *Hand-On Machine Learning with Scikit-Learn, Keras and TensorFlow*. ISBN 978-1-492-03264-9. Sebastopol: O'Reilly, 2019.

[12] TORRA I REVENTÓS, Vicenç y MASIP I RODÓ, David. *Aprendizaje*. Barcelona: UOC (s/f). Disponible en: http://cvapp.uoc.edu/autors/MostraPDFMaterialAction.do?id=200713

[13] FISHER, Ronald. *The use of multiple measurements in taxonomic problems*. 1936

[14] SIMONYAN, Karen y ZISSERMAN, Andrew. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2014.

[15] HE, Kaiming, ZHANG, Xiangyu, REN, Shaoqing y SUN, Jian. *Deep Residual Learning for Image Recognition*. 2015

[16] SZEGEDY, Christian, LIU, Wei, JIA, Yangqing, SERMANET, Pierre, REED, Scott, ANGUELOV, Dragomir, ERHAN, Dumitru, VANHOUCKE, Vincent y RABINOVICH, Andrew. *Going Deeper with Convolutions*. 2015.

[17] XIE, Ruitao, LIU, Libo, LIU, Jingxin y S QIU, Connor. Pathological Myopic Image Analysis with Transfer Learning. 2019.

[18] Rodrigues Freire, Cefas, Da Costa Moura, Julio Cesar, Montenegro da Silva Barros, Daniele y De Medeiros Valentim, Ricardo Alexsandro. Automatic lesion segmentation and Pathological Myopia classification in fundus images. 2019.

[19] *ISBI 2019* [en línea] [fecha de consulta: 18/03/2020]. Disponible en: https://biomedicalimaging.org/2019/

Anexo 1

Tabla 9 *Accuracy (SD)* de mejor a peor obtenida por cada combinación de hiperparámetros generada mediante *Randomized Search* para imágenes de entrada con tamaño 16x16 píxeles

Accuracy	Capas	Filtros L1	Tamaño filtros L1	Filtros n	Tamaño filtros n	Optimizer	Mom.	MaxPool n	MaxPool 1	LR	Epochs	Dropout	Batch
0.928 (0.02)	2	64	5x5	16	3x3	SGD	0.0	2x2	3x3	0.001	30	0.5	2
0.925 (0.03)	1	64	5x5	-	-	SGD	0.0	2x2	3x3	0.001	50	0.3	32
0.655 (0.14)	1	32	5x5	-	-	SGD	0.0	2x2	2x2	0.01	30	0.5	32
0.538 (0.07)	3	32	7x7	64	3x3	Adam	-	2x2	3x3	0.3	50	0.7	2
0.518 (0.08)	2	16	5x5	64	2x2	SGD	0.8	2x2	3x3	0.3	15	0.3	32
0.513 (0.08)	3	16	5x5	16	2x2	SGD	0.4	2x2	2x2	0.1	15	0.7	8
0.483 (0.08)	2	16	5x5	16	2x2	Adam	-	2x2	3x3	0.3	30	0.7	16
0.483 (0.079)	3	32	5x5	64	2x2	SGD	0.0	2x2	2x2	0.1	30	0.5	16
0.468 (0.07)	3	16	5x5	16	2x2	SGD	0.4	2x2	3x3	0.3	50	0.7	4
0.443 (0.06)	3	32	7x7	16	2x2	SGD	0.8	2x2	3x3	0.3	15	0.3	16

Tabla 10 *Accuracy (SD)* de mejor a peor obtenida por cada combinación de hiperparámetros generada mediante *Randomized Search* para imágenes de entrada con tamaño 32x32 píxeles

Accuracy	Capas	Filtros L1	Tamaño filtros L1	Filtros n	Tamaño filtros n	Optimizer	Mom.	MaxPool 1	MaxPool n	LR	Epochs	Dropout	Batch
0.918 (0.02)	3	64	5x5	16	2x2	Adam	-	2x2	2x2	0.001	30	0.7	16
0.915 (0.02)	1	64	5x5	-	-	Adam	-	3x3	2x2	0.001	15	0.3	16
0.898 (0.05)	2	16	7x7	32	3x3	SGD	0.8	3x3	2x2	0.001	50	0.3	32
0.848 (0.02)	1	16	5x5	-	-	SGD	0.4	2x2	2x2	0.001	30	0.7	8
0.835 (0.04)	3	32	7x7	16	2x2	Adam	-	3x3	2x2	0.01	30	0.7	8
0.575 (0.09)	1	32	7x7	-	-	Adam	-	3x3	2x2	0.01	15	0.7	4
0.533 (0.07)	1	16	5x5	-	-	Adam	-	2x2	2x2	0.3	50	0.3	8
0.533 (0.07)	3	16	5x5	16	2x2	Adam	-	2x2	2x2	0.1	30	0.3	32
0.483 (0.08)	1	16	7x7	-	-	SGD	0.8	3x3	2x2	0.1	30	0.7	8
0.480 (0.07)	1	32	5x5	-	-	SGD	0.0	3x3	2x2	0.3	30	0.5	32
0.468 (0.07)	2	64	5x5	64	3x3	SGD	0.0	3x3	2x2	0.3	50	0.7	2
0.443 (0.056)	3	32	7x7	16	2x2	SGD	0.4	2x2	2x2	0.1	50	0.7	4

Tabla 11 *Accuracy (SD)* de mejor a peor obtenida por cada combinación de hiperparámetros generada mediante *Randomized Search* para imágenes de entrada con tamaño 64x64 píxeles

Accuracy	Capas	Filtros L1	Tamaño filtros L1	Filtros n	Tamaño filtros n	Optimizer	Mom.	MaxPool 1	MaxPool n	LR	Epochs	Dropout	Batch
0.888 (0.04)	1	32	5x5	-	-	SGD	0.4	3x3	2x2	0.001	15	0.5	32
0.880 (0.05)	3	16	7x7	32	2x2	Adam	-	2x2	2x2	0.01	50	0.3	8
0.873 (0.05)	2	32	5x5	32	2x2	SGD	0.8	2x2	2x2	0.001	50	0.3	32
0.785 (0.21)	2	32	7x7	16	3x3	SGD	0.8	3x3	2x2	0.001	15	0.3	32
0.745 (0.19)	2	32	7x7	32	3x3	Adam	-	2x2	2x2	0.01	30	0.7	4
0.585 (0.12)	2	16	5x5	64	3x3	Adam	-	2x2	2x2	0.1	30	0.3	32
0.533 (0.07)	1	32	5x5	-	-	SGD	0.4	3x3	2x2	0.1	50	0.5	8
0.513 (0.08)	3	64	7x7	16	3x3	SGD	0.8	2x2	2x2	0.001	15	0.3	2
0.513 (0.08)	2	32	7x7	16	2x2	Adam	-	2x2	2x2	0.1	15	0.5	2
0.488 (0.08)	2	64	7x7	64	2x2	Adam	-	2x2	2x2	0.1	15	0.5	8
0.483 (0.08)	1	16	7x7	-	-	Adam	-	3x3	2x2	0.1	50	0.3	4
0.468 (0.07)	2	64	5x5	64	2x2	SGD	0.0	3x3	2x2	0.3	30	0.3	8

Tabla 12 *Accuracy (SD)* de mejor a peor obtenida por cada combinación de hiperparámetros generada mediante *Randomized Search* para imágenes de entrada con tamaño 128x128 píxeles

Accuracy	Capas	Filtros L1	Tamaño filtros L1	Filtros n	Tamaño filtros n	Optimizer	Mom.	MaxPool 1	MaxPool n	LR	Epochs	Dropout	Batch
0.948 (0.02)	3	16	5x5	16	2x2	Adam	-	3x3	2x2	0.001	'30	0.5	32
0.925 (0.02)	3	16	5x5	64	3x3	Adam	-	3x3	2x2	0.001	'50	0.7	32
0.895 (0.04)	1	64	5x5	-	-	SGD	0.0	3x3	2x2	0.001	'15	0.3	16
0.893 (0.04)	2	64	5x5	64	2x2	Adam	-	2x2	2x2	0.001	'30	0.7	2
0.748 (0.17)	3	16	5x5	64	3x3	Adam	-	2x2	2x2	0.01	'15	0.3	2
0.590 (0.09)	2	64	5x5	64	3x3	Adam	-	3x3	2x2	0.01	'30	0.5	2
0.573 (0.10)	1	16	5x5	-	-	SGD	0.0	2x2	2x2	0.01	'50	0.5	4
0.518 (0.08)	3	32	7x7	32	3x3	SGD	0.0	3x3	2x2	0.3	'50	0.7	16
0.513 (0.08)	2	32	5x5	32	3x3	SGD	0.8	3x3	2x2	0.1	'50	0.3	2
0.488 (0.08)	3	32	5x5	64	3x3	Adam	-	3x3	2x2	0.01	'30	0.7	2
0.468 (0.07)	2	64	7x7	64	2x2	SGD	0.8	2x2	2x2	0.3	'50	0.7	32
0.440 (0.05)	1	32	5x5	-	-	Adam	-	2x2	2x2	0.1	'15	0.3	2

Tabla 13 *Accuracy (SD)* de mejor a peor obtenida por cada combinación de hiperparámetros generada mediante *Randomized Search* para imágenes de entrada con tamaño 256x256 píxeles

Accuracy	Capas	Filtros L1	Tamaño filtros L1	Filtros n	Tamaño filtros n	Optimizer	Mom.	MaxPool 1	MaxPool n	LR	Epochs	Dropout	Batch
0.935 (0.01)	3	32	7x7	16	2x2	Adam	-	2x2	2x2	0.001	50	0.5	32
0.923 (0.01)	2	16	5x5	32	2x2	Adam	-	3x3	2x2	0.001	15	0.7	8
0.908 (0.03)	2	16	5x5	16	3x3	Adam	-	3x3	2x2	0.001	30	0.7	4
0.883 (0.03)	3	32	7x7	16	2x2	Adam	-	3x3	2x2	0.01	30	0.7	16
0.880 (0.047)	1	32	7x7	-	-	SGD	0.8	2x2	2x2	0.001	50	0.7	32
0.798 (0.18)	2	16	7x7	64	3x3	Adam	-	2x2	2x2	0.01	15	0.3	4
0.558 (0.06)	3	64	7x7	32	2x2	SGD	0.4	3x3	2x2	0.01	15	0.7	32
0.538 (0.08)	2	16	5x5	16	3x3	Adam	-	2x2	2x2	0.3	30	0.5	4
0.518 (0.08)	2	32	7x7	32	2x2	SGD	0.8	3x3	2x2	0.01	15	0.7	32
0.483 (0.08)	3	32	5x5	32	3x3	SGD	0.0	3x3	2x2	0.1	30	0.5	16
0.443 (0.06)	3	32	7x7	16	3x3	SGD	0.8	3x3	2x2	0.001	50	0.5	2
0.438 (0.05)	3	16	5x5	16	3x3	Adam	-	3x3	2x2	0.3	30	0.7	4