UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

TRABAJO FIN DE GRADO

Sistema para la detección precoz de cáncer de piel basado en tratamiento de imágenes

Cecilia Serrano Moreno
Tutor: Juan Carlos San Miguel Avedillo
Ponente: José María Martínez Sánchez

Junio 2018

Sistema para la detección precoz de cáncer de piel basado en tratamiento de imágenes

Cecilia Serrano Moreno
Tutor: Juan Carlos San Miguel Avedillo
Ponente: Jose María Martínez Sánchez

Video Processing and Understanding Lab
Departamento de Tecnología Electrónica y de las Comunicaciones
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Junio 2018

Resumen

El objetivo de este trabajo fin de grado es el estudio de un sistema basado en redes neuronales convolucionales desde un punto de vista práctico. Para ello, se tomará como punto de partida un caso concreto de aplicación: una competición internacional de diagnóstico de melanomas a través de técnicas de procesamiento de imágenes. Se comenzará explicando los conceptos más básicos sobre redes neuronales, profundizando en el caso de las redes neuronales convolucionales. A continuación, se dará una visión general del concurso, explorando las partes en que se divide y los recursos que ofrece, y acabando con un resumen de los mejores aportes que se presentaron a la competición en el año 2017.

En el siguiente capítulo, se propondrá un diseño para la tarea de diagnóstico precoz de cáncer de piel. Se utilizará la red convolucional AlexNet, con el fin de estudiar las distintas capas de su arquitectura base para finalmente adaptar la red al objetivo propuesto en el proyecto. En esta etapa se incorporarán algunas de las técnicas más útiles extraídas de la competición. También se resolverán algunos de los problemas que fueron surgiendo durante el desarrollo del diseño.

Por último, se realizará un estudio experimental de los distintos parámetros necesarios para configurar el entrenamiento de una red convolucional, poniendo especial interés en la repercusión de estos sobre las etapas de entrenaminto y de test. A partir de este análisis, se obtendrán unas conclusiones que servirán de base para la construcción de una o varias soluciones que proporcionen unos resultados óptimos.

Palabras clave

Redes neuronales, redes convolucionales, aprendizaje profundo, aprendizaje máquina, melanoma, cáncer de piel

Abstract

The main objective of this Degree's Final Thesis is the study of a system based on convolutional neural networks from a practical point of view. For this, a concrete case of application will be taken: an international challenge of melanoma diagnosis using image processing techniques. The first part will be an explanation of the most basic concepts of neural networks, more precisely in convolutional neural networks. The next step will give a general vision over the challenge, exploring the parts in which it is divided and the resources it provides and finishing with a summary of the participants with the best results in the previous competition (2017).

In the next chapter, a design for the early skin cancer diagnosis task will be proposed. The convolutional network AlexNet will be used for the purpose of studying the different layers of the base architecture to finally adapt the net to de objective suggested at this project. Some of the most useful techniques coming from the challenge will be included in this latter step. Also, some of the problems that were found in the development of the method will be solved.

Finally, an experiment study of the different parameters needed to configure the training of the convolutional network will be achieved, bringing special interest in its impact on the training and test steps. From this analysis, conclusions will be extracted and furtherly used for the construction of one or more solutions that will bring more optimal results.

Keywords

Neural networks, convolutional networks, deep learning, machine learning, melanoma, skin cancer

Agradecimientos

En primer lugar, quiero dar las gracias a mi tutor, Juan Carlos, por su dedicación y su paciencia durante este año.

Quiero dar las gracias a mis padres por apoyarme siempre en todo lo que han podido, y por aguantarse las ganas de matarme en los peores momentos de la carrera. Sin vosotros, jamás habría podido llegar tan lejos.

Tampoco puedo olvidarme de todas esas personas que han aparecido en mi vida gracias a esta locura que hemos emprendido juntos como es estudiar teleco; vosotros me habéis enseñado que hasta la última hora antes de una entrega de prácticas puede convertirse en un buen momento.

Finalmente, darle las gracias a mi novio, Jorge, por soportarme en mis peores momentos y hacer que, pase lo que pase, siempre acabe con una sonrisa en la cara.

Índice general

Resumen v

Abstract vii

Agradecimientos ix

1. Introducción **1**
1.1. Motivación 1
1.2. Objetivos 1
1.3. Organización de la memoria 2

2. Estado del arte **3**
2.1. Introducción 3
2.2. Redes neuronales convolucionales 3
2.2.1. Definición 3
2.2.2. Redes Neuronales Artificiales 3
2.2.3. Etapas de una red convolucional 4
2.3. Entrenamiento de una red neuronal 6
2.4. ISIC 2017: Skin Lesion Analysis Towards Melanoma Detection 9
2.4.1. Dataset 9
2.4.2. Parte 3: Clasificación de lesiones 10
2.4.3. Ranking de participantes en la fase de clasificación 11

3. Diseño y desarrollo **13**
3.1. Introducción 13
3.2. AlexNet 13
3.3. Transfer learning 15
3.4. Desbalanceo de clases 16

4. Evaluación **19**
4.1. Introducción 19
4.2. Marco de evaluación 20
4.2.1. *Dataset* 20
4.2.2. Métricas 20
4.3. Pruebas y resultados 21

4.3.1. Número de épocas 21
4.3.2. Tamaño de mini-bach 23
4.3.3. Learning rate 24
4.3.4. Regularización 25
4.3.5. Data augmentation 26
4.3.6. Conclusión 28
4.4. Resultados finales 28
4.4.1. Reducción de overfitting 29
4.4.2. Mejora de la sensibilidad 30

5. Conclusiones y trabajo futuro **33**
5.1. Conclusiones 33
5.2. Trabajo futuro 34

Bibliografía **35**

Índice de figuras

2.1. Comparación de las conexiones para una capa convolucional (arriba) y una fully-connected (abajo). 5
2.2. Función de activación de una capa ReLU. 6
2.3. Ejemplo de max pooling. 7
2.4. Ejemplos de las tres lesiones a clasificar en la parte 3 de la ISIC 2017. 10

3.1. Ejemplo de extracción de características en la primera capa convolucional de AlexNet. 14
3.2. Arquitectura de AlexNet. 15

4.1. Entrenamiento con configuración base durante 50 épocas 23
4.2. Comparativa de entrenamientos en función del tamaño de mini-batch (B) 24
4.3. Comparativa de entrenamientos en función de learning rate 25
4.4. Comparativa de entrenamientos en función de L_2 26
4.5. Comparativa de entrenamientos en función del tipo de data augmentation 27
4.6. Comparativa de entrenamientos en función del tipo de data augmentation 30
4.7. Comparativa de entrenamientos en función del tipo de data augmentation 31

Índice de tablas

2.1. Tabla comparativa de propuestas del concurso: 12

3.1. Arquitectura de AlexNet 14

4.1. Configuración base utilizada para los experimentos: 19
4.2. Matriz de confusión 20
4.3. Resultados en el conjunto de test para distintas épocas 22
4.4. Resultados en el conjunto de test para distintos tamaños de mini-batch 24
4.5. Resultados en el conjunto de test para distintos valores de learning rate 26
4.6. Resultados en el conjunto de test para distintos valores de L_2 27
4.7. Resultados en el conjunto de test para distintos tipos de data augmentation 28
4.8. Configuración utilizada en el experimento de reducción de overfitting: 29
4.9. Resultados en el conjunto de test para distintos tipos de data augmentation 30
4.10. Configuración utilizada en el experimento de reducción de overfitting: 31
4.11. Resultados en el conjunto de test para distintos tipos de data augmentation 32

Capítulo 1

Introducción

1.1. Motivación

El aprendizaje profundo es una técnica de machine learning que se encuentra en auge en estos momentos. Existe un sinfín de herramientas que soportan el uso de estos métodos, sin embargo, se presenta una gran dificultad al intentar entender todos los parámetros que intervienen en el entrenamiento de una red neuronal. La asignación de unos valores óptimos a estos parámetros no es trivial, de ello pueden depender tanto el correcto funcionamiento del entrenamiento como la obtención de unos resultados satisfactorios. Es por esto que la comprensión de la repercusión de estos parámetros sobre la evolución de un red es fundamental para cualquier proyecto basado en aprendizaje profundo.

1.2. Objetivos

El objetivo de este proyecto consiste en el estudio de los hiperparámetros de una red neuronal convolucional a fin de resolver el problema propuesto en la competición *ISIC 2017: Skin Lesion Analysis Towards Melanoma Detection.* Se realizará un estudio previo de otros aportes a la misma competición con el fin de elegir una configuración inicial sobre la que se efectuarán todos los entrenamientos posteriormente. Además, aunque este no es el objetivo principal, se intentará proporcionar una solución que resuelvan el reto planteado a partir de las conclusiones obtenidas a lo largo del proyecto.

El objetivo principal puede organizarse en varios subobjetivos:

1. Estudio del estado del arte

   Antes de abordar el proyecto, es de capital importancia hacer un estudio previo

del estado del arte. Se explorarán los conceptos básicos de redes neuronales y, en concreto, de redes convolucionales. También se estudiará la competición ISIC 2017, haciendo especial énfasis en las mejores propuestas presentadas.

2. Diseño de una propuesta

   Partiendo del estudio previo, se elegirá y comprenderá una red convolucional que será la que se utilice para experimentar sobre ella.

3. Realización de experimentos

   Utilizando el diseño elegido, se realizarán varios experimentos con el fin de observar la repercusión que tienen los hiperparámetros sobre el entrenamiento de la red. Posteriormente, se probarán una o varias configuraciones que apliquen las conclusiones extraidas para comprobar su utilidad.

1.3. Organización de la memoria

La memoria consta de los siguientes capítulos:

- Capítulo 1. Introducción, motivación y objetivos del proyecto.
- Capítulo 2. Exposición de los conceptos básicos sobre redes neuronales convolucionales. Descripción de la competición en que se basa este TFG y de los recursos proporcionados por esta.
- Capítulo 3. Descripción de la red que se utilizará para los experimentos y de los procedimientos relativos al entrenamiento
- Capítulo 4. Análisis del efecto de distintos parámetros en el ajuste de una red neuronal preentrenada.
- Capítulo 5. Conclusiones y trabajo futuro
- Bibliografía.

Capítulo 2

Estado del arte

2.1. Introducción

Para la realización de este trabajo, se han estudiado los fundamentos básicos de las redes neuronales

2.2. Redes neuronales convolucionales

2.2.1. Definición

Las redes neuronales convolucionales son un tipo concreto de redes neuronales que típicamente se utiliza para el procesamiento de imágenes. La peculiaridad de estas redes neuronales es que utilizan la operación de convolución como sustituto de la multiplicación de matrices. Una convolución es una operación lineal que, en el caso de una imagen, se aplica mediante una ventana conocida como *kernel*; el resultado de esta operación es un filtrado. Gracias al uso de esta operación, es posible reducir enormemente la dimensionalidad de la red mediante el uso de un kernel que tendrá un tamaño más pequeño que el de la imagen de entrada. El objetivo de la red será 'aprender' los valores de cada kernel utilizado, de forma que, tras la aplicación sucesiva de varios kernels, se extraigan características relevantes de la imagen.

2.2.2. Redes Neuronales Artificiales

Una red neuronal artificial es un sistema de aprendizaje automático supervisado que está inspirado en el funcionamiento de las conexiones neuronales del ser humano. Esto no quiere decir que reproduzca el funcionamiento de un cerebro humano, si bien, ambos tienen en común la interconexión de unidades que computan una salida a partir de las distintas entradas que reciben; estas unidades se conocen como *neuronas*. Así,

una red neuronal es un sistema compuesto por un grupo de neuronas organizadas en *capas* que reciben datos y los transforman en una salida.

El objetivo último de una red neuronal es la aproximación de una función f a través de un set de datos conocidos como *datos de entrenamento;* si se define la función como $y = f^*(x)$, los datos de entrenamiento proporcionan ejemplos de x evaluados para distintos puntos, mientras que y se correspondería, suponiendo un problema de clasificación, con la etiqueta que recibe cada entrada. El comportamiento de las capas intermedias de la red depende de unos valores o pesosΘ, pudiendo así reescribir la función objetivo como $y = f(x; \Theta)$; pero los datos de entrenamiento no definen cuáles son estos valores, por eso es necesario un algoritmo de aprendizaje.

Para realizar este aprendizaje, es necesario elegir una función de coste que describa la distancia entre la salida $\widehat{y}$ de la red y el objetivo y. La función de coste es la que da una medida del error cometido, es por eso que el objetivo será minimizarla; para ello, es necesario hacer uso de un algoritmo de optimización, que será el encargado de realizar el entrenamiento de la red. Uno de los ejemplos más utilizados de optimizador en el ámbito de machine learning es el de descenso por gradiente.

Capa fully-connected

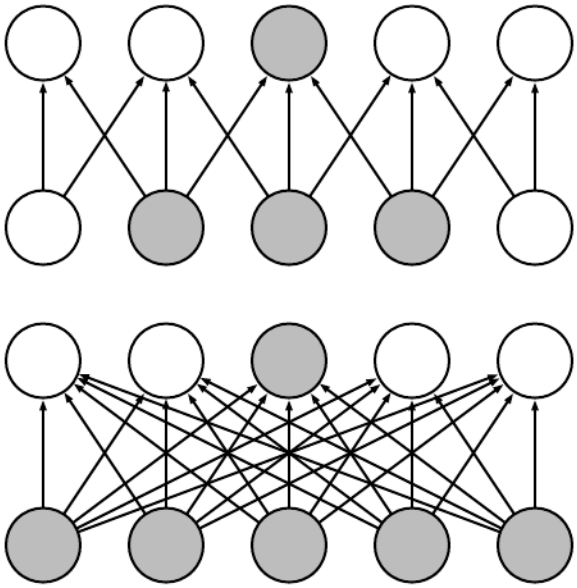
La capa totalmente conectada o *fully-connected* es la red más básica y una de las más utilizadas en redes neuronales. En este tipo de capas, todas las entradas se conectan a todas las salidas, de forma que la operación utilizada en este caso es una multiplicación matricial. Son útiles si se utilizan con entradas de tamaño pequeño, pero para entradas más grandes, pueden no ser prácticas debido al alto coste computacional que introduce la multiplicación de matrices. Son muy utilizadas en las capas finales a modo de clasificador.

2.2.3. Etapas de una red convolucional

Una capa convolucional está típicamente compuesta por tres etapas: la primera está dedicada a las operaciones de convolución, en la segunda el resultado pasa por una función de activación no lineal y en la última se efectúa una operación de pooling.

- Etapa convolucional: La etapa convolucional es la que se dedica a la extracción de características de la imagen de entrada, ya que cada convolución genera un mapa que se activará cuando a la entrada se encuentre con la característica que ha aprendido. El uso de la convolución en estas capas ocasiona que una neurona no dependa de todos los datos de la capa anterior, sino únicamente de una sección de estos, que tendrá el tamaño seleccionado para el kernel; de esta forma,

Figura 2.1: Comparación de las conexiones para una capa convolucional (arriba) y una fully-connected (abajo).

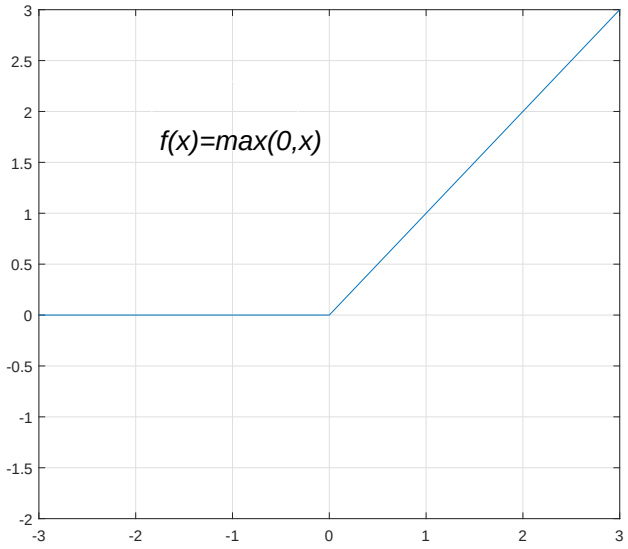


Extraído de
http://www.deeplearningbook.org/contents/convnets.html

la red mantiene la información espacial de la imagen de entrada y, además, al tener menor número de conexiones, disminuirá el número de pesos que es necesario almacenar. En la Figura 2.1 se muestra una comparativa entre una red convolucional y otra fully-connected donde se ve fácilmente la disminución de la cantidad de pesos para una capa convolucional.

- ReLU (Rectified Linear Unit): La capa ReLU o rectificadora es una de las opciones que se pueden utilizar como función de activación no lineal. A diferencia de otras funciones con la misma finalidad, como la sigmoide o la tangente hiperbólica, esta función no es saturante y es más similar a una función lineal, lo que hace que un modelo basado en este tipo de función sea mucho más fácil de optimizar y, por consiguiente, que el entrenamiento sea más rápido. En la Figura 2.2 se observa la expresión y el aspecto de la función de activación de este tipo de capas. Cabe señalar que en esta etapa también se pueden utilizar otros tipos de función de activación, como las ya mencionadas, pero la ReLU es muy utilizada gracias que agiliza en gran medida el entrenamiento sin dejar de dar buenos resultados.

Figura 2.2: Función de activación de una capa ReLU.

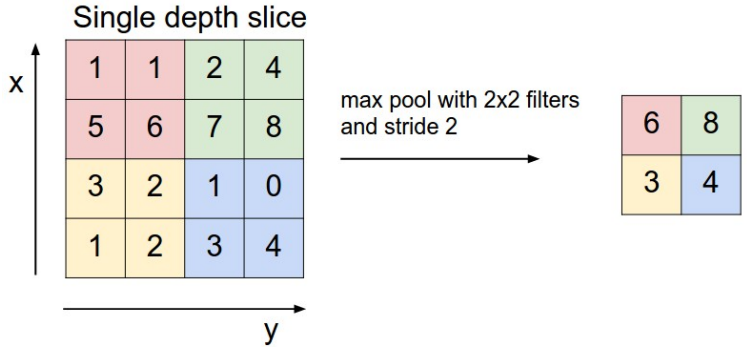


- Pooling: Esta etapa se utiliza para submuestrear la entrada, eliminando parte de la información espacial. Este tipo de capa toma una ventana de la entrada y la reduce a un sólo valor, que generalmente es el máximo o la media de los datos de entrada. El objetivo principal de esta etapa es la reducción de la dimensionalidad del resultado de la etapa anterior, de forma que, al reducir el número de parámetros, es posible ampliar la profundidad de las capas y obtener características más complejas. En la Figura 2.3 se puede ver un ejemplo de pooling utilizando el máximo de la ventana de entrada como salida.

2.3. Entrenamiento de una red neuronal

Durante el entrenamiento de una red neuronal se hace uso de un algoritmo de aprendizaje que optimice la función de coste elegida. Un algoritmo de este estilo modifica los pesos de la red iterativamente hasta llegar a un punto en que satisfaga unas condiciones previamente establecidas para la función de coste. Para realizar este proceso, es necesario calcular una función de coste que cuantifique el error cometido a la salida de la red y utilizar un optimizador que reduzca lo más posible el valor de esta función. El optimizador calcula el gradiente de la función de coste, que se utilizará para modificar ligeramente los pesos. Con los pesos actualizados, se computa nuevamente la salida de la red y se repite el proceso descrito.

Figura 2.3: Ejemplo de max pooling.



Extraído de http://cs231n.github.io/convolutional-networks/

Para que el proceso descrito funcione, es necesario disponer de una cantidad considerable de datos de entrenamiento a partir de los cuales se obtenga la salida de la red. Para denominar al set de datos que se utilizan durante el entrenamiento, es habitual el uso del término *batch*. Idealmente, se utilizaría el batch completo cada vez que se realiza el proceso de aprendizaje, sin embargo, es muy común que no se puedan utilizar todos los datos en una sóla iteración debido a las limitaciones del hardware —principalmente, se debe a la falta de memoria, ya que el espacio necesario para realizar un entrenamiento depende linealmente del tamaño del batch—; en estos casos, se toman subconjuntos más manejables que se puedan utilizar de forma simultánea. Un subconjunto de este tipo se suele denominar *mini-batch*.

De esta forma, una iteración, además de corresponderse con cada actualización de los pesos, también coincide con una pasada de un mini-batch por la red. Análogamente, cada vez que se utilice el batch completo, se dice que se ha completado una *época*. Para entrenar una red, es necesario repetir el proceso durante varias épocas, ya que con una sóla no es suficiente para que los pesos de la red lleguen a generalizar la función objetivo.

Además, existe un parámetro que afecta tanto a la rapidez con que avanza el entrenamiento como a la convergencia de los pesos de la red, este parámetro es la tasa de aprendizaje o *learning rate*. Este valor actúa directamente sobre la actualización de los pesos, ponderando la variación que se les aplica en cada iteración. Si la learning rate es muy pequeña, los pesos apenas cambiarán, con lo que el entrenamiento evolucionará de forma muy lenta; además, cabe la posibilidad de que la red se quede en un mínimo local y no aprenda más. Por el contrario, si la tasa es demasiado elevada, es posible que las modificaciones de los pesos sean demasiado grandes, haciendo que estos oscilen a lo largo del entrenamiento sin llegar a converger.

Al finalizar un entrenamiento, es necesario comprobar que la red que se ha al-

canzado se comporta de la forma esperada al recibir unos datos que no sean los de entrenamiento, para ello, es necesario utilizar un conjunto distinto con el que se testeará el funcionamiento de la red. Hay tres posibles situaciones al hacer esta comprobación:

- Red subentrenada: El subentrenamiento o *underfitting* se obtiene cuando el porcentaje de acierto es bajo tanto para los datos de entrenamiento como para los datos de test. Esto significa que la red no ha sido bien entrenada y muchas veces se solucionará aumentando el número de épocas del entrenamiento.

- Red correctamente entrenada: En este caso, el porcentaje de acierto será alto tanto para la fase de entrenamiento como para la de test. Esto quiere decir que la generalización de la función objetivo se ha conseguido con éxito, y el problema queda resuelto.

- Red sobreentrenada: El *overfitting* o sobreentrenamiento se identifica con un porcentaje de acierto muy alto en la fase de entrenamiento pero sensiblemente más bajo en la de test. Este problema suele deberse a que los pesos de la red se han ajustado demasiado a los datos de entrenamiento, por lo que no funcionan correctamente al utilizarlos para unos datos nuevos. Evitar este problema no es una tarea sencilla, sin embargo, existen muchos procedimientos que tratan de solucionarlo.

Para evitar el overfitting, hay una gran cantidad de técnicas denominadas de *regularización* que buscan soluciones a este problema. Una de las más utilizadas es la regularización L_2, cuyo objetivo es evitar una complejidad innecesaria en el modelo que pueda desembocar en sobreentrenamiento. Para ello, se calcula la norma L_2 de los pesos y se comprueba que no tome un valor demasiado elevado —el umbral de decisión se establece como un parámetro más del entrenamiento—.

Otro método muy útil a la hora de evitar el sobreentrenamiento es el de *data augmentation,* que consiste en un aumento artificial de los datos mediante transformaciones aleatorias en el conjunto de entrenamiento. Este procedimiento se suele utilizar cuando los datos de entrada son imágenes y, sobre ellas, se efectúan transformaciones geométricas como escalados o rotaciones. La principal motivación del data augmentation es que la red no se 'aprenda' las imágenes de entrada en su posición original.

2.4. ISIC 2017: Skin Lesion Analysis Towards Melanoma Detection

ISIC 2017: Skin Lesion Analysis Towards Melanoma Detection es una competición internacional de diagnóstico de melanomas organizada por la ISIC (International Skin Imaging Collaboration). La organización proporciona un dataset formado por 2000 imágenes de test, 150 imágenes de validación y 600 imágenes de test, reservadas para la evaluación final de cada trabajo —el groundtruth de estas últimas se proporciona una vez acabada la competición—.

El concurso se divide en tres partes independientes entre sí, cada una dedicada a una tarea:

- Parte 1: Segmentación de lesiones.
- Parte 2: Extracción de características.
- Parte 3: Clasificación de lesiones.

Este trabajo se centra en la parte 3, ya que es la única dedicada a la obtención de un diagnóstico para el cáncer de piel.

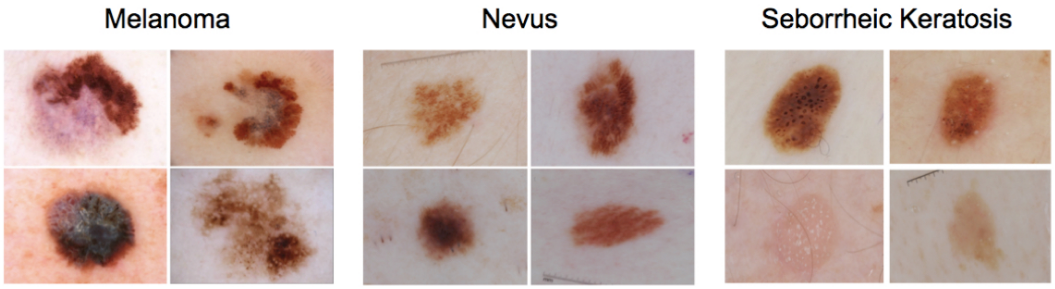
2.4.1. Dataset

El dataset proporcionado no está formado por imágenes corrientes, sino por fotografías dermatoscópicas, esto es, han sido tomadas con un aparato especial que permite visualizar capas más profundas de la piel. Esta técnica se utiliza para eliminar la reflexión y refracción de la epidermis y puede actuar mediante dos métodos: o bien un filtro de luz polarizada o bien un aceite de inmersión, ambos procedimientos con la ventaja de ser no invasivos, es decir, que no es necesario penetrar la piel del paciente.

Los recursos proporcionados por la competición son:

- 2000 imágenes dermatoscópicas de entrenamiento acompañadas de dos ficheros con datos clínicos del paciente (edad y sexo) y el ground truth data que las clasifica entre melanoma, nevus o queratosis seborréica.
- 150 imágenes dermatoscópicas de validación, utilizadas para testear durante el desarrollo de la competición. Una vez concluida la competición, se publica también del ground truth de este conjunto.
- 600 imágenes dermatoscópicas de test que son publicadas junto con su ground truth al finalizar la competición.

Figura 2.4: Ejemplos de las tres lesiones a clasificar en la parte 3 de la ISIC 2017.



Extraído de
https://challenge.kitware.com/#phase/5840f53ccad3a51cc66c8dab

2.4.2. Parte 3: Clasificación de lesiones

La tercera parte es la que se dedica propiamente al diagnóstico de lesiones, es por eso que este proyecto se centrará en la búsqueda de una solución para este apartado. Los tres tipos lesión a clasificar son:

- Melanoma: Es un tumor maligno y derivado de melanocitos. El dataset proporcionado dispone de 374 imágenes de melanomas.

- Nevus: Tumor benigno derivado de melanocitos. El dataset dispone de 1372 imágenes correspondientes a nevus.

- Queratosis seborréica: Tumor benigno no derivado de melanocitos. El dataset dispone de 254 imágenes clasifidas como queratosis.

En la Figura 2.4 se pueden ver los tres tipos distintos de lesiones a clasificar.

La tarea se divide en dos clasificaciones binarias: la primera de ellas trata de aislar los melanomas del resto de lesiones y la segunda hace lo mismo para el caso de queratosis seborréica. En este proyecto no se abarcará toda la parte 3, sino únicamente una de las dos clasificaciones, ya que el principal interés de este TFG es el estudio de los elementos que influyen en el entrenamiento de una red neuronal convolucional; si quisiéramos realizar la segunda clasificación, bastaría con seguir el mismo procedimiento para un dataset distribuido de forma distinta. En concreto, se ha elegido abordar la clasificación de melanomas vs. resto de lesiones, ya que la única lesión cancerígena, y la más letal del conjunto, es el melanoma y un diagnóstico precoz puede marcar la diferencia a la hora de salvar la vida de un paciente.

2.4.3. Ranking de participantes en la fase de clasificación

Antes de abordar el diseño de una solución, el primer paso ha sido el estudio de las mejores aproximaciones que se presentaron en la competición de 2017. En la tabla 2.1 se puede observar un resumen de las características más notables de cada uno de los proyectos estudiados.

Del estudio de estas propuestas se extraen algunas conclusiones a tener en cuenta en el desarrollo del proyecto:

- Todos los modelos estudiados utilizan redes neuronales convolucionales y residuales. Además, estas no se entrenan de cero, sino que toman una red base ya entrenada y ajustan las últimas capas para adaptar el modelo al problema que se aborda. Esta técnica se llama *transfer learning* y se explicará en más detalle en el apartado .

- De seis participantes, sólo uno introduce una etapa de extracción de estructuras de interés. Esto se debe a que no es necesario insertar una fase de extracción de características si se utiliza una red convolucional, ya que estas realizan ese trabajo de forma automática.

- Puede interesar el uso de técnicas de data augmentation o de normalización previa de las imágenes.

- El uso de los datos clínicos adicionales no es demasiado útil para la clasificación de melanomas, ya que únicamente lo utiliza un participante de los seis seleccionados.

- La ampliación del dataset mediante otras imágenes adicionales puede ser de gran ayuda, sin embargo, la cantidad de datos que se utilicen para el entrenamiento influye directamente sobre el tiempo necesario para este, lo cual puede ser un gran problema.

Tabla 2.1: Tabla comparativa de propuestas del concurso:

	Casio & Shishu [1]	Multimedia Processing Group - UC3M [2]	RECOD Titans [3]	USYD-BMIT [4]	IHPC [5]	University of Guelph [6]
Clasificación en parte 3	#1	#2	#3	#4	#5	#6
Dataset adicional	Sí	Adición de anotaciones sobre la estructura de las lesiones	Sí	Sí	No	Sí
Pre-procesamiento	Normalización de color y luminancia	Coordenadas polares	Normalización mediante la media	Reescalado	Normalización mediante media y desviación estándar	Reescalado
Segmentación	No	Sí	No	Sí	Sí	No
Data Augmentation	Sí	Sí	Sí	Sí	Sí	Sí
Extracción de características adicional	No	Estructuras de interés	No	No	No	No
Red base	ResNet-50	ResNet-50	ResNet-101 e Inception-v4	ResNet	GoogleNet y U-Net	Inception-v3
Uso de datos edad/sexo	Sólo en queratosis	Sí	No	No	No	No

Capítulo 3

Diseño y desarrollo

3.1. Introducción

3.2. AlexNet

AlexNet [7] es una red convolucional que se presentó a la competición ILSVRC-2012[1], alcanzando el top-5 con una tasa de error de 15.3 %. La arquitectura de la red, que se puede ver en la Tabla 3.1, consta de cinco capas convolucionales y tres capas fully-connected que se explicarán con más detalle a continuación.

La primera capa de la red es la que recibe las imágenes de entrada, cuya dimensión debe ser de 227x227x3. Se debe destacar que en esta capa se realiza una normalización de tipo *zero center,* que consiste en la extracción de la imagen media del dataset.

Las cinco capas convolucionales tienen como objetivo la extracción de características. Las dos primeras incluyen una etapa de pooling de 3x3 con solapamiento que se utiliza para reducir las dimensiones de la salida de la red, de forma que en la siguiente etapa se aumente la profundidad de la capa de convolución. Esto ayuda a conseguir unas características más complejas, aunque esto implique eliminar parte de la información espacial. Para entender la extracción de características, se puede observar la Figura

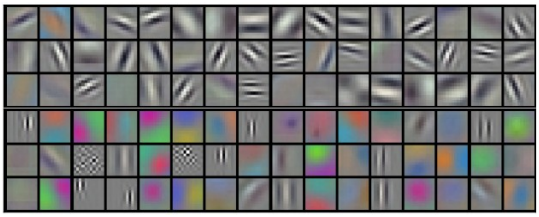
Las dos capas siguientes tienen la función de efectuar la clasificación. Para ello, se utilizan capas fully-connected donde las neuronas están totalmente conectadas entre sí. Las dos capas de *dropout* se utilizan como método para reducir el overfitting. El funcionamiento del dropout consiste en poner a cero la salida de algunas neuronas pertenecientes a las capas ocultas. De esta forma, se evita que los pesos se adapten

[1]ImageNet Large Scale Visual Recognition Challenge es una competición que consiste en la clasificación automática de más de 15 millones imágenes previamente etiquetadas por seres humanos entre unas 22000 categorías

Tabla 3.1: Arquitectura de AlexNet

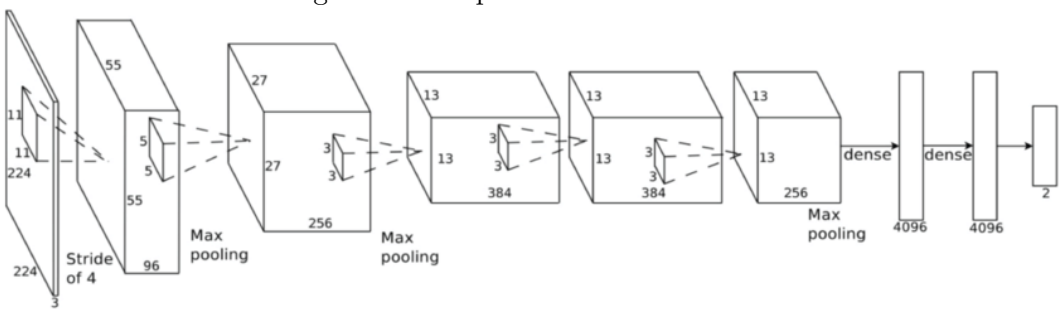
Capa	Etapas	Dismensionalidad
Capa de entrada	Normalización zerocenter	277x277x3
Capa convolucional	Convolucional	96 convoluciones de 11x11x3
	ReLU	—
	Normalización entre canales	Normalización con 5 canales
	Pooling	3x3
Capa convolucional	Convolucional	256 convoluciones de 5x5x48
	ReLU	—
	Normalización entre canales	Normalización con 5 canales
	Pooling	3x3 max pooling
Capa convolucional	Convolucional	384 convoluciones de 3x3x256
	ReLU	—
Capa convolucional	Convolucional	384 convoluciones de 3x3x192
	ReLU	—
Capa convolucional	Convolucional	256 3x3x192
	ReLU	—
	Pooling	3x3
Capa fully-connected	Fully-connected	4096
	ReLU	—
	Dropout 50 %	—
Capa fully-connected	Fully-connected	4096
	ReLU	—
	Dropout 50 %	—
Capa final	Fully-connected	1000
	Softmax	—
	Clasificación	—

Figura 3.1: Ejemplo de extracción de características en la primera capa convolucional de AlexNet.



Extraído de http://cs231n.github.io/convolutional-networks/

Figura 3.2: Arquitectura de AlexNet.



Extraído de https://www.researchgate.net/figure/AlexNet-CNN-architecture-layers_fig1_318168077

en exceso a los datos de entrenamiento, ya que algunos de ellos se eliminan de forma aleatoria. La probabilidad de eliminar un peso utilizada en este caso es del 50 %, un valor utilizado muy habitualmente para este parámetro.

La capa final utiliza una fully-connected con un tamaño de salida de 1000, ya que este es el número de clases para el que fue entrenada. Después de esta capa, se encuentra una softmax, que calcula la probabilidad del dato de entrada de pertenecer a cualquiera de las clases. Finalmente, la capa de clasificación que se encargan de dar una solución final a partir de las probabilidades obtenidas de la softmax.

En la Figura 3.2 se puede ver gráficamente cómo funciona cada capa de AlexNet. Por ejemplo, las capas convolucionales se representan como cajas más anchas que las fully-connected, lo que da información sobre la profundidad de la capa.

Se ha elegido esta red para el desarrollo de los experimentos debido a que es una red relativamente pequeña y sencilla de entender, de cara a la obtención de unas conclusiones consistentes. Además, por limitaciones del hardware disponible, resulta más eficiente el entrenamiento de una red pequeña.

3.3. Transfer learning

Una de las cuestiones más problemáticas que hay que enfrentar en deep learning es la gran cantidad de tiempo que se necesita para entrenar una red; dependiendo de la profundidad de la red escogida y de las características del hardware del que se disponga, un entrenamiento puede durar de semanas a meses. Es por esto que, en lugar de entrenar una red de cero, se ha optado por utilizar la técnica de *transfer learning*.

El transfer learning es una técnica englobada dentro del deep learning en la cual se utiliza una red base que haya sido entrenada previamente para una tarea similar

a la que se quiere abordar, de forma que, en lugar de entrenar una red de cero, la tarea se simplifica al ajuste fino de las últimas capas de la red seleccionada para adaptarla al problema en cuestión y se comienza el entrenamiento partiendo de los pesos proporcionados por la red original.

En este proyecto se utilizarán los pesos de la red AlexNet que se entrenó para la participación en el concurso ILSVRC-2012, ya que esta desempeñaba una tarea de clasificación de imágenes. La principal diferencia entre la AlexNet preentrenada y la red que se quiere obtener es que la primera está entrenada para clasificar entre 1000 clases distintas de objetos, mientras que en este proyecto la clasificación es binaria. Es por esto que habrá que cambiar la capa final de la red de forma que la etapa de clasificación esté dedicada únicamente a dos clases. De esta forma, la extracción de características se realiza de forma idéntica a la red inicial, sin embargo, la última capa se adapta al problema de diagnóstico de cáncer de piel.

3.4. Desbalanceo de clases

Antes de comenzar con los experimentos, surge la necesidad de solucionar un problema con el dataset: el desbalanceo de clases. Si se observa la distribución del set de entrenamiento, se puede advertir que sólo hay 374 imágenes pertenecientes a la clase melanoma frente a 1626 imágenes de queratosis y nevus. Este problema ya ha sido estudiado en otros trabajos [8] y de las conclusiones se extrae que si se utiliza un dataset desbalanceado para una tarea de clasificación, ya sea binaria o de múltiples clases, el resultado favorecerá a la correcta clasificación de las clases más numerosas y proporcionará peores resultados en las clases menos numerosas.

En el caso del dataset utilizado, el desbalanceo de clases provoca que la clase melanoma, obtenga unos resultados de clasificación muy malos en comparación con el rendimiento global. este problema es especialmente grave, ya que el objetivo principal es una buena clasificación de melanomas —es preferible dar un diagnóstico positivo erróneo que dejar pasar un caso de cáncer de piel, ya que la vida del paciente está en juego—.

A continuación, se explicarán las dos técnicas propuestas para solucionar el problema del desbalanceo de clases. Ambas trabajan a nivel de datos, es decir, modifican el dataset sin necesidad de alterar la red a entrenar.

Sobremuestreado de la clase minoritaria

Este método es ampliamente utilizado para balancear un dataset, y el procedimiento sencillamente consiste en la replicación aleatoria de los datos pertenecientes a

la clase minoritaria hasta conseguir una distribución uniforme de clases. El problema que puede surgir con el uso de este método es un potencial sobreentrenamiento de la red, ya que favorece a la pérdida de generalidad por parte de la red. Es por esto que el uso de métodos de regularización será fundamental.

Submuestreado de la clase mayoritaria

Este método hace exactamente lo contrario que el anterior: en lugar de tomar la totalidad de los datos de la clase mayoritaria, se queda únicamente con una subsección aleatoria de esta que tenga un tamaño similar al de la clase minoritaria. Esta técnica es útil cuando se dispone de un dataset lo suficientemente grande como para que la eliminación de muestras no afecte al rendimiento del entrenamiento de la red. Lamentablemente, en el caso de este proyecto disponemos de un set de datos relativamente pequeño, por lo que el submuestreado no da buenos resultados.

En vista de los dos métodos expuestos, se optará por el uso del sobremuestreado de la clase minoritaria, ya que el dataset disponible es bastante reducido, con lo que un submuestreo agravaría este problema.

Capítulo 4

Evaluación

4.1. Introducción

En este capítulo se mostrarán los distintos experimentos a partir de los cuales se trata de entender los hiperparámetros más relevantes de una red neuronal convolucional y el impacto que tienen sobre el comportamiento de esta. Para ello, se ha tomado una configuración base, de forma que, para cada experimento, se utilizan los mismos hiperparámetros cambiando únicamente aquellos que sean objeto de estudio en ese caso; de esta forma es más sencillo comparar los resultados finales. La configuración base que se ha utilzado se muestra en la tabla 4.1.

Para realizar estos experimentos se ha utilizado como herramienta Matlab, ya que el toolbox Neural Networks permite cargar rápidamente una red AlexNet preentrenada. Se ha utilizado una GTX 850M para la ejecución de los experimentos y se ha estimado que, para el caso de 25 épocas, un entrenamiento tiene una duración de entre 7 y 8 horas. Durante todos los experimentos, el objetivo será la clasificación de melanomas frente al resto de lesiones.

Tabla 4.1: Configuración base utilizada para los experimentos:

Tamaño de mini-batch	Número de épocas	Learning rate	Factor L_2	Data augmentation
128	25	0.001	0.0001	No

4.2. Marco de evaluación

4.2.1. *Dataset*

El *dataset* utilizado ha sido el que proporciona la competición ISIC 2017 y, para evitar el problema de desbalanceo de clases, se le ha aplicado un sobremuestreo a la clase minoritaria. En el caso de disponer de un dataset pequeño, como es este caso, es habitual utilizar particiones aleatorias para cada etapa (entrenamiento, validación y test) con el fin de no obtener un resultado sesgado debido a la división del dataset, sin embargo, en este caso se han utilizado las particiones fijas establecidas por la competición. Se asume que la organización del ISIC 2017 se ha encargado de proporcionar unos subsets con una distribución uniforme, de forma que este factor no afecte a los resultados del proyecto. Esta decisión permite evitar la repetición de entrenamiento y obtención de resultados, lo que agiliza sensiblemente el trabajo.

Por otra parte, por simplicidad, no se ha utilizado el set de validación, ya que el objetivo de este proyecto no es la selección de los mejores modelos obtenidos, sino el estudio de varias configuraciones para el entrenamiento de una red; es por esto que las etapas se han reducido a entrenamiento y test.

4.2.2. Métricas

Dentro del ámbito de machine learning, en un problema de clasificación, es común evaluar los resultados dividiéndolos mediante una matriz de confusión como la de la Tabla 4.2, donde un Verdadero Positivo (VP) es un dato correspondiente a la clase positiva que ha sido correctamente clasificado, Falso Positivo (FP) es un dato correspondiente a la clase negativa que ha sido erróneamente clasificado, Verdadero Negativo (VN) es un dato correspondiente a la clase negativa que ha sido correctamente clasificado y Falso Negativo (FN) es un dato correspondiente a la clase positiva que ha sido erróneamente clasificado.

Tabla 4.2: Matriz de confusión

		Clasificación	
		Positivos	Negativos
Ground-truth	Positivos	VP	FN
	Negativos	FP	VN

Por ser este un problema de diagnóstico médico, la clase positiva corresponderá

a las imágenes de melanomas y la negativa al resto de lesiones cutáneas. Dada esta nomenclatura, las tres medidas que utilizaremos son:

- Accuracy: Mide la proporción entre las muestras bien clasificadas y el total de datos. Arroja una idea sobre el rendimiento global del sistema a evaluar. Nótese que el rango de valores para esta medida estará entre 0.5 y 1.

$$Accuracy = \frac{VP + VN}{VP + VN + FP + FN} \tag{4.1}$$

- Sensibilidad: Mide la proporción entre los verdaderos positivos y el total de positivos. Esta medida se utilizará para evaluar la efectividad de la clasificación de los melanomas. El rango de valores de esta medida estará entre 0 y 1.

$$Sensibilidad = \frac{VP}{VP + FN} \tag{4.2}$$

- Fall-out: Mide la proporción entre los falsos positivos y el total de casos negativos. Se utilizará para saber qué porcentaje de lesiones no cancerígenas se han diagnisticado erróneamente como melanoma. El rango de valores de esta medida estará entre 0 y 1.

$$Fall - out = \frac{FP}{FP + TN} \tag{4.3}$$

4.3. Pruebas y resultados

En los siguientes apartados, se muestran los distintos experimentos que se han llevado a cabo. Para exhibir los resultados, se muestran figuras de la accuracy y de la función de coste o pérdida (también llamada función de loss) para el conjunto de entrenamiento. Además, se proporciona una tabla en cada experimento con los resultados en la fase de test.

4.3.1. Número de épocas

Es importante utilizar un número de épocas adecuado para asegurar la convergencia de la red. Por otra parte, estudiar este parámetro puede ser de utilidad para identificar la aparición de sobreentrenamiento. En este experimento, se ha entrenado la red durante cincuenta épocas utilizando la configuración base. En la Figura 4.1, se muestra la accuracy de la clasificación de los datos durante la fase de entrenamiento y el valor de la función de pérdida.

Tabla 4.3: Resultados en el conjunto de test para distintas épocas

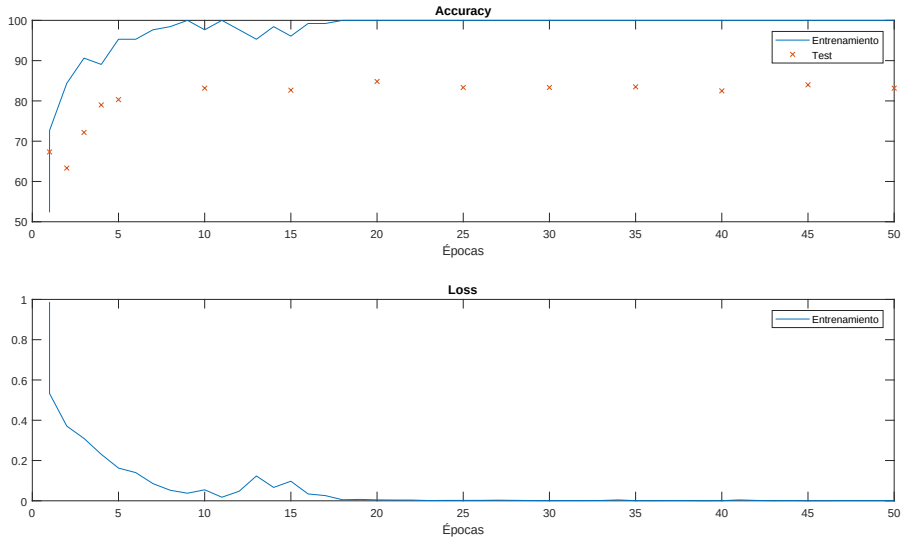
Época	Accuracy	Sensibilidad	Fall-out
5	0.8033	0.7094	0.1739
10	0.8317	0.5299	0.0952
15	0.8267	0.2308	0.0290
20	0.8483	0.4274	0.0497
25	0.8333	0.4530	0.0745
30	0.8333	0.4103	0.0642
35	0.8350	0.3932	0.0580
40	0.8250	0.4615	0.0870
45	0.8400	0.4274	0.0600
50	0.8317	0.3932	0.062

El primer objetivo de este experimento es identificar a partir de qué época se puede considerar que la red ha convergido, es decir, la función de loss no cambia significativamente. Como se ve en la Figura 4.1, a partir de la época 20 la función de loss se queda estancada en un valor muy cercano al cero (inferior a 0.005), por lo que se puede considerar que está convergiendo a partir de ese punto. Además, se observa que la accuracy tiene un comportamiento similar, alcanzando valores muy cercanos al 100 % a partir de la época 20.

Para identificar fácilmente cuándo la red comienza a presentar overfitting, es necesario fijarse en los resultados que se obtienen a diferentes épocas en el conjunto de test. Como se observa en la Figura 4.1, la red comienza a sobreentrenarse prácticamente desde la segunda época, ya que la accuracy para los resultados de test es inferior a la que se obtiene en el entrenamiento.

Para obtener los resultados del test en distintos puntos, se han establecido *checkpoints* para almacenar los pesos de la red en cada época, y posteriormente se ha probado su funcionamiento para el set de test. Este procedimiento es muy práctico cuando se trata de evitar el overfitting mediante *early stopping* —el early stopping es un método de regularización en el cual se detiene el entrenamiento antes de que la red empiece a sobreentrenarse—, ya que se puede elegir *a posteriori* la red más óptima. Un ejemplo de esta práctica para el caso de estudio sería la selección de la red entrenada únicamente hasta las cinco épocas, ya que esta obtiene una sensibilidad más alta que la de los resultados posteriores y, aunque la medida de fall-out es peor, puede considerarse que no es algo especialmente grave.

Figura 4.1: Entrenamiento con configuración base durante 50 épocas



4.3.2. Tamaño de mini-bach

En este experimento se han probado varios valores para el tamaño de mini-batch para comprobar su repercusión en la convergencia de la red. Es importante señalar que los valores mostrados para la accuracy y la función de loss han sido tomados del último mini-batch entrenado en cada época y no del batch completo, ya que es lo habitual durante el entrenamiento de una red neuronal —la obtención de estas medidas para todo el batch ralentizaría significativamente el experimento—, de ahí que algunas oscilaciones sean especialmente abruptas. Lo ideal para conseguir un estudio completo sobre este parámetro sería probar valores de mini-batch más grandes, llegando a igualar el tamaño del batch, sin embargo, esto no ha sido posible debido a las limitaciones del hardware disponible.

Al observar la Figura 4.2, se puede apreciar que, a menor mini-batch, las oscilaciones de la accuracy y la función de pérdida son mucho más pronunciadas; esto quiere decir que los pesos fluctúan más en estos casos debido a que la cantidad de datos es insuficiente para poder obtener un resultado generalizado. En el caso del mini-batch más grande que se ha probado, de 128, las oscilaciones son mucho más suaves y esto hace que la convergencia más lenta. En la Tabla 4.4 se puede comprobar que, a pesar de que los entrenamientos con mini-batch de 16 y 64 converjan más rápido, las tres medidas de evaluación son perceptiblemente mejores en el caso de 128.

En vista de estos resultados, se desprende que será siempre recomendable utilizar el batch más grande que se pueda en función de los recursos disponibles.

Figura 4.2: Comparativa de entrenamientos en función del tamaño de mini-batch (B)

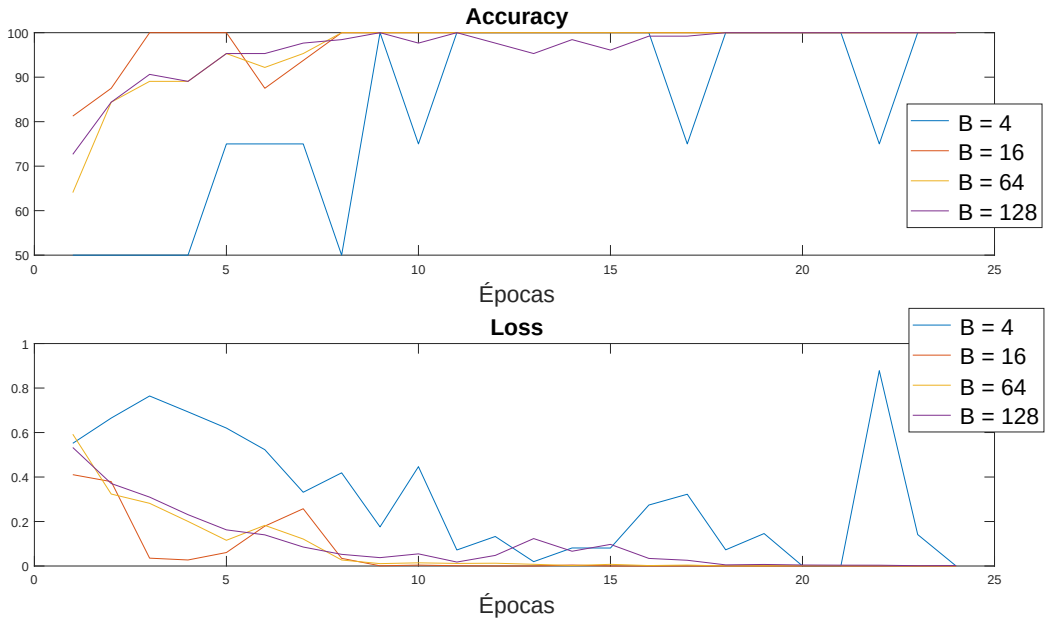


Tabla 4.4: Resultados en el conjunto de test para distintos tamaños de mini-batch

B	Accuracy	Sensibilidad	Fall-out
4	0.7667	0.2479	0.1077
16	0.8017	0.2479	0.0642
64	0.8233	0.2735	0.0435
128	0.8317	0.3932	0.0621

4.3.3. Learning rate

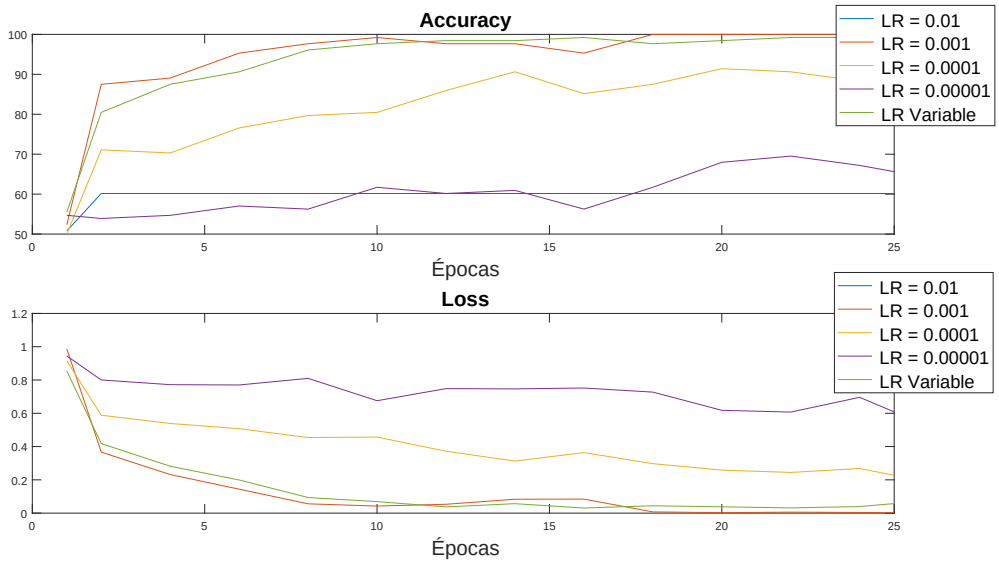
En este experimento se utilizan distintos valores para la tasa de aprendizaje o *learning rate* para comprobar su efecto sobre la convergencia de la red. En la Figura 4.3 se observa que, al utilizar un valor muy bajo, la red aprende de forma muy lenta, por lo que el entrenamiento necesita más épocas para que los pesos de la red comiencen a converger. Además, al permitir una variabilidad tan baja, aumenta la probabilidad de que la función de loss converja a un falso mínimo, con lo que se quedaría estancada entorno a un valor que podría no ser el deseado.

Otro efecto relevante es el de una tasa demasiado alta, que provoca una oscilación tan grande que nunca llega a alcanzar un punto óptimo. Esto es lo que sucede al elegir

el valor de 0.01: aunque no se aprecia bien en la figura, debido a que no se muestra el comportamiento durante todas las épocas, la accuracy está oscilando entre 50 % y 60 %, lo que quiere decir que no se consigue ningún aprendizaje —una posible explicación es que todos los datos estén siendo clasificados como positivos en unas iteraciones o como negativos en otras, sin llegar a un punto intermedio donde se pueda alcanzar un mejor resultado, por eso la accuracy es siempre tan baja.

Por último, es interesante fijarse en el caso de una tasa de aprendizaje variable. En la que se muestra en la Figura 4.3, se ha comenzado con un valor de 0.001 que se ha dividido por 10 cada ocho épocas. El resultado es una curva con una pendiente pronunciada en las primeras épocas, pero con pocas oscilaciones hacia épocas más avanzadas. Este efecto puede ser de gran ayuda tanto para acelerar el entrenamiento en las primeras épocas como para evitar el overfitting en las últimas, ya que el uso de una learning rate pequeña puede forzar un punto de convergencia distinto. Esto se puede comprobar si se comparan los resultados de test en la Tabla 4.5, donde se obsera que la accuracy final de la red con learning rate 0.001 es idéntica a la que utiliza una tasa variable; además, en el caso de learning rate variable se mejora un 10 % la sensibilidad.

Figura 4.3: Comparativa de entrenamientos en función de learning rate



4.3.4. Regularización

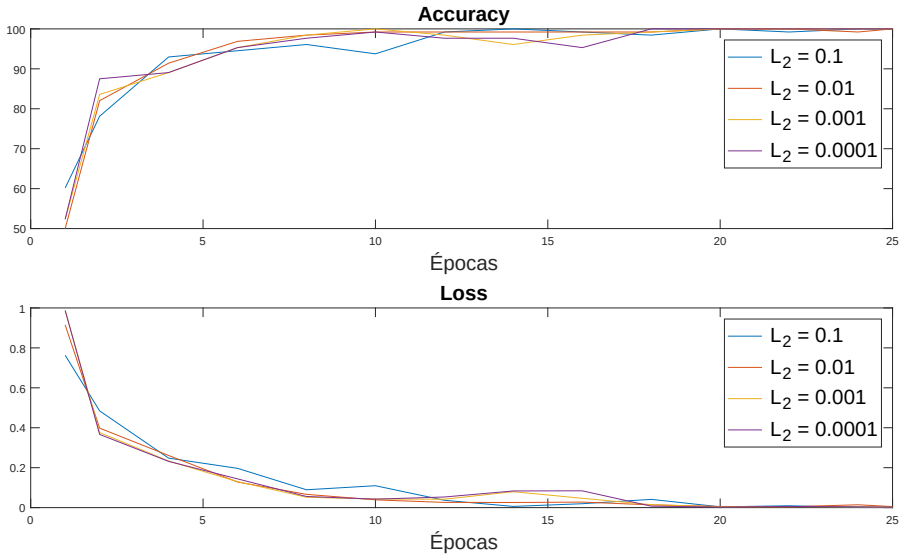
Este experimento consiste en la comparación de distintos valores del parámetro L_2para aplicar regularización. En la Figura 4.4 se puede comprobar que el impacto

Tabla 4.5: Resultados en el conjunto de test para distintos valores de learning rate

Learning Rate	Accuracy	Sensibilidad	Fall-out
0.01	0.5	0	0
0.001	0,8350	0,43590	0,0683
0.0001	0,7967	0,6838	0,1760
0.00001	0,6833	0,7521	0,3333
Variable	0,8350	0,5385	0,0932

de este parámetro es más bien reducido, ya que todas las curvas de aprendizaje son similares. Además, en la Tabla 4.6 se observan unos valores muy similares a los de experimentos anteriores, con una accuracy en test en torno a un 83 % frente a casi un 100 % en entrenamiento, lo cual es un claso signo de overfitting. Si bien, en el caso de un L_2más alto, como es el caso de 0.1, se aprecia que la red aprende ligeramente más despacio, lo que podría indicar la necesidad de probar valores de L_2incluso más altos.

Figura 4.4: Comparativa de entrenamientos en función de L_2



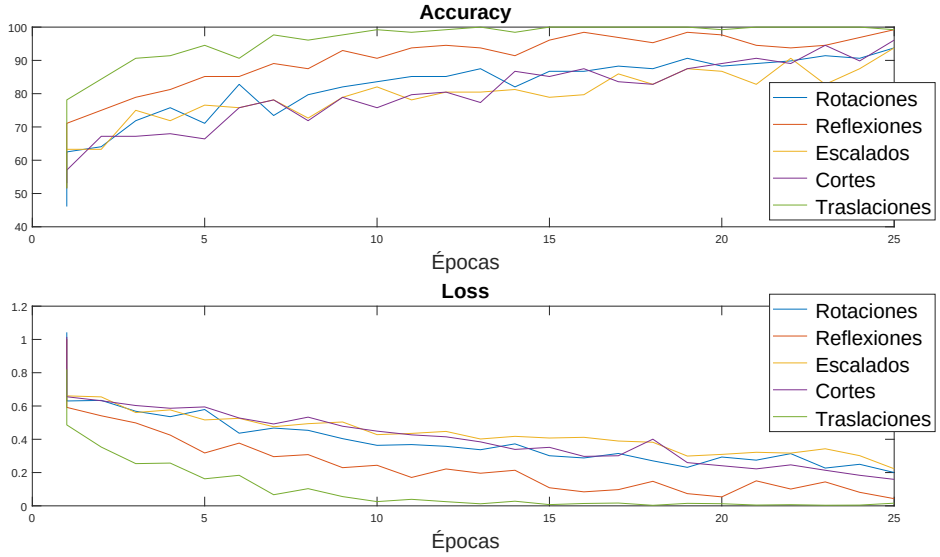
4.3.5. Data augmentation

En este experimento se muestran distintas variantes que se pueden utilizar a la hora de aplicar aumento artifical de datos, más conocido como *data augmentation.* Se han probado distintos tipos de transformaciones por separado, de forma que se pudiera comprobar cuál es el efecto de cada una de ellas en el entrenamiento de la red.

Tabla 4.6: Resultados en el conjunto de test para distintos valores de L_2

L_2	Accuracy	Sensibilidad	Fall-out
0.1	0,8317	0,4615	0,0787
0.01	0,8350	0,4872	0,0807
0.001	0,8283	0,3932	0,0663
0.0001	0,8350	0,43590	0,0683

Figura 4.5: Comparativa de entrenamientos en función del tipo de data augmentation



Las cinco transformaciones aplicadas son rotaciones, reflexiones, escalados, cortes y traslaciones.

Como se puede ver en la Figura 4.5, el empleo de data augmentation causa que el entrenamiento sea más lento, lo que puede resultar incómodo por cuestión de tiempo, pero también podría ser buena señal de cara a evitar el sobreentrenamiento. En concreto, las transformaciones de rotación, corte y escalado son las que obtienen una curva de entrenamiento más suave; de hecho, al cabo de las 25 épocas no terminan de converger.

La accuracy obtenida en la etapa de test, que se puede ver en la Tabla 4.7, es bastante similar a la que se ha conseguido en los experimentos anteriores, si bien, la sensibilidad es algo más alta, especialmente en los casos de rotaciones y cortes, lo que podría ser un punto a tener en cuenta para futuros entrenamientos donde se desee hacer énfasis en la optimización del diagnóstico.

Tabla 4.7: Resultados en el conjunto de test para distintos tipos de data augmentation

Tipo de data augmentation	Accuracy	Sensibilidad	Fall-out
Rotaciones	0.8333	0.5556	0.0994
Reflexiones	0.8333	0.4701	0.0787
Escalados	0.8200	0.4359	0.0870
Cortes	0.8167	0.5043	0.1077
Traslaciones	0.8250	0.4359	0.0807

4.3.6. Conclusión

A partir de los experimentos, se pueden inferir algunas conclusiones que serán de gran utilidad para experimentos futuros:

Se han aprendido varias técnicas que se pueden emplear en la reducción del overfitting. La primera de ellas ha sido el uso de una learning rate variable a lo largo del entrenamiento, que permite avanzar rápidamente en las primeras épocas y dedicar las siguientes a una estabilización más fina de los pesos. Esto permite forzar que el punto donde se estabilicen los pesos no esté muy adaptado a los datos de entrenamiento y así funciones mejor a la hora de generalizar.

La segunda técnica que puede contribuir a evitar el sobreentrenamiento es el *early stopping,* concretamente en el caso de establecer *checkpoints.* Al guardar los pesos periódicamente, se podrá entrenar toda la red y decidir después cuál de los resultados es el más favorable.

Por último, el uso de *data augmentation* también puede ser de gran ayuda para combatir el overfitting y ayudar a una mejor generalización.

En el caso de la regularización L_2, no ha tenido una gran repercusión en el entrenamiento, ya que no ha ayudado a disminuir el overfitting de la red, contradiciendo las espectativas que se tenían de este procedimiento. La solución a este problema podría ser el uso de un valor más elevado para L_2.

4.4. Resultados finales

En este apartado se aplicarán algunas de las técnicas exploradas previamente para intentar resolver los problemas mencionados durante los experimentos. En el primer experimento, el objetivo será la minimización del overfitting, mientras que en el segundo se tratará de mejorar la sensibilidad.

Lo ideal en este apartado sería disponer de un nuevo set que se utilice para testear los modelos finales elegidos, sin embargo, el dataset de test se ha utilizado en las etapas anteriores —en este contexto, se podría decir que el conjunto de test ha

actuado como conjunto de validación—. Por esta razón, y dado que este apartado tiene el principal objetivo de demostrar la utilidad práctica de los porcedimientos presentados, se utilizará únicamente el set de test.

4.4.1. Reducción de overfitting

Para el entrenamiento de esta red se ha utilizado la configuración de la Tabla 4.8. Como se ha visto antes, la elección de una learning rate variable puede ser de gran ayuda para tratar de reducir el sobreentrenamiento; por esta razón, para este experimento, se ha elegido una learning rate variable que comienza en 0.001 y se divide por 10 cada 5 épocas. El tamaño de mini-batch y el número de épocas no han sido modificados con respecto a la configuración base. Aunque ya se vio que no tenía tanto impacto como se hubiera deseado, se ha utilizado un factor L_2de 0.01. En cuanto al data augmentation, se han utilizado las dos variantes que, según la Tabla 4.7, conseguían mejor accuracy.

En la Figura 4.6, se muestran tanto la accuracy como los valores para la función de pérdida durante el entrenamiento. Además, se muestra también la accuracy obtenida en la fase de test para las redes que se han guardado como checkpoint durante el proceso. Como se puede observar, la accuracy obtenida en test es bastante similar a la de la fase de entrenamiento, lo cual quiere decir que la red no está sobreentrenada. En parte, esto se debe a que se ha sacrificado el objetivo de un resultado más exacto por el de la eliminación de overfitting, ya que la accuracy se queda en torno al 80 % tanto en la fase de entrenamiento como en la de resultados.

En la Tabla 4.9 se detallan los resultados de cada caso de la fase de test junto con las medidas de sensibilidad y fallout. Se puede advertir que los resultados para la sensibilidad son bastante peores que los de la accuracy; esto quiere decir que el diagnóstico no es muy bueno si la evaluación se centra únicamente en los pacientes enfermos.

Tabla 4.8: Configuración utilizada en el experimento de reducción de overfitting:

Tamaño de mini-batch	Número de épocas	Learning rate inicial	Factor L_2	Data augmentation
128	25	0.001	0.01	Rotaciones y reflexiones

Figura 4.6: Comparativa de entrenamientos en función del tipo de data augmentation

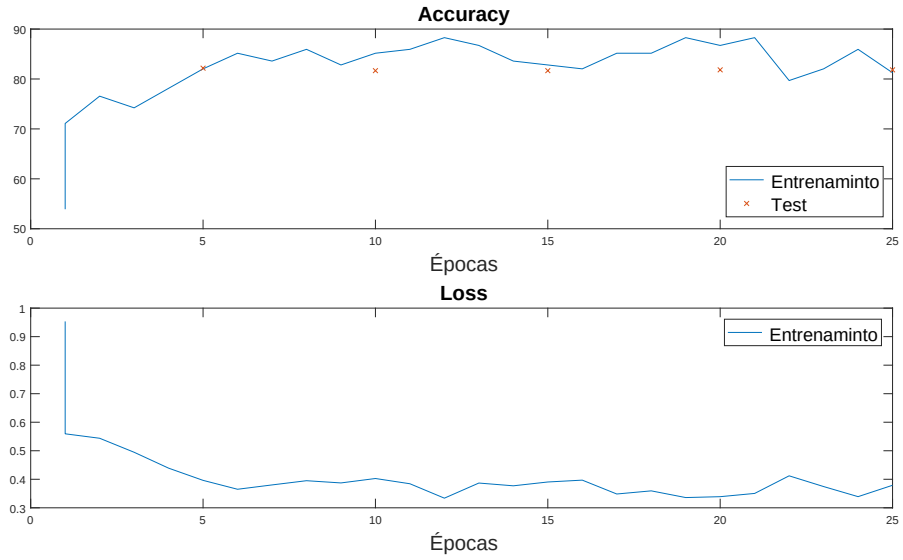


Tabla 4.9: Resultados en el conjunto de test para distintos tipos de data augmentation

Época	Accuracy	Sensibilidad	Fall-out
5	0.8217	0.5299	0.1077
10	0.8167	0.5641	0.1222
15	0.8167	0.5556	0.1201
20	0.8183	0.5556	0.1180
25	0.8183	0.5556	0.1180

4.4.2. Mejora de la sensibilidad

La motivación de este experimento es la mejora de la sensibilidad, ya que este es uno de los problemas más graves a los que se enfrenta la red —el correcto diagnóstico de las personas enfermas es prioridad frente al de las personas sanas, debido a que conlleva mayor riesgo—. Para realizar esta tarea, se ha tomado como referencia la Tabla 4.5, donde se observa que, para bajas tasas de entrenamiento, la sensibilidad toma valores más altos[1]. Por esta razón, se ha vuelto a partir de una learning rate de 0.001, pero en este caso se divide por 10 cada tres épocas, por lo que caerá rápidamente. Al utilizar una learning rate que fuerza una convergencia más rápida, se ha reducido el número de épocas a 20. Además, se han utilizado los dos tipos de data augmentation que obtenían mejores resultados según la Tabla 4.7, que son rotaciones y cortes.

[1]Nótese que esta inferencia sólo aplica a este caso concreto de estudio, no es generalizable para otras ocasiones.

En la Figura 4.7, se pueden ver la accuracy y la función de pérdida durante el entrenamiento y la accuracy en el conjunto de test para las redes guardadas mediante checkpoint. En este caso, no se elimina por completo el overfitting, ya que la accuracy en test no llega a igualar a la que se alcanzaba en el entrenamiento.

En la Tabla 4.11 se puede comprobar que no se han logrado unos valores mucho mejores que los que ya se tenían anteriormente para la sensibilidad y el fallout. El mejor caso de sensibilidad se alcanza a las cinco épocas, con un 61.54 %, sin embargo, el fallout que consigue esta red es de 19.46 %, un valor demasiado alto si se tiene en cuenta que, de cien pacientes sanos, unos veinte serían diagnosticados erróneamente con cáncer de piel.

Tabla 4.10: Configuración utilizada en el experimento de reducción de overfitting:

Tamaño de mini-batch	Número de épocas	Learning rate inicial	Factor L_2	Data augmentation
128	20	0.001	0.01	Rotaciones y cortes

Figura 4.7: Comparativa de entrenamientos en función del tipo de data augmentation

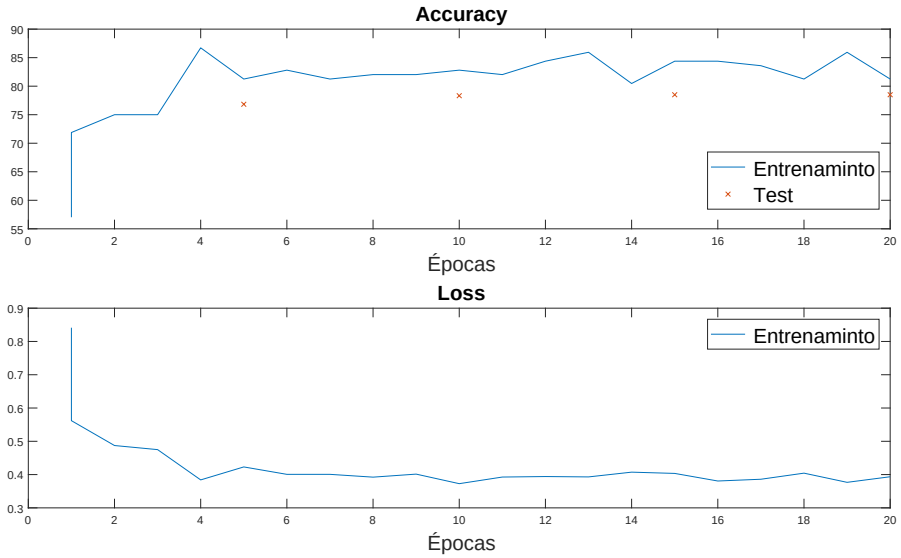


Tabla 4.11: Resultados en el conjunto de test para distintos tipos de data augmentation

Época	Accuracy	Sensibilidad	Fall-out
5	0.7683	0.6154	0.1946
10	0.7833	0.5812	0.1677
15	0.7850	0.5812	0.1656
20	0.7850	0.5812	0.1656

Capítulo 5

Conclusiones y trabajo futuro

5.1. Conclusiones

Durante este proyecto se han llevado a cabo una serie de tareas que han contribuido a cumplir el objetivo principal de este trabajo de fin de grado, que era el estudio de los parámetros que intervienen en el entrenamiento de una red convolucional a través de un problema de diagnóstico precoz de melanomas. Los puntos más importantes que se han completado son:

- El estudio del estado del arte ha sido determinante en este proyecto, ya que unos conocimiento básicos sobre redes convolucionales han resultado muy útiles para la comprensión del trabajo. Además, el análisis de otras propuestas para el problema que se enfrentaba ha sido de gran ayuda para aportar ideas a la fase de diseño y desarrollo.

- Se ha realizado un diseño basado en una red preexistente, la AlexNet, a partir de la cual se ha efectuado un entrenamiento basado en transfer learning. Además, se han solucionado el problema de desbalanceo de clases en el dataset.

- Se han llevado a cabo una serie de experimentos a través de los cuales se ha podido comprender la repercusión de los hiperparámetros más relevantes sobre el entrenamiento de una red convolucional. Además, estos resultados se han utilizado para tratar de reducir algunos problemas habituales en machine learning como el overfitting.

5.2. Trabajo futuro

Los resultados obtenidos durante este trabajo son una enorme ayuda a la hora de abordar una tarea que se quiera resolver mediante deep learning, sin embargo, este es un campo tan vasto que no es posible abarcar todas sus vertientes en un trabajo de estas características. A continuación se ven algunas de las propuestas que podrían abordarse en un trabajo futuro:

- En este proyecto se ha utilizado una red convolucional bastante básica con el fin de entender su arquitectura y su funcionamiento, sin embargo, en la actualidad existen muchas otras redes con arquitecturas más complejas que se podrían estudiar. En un trabajo futuro, sería deseable utilizar una red de mayor profundidad o que incluya capas residuales, como por ejemplo una ResNet.

- Durante los experimentos se han probado varios métodos para evitar el overfitting, que ha sido uno de los problemas más notorios a lo largo de los experimentos, sin embargo, en un futuro deberían probarse otros tipos de regularización, como la técnica de dropout, para completar esta investigación.

- Otro experimento que sería interesante realizar es el entrenamiento de una red desde cero, sin utilizar la técnica de transfer learning, de forma que se puedan comparar ambos métodos y ver las ventajas de cada uno.

- En el experimento relativo al tamaño de mini-batch, no se han conseguido probar valores muy altos debido a las limitaciones del hardware, lo ideal para un trabajo futuro sería dar mayor completitud a ese apartado probando con valores más altos de mini-batch.

Bibliografía

[1] K. Matsunaga, A. Hamada, A. Minagawa, and H. Koga, "Image classification of melanoma, nevus and seborrheic keratosis by deep neural network ensemble," *arXiv preprint arXiv:1703.03108*, 2017. 12

[2] I. G. Díaz, "Incorporating the knowledge of dermatologists to convolutional neural networks for the diagnosis of skin lesions," *arXiv preprint arXiv:1703.01976*, 2017. 12

[3] A. Menegola, J. Tavares, M. Fornaciali, L. T. Li, S. Avila, and E. Valle, "Recod titans at isic challenge 2017," *arXiv preprint arXiv:1703.04819*, 2017. 12

[4] L. Bi, J. Kim, E. Ahn, and D. Feng, "Automatic skin lesion analysis using large-scale dermoscopy images and deep residual networks," *arXiv preprint arXiv:1703.04197*, 2017. 12

[5] X. Yang, Z. Zeng, S. Y. Yeo, C. Tan, H. L. Tey, and Y. Su, "A novel multi-task deep learning model for skin lesion segmentation and classification," *arXiv preprint arXiv:1703.01025*, 2017. 12

[6] T. DeVries and D. Ramachandram, "Skin lesion classification using deep multi-scale convolutional neural networks," *arXiv preprint arXiv:1703.01402*, 2017. 12

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012. 13

[8] M. Buda, A. Maki, and M. A. Mazurowski, "A systematic study of the class imbalance problem in convolutional neural networks," *arXiv preprint arXiv:1710.05381*, 2017. 16