Proyecto Fin de Carrera
Ingeniería de Telecomunicación

Redes neuronales profundas para el análisis de patrones en lesiones pigmentadas de la piel

Autor: Paula Torres Rodríguez

Tutoras: María del Carmen Serrano Gotarredona y Begoña Acha Piñero

Dpto. Teoría de la Señal y Comunicaciones
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2021

Departamento de Teoría de la Señal y Comunicaciones

Proyecto Fin de Carrera
Ingeniería de Telecomunicación

Redes neuronales profundas para el análisis de patrones en lesiones pigmentadas de la piel

Autor:

Paula Torres Rodríguez

Tutores:

María del Carmen Serrano Gotarredona

Begoña Acha Piñero

Catedrática de
Universidad

Dpto. Teoría de la Señal y Comunicaciones
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2021

Proyecto Fin de Carrera: Redes neuronales profundas para el análisis de patrones en lesiones pigmentadas de la piel.

Autor: Paula Torres Rodríguez
Tutoras: Maria del Carmen Serrano Gotarredona y Begoña Acha Piñero

El tribunal nombrado para juzgar el trabajo arriba indicado, compuesto por los siguientes profesores:

Presidente:

Vocal/es:

Secretario:

acuerdan otorgarle la calificación de:

El Secretario del Tribunal

Fecha:

A mi madre, la estrella que más brilla en el cielo.

Agradecimientos

A mi madre, por siempre creer en mí. A Manu, por siempre estar ahí. A mi familia, incluida mi familia de verano. A mis amigos, sin olvidarme de mis compañeros de la universidad, que algunos son más que compañeros. A mis tutoras, Carmen y Begoña, por su colaboración, consejos y su paciencia.

Paula Torres Rodríguez

Paradas, 2021

Resumen

Hay muchos avances tecnológicos que van modificando el concepto de salud debido a las necesidades sanitarias. El cáncer ha aumentado mucho en las últimas décadas, por lo tanto, presenta un importante problema de salud pública. La detección precoz de tumores malignos, es un aspecto muy importante para prevenir el riesgo de los mismos. Disponemos de muchas técnicas por imagen empleadas para la detección precoz, entre ellas está la dermatoscopia, la cual ha revelado una nueva forma de tratar las lesiones pigmentadas de la piel, debido a que es una técnica in vivo no invasiva.

La Inteligencia Artificial es una rama de la ciencia, en concreto de la informática, que hace referencia a la capacidad de una máquina de resolver un problema como lo haría un ser humano. Dentro de la inteligencia artificial existen dos ramas: Machine Learning y Deep Learning.

Este trabajo pretende realizar la clasificación de lesiones pigmentadas en la piel mediante Inteligencia Artificial más específicamente usando aprendizaje profundo (Deep Learning), utilizando el lenguaje de programación Python. Vamos a usar una red neuronal sencilla para el tratamiento digital de imágenes dermatoscópicas y así poder detectar patrones del tipo globular, homogéneo y reticulado. La detección de estos tres patrones forma parte de los criterios diagnósticos de las lesiones de la piel utilizados por los dermatólogos para el diagnóstico.

Abstract

There are many technological advances that are modifying the concept of health due to health needs. Cancer has increased greatly in recent decades, therefore it presents a major public health problem. The early detection of malignant tumors is a very important aspect to prevent their risk. We have many imaging techniques used for early detection, among them is dermoscopy, which has revealed a new way of treating pigmented skin lesions, since it is a non-invasive in vivo technique.

Artificial Intelligence is a branch of science, specifically computer science, which refers to the ability of a machine to solve a problem as a human being would. Within artificial intelligence there are two branches: Machine Learning and Deep Learning.

This work aims to perform the classification of pigmented skin lesions through Artificial Intelligence more specifically using Deep Learning, using the Python programming language. We are going to use a simple neural network for the digital treatment of dermoscopic images and thus be able to detect globular, homogeneous and reticulated patterns. The detection of these three patterns is part of the diagnostic criteria for skin lesions used by dermatologists for diagnosis.

Índice

Resumen III
Abstract V

1 Introducción al problema clínico 1
1.1 Modelos de diagnóstico 2
1.1.1 Primera fase: Análisis de patrones 2
1.1.2 Segunda fase: Algoritmos de diagnóstico 3
1.2 Sistemas CAD 5

2 Estado del arte 6

3 Procesamiento de imagen 9
3.1 Aprendizaje máquina 9
3.2 Aprendizaje profundo 10
3.2.1 Tipos de redes 11
3.2.2 Capas 12
3.2.3 Funciones de activación 13
3.2.4 Parámetros e hiperparámetros 15
3.2.5 Entrenamiento de la red 16
3.2.6 Conjuntos de datos 16
3.2.7 Regularización 16
3.2.8 Aprendizaje transferido / redes 'from scratch' 18

4 Metodología propuesta 20
4.1 Material: base de datos 20
4.2 Implementación de la red (Google Colab, Python, Keras) 20
4.3 Red propuesta 22
4.3.1 Entrenamiento de las imágenes con la red neuronal convolucional 24
4.3.2 Compilación del modelo y resultados 27
4.4 Ajustes de la red 27
4.4.1 Redimensionamiento y cambio de formato de las imágenes 27
4.4.2 Balanceo de datos 28
4.5 Evaluación 29
4.5.1 Conjuntos de entrenamiento y test 29
4.5.2 Medidas de rendimiento 29
4.6 Pruebas 31

5 Resultados 33
5.1 Pruebas sin aumento ni reducción de casos 33
5.2 Pruebas con reducción de casos en patrón globular 34
5.3 Pruebas con aumento de casos en patrón reticular 38
5.4 Pruebas con reducción de casos en patrón globular y aumento en patrón reticular 39
5.5 Interpretación final de los resultados 41

6 Conclusiones 43

7 Referencias 45

1 Introducción al problema clínico

El melanoma es el nombre genérico de los tumores melánicos o pigmentados. La importancia del melanoma reside, más que en su frecuencia, en su rápido aumento en los países desarrollados desde los años 50 y a que este aumento está relacionado directamente con la exposición solar por motivos estéticos y de ocio. Alrededor del 81% de los casos se localizan en países desarrollados.

A pesar de que la mayoría de los melanomas se originan en la piel, también pueden aparecer en otras superficies del cuerpo. Cuando el melanoma aparece en la piel, la enfermedad se denomina melanoma cutáneo.

Se diagnosticaron 324.635 casos de melanoma cutáneos en 2020, según estimaciones de la IARC (Centro Internacional de Investigaciones sobre el Cáncer), dándose 173.844 diagnósticos en hombres y 150.791 en mujeres.

Actualmente las ratios de incidencia estandarizada por edad y sexo son mayores en los hombres que en mujeres, salvo en los países del norte de Europa donde predomina la incidencia en el sexo femenino.

En España se diagnostican cerca de 5.500 casos anuales. En nuestro país es un tumor más frecuente entre las mujeres 3.182 que entre los hombres 1.207. Se registran la mayoría de casos entre los 40 y los 70 años. [1]

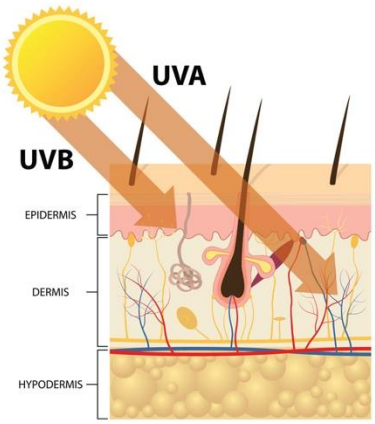

Figura 1-1. Incidencia de los rayos del sol en la piel. [1]

Existen cuatro tipos principales de melanoma [1]:

- Melanoma de extensión superficial: se da en cualquier punto de la piel, pero es más común en espalda y en miembros inferiores en mujeres, y en tronco en hombres. Es el tipo más frecuente en personas de raza blanca.
- Léntigo maligno melanoma: se da en la piel dañada por el sol, sobre todo en cara, cuello y brazos.
- Melanoma lentiginoso acral: se da en las palmas de las manos, las plantas de los pies y por debajo de las uñas. Es más frecuente en personas de raza negra.
- Melanoma nodular: aparece en tronco, cabeza o cuello. Últimamente se han detectado muchas mutaciones en este tipo de cáncer, entre ellas la más destacada es la mutación BRAF. Esta detección puede cambiar la estrategia terapéutica en algunos pacientes portadores de esta mutación.

Debido al aumento de casos que existe tenemos la necesidad de estudiar estas lesiones para detectarlas a tiempo, esto ha dado lugar al avance de las tecnologías para mejorar la sensibilidad y el diagnóstico clínico y causar las mínimas molestias en los pacientes. La dermatoscopia es una técnica de diagnóstico no invasivo que usa aumento óptico para permitir la visualización de características morfológicas que no son visibles a simple vista, por consiguiente, forman un enlace entre la dermatología macroscópica y la microscópica. Esta observación "submacroscópica" de las lesiones de piel pigmentadas (PSL) se suma a las herramientas de diagnóstico en vivo disponibles, al proporcionar nuevas características morfológicas para la diferenciación del melanoma de otros melanocíticos y no melanocíticos en lesiones de piel pigmentadas. Como resultado, la práctica de la dermatoscopia está convirtiéndose en muy popular, no solo entre los dermatólogos, también entre los oncólogos, cirujanos o pediatras. El diagnóstico convencional en dermatoscopia se basa en la simulación simultánea de los criterios morfológicos. En consecuencia, la reproducibilidad y validez de los criterios dermatoscópicos es de suma importancia. [2]

1.1 Modelos de diagnóstico

El procedimiento de diagnóstico en dermatoscopia que ha sido usado por la mayoría de los trabajadores de este campo es un método de dos fases. En la primera fase se debe reconocer si es una lesión melanocítica o no melanocítica y en la segunda fase se debe discernir si es de naturaleza benigna o maligna la lesión anteriormente estudiada, si hemos llegado a la conclusión de que es melanocítica.

1.1.1 Primera Fase: Análisis de Patrones

En la primera fase se lleva a cabo el análisis de patrones, además de los parámetros dermatoscópicos individuales, se valora también el patrón global que presenta cada lesión melanocítica. Existen diferentes patrones globales: (a) reticulado, (b) globular, (c) empedrado, (d) homogéneo, (e) en estallido de estrellas, (f) paralelo, (g) lacunar, (h) multicomponente e (i) inespecífico. La lesión melanocítica se encasilla en el patrón que presente el parámetro dermatoscópico predominante, puesto que suelen presentar diferentes estructuras. [3]

Vamos a describir los patrones globales [3]:

a) Patrón reticulado atípico: la lesión está formada en su mayor parte por un retículo pigmentado atípico.
b) Patrón globular atípico: formado por puntos y glóbulos de forma, distribución y tamaño irregulares.
c) Patrón empedrado: es muy similar al anterior, pero los glóbulos que lo componen son más grandes y están distribuidos de forma muy cercana. Su forma recuerda a la de los mosaicos o empedrados, por eso este nombre.
d) Patrón homogéneo: que puede presentarse en algunas metástasis de melanoma.
e) Patrón en estallido de estrellas: el patrón en estallido de estrellas se caracteriza por la presencia de proyecciones que adoptan una distribución radial y regular en toda la periferia de la lesión pigmentada a estudiar.
f) Patrón paralelo de la cresta: es el típico de lesiones palmo-plantares.
g) Patrón lacunar: caracterizado por la presencia de estructuras de varios tamaños, redondeadas u ovaladas con bordes poco definidos llamados lagos rojos y con la característica de la coloración rojiza, azul, lila o negra.
h) Patrón multicomponente: aquel que resulta de la combinación de tres o más patrones dermatoscópicos en la misma lesión.

i) Patrón inespecífico: presente en lesiones que no pueden clasificarse en ninguno de los patrones anteriores. Siempre debe descartarse la existencia de un posible melanoma en este patrón.

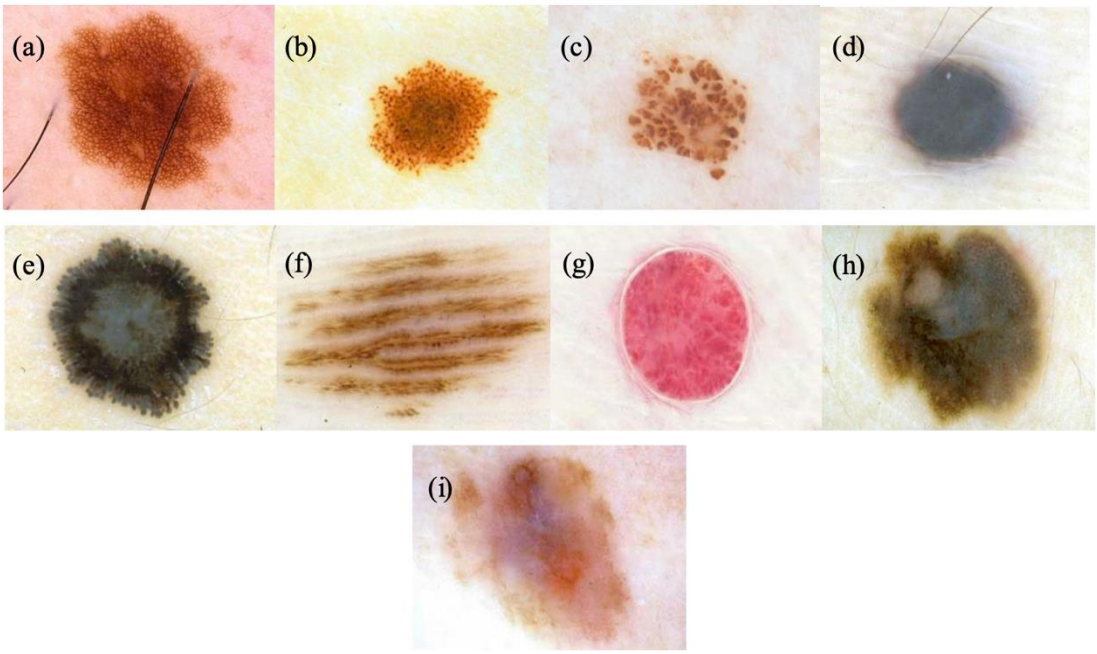

Figura 1-2. Patrones globales. [3]

1.1.2 Segunda Fase: Algoritmos de diagnóstico

En la segunda fase vamos a estudiar si la lesión melanocítica es benigna o no, a continuación, vemos cuatro algoritmos de diagnóstico diferentes: (1) análisis de patrones modificados, (2) regla ABCD de dermoscopia, (3) método de Menzies, y (4) lista de verificación de 7 puntos.

1) Análisis de patrones modificados: este método es muy completo y con él se obtienen buenos resultados. Probablemente esto es debido a que es el método que mejor refleja cómo trabaja el cerebro humano al clasificar imágenes morfológicas. Es un método que se basa en el juicio cualitativo, llevado a cabo por un experto, de varios criterios de dermatoscopia, que pueden ser globales o locales. Para la clasificación de esta fase nos centraremos en los patrones locales, que son las características dermatoscópicas que se presentan en una región específica de la lesión. Estas características son: retículo pigmentado prominente o atípico, puntos y glóbulos irregulares, áreas desestructuradas, velo azul-gris o azul-blanquecino, estructuras de regresión, proyecciones irregulares y estructuras vasculares asociadas a malignidad. Estos patrones pueden presentarse con naturaleza regular o irregular, haciendo así ver si una lesión es benigna o maligna, respectivamente. [4]

2) Regla ABCD de dermoscopia: esta regla es un método muy aplicado y conocido para el estudio de las lesiones melanocíticas. Las letras A: asimetría, B: borde, C: color, D: diferencias estructurales. A partir de la observación de dichas características se asigna a la lesión una puntuación semicuantitativa que permite determinar con bastante precisión si se trata de un melanoma o una lesión benigna. La valoración de cada característica se realiza atendiendo a las siguientes consideraciones:
   - Asimetría: se valora teniendo en cuenta su color y estructura. Se colocan dos ejes perpendiculares de tal forma que se minimice la asimetría. Si los ejes son

completamente simétricos, la puntuación es 0. Si hay asimetría respecto de un único eje, la puntuación es 1. Si existe asimetría respecto a los dos ejes la puntuación es 2.

- Borde: se divide el borde de la lesión en 8 segmentos, y se añade un punto por cada segmento en el que se observe un cambio brusco.
- Color: se suman los diferentes colores presentes en la lesión.
- Diferencias estructurales: se suma un punto por cada patrón estructural encontrado en la lesión pigmentada. Las posibilidades son: red pigmentada, glóbulos, áreas homogéneas, manchas y streaks.

A continuación, cada puntuación individual es multiplicada por un factor que varía según de qué característica provenga, finalmente se suman todas las puntuaciones parciales obteniendo el índice dermatoscópico total y se clasifica la lesión en benigna sospechosa y maligna según los resultados. [5]

Figura 1-3. Regla ABCD.

3) Método de Menzies: es un método sencillo que valora 11 características dermatoscópicas, divididas en 9 positivas y 2 negativas. Las positivas son características del melanoma y las negativas definen la lesión como benigna. Para que una lesión sea clasificada como melanoma no debe tener ninguna característica negativa y tener al menos, una positiva.[6]

4) Lista de verificación de 7 puntos de Argenziano: es un método de diagnóstico semicuantitativo. Se toman 7 patrones y se dividen en dos tipos, mayores y menores, cada uno con una puntuación distinta, según su relevancia en las lesiones malignas. La imagen dermatoscópica es analizada y se evalúa. Si finalmente la puntuación obtenida es superior a 3, la lesión es clasificada como melanoma.[7]

La concordancia intraobservador es de buena a excelente para todos los criterios dermatoscópicos explicados anteriormente.

1.2 Sistemas CAD

Los sistemas de ayuda al diagnóstico mediante computador (Computer Aided Diagnosis, CAD) intentan imitar esta forma de diagnóstico clínico implementando algoritmos matemáticos que encuentran las características explicadas en las fases anteriores, como son color, asimetría, patrones, etc. En todos estos años numerosos trabajos llevados a cabo por científicos han diseñado sistemas de aprendizaje máquina (Machine Learning, ML) que consiguen unas tasas de acierto muy elevadas. Aunque en los últimos años la tendencia es usar aprendizaje profundo (Deep Learning, DL) esto se debe a que puede procesar grandes bases de datos de imágenes de lesiones pigmentadas en la piel y se tienen disponibles máquinas de gran capacidad computacional como son las GPU y TPU. Con el Deep Learning se tiene también la posibilidad de usar redes muy complejas que ya han sido previamente implementadas y preentrenadas con bases de datos muy grandes de imágenes generales. Todo esto ha hecho que se avance mucho en los estudios de lesiones pigmentadas en la piel con estas tecnologías.

2 Estado del arte

En este trabajo nos vamos a centrar en el diseño de un sistema CAD que clasifique una lesión en tres tipos de patrones globales de los anteriormente descritos: globular, homogéneo y reticulado. Vamos a ver de manera resumida los estudios que se han llevado a cabo en este campo:

Los algoritmos de aprendizaje profundo, especialmente las redes neuronales convolucionales (CNN), se han utilizado para tareas de visión por computadora durante décadas. Sin embargo, el poder real de las redes neuronales profundas para las tareas de reconocimiento visual no se había descubierto hasta la competencia ImageNet [8] en 2012, cuando los investigadores comenzaron a lograr resultados sobresalientes combinando el uso eficiente de unidades de procesamiento de gráficos con nuevos términos de regularización y datos efectivos como las técnicas de aumento [9]. Posteriormente, las CNN mostraron un rendimiento excelente en varias tareas de reconocimiento visual, incluido el reconocimiento de señales de tráfico, dígitos escritos a mano y rostros.

Recientemente, la disponibilidad de conjuntos de datos a gran escala y el desarrollo de GPU muy potentes han permitido a los investigadores profundizar mucho en las CNN. Por ejemplo, una red de CNN profunda llamada AlexNet, propuesta por Krizhevsky et al. [10], logró un rendimiento excelente (tasas de error entre los primeros y los cinco primeros del 37,7% y 17,0%) en el 2012 ImageNet Large Scale Visual Recognition Challenge. No había una comprensión clara de por qué las CNN profundas muestran un rendimiento de clasificación muy bueno hasta que la técnica de visualización propuesta por Zeiler et al. [11] proporcionó una comprensión intuitiva de las funciones de las capas de características intermedias. Además, algunos otros algoritmos basados en CNN profundos también han demostrado un rendimiento muy bueno en la clasificación de imágenes.

A pesar de que recientemente se ha descubierto el poder real de las CNN, las aplicaciones de las CNN en el análisis de imágenes médicas se remontan a la década de 1990, cuando se usaban para la detección asistida por computadora de microcalcificaciones en mamografía digital [12] y la detección asistida por computadora de nódulos pulmonares en conjuntos de datos de Tomografía Computerizada (TC) [13]. Con el resurgimiento de las CNN debido al desarrollo de una potente computación GPU, la literatura sobre imágenes médicas ha sido testigo de una nueva generación de sistemas de detección asistida por computadora que muestran un rendimiento superior en muchas tareas, incluida la detección asistida por computadora de los ganglios linfáticos en las imágenes de TC [14] y la detección asistida por computadora de embolia pulmonar (EP) en conjuntos de datos de TC [15]. Las investigaciones y los desarrollos de las CNN en el análisis de imágenes médicas no se limitan solo a los sistemas de detección de enfermedades, sin embargo, las CNN se han utilizado recientemente para problemas de clasificación de lesiones cutáneas [16], segmentación del páncreas en imágenes de TC [17] y otros tipos de segmentaciones clínicas.

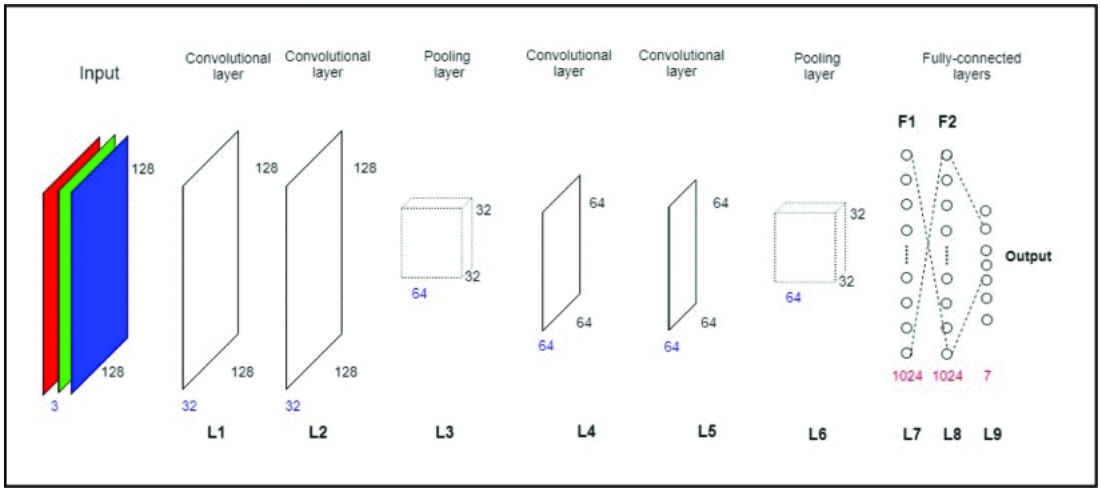

Figura 2-1. Arquitectura de la CNN profunda para la clasificación de enfermedades de lesiones cutáneas. La red consta de nueve capas. [16]

Issai Gola et al. [18] presentaron una herramienta dermatológica automatizada para la parametrización de melanomas. Se basa en la regla ABCD estándar y en los protocolos de reconocimiento de patrones dermatológicos. Por un lado, desarrollaron una pila completa de algoritmos para la parametrización de asimetría, borde, color y diámetro, y por otro, desarrollaron tres algoritmos automáticos para el procesamiento de imágenes digitales con el fin de detectar los patrones adecuados. Estos permiten calcular algunas características cuantitativas basadas en el aspecto y los patrones internos del melanoma utilizando algoritmos de operación simple, con el fin de minimizar el tiempo de respuesta. La base de datos utilizada constaba de 160 imágenes RGB (20 imágenes por patrón) catalogadas por dermatólogos, y los resultados han resultado ser exitosos según la evaluación de expertos médicos. Por tanto, demuestra ser un sistema fiable a la hora de realizar un diagnóstico.

Mahmoud et al. [19] propusieron un nuevo sistema CAD para diferenciar entre lesiones cutáneas de melanoma maligno y no melanoma. Con un conjunto de datos que constaba de tres grupos de imágenes dermatoscópicas, donde 40 eran melanoma, 80 eran nevos comunes y 80 eran nevos displásicos. Eliminaron el vello de las imágenes dermatoscópicas para mejorar el rendimiento del clasificador, y luego aplicaron un filtro mediano con 7x7. En el paso de análisis de textura se han utilizado varios métodos de análisis de textura, como filtros Gabor, matriz de coincidencia de niveles de grises, histograma de gradientes orientados, patrón binario local y patrón de número direccional local. Se ha elegido el clasificador de perceptrón multicapa para tomar una decisión con el caso de la imagen (caso positivo o caso negativo). El mejor rendimiento del sistema CAD propuesto se logra cuando usamos histograma de gradientes orientados como extracción de características. El histograma de gradientes orientados obtuvo una relación entre la sensibilidad y especificidad de 0,9783 con la tarea de clasificación de melanoma / nevos comunes y una relación entre la sensibilidad y especificidad de 0,9439 con la clasificación de melanoma / nevos displásicos, siendo este el mejor método de análisis de textura.

A. Sáez et al [20] propusieron diferentes métodos basados en modelos de clasificación de patrones globales en imágenes dermatoscópicas. La identificación de patrones globales se incluye en el marco de análisis de patrones. El modelado lo realizaban en dos sentidos: primero, modelaban una imagen dermatoscópica mediante un modelo de Markov condicional simétrico finito aplicado al espacio de color L * a * b * y los parámetros estimados del modelo los trataban como características. A su vez, suponían que la distribución de estas características seguía diferentes modelos a lo largo de una lesión: un modelo gaussiano, un modelo mixto gaussiano y un modelo de histograma de bolsa de características. Para cada caso, la clasificación se llevaba a cabo mediante un enfoque de recuperación de imágenes con diferentes métricas de distancia. El

objetivo principal era clasificar una lesión pigmentada completa en tres patrones posibles: globular, homogéneo y reticular. Realizando una evaluación exhaustiva del rendimiento de cada método en una base de datos de imágenes extraída de un Atlas público de dermatoscopia. La mejor tasa de éxito de clasificación la lograban mediante el método basado en el modelo de mezcla gaussiana con una tasa de éxito del 78,44% en promedio. En una evaluación adicional analizaron el patrón multicomponente obteniendo una tasa de éxito del 72,91%.

Serrano et al [21] presentaron un método para el análisis de patrones en imágenes dermatoscópicas de piel anormalmente pigmentada (lesiones melanocíticas). Para el diagnóstico de un posible cáncer de piel, los médicos evalúan la lesión de acuerdo con diferentes reglas. La nueva tendencia en Dermatología es clasificar la lesión por patrón de irregularidad. Para analizar el patrón de turbulencia, las lesiones deben segmentarse en regiones de patrón único. El método de clasificación presentado, cuando se aplica en parches de lesiones superpuestos, proporciona un gráfico de patrones que, en última instancia, podría permitir el análisis de turbulencia de textura única en la región. Debido a la apariencia de textura de color de estos patrones, presentaron un método novedoso basado en una extensión de color de Markov Random Field (MRF) del Modelo Simétrico Condicional Finito (FSCM) para la caracterización y discriminación de muestras de patrones.

En el challenge ISIC 2018[22], estudiaron el papel de las funciones de activación en las capas ocultas de las redes neuronales profundas para la clasificación de imágenes médicas. Y proponen un algoritmo CNN profundo para la clasificación de posibles diagnósticos de melanoma utilizando imágenes de lesiones cutáneas. Usan datos de imágenes extraídos de la CIIU 2018: análisis de lesiones cutáneas hacia la detección del melanoma en conjuntos de datos de gran desafío [23] [24]. Hay siete categorías de enfermedades posibles: melanoma, nevo melanocítico, carcinoma de células basales, queratosis actínica, queratosis benigna, dermatofibroma y lesión vascular. Los investigadores crearon una CNN simple para la clasificación del melanoma utilizando algunas de las ideas fundamentales de LeNet [25], que se aplicó para reconocer caracteres escritos a mano. Necesitaban una red más profunda, ésta constaba de cuatro capas convolucionales y dos agrupadas para la extracción de características, y tres capas completamente conectadas, al final, para la clasificación. El conjunto de datos de entrenamiento tenía 10015 imágenes de piel y evaluaron usando un conjunto de prueba que disponía de 1500 imágenes. Llegaron a la conclusión de que con la ayuda del método de aprendizaje profundo se puede predecir un diagnóstico de alta calidad para las enfermedades de lesiones cutáneas. Debido a que crearon una CNN profunda para la clasificación de lesiones cutáneas agregando la función lineal adaptativa por partes, que facilita notables mejoras de rendimiento del modelo, sin aumentar significativamente el número de parámetros que se pueden aprender.

Sadeghi et al. [26] llevaron a cabo la detección y clasificación de cinco tipos de patrones globales: reticular, globular, empedrado, homogéneo y paralelo. Su estudio fue llevado a cabo mediante técnicas de análisis de texturas basadas en aproximaciones estadísticas, la clasificación de las imágenes la realizaban mapeando y comparándola con la distribución de los modelos aprendidos durante todo el proceso. Sus experimentos muestran que la distribución conjunta del color en el espacio de color L*a*b* supera a los otros espacios de color con una tasa del 86.8%.

3 Procesamiento de imagen

Después de haber introducido los sistemas CAD vamos a ver lo que es el Machine Learning y el Deep Learning dos ramas que surgen de la Inteligencia Artificial.

La Inteligencia Artificial (IA) surge en la década de los 50, cuando investigadores del campo de la informática se plantean si se puede hacer pensar a un ordenador, es decir, se plantean que los ordenadores hagan el esfuerzo por automatizar tareas intelectuales normalmente llevadas a cabo por los humanos. Por definición, la IA es un campo general que incluye el Machine Learning y el Deep Learning que ahora los vamos a explicar en profundidad.

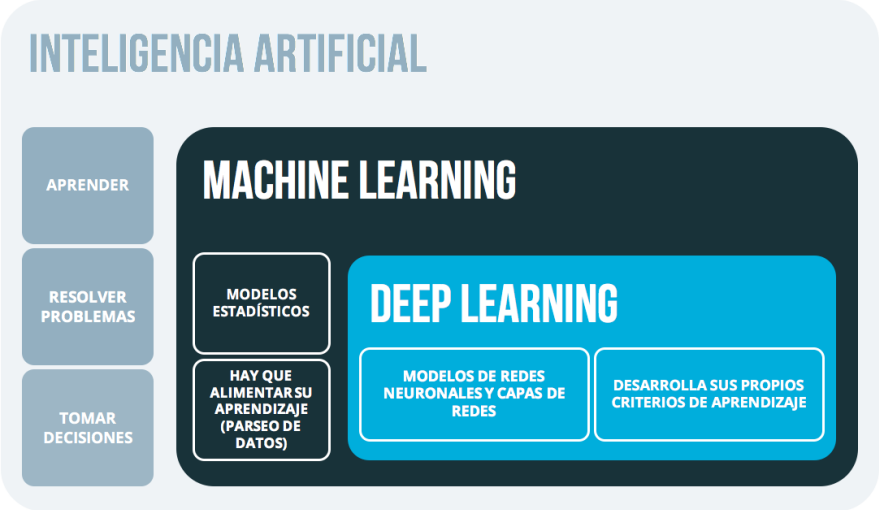

Figura 3-1. IA, ML, DL. [27]

3.1 Aprendizaje Máquina

El Machine Learning es una disciplina que proviene de las ciencias y la ingeniería que se ocupa de la construcción y el estudio de los algoritmos que pueden aprender a partir de datos, es un subcampo de la Inteligencia Artificial. Los algoritmos de Machine Learning intentan construir modelos basándose en los datos con el objetivo de hacer predicciones o tomar decisiones, como si fueran verdaderos expertos, en lugar de seguir las instrucciones de manera explícita para lo que han sido programados. [28]

Es decir, con el Machine Learning, los humanos introducen los datos, también las respuestas esperadas a partir de esos datos, y se generan las reglas. Estas reglas pueden aplicarse entonces a nuevos datos para producir respuestas originales. En este sistema se entrena más que programar de forma explícita. Se le proporcionan muchos ejemplos relevantes para una tarea y el sistema encuentra una estructura estadística en esos ejemplos que al final le permite crear unas reglas para automatizar la tarea.

El Machine Learning tiende a manejar conjuntos de datos grandes y complejos.

Para llevarlo a cabo necesitamos tres cosas [29]:

1. Introducción de puntos de datos: por ejemplo, si la tarea es el etiquetado de imágenes, los datos son las fotografías.

2. Ejemplos de resultados esperados: en una tarea con imágenes, los resultados esperados podrían ser etiquetas como “perro”, “gato”, etc.
3. Una manera de medir si el algoritmo está haciendo un buen trabajo: es necesario para medir la distancia entre el resultado actual del algoritmo y su resultado esperado. La media se utiliza como señal de retroalimentación para ajustar la manera en que funciona el algoritmo. Este paso es a lo que se le llama aprendizaje.

Todos los modelos de Machine Learning tienen que ver con encontrar representaciones apropiadas para sus datos de entrada, transformaciones de los datos que los hacen más fáciles de trabajar con ellos en una tarea determinada, como por ejemplo una tarea de clasificación.

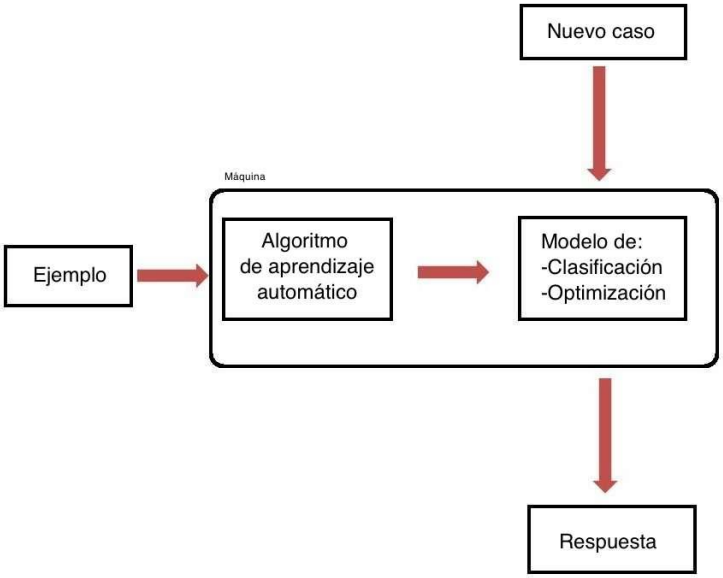

Figura 3-2. Esquema de modelo de aprendizaje automático.

3.2 Aprendizaje Profundo

El Deep Learning es un subcampo específico del Machine Learning, una nueva interpretación del aprendizaje de representaciones a partir de datos que pone el énfasis en aprender capas sucesivas de representaciones cada vez más significativas. “Deep” que significa profundo, se refiere a esta idea de capas sucesivas de representaciones. La cantidad de capas que contribuye al modelo de datos se llama “profundidad” del modelo.

En Deep Learning, las representaciones en capas se aprenden mediante modelos llamados “redes neuronales”, estructuradas en capas literales apiladas unas encima de otras. La especificación de lo que hace una capa a los datos de entrada se almacena en los “pesos” de la capa.

El Deep Learning es una manera de aprender representaciones de datos en múltiples etapas. Aprender significa encontrar un conjunto de valores para los pesos de todas las capas de una red, de manera que la red asigne correctamente las entradas de ejemplos a sus objetivos asociados.

Para controlar el resultado de una red neuronal, tenemos que ser capaces de medir cuánto se aleja ese resultado de lo que esperábamos, esto es llamado función de pérdida. La función de pérdida coge las predicciones de la red y el objetivo verdadero y computa una puntuación de distancia, capturando lo bien que ha funcionado la red en este ejemplo específico. Algo

fundamental en el Deep Learning es utilizar esta puntuación como señal de retroalimentación para ajustar un poco el valor de los pesos en una dirección que reducirá la puntuación de pérdida para el estudio que estemos llevando a cabo, a esto se le llama optimizador. El optimizador implementa una variante específica del descenso del gradiente estocástico.

Al principio se suelen asignar pesos aleatorios a la red, así que la red implementa transformaciones aleatorias, esto suele estar lejos de ser la forma ideal y, en consecuencia, la puntuación de la pérdida es alta. Esta pérdida se reduce entrenando la red. [29]

3.2.1 Tipos de redes

La red neuronal la podemos definir como un grafo, una capa de entradas que reciben la señal de entrada, la envían mediante estímulos a la siguiente capa oculta, la cual se encarga de procesar la información y transmitirla a la siguiente capa, así hasta que se llega a la última capa, que es llamada capa de salida y es la que transmite la respuesta.

Vamos a definir los principales tipos de redes neuronales:

- Redes Generativas Antagónicas (GAN): se presentaron en 2014, son redes neuronales profundas que comprenden dos redes enfrentadas la una con la otra, de ahí el nombre de antagónicas. Tenemos una red que se llama generadora, que genera nuevas instancias de datos y una red llamada antagonista o discriminadora que evalúa las nuevas instancias generadas por la red generadora que le llegan como entrada para determinar su autenticidad. Devuelve 1 si es auténtica y 0 si es falsa. [30]
- Redes Neuronales Recurrentes (RNN): son redes neuronales en las que los datos pueden fluir en cualquier dirección. El concepto básico que precede a las RNN es el uso de información secuencial. Son redes recurrentes debido a que repiten la misma tarea para cada elemento de una secuencia y la salida se basa en los cálculos anteriormente realizados, es decir, tienen una memoria que captura información sobre lo calculado previamente. [30]
- Redes Neuronales Totalmente Convolucionales (FCN): son redes que como su nombre indica están compuestas básicamente por operaciones convolutivas. Éstas permiten clasificar una imagen de entrada, la principal característica de las FCN es que permiten preservar información espacial y responder a dónde se encuentra el objeto en una imagen. Además, hacen uso de la deconvolución su objetico es extraer una determinada característica dados unos datos alterados y que enmascaran a esa característica buscada. [31]
- Redes Neuronales Profundas Convolucionales (CNN): son redes neuronales de múltiples capas. La idea detrás de las redes neuronales convolucionales es la de un filtro en movimiento que pasa a través de la imagen, éste filtro en movimiento o convolución, se aplica a una cierta vecindad de nodos. Éste tipo de redes reducen drásticamente la cantidad de parámetros que deben ajustarse. Manejan de manera eficiente la alta dimensionalidad de las imágenes en bruto. Yann LeCun fue pionero en este tipo de redes. [30]
- Teniendo en cuenta el flujo de datos, podemos distinguir entre redes unidireccionales (feedforward) y redes recurrentes o realimentadas (feedback). Mientras que en las redes unidireccionales la información circula en un único sentido, en las redes recurrentes o realimentadas la información puede circular entre las distintas capas de neuronas en cualquier sentido, incluso en el de salida-entrada.[32]

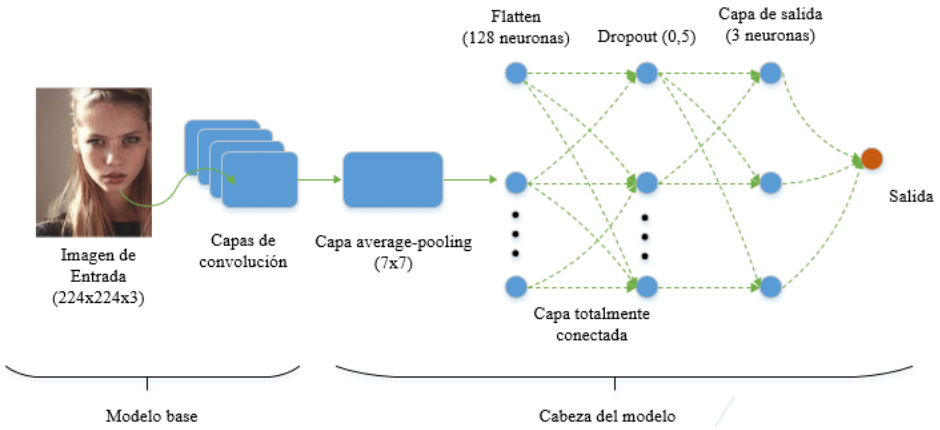

Figura 3-3. Red Neuronal Convolucional.

3.2.2 Capas

El bloque de construcción principal de las redes neuronales es la capa. Ésta es un módulo de procesamiento de datos que recibe como entrada uno o más tensores y devuelve como salida uno o más tensores. Si usamos datos vectoriales sencillos, almacenados en tensores 2D con la forma muestras y características, son procesados a menudo por capas densamente conectadas de la clase Dense de Keras, que explicaremos lo que es en el siguiente punto.

Cada capa solo aceptará a tensores de entrada con una determinada forma específica y devolverá tensores de salida con una forma determinada.

Al utilizar Keras, no tenemos que preocuparnos por la compatibilidad, porque las capas que añadimos a los modelos se construyen de manera dinámica para corresponderse con la forma de la capa entrante.

3.2.2.1 Distintos tipos de capas

Se distinguen tres tipos de capas: de entrada, de salida y ocultas. [32]

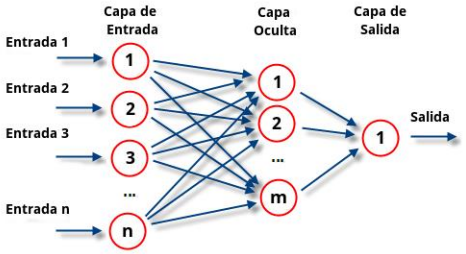

Figura 3-4. Capas en una Red Neuronal Artificial. [33]

- Una capa de entrada, también denominada sensorial, está compuesta por neuronas que reciben datos o señales procedentes del entorno. [32]
- Una capa de salida está compuesta de neuronas que proporcionan la respuesta de la red neuronal. [32]
- Una capa oculta no tiene una conexión directa con el entorno, es decir, no se conecta directamente ni a órganos sensores ni a efectores. Este tipo de capa oculta proporciona grados de libertad a la red neuronal gracias a los cuales es capaz de representar más fehacientemente determinadas características del entorno que trata de modelar. [32]

Teniendo en cuenta diversos conceptos se pueden establecer diferentes tipos de arquitecturas neuronales. Así considerando su estructura podemos hablar de redes monocapa –compuestas por una única capa de neuronas– o redes multicapa –las neuronas se organizan en varias capas–.[32]

3.2.3 Funciones de activación

Todas las variables independientes de la capa de entrada, pertenecen a una única observación, una única muestra.[34]

En el núcleo de la neurona es donde se procesan las señales de entrada y los pesos. Una manera posible de hacerlo es multiplicar cada señal de entrada por su peso y sumarlo todo, es decir, hacer una combinación lineal de los pesos de las conexiones y las entradas.[34]

$$w0x0 + w1x1 + w2x2 + \cdots + wNxm$$

Después se aplica una función de activación al resultado del primer paso, y el resultado final se propaga hacia la salida.

$$y = \phi\ (w0x0 + w1x1 + w2x2 + \cdots + wNxm)$$

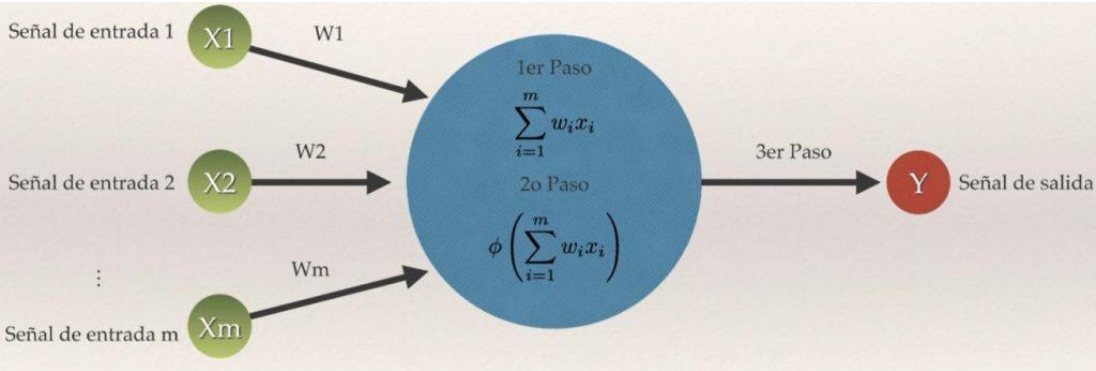

Figura 3-5. Procesado de información en una neurona.

Las funciones de activación [34], son la manera de transmitir la información por las conexiones de salida. Si nos interesa transmitir la información sin modificaciones usamos la función identidad. Hay funciones de activación del tipo lineal que son la clasificación binaria y la multiclase. Sin embargo, las funciones de activación se suelen usar para dar una no linealidad al modelo y que la red sea capaz de resolver problemas más complejos. Si todas las funciones de activación fueran lineales, la red resultante sería como una red sin capas ocultas.

Las funciones de activación no lineales más usadas son [34]:

- Función escalón (threshold): se puede ver en la figura que en el eje x está representado el valor ya ponderado de las entradas y sus respectivos pesos y en el eje y tenemos el valor de la función escalón. Ésta da un 0 si el valor de x es negativo y un 1 si es positivo. Es la función más estricta, dado que no hay casos intermedios.

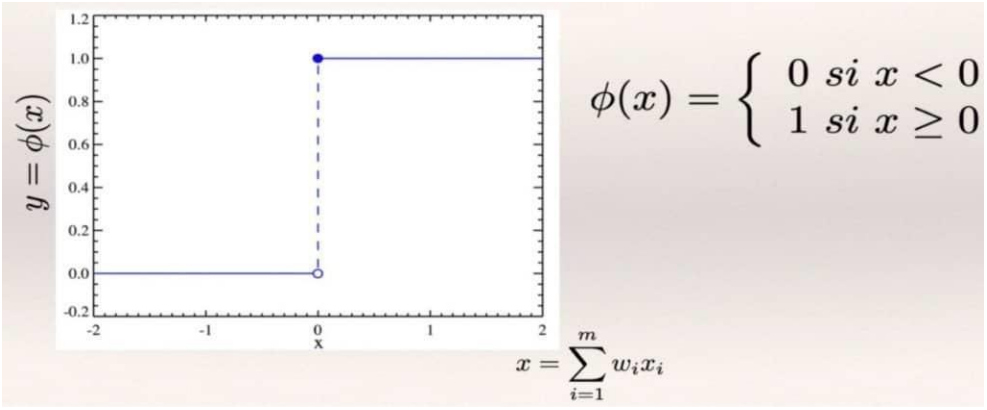

Figura 3-6. Función escalón.

– Función sigmoide: ésta no es tan estricta como el escalón, hace el cambio entre 0 y 1 de una forma más suave. Es usada en Machine Learning en regresión logística, una de las técnicas más usadas. La función no tiene aristas, es más suave. Cuanto más positivo es el valor de x más nos acercamos a 1 y por el contrario, cuanto más negativo es x más nos acercamos a 0. Es muy útil en la capa final de salida cuando acaba la red neuronal, no solo para clasificar con valores categóricos, sino también para intentar predecir las posibilidades de pertenecer a cada categoría, donde se sabe que la probabilidad de un suceso seguro es 1 y la de imposible es 0.

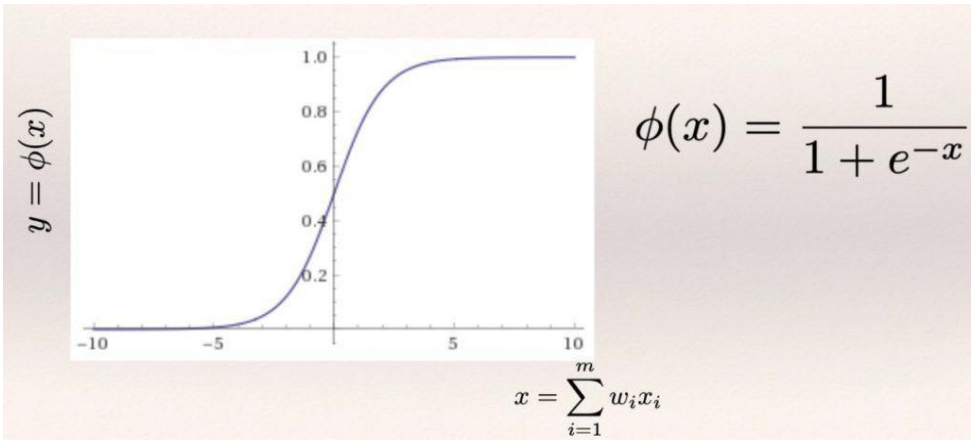

Figura 3-7. Función sigmoide.

– Función rectificadora (ReLu): volvemos a tener en esta función una arista. Su valor es 0 para cualquier valor negativo de x. Si x es positivo la función retorna el propio valor de x. Es decir, se queda con el valor exacto de las positivas y todos los demás valores negativos los hace 0. Los valores positivos no se restringen a 1

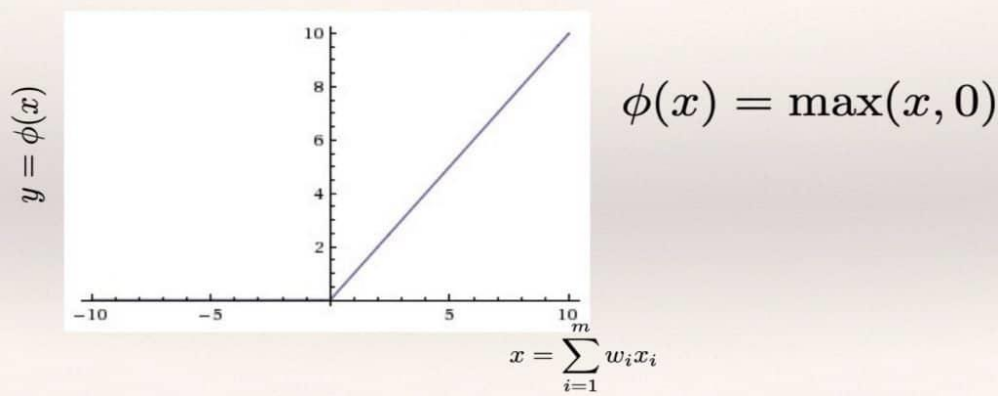

Figura 3-8. Función ReLu.

- Función tangente hiperbólica: es muy parecida a la función sigmoide, pero en este caso, como se ve en la figura empieza en -1 y termina en 1.

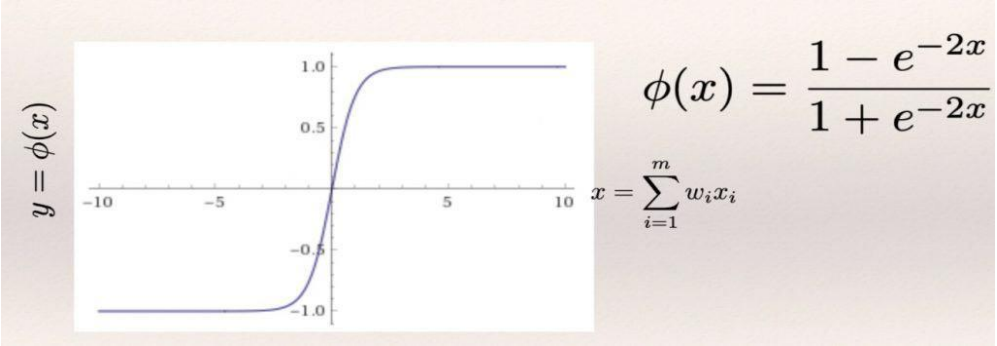

Figura 3-9. Función tangente hiperbólica

3.2.4 Parámetros e Hiperparámetros

Parámetro e hiperparámetro parecen dos conceptos parecidos, pero no tienen nada que ver.

Los parámetros son las variables que se estiman cuando se lleva a cabo el proceso de entrenamiento con los conjuntos de datos. En los modelos de aprendizaje automático son la parte más importante, es la parte de los modelos que se aprende de los datos. Son necesarios para realizar predicciones. También definen la capacidad del modelo para resolver un problema. Estimarlos de manera correcta es muy importante. La forma más habitual de estimar los parámetros de los modelos es mediante algoritmos de optimización. [35]

Algunos ejemplos de parámetros son: los coeficientes en una regresión lineal o logística, los pesos en una red neuronal artificial y los vectores de soporte en una máquina de vectores de soporte.

Los hiperparámetros son los valores de las configuraciones utilizadas durante el proceso de entrenamiento. Estos valores son indicados por un científico de datos. El valor óptimo de un hiperparámetro no se conoce a priori en un problema dado. Hay que usar valores genéricos o que han funcionado bien en problemas parecidos, otra opción es buscar los mejores valores mediante prueba y error. Al entrenar un modelo de aprendizaje automático se fijan los valores de los hiperparámetros para que con ellos se obtengan los parámetros. [35]

Algunos ejemplos de hiperparámetros son: la ratio de aprendizaje en el algoritmo del descenso del gradiente, el número de vecinos en k-vecinos más cercanos (k-nn) y la profundidad máxima en un árbol de decisión.

3.2.5 Entrenamiento de la red

El entrenamiento de una red neuronal se lleva a cabo ajustando los pesos de todas las entradas de las neuronas que forman la red, para que las respuestas de la capa de salida se ajusten lo más posible a el conjunto de datos que conocemos. A medida que se ajustan los pesos, el error disminuye. Debemos poner un elevado número de imágenes del tipo que queremos estudiar y de otros tipos intentando poner la mayor variedad posible en el entrenamiento para que la red neuronal sea capaz de generalizar e identificar dicho tipo de imágenes.

En cada ciclo (epoch) todos los datos de entrenamiento pasan a través de la red neuronal para que esta aprenda sobre ellos. Si hay 10 ciclos y 1000 datos, entonces 1000 datos en cada ciclo pasarán por la red neuronal. Si se especifica el parámetro tamaño del lote (batch size), cada epoch tendrá más ejecuciones internas. Estas ejecuciones se denominan iteraciones. Si nuestro batch size es de 100, habrá 10 iteraciones para completar un ciclo.

3.2.6 Conjunto de datos

Debido a la elevada personalización de la que dispone un modelo de Deep Learning y de los numerosos parámetros que puede contener, es muy normal construir un modelo que pueda "sobreajustar" o "subajustar", siempre y cuando se deje entrenando al modelo con las suficientes epochs.

En el entrenamiento cuanto más baja sea la pérdida de los datos de entrenamiento, más baja es también la de los datos de prueba. Cuando sucede eso decimos que el modelo está "subajustado": todavía hay progreso por hacer; la red aún no ha modelado todos los patrones relevantes en los datos de entrenamiento. Pero, pasado un número determinado de iteraciones por los datos de entrenamiento, la generalización deja de mejorar y la métrica de validación se para y empieza a bajar: el modelo está empezando a "sobreajustarse". Es decir, está empezando a aprender patrones que son específicos para los datos de entrenamiento, pero que son engañosos o irrelevantes si pensamos en los datos nuevos.

Para evitar esto lo mejor es conseguir más datos de entrenamiento. Cuando no es posible conseguir más datos, la siguiente mejor solución será modular la cantidad de información que puede almacenar nuestro modelo o añadir restricciones a qué tipo de información puede almacenar. Si una red puede memorizar una pequeña cantidad de patrones, el proceso de optimización la obligará a concentrarse en los más destacados, esto mejora la probabilidad de generalizar bien. Por tanto, el proceso de combatir así el "sobreajuste" se denomina regularización.

3.2.7 Regularización

Las técnicas más usadas para la regularización son [36]:

3.2.7.1 Reducir el tamaño de la red

El sobreajuste se puede evitar de una forma muy sencilla que es reducir el tamaño del modelo, el número de parámetros aprendidos del mismo, éste viene determinado por el número de capas y el de unidades por capa. En Deep Learning, normalmente el número de parámetros aprendidos de un modelo recibe el nombre de capacidad del modelo. Un modelo con más parámetros tiene más capacidad de memorización y, por consiguiente, puede aprender con mayor facilidad una asignación perfecta de tipo diccionario entre las muestras de entrenamiento y sus objetivos, una asignación sin poder de generalización.

Por otro lado, si los recursos de memoria de la red son limitados, esta asignación no se puede aprender fácilmente; por lo tanto, para minimizar su pérdida, debemos recurrir al aprendizaje de

representaciones comprimidas que sean predictivas del objetivo. Es el tipo de expresión que nos interesa. Al mismo tiempo, sabemos que debemos usar un modelo con parámetros suficientes para no subajustarse: el modelo no debe carecer por completo de recursos de memoria. Debemos encontrar un equilibrio entre el exceso de capacidad y la capacidad insuficiente. Desafortunadamente, no existe una fórmula para determinar el número correcto de capas o el tamaño de cada capa. Tendremos que usar datos de validación para evaluar muchas arquitecturas diferentes para encontrar un tamaño de modelo que se ajuste a nuestros datos. El flujo de trabajo general para encontrar el tamaño de modelo correcto comienza con relativamente pocas capas y parámetros y aumenta el tamaño de las capas o añade nuevas hasta que vemos que disminuyen las salidas respecto a la pérdida de validación.

3.2.7.2 Añadir regularización de pesos L1 y L2

El principio de la navaja de Occam nos dice que dadas dos explicaciones para algo, la que más probabilidad tiene de ser la acertada es la más sencilla, la que ha hecho menos elucubraciones. Esta es la idea que se aplica a los modelos que aprenden de las redes neuronales. Si nos dan unos datos de entrenamiento y una arquitectura de red, unos cuantos de conjuntos de valores de peso podrían explicar los datos. Los modelos más simples tienen menos probabilidades de sufrir "sobreajuste" que los más complejos.

Es decir, un modelo sencillo es uno en el que la distribución de los valores de parámetro tiene menos entropía. Por consiguiente, una forma usual de mitigar el "sobreajuste" es limitar la complejidad de la red forzando a sus pesos a tomar sólo los valores pequeños, lo que conlleva a que la distribución de los valores de peso sea más regular. Esto es llamado regularización de peso y se hace añadiendo a la función de pérdida de la red un coste asociado a tener pesos grandes.

La penalización puede ser de dos maneras, regularización L1 o L2, que vamos a explicar a continuación.

En la regularización L1, el coste añadido es proporcional al valor absoluto de los coeficientes de peso, la norma L1 de los pesos.

En la regularización L2, el coste añadido es proporcional al cuadrado del valor de los coeficientes de peso, la norma L2 de los pesos.

En Keras, la regularización del peso se añade pasando instancias regularizadoras de peso a las capas como argumento clave.

3.2.7.3 Añadir Dropout

Es una de las técnicas más efectivas y más usadas. Aplicarla a una capa reside en retirar, poner a cero, aleatoriamente un número de características de salida de la capa mientras se hace el entrenamiento. La tasa de dropout es la fracción de las características que se ponen a cero, normalmente está entre 0.2 y 0.5. Mientras que realizamos la prueba no retiramos ninguna unidad, en su lugar, los valores de salida de la capa se reducen en un factor igual a la tasa de dropout, para igualar el hecho de que haya más unidades activas en el entrenamiento.

En Keras, el dropout se introduce a través de una capa dropout, que se aplica a la salida de la capa justo delante de ella.

3.2.7.4 Utilizar el aumento de datos

El "sobreajuste" es consecuencia de tener muy pocas muestras de las que aprender. El aumento de datos opta por generar más datos de entrenamiento a partir de las muestras que ya tenemos, aumentando las muestras mediante transformaciones aleatorias que producen imágenes con aspecto creíble. Lo que intentamos es que en el momento del entrenamiento el modelo no vea nunca dos veces la misma imagen idéntica. Esto ayuda a exponer al modelo a más aspectos de los datos y a generalizar mejor.

En Keras, puede hacerse el aumento de datos configurando transformaciones aleatorias y operaciones de normalización que se realizarán en sus datos de imagen durante el entrenamiento. Para usar ImageDataGenerator tenemos que importar la clase de Keras de procesamiento de imagen.

Figura 3-10. Aumento de datos.

3.2.8 Aprendizaje transferido / redes 'from scratch' (desde cero)

El aprendizaje transferido es el proceso de entrenar un modelo en un conjunto de datos a gran escala y luego usar ese modelo previamente entrenado para llevar a cabo el aprendizaje para otra tarea posterior (es decir, la tarea objetivo). El aprendizaje por transferencia se popularizó en el campo de la visión por computadora gracias al conjunto de datos ImageNet [14].

Se pueden distinguir 3 tipos de arquitecturas [37]:

- CNN preentrenada como un extractor de características fijo: consiste en extraer características de redes para clasificar imágenes, como Alexnet, VGG u otras, y luego clasificar mediante el enfoque clásico de segmentación semántica. El enfoque clásico consiste el uso de algún tipo de superpixeles más un clasificador y, de forma opcional, un refinamiento mediante algún método basado en modelos de probabilísticos de grafos. Esto funciona si las características son generales, es decir, no son específicas para la tarea original.
- Ajuste fino de una CNN preentrenada (fine tuning): es otra forma de transferir el conocimiento que consiste en no tocar las capas convolucionales cercanas a la entrada y entrenar las capas cercanas a la salida y las capas completamente conectadas.
- Modelos preentrenados: se lleva a cabo usando el modelo completo ya entrenado y se ajusta con más precisión con el nuevo conjunto de datos. Es habitual en la comunidad publicar modelos en un punto de control que funcionan correctamente y que otros usuarios puedan usar esas redes para realizar ajustes.

Cuando usamos una red neuronal profunda, primero se escoge la arquitectura más adecuada para el trabajo en particular. Después se tiene que inicializar la arquitectura con todos los parámetros del modelo pre-entrenado, es decir, elegir el valor inicial de todas las constantes que se multiplicarán o se sumarán a las variables de entrada de cada capa de nuestra red, de forma que los resultados finales sean mejores y se alcancen mucho antes. A continuación, se quita la capa de salida de la arquitectura base y se añaden capas adicionales, de forma que se pueda retocar el modelo original para encaminarlo a nuestro problema concreto.

Se mejoran mucho los resultados si a la hora de aplicar aprendizaje transferido se entrena primero el modelo con versiones reducidas de las imágenes originales y luego se entrena con imágenes más grandes. Esto se conoce como redimensionamiento progresivo y funciona usando un tamaño mínimo de 64x64 píxeles.

A la hora de entrenar cualquier red usando aprendizaje transferido se puede usar un procedimiento casi genérico:

- 1ª etapa: se dejan los parámetros correspondientes a la arquitectura base inamovibles. Y se elige una tasa de aprendizaje adecuada para el ajuste de los parámetros de las capas adicionales. El tiempo de entrenamiento se pone pequeño porque hay pocos parámetros que ajustar, es decir, pocas epochs, pero sobre la totalidad de los datos.
- 2ª etapa: se realiza un ajuste fino para mejorar el modelo. Se ajustan los parámetros del modelo base que antes habíamos dejado inamovibles. Se elige un rango para la tasa de aprendizaje máxima en las distintas capas, de manera que los pesos iniciales varíen poco y que los pesos finales tengan mayor margen de modificación. Las epochs serán más en esta etapa en un principio y hasta que el error de validación empeore de forma sostenida. [38]

La base de datos desde donde se desean extraer las características debe ser bastante más grande que base donde se van a transferir. A su vez ambos problemas deben tener cierta relación, siendo el caso ideal donde el problema al que se buscan transferir las características sea un sub-problema dentro de la base de datos de la red donante de características. Este es el caso de muchas de las bases de datos de imágenes pequeñas donde la gran mayoría de las clases de estas bases de datos están ya en Imagenet. Existen casos donde no se da esta relación directa de las clases, pero gracias a la inmensa cantidad de clases de Imagenet, se logran generar características útiles para estos problemas.

4 Metodología propuesta

4.1 Material: base de datos

La base de datos ha sido obtenida de Argenziano, G., Soyer, H. P., De Giorgi, V., Piccolo, D., Carli, P., & Delfino, M. (2002). Dermoscopy: a tutorial. EDRA, Medical Publishing & New Media, 35. Nos proporcionan una base de datos principal que contiene imágenes en formato BMP con tamaños 60x60 y de 80x80, las de tamaño 80x80 las redimensionamos a 60x60 que son de las que más hay. En la base de datos se identifican tres tipos de patrones: globular, homogéneo y reticulado. En total tenemos 1846 imágenes.

	Casos	**% sobre total**
Globular	1240	67%
Homogéneo	343	19%
Reticulado	262	14%
Total	1845	100%

Tabla 4-1. Base de datos inicial.

Observamos que hay un desbalanceo de casos a favor del conjunto de datos globular, ya que él sólo presenta dos tercios del total de casos. Esto puede influir negativamente en los resultados.

A continuación, mostramos imágenes de ejemplo de los tres patrones:

Tabla 4-2. Tipos de patrones.

4.2 Implementación de la red

Python cuenta con facilidades para la programación orientada a objetos, imperativa y funcional, por lo que se considera un lenguaje multi-paradigmas. Fue basado en el lenguaje ABC y se dice que fue influenciado por otros como C, Algol 60, Modula-3 e Icon según su propio autor.

Es un lenguaje de alto nivel ya que contiene implícitas algunas estructuras de datos como listas, diccionarios, conjuntos y tuplas, que permiten realizar algunas tareas complejas en pocas líneas de código y de manera legible. [39]

La sintaxis de Python es muy sencilla, tanto que en algunas ocasiones parece pseudocódigo. Es muy interesante observar las diferencias que existen entre el programa Hola Mundo de Python y el de otro lenguaje de alto nivel como C++:

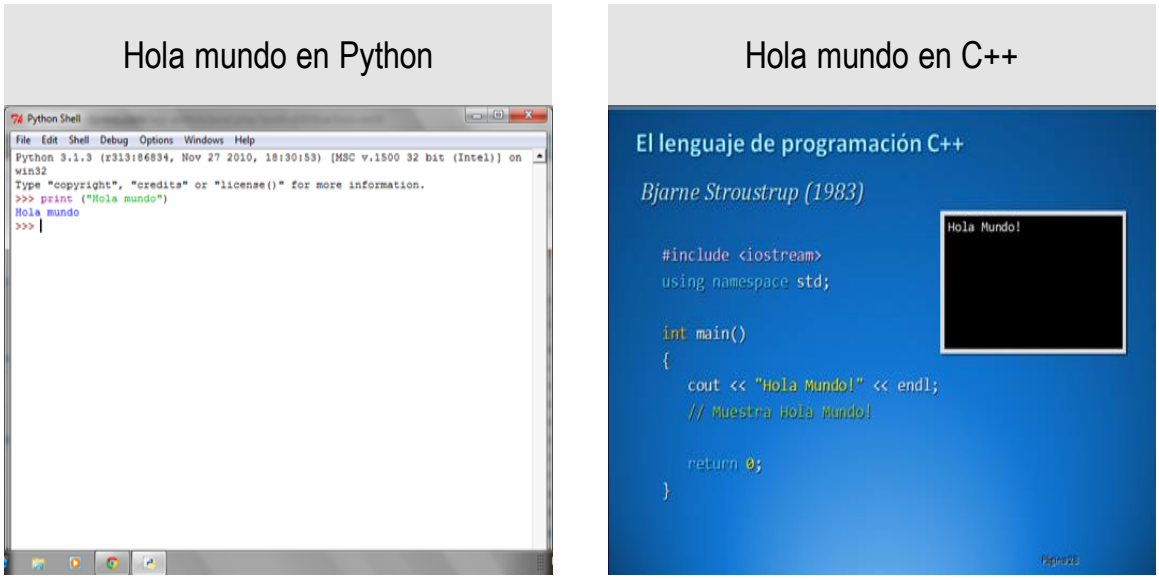

Tabla 4-3. Comparación Python y C++.

Keras es una interfaz de programación de aplicaciones (API) de aprendizaje profundo escrita en Pyhton, se ejecuta sobre la plataforma de aprendizaje automático TensorFlow. Keras fue desarrollado para permitir una experimentación rápida. Las estructuras de datos centrales de Keras son capas y modelos. El tipo más simple es el modelo secuencial, una pila lineal de capas.

Las características de Keras son:

- Simplicidad: reduce la carga cognitiva del desarrollador para que pueda concentrarse en las partes del problema que realmente importan.
- Flexibilidad: los flujos de trabajo simples deben ser rápidos y fáciles, mientras que los flujos de trabajo arbitrariamente avanzados deben ser posibles a través de una ruta clara que se base en lo ya aprendido.
- Potencia: rendimiento y escalabilidad de la fuerza de la industria, es usado por organizadores y empresas como la NASA o YouTube. [40]

Por tanto, como ya se ha podido intuir, el lenguaje elegido para el desarrollo de este proyecto es Python, específicamente usaremos Keras encima de TensorFlow. Para entrenar redes neuronales es importante tener mucha potencia de procesamiento, es preciso cuando se hace uso

de procesamiento de imágenes usar unidades de procesamiento gráfico (GPUs), las cuales han aumentado mucho la velocidad de la que se disponía con una CPU.

Para entrenar el código hemos usado Google Colab que es una herramienta para programar y ejecutar programas on-line, utilizando máquinas que pone a disposición Google, que ya tienen instalados los compiladores y paquetes necesarios para programar modelos de Machine Learning en la nube.

Para usarla hay teclear https://colab.research.google.com/notebooks/intro.ipynb en la línea de direcciones del navegador. En esa página, debemos entrar con nuestro usuario de Google, es necesario tener cuenta en Google para usar Google Colab. En el menú elegimos "Archivo/Nuevo" y ya podemos empezar a programar en Python, aquí ejecutamos el código en el cual ya hacemos la importación de la librería de Keras y el modelo secuencial que es el que estamos usando.

Como estamos usando Google Colab tenemos las imágenes guardadas en la nube de google, en Google Drive.

Para el uso de Keras, en nuestro código de Google Colab, debemos añadir las siguientes líneas:

```
import keras
print(keras.__version__)
#preparamos para el modelo secuencial de keras
from keras import layers
from keras import models
from keras.utils.vis_utils import plot_model
from keras.preprocessing.image import ImageDataGenerator
import tensorflow.python.keras
from PIL import Image
```

Figura 4-1. Importaciones para el uso de Keras.

4.3 Red propuesta

Una red convolucional básica está formada por una pila de capas conv2D y MaxPooling2D.

Lo que toma como entrada esta red son tensores de la forma: altura, anchura y canales de la imagen, pero no incluye la dimensión del lote.

En el caso de este trabajo tenemos la red neuronal convolucional para que procese tensores de entrada de tamaño (60,60,3) que es el formato de las imágenes usadas, 60x60 y 3 porque están en RGB. Para pasar esto usamos el argumento input_shape a la primera capa.

MaxPooling2D reduce de manera drástica los mapas de características, al igual que las convoluciones con pasos de avance.

La salida de cada capa Conv2D y MaxPooling2D es un tensor 3D con la forma altura, anchura y canales.

Las dimensiones de altura y anchura se van viendo reducidas normalmente a medida que profundizamos en la red.

El número de canales que se pasa en las capas Conv2D, es 32 o 64, éste es el número de filtros de convolución empleado en cada capa y que el tamaño de la máscara de convolución de cada filtro es 3x3.

Antes de nada, tenemos que definir el modelo de Keras utilizando la clase Sequential. Esta clase es sólo para pilas lineales de capas, que es lo que estamos usando.

```
CNN = models.Sequential()
CNN.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(60, 60, 3)))
CNN.add(layers.MaxPooling2D((2, 2)))
CNN.add(layers.Conv2D(64, (3, 3), activation='relu'))
```

Figura 4-2. Creación del modelo secuencial.

Lo siguiente es introducir el último tensor de entrada en una red clasificadora densamente conectada, es decir, una pila de capas Dense. Estos clasificadores procesan vectores, que son 1D, mientras que la salida es un tensor 3D.

Primero tenemos que aplanar con la orden Flatten las salidas de 3D a 1D y después añadir capas Dense encima.

En este caso tenemos una primera capa Dense con 64 unidades de salida, Relu aplica una función de activación de unidad lineal rectificada.

La segunda capa Dense utiliza la función de activación Softmax, que generará como salida una distribución de probabilidad sobre las 3 clases de salida diferentes, para cada muestra de entrada, la red producirá un vector de salida de 3 dimensiones.

```
CNN.add(layers.Flatten())
CNN.add(layers.Dense(64, activation='relu'))
CNN.add(layers.Dense(3, activation='softmax'))
```

Figura 4-3. Creación de capas.

Con la línea de código:

```
plot_model(CNN, to_file='CNN2_plot.png', show_shapes=True, show_layer_names=True)
CNN.summary()
```

Figura 4-4. Impresión del modelo.

Podemos ver en una representación en pantalla, cuando ejecutamos todo el código anterior, que las salidas se han aplanado para convertirse en vectores (11,11,8) con la forma (968,) antes de pasar a través de las dos capas Dense.

```
2.4.3
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 58, 58, 8)         224
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 29, 29, 8)         0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 27, 27, 8)         584
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 13, 13, 8)         0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 11, 11, 8)         584
_________________________________________________________________
flatten (Flatten)            (None, 968)               0
_________________________________________________________________
dense (Dense)                (None, 8)                 7752
_________________________________________________________________
dense_1 (Dense)              (None, 3)                 27
=================================================================
Total params: 9,171
Trainable params: 9,171
Non-trainable params: 0
_________________________________________________________________
```

Figura 4-5. Ejecución del modelo.

4.3.1 Entrenamiento de las imágenes con la red neuronal convolucional

Ahora vamos a entrenar la red neuronal convolucional con las imágenes de lesiones pigmentadas.

Como estamos usando google colab tenemos las imágenes guardadas en la nube de google, en google drive.

4.3.1.1 Imágenes train

Le pasamos la ruta donde tenemos guardadas las imágenes de "train" entrenamiento, en carpetas separadas por según el tipo de patrón que queremos estudiar.

Con el método os.listdir, devolvemos una lista que contiene los nombres de las entradas en el directorio dado por la ruta, en nuestro caso exploramos el directorio path_train. La lista devuelta está en orden arbitrario.

Imprimimos el total de las imágenes por pantalla para asegurarnos de que las ha cogido bien.

```
path_imagenes1= '/content/drive/My Drive/lesionespigmentadas/train/globular'
path_imagenes2= '/content/drive/My Drive/lesionespigmentadas/train/homogeneo'
path_imagenes3= '/content/drive/My Drive/lesionespigmentadas/train/reticulado'
listaFicheros1= os.listdir(path_imagenes1)
listaFicheros2= os.listdir(path_imagenes2)
listaFicheros3= os.listdir(path_imagenes3)

patrones2= ['globular','homogeneo','reticulado']

totalImagenes1 = len(listaFicheros1)
print(totalImagenes1)
totalImagenes2 = len(listaFicheros2)
print(totalImagenes2)
totalImagenes3 = len(listaFicheros3)
print(totalImagenes3)
totalImagenes = totalImagenes1 + totalImagenes2 + totalImagenes3
print(totalImagenes)
```

Figura 4-6. Acceso a las imágenes de train.

Primero le pasamos las imágenes separadas por patrones del tipo globular, luego las del tipo homogéneo y por último las de reticulado.

En range vamos sumando exhaustivamente las que tenemos de cada tipo para ir creando la matriz.

Por último, imprimimos el valor de la matriz Y_training y nos tiene que dar la suma de los elementos de la matriz, es decir, el total conjunto de las imágenes de los tres tipos.

```
Y_training=np.zeros((totalImagenes,len(patrones2)))

for k in range(1116):
  Y_training[k,0]=1
for k in range(242):
  Y_training[k+1116,1]=1
for k in range(236):
  Y_training[k+1358,2]=1
print(Y_training)
```

Figura 4-7. Crear la matriz de datos train.

Usamos la función enumerate que toma una colección, por ejemplo, una tupla y la devuelve como un objeto enumerado. Esta función agrega un contador como clave del objeto enumerado.

A continuación, abrimos con el método path.join, que une los segmentos de ruta especificados en la ruta. Los segmentos de ruta deben ser cadenas separadas por comas.

Por último el método append agrega un elemento al final de la lista y con np.asarray(im) le damos im como entrada y lo convierte en una matriz.

```
a = []
for i,imagen in enumerate(listaFicheros):

  im = Image.open(os.path.join(path_imagenes,imagen))
  a.append(np.asarray(im))


train_set_x =np.asarray(a)
print(train_set_x.shape)
print(len(a))
```

Figura 4-8. Recorrer la matriz de train.

Por último, vectorizamos las etiquetas para acabar con valores de punto flotante en el rango 0 a 1.

La función astype crea una copia de la matriz y le permite especificar el tipo de datos como parámetro, es decir hacemos una copia de la matriz y la convertimos a float32 dividiendo entre 255 para normalizar.

```
train_set_x = train_set_x.astype('float32') / 255

print(train_set_x)
```

Figura 4-9. Conversión y normalización de datos train.

4.3.1.2 Imágenes test

Para las imágenes de test hacemos lo mismo:

Le damos la ruta y nos devuelve una lista con todas las imágenes de test.

```
path_test1= '/content/drive/My Drive/lesionespigmentadas/test/globular'
lista_test1= os.listdir(path_test1)
path_test2= '/content/drive/My Drive/lesionespigmentadas/test/homogeneo'
lista_test2= os.listdir(path_test2)
path_test3= '/content/drive/My Drive/lesionespigmentadas/test/reticulado'
lista_test3= os.listdir(path_test3)

total_test1 = len(lista_test1)
total_test2 = len(lista_test2)
total_test3 = len(lista_test3)
total_test = total_test1 + total_test2 + total_test3

print(total_test1)
print(total_test2)
print(total_test3)
```

Figura 4-10. Acceso a las imágenes de test.

Hacemos una matriz de ceros que vamos llenando con las imágenes como filas y los patrones como columnas.

```
Y_test=np.zeros((total_test,len(patrones2)))
for k in range(124):
 Y_test[k,0]=1
for k in range(27):
 Y_test[k+124,1]=1
for k in range(26):
 Y_test[k+151,2]=1
```

Figura 4-11. Crear la matriz de datos test.

Creamos una matriz y aplicamos el bucle for para irla recorriendo.

```
b = []
for i,imagen in enumerate(lista_test):

  im = Image.open(os.path.join(path_test,imagen))
  b.append(np.asarray(im))


test_set_x = np.asarray(b)
```

Figura 4-12. Recorrer la matriz de test.

Convertimos los valores al tipo deseado y normalizamos.

```
test_set_x = test_set_x.astype('float32') / 255
```

Figura 4-13. Conversión y normalización de datos test.

4.3.2 Compilación del modelo y resultados

El método compile configura el modelo para el entrenamiento. A continuación, usamos el método fit para entrenar la red.

Luego evaluamos las pérdidas y la precisión y hacemos una representación por pantalla de lo que nos devuelve el entrenamiento.

```
CNN.compile(loss='categorical_crossentropy',optimizer='adam',
            metrics=['accuracy'])

modelo_history = CNN.fit(x=train_set_x,y=Y_training,epochs=60,verbose=1,shuffle=1)


test_loss, test_acc = CNN.evaluate(test_set_x, Y_test)

test_acc

y_pred = CNN.predict(test_set_x)
y_pred_decimal=np.argmax( y_pred ,axis=1)
y_real=np.argmax( Y_test ,axis=1)
print(y_pred[:5],Y_test[:5],y_pred_decimal[:5],y_real[:5])
```

Figura 4-14. Compilación del modelo.

4.4 Ajustes de la red

4.4.1 Redimensionamiento y cambio de formato de las imágenes

El paquete de recortes que nos proporcionan no está todo en las mismas dimensiones, entonces debemos redimensionarlo empleando el siguiente código y cambiar el formato de las imágenes, porque todas no estaban en el formato '.bmp'

```
#Redimensionar imágenes
os.getcwd()
collection="/content/drive/My Drive/lesionespigmentadas/train"
j=0
for j,filename in enumerate(os.listdir(collection)):
  img=Image.open(filename)
  new_img=img.resize((60,60))
  new_img.save=(filename, 'bmp')
  j=j+1
################################
#cambiar formato de imagenes
im=Image.open('Acl225.jpg')
im.save('Acl225.bmp')
```

Figura 4-15. Redimensionamiento y cambio de formato de las imágenes.

4.4.2. Balanceo de casos

Como comentamos en el epígrafe "4.1. Material: base de datos", existe un desbalanceo de casos en favor del conjunto de imágenes del patrón globular a tener en cuenta. A fin de corregirlo y observar si esto influye en los resultados, proponemos las siguientes soluciones:

1. Aumentar los casos de los conjuntos con patrones homogéneo y reticulado. Esto se aplica solo al conjunto de entrenamiento una vez separados los conjuntos de entrenamiento y test.
2. Reducir los casos del conjunto patrón globular.

4.4.2.1. Aumento de datos patrón homogéneo y reticulado

Para llevar a cabo el aumento de los datos de los conjuntos usamos Data Augmentation en los conjuntos de entrenamiento con patrones homogéneo y reticulado.

```
repetitions=4
def data_augmentation(repetitions):


  data_gen=ImageDataGenerator(rotation_range=90, horizontal_flip=True, vertical_flip=True, zoom_range=[0.5,1.0],
  rescale=1./255)
  save_here='/content/drive/MyDrive/lesionespigmentadas/DataAugmentation/reticuladoDA'
    #aqui arriba guardo las imagenes que he aumentado
  print(save_here)
  image_path=glob.glob('/content/drive/MyDrive/lesionespigmentadas/train/reticulado'+'/*.bmp')
  print(image_path)
  L=len(image_path)
  for i in range(L):
      #print(i)
      image=np.expand_dims(plt.imread(image_path[i]),0)
      data_gen.fit(image)
      for x,val in zip(data_gen.flow(image, save_to_dir=save_here, save_prefix='DA'+'_'+'reticulado', save_format='bmp'),
       range(repetitions)):
        pass
```

Figura 4-16. Data Augmentation.

Tras aplicar lo anterior en ambos conjuntos con un valor de prueba, observamos que los casos nuevos generados en el conjunto patrón homogéneo presentan unos colores que no son propios de las imágenes originales de este patrón. Hemos realizado varias pruebas, pero siempre pasa esto. Es por ello que descartamos el aumento de datos en este conjunto.

4.4.2.2. Reducción de datos del patrón globular

Proponemos una reducción de dividir entre tres los casos del conjunto patrón globular a fin de que este no represente un porcentaje tan elevado de los casos sobre el total de ellos. Y sea más parecido a los casos de los conjuntos patrones homogéneo y reticulado. Pasamos de tener 1240 imágenes entre entrenamiento y test, a tener 413 en total, hemos reducido un total de 827 imágenes.

4.5. Evaluación

4.5.1. Conjuntos de entrenamiento y de test

Para las pruebas vamos a utilizar un 90% de los casos como conjunto de entrenamiento y el 10% restante como conjunto de test. Si se lleva a cabo la reducción de datos, la división de los conjuntos de entrenamiento y test se producirá a posteriori. Sin embargo, si se lleva a cabo el

aumento de los datos, la división de los conjuntos de producirá a priori, llevando a cabo, por tanto, el aumento de datos únicamente en el conjunto de entrenamiento.

4.5.2. Medidas de rendimiento

Hay muchas métricas posibles para medir la calidad de la clasificación de un modelo, tomar una métrica u otra depende de lo que queramos valorar o de dar más importancia a ciertos tipos de errores o aciertos. [41]

Vamos a ver cuatro métricas diferentes, pero antes vamos a introducir los parámetros con los que se calculan dichas métricas:

- Falso Positivo (FP): el valor real es negativo y el modelo predice un positivo.
- Falso Negativo (FN): el valor real es positivo y el modelo predice un negativo.
- Verdadero Positivo (TP): el valor real es positivo y el modelo predice un positivo.
- Verdadero Negativo (TN): el valor real es negativo y el modelo predice un negativo.

La matriz de confusión es un método de evaluación de rendimiento de un modelo de clasificación. La matriz compara los valores reales con los predichos por el modelo de aprendizaje. Esto nos ayuda a ver cómo de bien está funcionando el modelo. Los parámetros anteriores los podemos extraer haciendo la matriz de confusión en nuestro modelo.

Figura 4-17. Matriz de confusión binaria.

De la matriz de confusión surgen las métricas:

Sensibilidad (sensitivity): se conoce también como tasa de verdaderos positivos, es la proporción de casos verdaderos positivos que fueron correctamente identificados por el modelo.

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

Especificidad (specificity): se conoce también como tasa de verdaderos negativos, es la proporción de casos verdaderos negativos que fueron correctamente identificados por el modelo.

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

Tasa de acierto (accuracy): se refiere a lo cerca que está el resultado de una medición del valor verdadero. En términos estadísticos, la exactitud está relacionada con el sesgo de una estimación. Se representa como la proporción de resultados verdaderos dividido entre el número total de casos examinados. Es decir, es la cantidad de predicciones positivas que fueron verdaderas.

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

Precisión (precisión): es la dispersión del conjunto de valores obtenidos a partir de mediciones repetidas de una magnitud. Mientras menor es la dispersión mayor es la precisión. Se representa como la proporción de verdaderos positivos dividido entre todos los resultados positivos. Es decir, es el porcentaje de casos positivos detectados, es la probabilidad de que la lesión pertenezca a una categoría, si el programa la ha clasificado como perteneciente a la misma categoría, a esto también se le llama valor predictivo positivo (Positive Predictive Value, PPV)

$$Precisión = PPV = \frac{TP}{TP + FP}$$

Valor predictivo negativo (Negative Predictive Value, NPV): es la probabilidad de que la lesión no pertenezca a una categoría, si el programa no la ha clasificado como perteneciente a la misma categoría.

$$NPV = \frac{TN}{TN + FN}$$

En nuestro trabajo como tenemos 3 tipos de patrones la matriz de confusión es de 3x3, para el cálculo de las métricas lo hacemos así:

		Valores predichos		
		Globular	Homogéneo	Reticulado
Valores reales	Globular	Celda 1	Celda 2	Celda 3
	Homogéneo	Celda 4	Celda 5	Celda 6
	Reticulado	Celda 7	Celda 8	Celda 9

Tabla 4-4. Cálculo de métricas en matriz de confusión.

En el caso de Globular:

- TP= celda 1
- FP= celda 2 + celda 3
- TN= celda 5 + celda 6 + celda 8 + celda 9
- FN= celda 4 + celda 7

En el caso de Homogéneo:

- TP= celda 5
- FP= celda 4 + celda 6

- TN= celda 1 + celda 3 + celda 7 + celda 9
- FN= celda 2 + celda 8

En el caso de Reticulado:

- TP= celda 9
- FP= celda 7 + celda 8
- TN= celda 1 + celda 2 + celda 4 + celda 5
- FN= celda 3 + celda 6

Consideraremos que los resultados son satisfactorios si los verdaderos positivos (TP), los cuales están en la diagonal de la matriz de confusión, y tenemos uno para cada patrón, son superiores al 70% de todos los datos que tenemos en el conjunto test de cada caso.

4.6. Pruebas

Vamos a llevar a cabo cuatro pruebas (a 50, 60, 70 y 80 epochs, respectivamente) para cada uno de los siguientes casos:

1. Pruebas sin aumento ni reducción de imágenes en ninguno de los conjuntos de patrones.
2. Pruebas sin aumento de imágenes en el conjunto patrón reticular y con reducción quedándonos con 413 imágenes en el patrón globular.
3. Pruebas con aumento de imágenes (Data Augmentation = 1) en el conjunto patrón reticular y sin reducción de imágenes en el patrón globular.
4. Pruebas con aumento de imágenes (Data Augmentation = 1) en el conjunto patrón reticular y con reducción de imágenes del patrón globular teniendo 413 imágenes.

Esto hace un total de 16 pruebas, las cuales serán evaluadas conforme lo descrito en el epígrafe "4.5. Evaluación" y valoradas en el epígrafe "5. Resultados".

Tabla de balanceo de casos según las pruebas propuestas anteriormente:

	% de patrón globular sobre el total	**% de patrón homogéneo sobre total**	**% de patrón reticular sobre total**
Sin aumento ni reducción de casos	67 %	18.6 %	14.2 %
Reducción de casos en patrón globular	40.6 %	33.7 %	25.7%
Aumento de casos en patrón reticular	60.8 %	16.8 %	22.4 %
Reducción de casos en patrón globular y aumento de casos en patrón reticulado	34 %	28 %	38 %

Tabla 4-5. Balanceo de casos.

A continuación, se muestra como quedarían los conjuntos de entrenamiento (train) y validación (test) en cada una de las pruebas propuestas:

	Globular	Homogéneo	Reticular
Sin aumento ni reducción de casos	1116 train/ 124 test	316 train/ 27 test	234 train/ 28 test
Reducción de casos en patrón globular	369 train/ 44 test	316 train/ 27 test	234 train/ 28 test
Aumento de casos en patrón reticular	1116 train/ 124 test	316 train/ 27 test	457 train/ 28 test
Reducción de casos en patrón globular y aumento de casos en patrón reticulado	369 train/ 44 test	316 train/ 27 test	457 train/ 28 test

Tabla 4-6. Train y test de pruebas propuestas.

5 Resultados

5.1. Pruebas sin aumento de casos

A continuación, se muestran las matrices de confusión obtenidas (Tabla 5-1), los parámetros de entrenamiento usados (Tabla 5-2), las parámetros de rendimiento obtenidos (Tabla 5-3), las métricas calculadas (Tabla 5-4) y por último el rendimiento global de la red (Figura 5-1) una vez realizadas las pruebas.

	50 epochs			60 epochs			70 epochs			80 epochs		
Sin aumento de casos	98	7	19	114	2	8	119	2	3	115	1	8
	4	21	2	6	20	1	10	16	1	6	17	4
	5	1	22	10	1	17	19	0	9	7	0	21

Tabla 5-1. Matrices de confusión sin aumento de casos.

Parámetro	Valor
Dimensiones de entrada	60x60x3
Tamaño del conjunto de datos	Total de imágenes iniciales
Epochs	50
Tipo de optimizador	Adam

Tabla 5-2. Parámetros entrenamiento prueba 1.

Parámetros	Globular	Homogéneo	Reticulado
Verdadero Positivo	98	21	22
Falso Positivo	26	6	6
Verdadero Negativo	46	144	130
Falso Negativo	9	8	21

Tabla 5-3. Parámetros sin aumento de casos.

Métricas	Globular	Homogéneo	Reticulado	Total
Sensibilidad	0.92	0.72	0.51	0.72
Especificidad	0.64	0.96	0.96	0.85
Accuracy	0.80	0.92	0.85	0.86
Precisión o PPV	0.79	0.77	0.79	0.78
NPV	0.84	0.95	0.86	0.89

Tabla 5-4. Métricas sin aumento de casos.

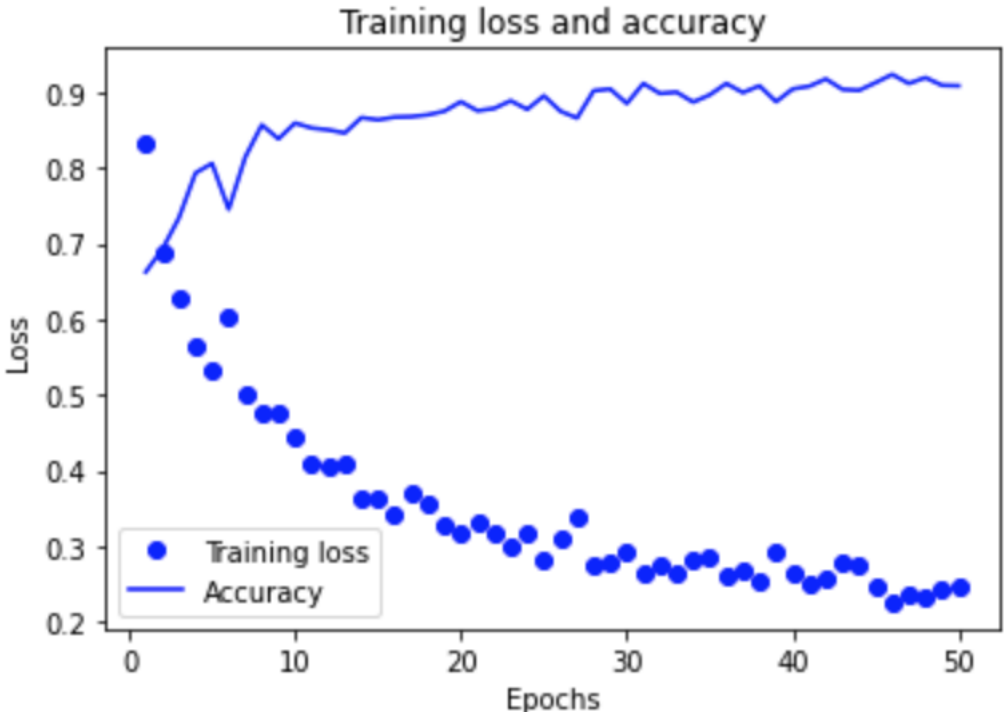

Figura 5-1. Training loss y accuracy prueba 1.

Se observa lo siguiente:

- Una precisión del 78% de media de los tres conjuntos, por lo que los resultados son satisfactorios.
- Un elevado número de FP en globular y un elevado número de FN en reticular, probablemente debido al desbalanceo de casos.
- Una sensibilidad del 72% y una especificidad del 85%.
- Una tasa de acierto del 86% de media en los tres conjuntos.
- El rendimiento global de la red en cuanto a pérdidas es 24% y tasa de acierto es de 91%.

El tiempo de ejecución de nuestro trabajo ha sido de 12min 50seg.

5.2. Pruebas con reducción de casos en patrón globular

A continuación, se muestran las matrices de confusión obtenidas (Tabla 5-5), los parámetros de entrenamiento usados (Tabla 5-6), las parámetros de rendimiento obtenidos para 50 y 60 epochs respectivamente (Tabla 5-7) y (Tabla 5-9), las métricas calculadas para 50 y 60 epochs (Tabla 5-8) y (Tabla 5-10) y por último el rendimiento global de la red (Figura 5-2) y (Figura 5-3) una vez realizadas las pruebas.

	50 epochs			60 epochs			70 epochs			80 epochs		
Reducción de casos patrón globular	31	2	11	32	3	9	34	3	7	39	1	4
	1	22	4	1	24	2	4	20	3	5	19	3
	4	1	23	7	2	19	8	2	18	9	1	18

Tabla 5-5. Matrices de confusión reducción casos globular.

Parámetros iniciales para el entrenamiento definitivo de la red:

Parámetro	Valor
Dimensiones de entrada	60x60x3
Tamaño del conjunto de datos	Reducción de casos en patrón globular y patrones homogéneo y reticulado inciciales
Epochs	50 y 60
Tipo de optimizador	Adam

Tabla 5-6. Parámetros de entrenamiento prueba 2.

Hacemos los cálculos de las medidas de rendimiento para la matriz de confusión de 50 y 60 epochs:

1. Cálculos usando 50 epochs:

Parámetros	**Globular**	**Homogéneo**	**Reticulado**
Verdadero Positivo	31	22	23
Falso Positivo	12	5	5
Verdadero Negativo	50	69	56
Falso Negativo	5	3	15

Tabla 5-7. Parámetros reducción casos globular 50 epochs.

Métricas	**Globular**	**Homogéneo**	**Reticulado**	**Total**
Sensibilidad	0.86	0.88	0.61	0.78
Especificidad	0.81	0.93	0.92	0.89
Accuracy	0.83	0.91	0.79	0.84
Precisión o PPV	0.72	0.81	0.82	0.78
NPV	0.90	0.96	0.8	0.89

Tabla 5-8. Métricas reducción casos globular 50 epochs.

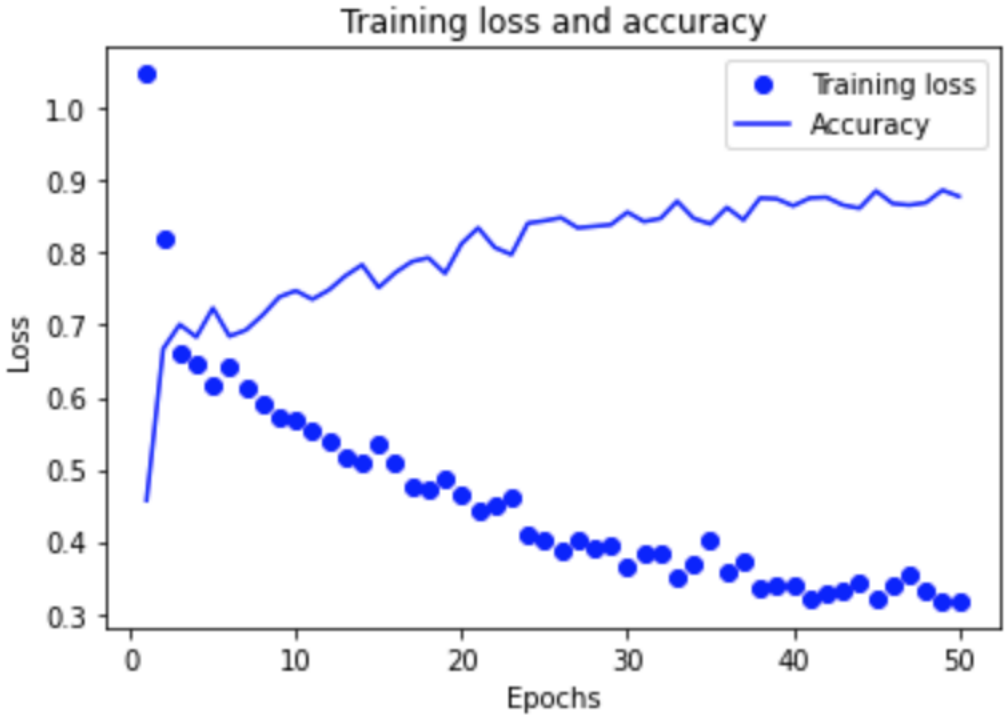

Figura 5-2. Training loss y accuracy prueba 2 con 50 epochs.

Se observa lo siguiente:

- Una precisión del 78% de media de los tres conjuntos, por lo que los resultados son satisfactorios.
- Un elevado número de FP en globular y un elevado número de FN en reticular.
- Una sensibilidad del 78% y una especificidad del 89%.
- Una tasa de acierto del 84% de media en los tres conjuntos.
- Con respecto al caso mencionado en el epígrafe "5.1 Pruebas sin aumento de casos" vemos que la tasa de acierto disminuye levemente en un 2%, mientras que la precisión se mantiene constante, pero la sensibilidad y especificidad sí que aumentan en un 6% y 4% respectivamente.
- El redimiento global de la red vemos que es satisfactorio en cuanto a las pérdidas que tenemos un 32% y la tasa de acierto del 87%.

2. Cálculos usando 60 epochs:

Parámetros	Globular	Homogéneo	Reticulado
Verdadero Positivo	32	24	19
Falso Positivo	12	3	9
Verdadero Negativo	47	67	60
Falso Negativo	8	5	11

Tabla 5-9. Parámetros reducción casos globular 60 epochs.

Métricas	Globular	Homogéneo	Reticulado	Total
Sensibilidad	0.8	0.83	0.63	0.75
Especificidad	0.8	0.96	0.87	0.88
Accuracy	0.8	0.92	0.8	0.84
Precisión o PPV	0.73	0.88	0.68	0.76
NPV	0.86	0.93	0.85	0.88

Tabla 5-10. Métricas reducción casos globular 60 epochs.

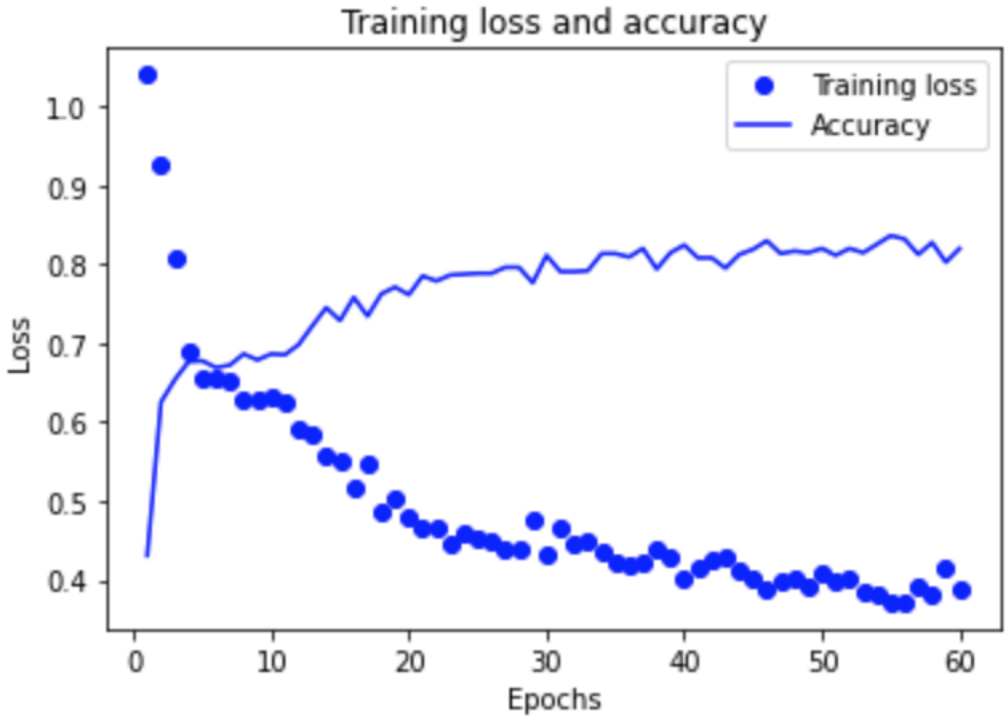

Figura 5-3. Training loss y accuracy prueba 2 con 60 epochs.

Se observa lo siguiente:

- Una precisión del 76% de media de los tres conjuntos, por lo que los resultados son satisfactorios.
- Un elevado número y similar de FP y FN en globular y en reticulado, probablemente sea debido a que los casos están más balanceados.
- Una sensibilidad del 75% y una especificidad del 88%.
- Una tasa de acierto del 84% de media en los tres conjuntos.
- Con respecto al caso mencionado en el epígrafe “5.1 Pruebas sin aumento de casos” vemos que la tasa de acierto y la precisión disminuyen levemente en un 2%.
- El redimiento global de la red vemos que es satisfactorio en cuanto a las pérdidas que tenemos un 39% y la tasa de acierto del 82%. Vemos por tanto que en 50 epochs era mejor.

El tiempo de ejecución ha sido de 7 minutos y 38 segundos.

5.3. Pruebas con aumento de casos en patrón reticular

A continuación, se muestran las matrices de confusión obtenidas (Tabla 5-11), los parámetros de entrenamiento usados (Tabla 5-12), las parámetros de rendimiento obtenidos (Tabla 5-13), las métricas calculadas (Tabla 5-14) y por último el rendimiento global de la red (Figura 5-4) una vez realizadas las pruebas.

	50 epochs			60 epochs			70 epochs			80 epochs		
Aumento de casos patrón reticular	112	3	9	107	7	10	113	9	2	110	5	9
	9	18	0	8	19	0	5	22	0	7	18	2
	23	1	4	18	1	9	22	1	5	16	1	11

Tabla 5-11. Matrices de confusión aumento casos reticular.

Parámetro	Valor
Dimensiones de entrada	60x60x3
Tamaño del conjunto de datos	Aumento de casos patrón reticular y patrones homogéneo y globular iniciales
Epochs	80
Tipo de optimizador	Adam

Tabla 5-12. Parámetros de entrenamiento prueba 3.

Parámetros	Globular	Homogéneo	Reticulado
Verdadero Positivo	110	18	11
Falso Positivo	14	9	17
Verdadero Negativo	32	146	140
Falso Negativo	23	6	11

Tabla 5-13. Parámetros aumento casos reticular.

Métricas	**Globular**	**Homogéneo**	**Reticulado**	**Total**
Sensibilidad	0.83	0.75	0.5	0.69
Especificidad	0.7	0.94	0.89	0.84
Accuracy	0.79	0.92	0.84	0.85
Precisión o PPV	0.89	0.66	0.39	0.65
NPV	0.58	0.96	0.93	0.82

Tabla 5-14. Métricas aumento casos reticular.

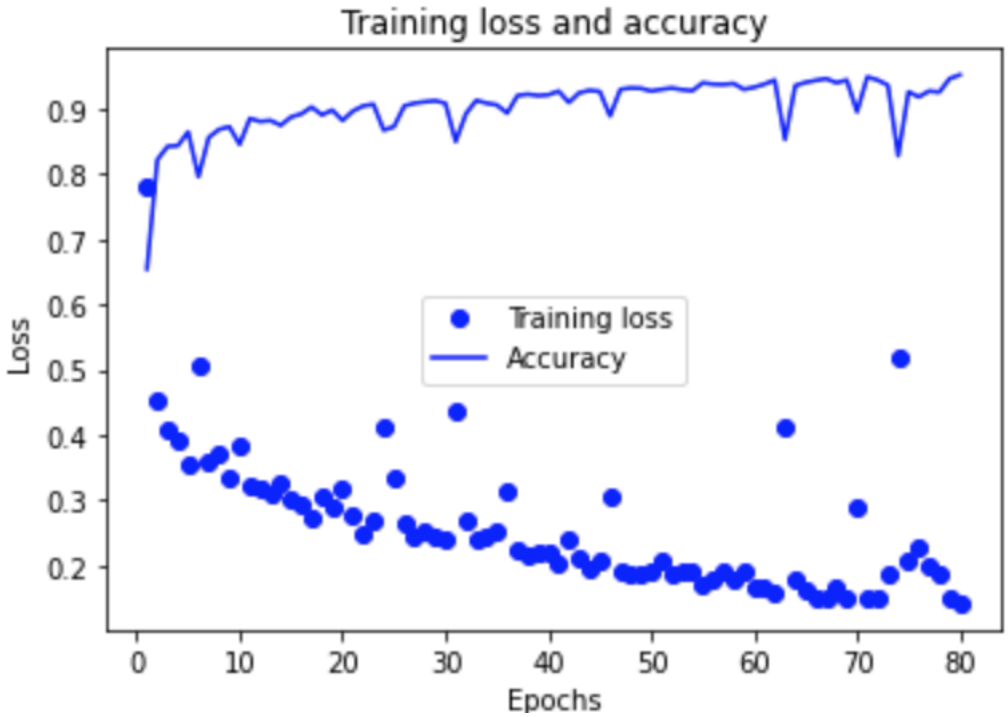

Figura 5-4. Training loss y accuracy prueba 3.

Se observa lo siguiente:

- Una precisión del 65% de media de los tres conjuntos, por lo que los resultados no son tan satisfactorios como en las pruebas anteriores.
- Un elevado número de FP y FN en globular y FP en reticulado, incluso siendo más alto en reticulado, probablemente sea debido a que hemos hecho aumento de datos en reticulado.
- Una sensibilidad del 69% y una especificidad del 84%.
- Una tasa de acierto del 85% de media en los tres conjuntos.
- La precisión disminuye en un 13% con respecto al epígrafe "5.1 Pruebas sin aumento de datos y 5.2 Pruebas con reducción de casos en patrón globular con 50 epochs " y un 11% respecto a lo estudiado en el epígrafe "5.2 Pruebas con reducción de casos en patrón globular con 60 epochs". Mientras que la tasa de acierto apenas disminuye un 1%.
- El redimiento global de la red vemos que es satisfactorio en cuanto a las pérdidas que tenemos un 14% y la tasa de acierto del 95%.

El tiempo de ejecución ha sido de 8 minutos y 16 segundos.

5.4. Pruebas con reducción de casos en patrón globular y aumento en patrón reticular

A continuación, se muestran las matrices de confusión obtenidas (Tabla 5-15), los parámetros de entrenamiento usados (Tabla 5-16), las parámetros de rendimiento obtenidos (Tabla 5-17), las métricas calculadas (Tabla 5-18) y por último el rendimiento global de la red (Figura 5-5) una vez realizadas las pruebas.

	50 epochs			60 epochs			70 epochs			80 epochs		
Reducción de casos en patrón globular y aumento en reticular	39	0	5	37	3	4	41	0	3	35	2	7
	6	21	0	4	23	0	5	20	2	3	22	2
	18	3	7	19	1	8	17	2	9	14	2	12

Tabla 5-15. Matrices de confusión reducción casos globular y aumento casos reticular.

Parámetro	Valor
Dimensiones de entrada	60x60x3
Tamaño del conjunto de datos	Reducción de casos patrón globular, aumento de casos patrón reticular y casos inciales patrón homogéneo
Epochs	80
Tipo de optimizador	Adam

Tabla 5-16. Parámetros de entrenamiento prueba 4.

Parámetros	Globular	Homogéneo	Reticulado
Verdadero Positivo	35	22	12
Falso Positivo	9	5	16
Verdadero Negativo	38	68	62
Falso Negativo	17	4	9

Tabla 5-17. Parámetros reducción casos globular y aumento reticular.

Métricas	**Globular**	**Homogéneo**	**Reticulado**	**Total**
Sensibilidad	0.67	0.85	0.58	0.7
Especificidad	0.8	0.93	0.79	0.84
Accuracy	0.74	0.91	0.75	0.8
Precisión o PPV	0.8	0.81	0.42	0.68
NPV	0.69	0.94	0.87	0.83

Tabla 5-18. Métricas reducción casos globular y aumento reticular.

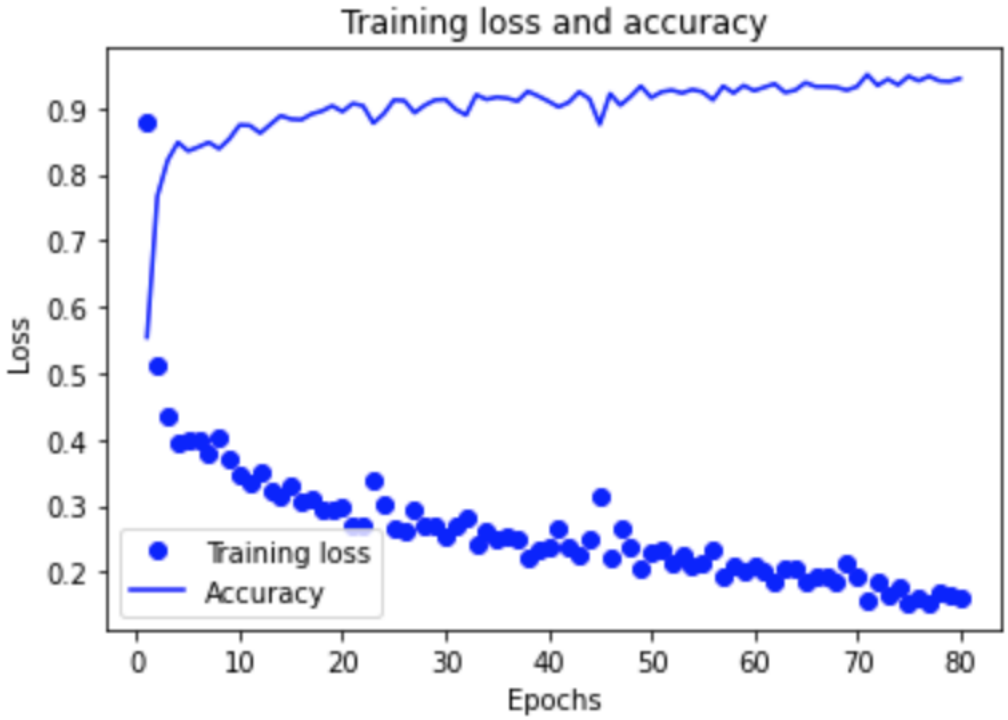

Figura 5-5. Training loss y accuracy prueba 4.

Se observa lo siguiente:

- Una precisión del 68% de media de los tres conjuntos, por lo que los resultados son moderadamente satisfactorios.
- Un elevado número de FN en globular y de FP en reticulado, probablemente sea debido a que los casos están más balanceados.
- Una tasa de acierto del 80% de media en los tres conjuntos.
- La precisión disminuye en un 10% con respecto a los epígrafes "5.1 Pruebas sin aumento de datos y 5.2 Pruebas con reducción de casos en patrón globular 50 epochs", un 8% respecto a lo estudiado en el epígrafe "5.2 Pruebas con reducción de casos en patrón globular 60 epochs" y aumenta un 3% respecto a lo estudiado en el epígrafe "5.3 Pruebas con reducción de casos en patrón globular".
- La tasa de acierto disminuye en un 6% con respecto al epígrafe "5.1 Pruebas sin aumento de datos", un 4% respecto a lo estudiado en el epígrafe "5.2 Pruebas con reducción de casos en patrón globular" y un 5% respecto a lo estudiado en el epígrafe "5.3 Pruebas con reducción de casos en patrón globular".
- El redimiento global de la red vemos que es satisfactorio en cuanto a las pérdidas que tenemos un 16% y la tasa de acierto del 94%.

El tiempo de ejecución ha sido de 9 minutos y 12 segundos.

5.5. Interpretación final de los resultados

Los resultados son muy parecidos en cuanto a tasa de acierto y precisión si el modelo trabaja con 50 epochs en las pruebas sin aumento de casos y las pruebas con reducción de casos en patrón globular, estudiadas en los epígrafes 5.1 y 5.2. La diferencia la encontramos en sensibilidad y especificidad, que es mejor el modelo reducido. Esto es debido a que los casos están más balanceados, en la base de datos original estudiada en el epígrafe 5.1 tenemos un 67% de los casos en globular, un 18.6% en homogéneo y un 14.2% en reticulado, mientras que en el 5.2 tenemos un 40.6% de los casos en globular, un 33.7% en homogéneo y un 25.7% en reticulado.

En cuanto al epígrafe “5.3 Pruebas con aumento de casos en patrón reticular” vemos que no se obtienen unos valores tan buenos como en los dos casos anteriores, aunque siguen siendo bastante buenos.

En el epígrafe “5.4 Pruebas con reducción de casos en patrón globular y aumento en reticular” vemos que no se obtienen unos valores tan buenos como en los casos anteriores, aunque es la prueba más balanceada de todas. Esto puede ser debido a que el aumento de datos no es satisfactorio en el caso de nuestros patrones.

Finalmente, concluimos que lo más favorable para nuestro estudio es hacer el entrenamiento sin aumento de datos, pero reduciendo las imágenes del tipo globular, es decir, la prueba realizada en el epígrafe “5.2 Pruebas con reducción de casos en patrón globular” debido a que tenemos un entrenamiento más balanceado y por tanto las métricas de nuestra prueba dan resultados más favorables.

6 Conclusiones

El aprendizaje profundo o Deep Learning ya es utilizado en la mayoría de los casos para el estudio de imágenes médicas.

El trabajo realizado tiene como objetivo el estudio en profundidad de la ingeniería biomédica y su capacidad para ayudar a prevenir enfermedades. En el caso estudiado, el melanoma, se ha desarrollado un procedimiento que no pretende brindar un diagnóstico completo de la enfermedad, pero sí pretende dar un paso más en ella.

Incluye la implementación de un programa de procesamiento de imágenes que estima ciertas características de la textura y las clasifica en los tres patrones globales más importantes del análisis de patrones [3], que es el método más citado y utilizado para detectar el melanoma. Hemos podido estudiar los métodos de selección de características, clasificación y verificación de los resultados.

Premisas:

- El melanoma es el tipo de cáncer de piel más agresivo y representa un alto porcentaje de las muertes por cáncer de piel.
- Éste es un tema importante porque el número de personas afectadas por esta enfermedad aumenta aproximadamente un 10% cada año.
- Por ello, la detección precoz es fundamental, para hacerlo realidad existen técnicas como la dermatoscopia, además de ayudar en este sentido, también reduce considerablemente el número de intervenciones en el campo de la dermatología.

Las conclusiones extraídas son las siguientes:

- Hay pocas formas de detectar y clasificar patrones globales en la literatura.
- El diseño propuesto es un sistema autónomo debido a que los parámetros utilizados por todas las imágenes son fijos, por lo que no dependen de la imagen de entrada.
- Vemos que en este trabajo de fin de grado se ha hecho un estudio de Deep Learning y se ha aplicado a la idea de una mejor clasificación de las imágenes con los tres tipos de patrones: globular, homogéneo y reticulado.

Finalmente, a la vista de los resultados, cabe señalar que el diseño cumplió con su función diseñada: detectar patrones para ayudar en el campo médico.

Como líneas futuras, además de esta funcionalidad elegida en este proyecto, hay posibilidad de añadir un gran número de mejoras.

La principal es balancear los datos con imágenes tomadas de una base de datos más amplia para que en homogéneo y en reticulado tengamos más o menos las mismas imágenes que en el patrón globular, para que nuestro estudio aumentara en cuanto a los valores de precisión.

Nuestro sistema está centrado en la detección de tres patrones, pero la red neuronal podría ser entrenada para detectar un mayor número de patrones. Para esto sólo deberíamos de entrenar la red con un mayor número de imágenes etiquetadas con patrones de estos tipos. Además, podríamos dotarle de la capacidad de detectar también si la lesión es melanocítica o no, pues estos factores son importantes a tener en cuenta para el estudio de una lesión pigmentada.

7 Referencias

[1] AECC, “Melanoma maligno en la piel: lo que necesitas saber”, 2020. [En línea] Disponible: https://www.aecc.es/es/todo-sobre-cancer/tipos-cancer/cancer-piel/melanoma

[2] Argenziano G, Soyer H.P, Chimenti S, Talamini R, Corona R, Sera F, Binder M, Cerroni L, Kopf A.W.,”Dermoscopy of pigmented skin lesions: Results of a consensus meeting via the Internet” , J. Am. Acad. Dermatol., vol. 48, no 5, pp. 680, 2003.

[3] Zaballos Pedro, Carrera Cristina, Puig Susana, Malvehy Josep, “Dermoscopic Criteria in the Diagnosis of Melanoma”, Colegio Ibero-Latino-Americano de Dermatología., pp. 7, 2004.

[4] Pehamberger H, Steiner A, Wolff K.,”in vivo epiluminescence microscopy of pigmented skin lesions. I. Pattern analysis of pigmented skin lesions.,” American Academy of Dermatology, vol. 17, no 4, pp. 571- 583, 1987.

[5] Stolz W, Riemann A, Cognetta AB, Pillet L, Abmayr W, Hoelzel D, “ABCD rule of dermatoscopy: a new practical method for early recognition of malignant melanoma,” Eur J Dermatol, Vols. %1 de %24:521-7, 1994.

[6] Menzies, S.W., Ingvar, C., Crotty, K.A., McCarthy, WH., “Frequency and morphologic characteristics of invasive melanomas lacking specific surface microscopic features,” Arch. Dermatol. , Vols. %1 de %2132: 1178-82, 1996.

[7] Argenziano,G.,Fabbrocini,G.,Carli,P.,DeGiorgi,V.,Sammarco,E.,Delfino,M., “Epiluminescence microscopy for the diagnosis of doubtful melanocytic skin lesions. Comparison of the ABCD rule of dermatoscopy and a new 7-point checklist based on pattern analysis,” Arch. Dermatol., Vols. %1 de %2134:1563-70, 1998.

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li y Li FeiFei, "Imagenet: A large-scale hierarchical image database", Computer Vision and Pattern Recognition 2009. CVPR 2009. IEEE Conference on , pp. 248-255, 2009.

[9] Yann LeCun, Yoshua Bengio y Geoffrey Hinton, "Deep Learning", naturaleza , vol. 521, no. 7553, págs.436, 2015.

[10] Alex Krizhevsky, Ilya Sutskever y Geoffrey E. Hinton, "Imagenet Classification with Deep Convolutional Neural Networks", Avances en los sistemas de procesamiento de información neuronal , págs. 1097-1105, 2012.

[11] Matthew D. Zeiler y Rob Fergus, "Visualize and understand convolutional networks", conferencia europea sobre visión por computadora , págs. 818-833, 2014.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren y Jian Sun, "Going deep into rectifiers: outperforming human-level Image sorting", Actas de la conferencia internacional IEEE sobre visión por computadora , págs. 1026-1034, 2015.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren y Jian Sun, "Deep residual learning for image recognition", Actas de la conferencia IEEE sobre visión por computadora y reconocimiento de patrones , págs. 770-778, 2016.

[14] MB Lens y M. Dawes, "Global perspectives on contemporary epidemiological trends in cutaneous malignant melanoma", British Journal of Dermatology , vol. 150, no. 2, págs. 179-185, 2004.

[15] Cristina Nader Vasconcelos y Bárbara Nader Vasconcelos, "Comités de redes neuronales convolucionales para la clasificación del melanoma con aumento de datos de transformaciones de imágenes clásicas y de conocimiento experto", preprint arXiv arXiv: 1702.07025 , 2017.

[16] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato y Lior Wolf, "Deepface: Closing the gap to human-level performance in facial verification", Actas de la conferencia

IEEE sobre visión por computadora y reconocimiento de patrones , págs. 1701-1708, 2014.

[17] W. Zhang, K. Doi, ML Giger, Y. Wu, RM Nishikawa y RA Schmidt, “Computerized detection of pooled microcalcifications on digital mammograms using a displacement invariant artificial neural network.”

[18] Isasi A Gola, Zapirain BG, Zorrilla AM. “Melanomas non-invasive diagnosis application based on the ABCD rule and pattern recognition image processing algorithms.” Comput Biol Med. 2011 Sep;41(9):742-55.

[19] H. Mahmoud, M. Abdel-Nasser y OA Omer, "Computer-aided diagnostic system for the detection of skin lesions using texture analysis methods ", Conferencia Internacional 2018 sobre Tendencias Innovadoras en Ingeniería Informática (ITCE) , 2018, págs. 140-144, doi : 10.1109 / ITCE.2018.8327948.

[20] Sáez Manzano, Aurora, Serrano Gotarredona, Maria Carmen, Acha Piñero, Begoña: “Model-based Classification Methods of Global patterns in dermoscopic images”. En: IEEE Transactions on Medical Imaging. 2014. Vol. 33. Núm. 5. Pag. 1137-1147. 10.1109/Tmi.2014.2305769

[21] Serrano Gotarredona, Maria Carmen, Acha Piñero, Begoña: “Pattern Analysis of Dermoscopic Images Based on Fscm Color Markov Random Fields”. En: Lecture Notes in Computer Science. 2009. Vol. 5807/2009. Pag. 676-685

[22] A.Namozov, D. Ergashev y YI Cho, "Adaptative Activation Functions for Classification of Skin Lesions Using Deep Neural Networks", 10a Conferencia internacional conjunta sobre Soft Computing y sistemas inteligentes (SCIS) de 2018 y XIX Simposio internacional sobre sistemas inteligentes avanzados (ISIS) , 2018, págs.232-235, doi: 10.1109 / SCIS-ISIS.2018.00048.

[23] Alex Krizhevsky, Ilya Sutskever y Geoffrey E. Hinton, "Imagenet Classification with Deep Convolutional Neural Networks", Avances en los sistemas de procesamiento de información neuronal , págs. 1097-1105, 2012.

[24] Matthew D. Zeiler y Rob Fergus, "Visualize and understand convolutional networks", conferencia europea sobre visión por computadora , págs. 818-833, 2014.

[25] DC Cires¸an, A. Giusti, LM Gambardella y J. Schmidhuber, "Detection of mitosis in breast cáncer histology images with Deep neural networks", Computación de imágenes médicas e intervención asistida por computadora - MICCAI 2013 , págs. 411-418, 2013.

[26] Sadeghi Maryam, Lee Timm K. , McLean David, Lui Harvey, y Atkins M. Stella “Analysis of the global pattern and classification of images using dermoscopic textons”, Proc. SPIE 8314, Medical Imaging 2012.

[27] Corrales Jose Carlos, “Deep Learning en ciberseguridad: la herramienta definitiva” [En línea] Disponible:https://www.revelock.com/es/blog/deep-learning-en-ciberseguridad-la-herramienta-definitiva

[28] Agudelo Marcela, Moleón-Getino Antonio, “El impacto del Big-Data en la sociedad de la información. Significado y utilidad”, Corporación universitaria minuto de Dios, Bogotá, Vol. 20, Núm. 2, (2015) pp 427.

[29] Chollet Francois, “Deep Learning with Python”.

[30] Código Fuente, “Redes neuronales profundas”[En línea] Disponible: https://www.codigofuente.org/redes-neuronales-profundas-tipos-caracteristicas/

[31] Term 3 Udacity Self-Driving Car Engineer Nanodegree. “Semantic segmentation”, October 2017.

[32] Larranaga Pedro, Inza Iñaki, Abdelmalik Moujahid, “Redes Neuronales” [En línea] Disponible: http://www.sc.ehu.es/ccwbayes/docencia/mmcc/docs/t8neuronales.pdf, Departamento de Ciencias de la Computación e Inteligencia Artificial, Universidad del País Vasco.

[33] Wikipedia, ”Red neuronal artificial”, [En línea] Disponible: https://es.wikipedia.org/wiki/Perceptr%C3%B3n_multicapa#/media/Archivo:RedNeuronalArtificial.png
[34] Villanueva García Jose David, “Redes Neuronales desde cero”, [En línea] Disponible:https://www.iartificial.net/redes-neuronales-desde-cero-i-introduccion/
[35] Rodríguez Daniel, ”¿cuál es la diferencia entre parámetro e hiperparámetro?” [En línea] Disponible:https://www.analyticslane.com/2019/12/16/cual-es-la-diferencia-entre-parametro-e-hiperparametro/
[36] Durán Jaime, “Técnicas de regularización básicas para redes neuronales”, [En línea] Disponible: https://medium.com/metadatos/t%C3%A9cnicas-de-regularizaci%C3%B3n-b%C3%A1sicas-para-redes-neuronales-b48f396924d4
[37] Orellana Rueda, Pedro. “Segmentación semántica y reconocimiento de lugares usando características CNN pre-entrenadas” [En línea]. Santiago, Chile: Universidad de Chile - Facultad de Ciencias Físicas y Matemáticas, 2019 [Fecha consulta: 12 de agosto 2021]. Disponible en “http://repositorio.uchile.cl/handle/2250/173733”
[38] Durán Jaime, “Qué es la transferencia de aprendizaje y cómo aplicarla a tu red neuronal”[En línea] Disponible:https://medium.com/metadatos/qu%C3%A9-es-la-transferencia-de-aprendizaje-y-c%C3%B3mo-aplicarla-a-tu-red-neuronal-e0e120156e40
[39] Challenger-Pérez Ivet, Díaz-Ricardo Yanet, Becerra-García Roberto Antonio, ”El lenguaje de programación Python” Ciencias Holguín, vol. XX, núm. 2, abril-junio, 2014, pp. 1-13.
[40] Chollet Francois, Keras, [En línea] Disponible: https://keras.io/guides/sequential_model/
[41] Barrios Arce Juan Ignacio en Big Data, Ciencia de datos, Informática Médica, Inteligencia Artificial, Machine Learning, Julio 2019.

Índice de Figuras

Figura 1-1. Incidencia de los rayos del sol en la piel.
Figura 1-2. Patrones globales.
Figura 1-3. Regla ABCD.
Figura 2-1. Arquitectura de la CNN profunda para la clasificación de enfermedades de lesiones cutáneas. La red consta de nueve capas.
Figura 3-1. IA, ML,DL.
Figura 3-2. Esquema de modelo de aprendizaje automático.
Figura 3-3. Red Neuronal Convolucional.
Figura 3-4. Capas en una Red Neuronal Artificial. [33]
Figura 3-5. Procesado de información en una neurona.
Figura 3-6. Función escalón.
Figura 3-7. Función sigmoide.
Figura 3-8. Función ReLu.
Figura 3-9. Función tangente hiperbólica.
Figura 3-10. Aumento de datos.
Figura 4-1. Importaciones para el uso de Keras.
Figura 4-2. Creación del modelo secuencial.
Figura 4-3. Creación de capas.
Figura 4-4. Impresión del modelo.
Figura 4-5. Ejecución del modelo.
Figura 4-6. Acceso a las imágenes de train.
Figura 4-7. Crear la matriz de datos train.
Figura 4-8. Recorrer la matriz de train.
Figura 4-9. Conversión y normalización de datos train.
Figura 4-10. Acceso a las imágenes de test.
Figura 4-11. Crear la matriz de datos test.
Figura 4-12. Recorrer la matriz de test.
Figura 4-13. Conversión y normalización de datos test.
Figura 4-14. Compilación del modelo.
Figura 4-15. Redimensionamiento y cambio de formato de las imágenes.
Figura 4-16. Data Augmentation.
Figura 4-17. Matriz de confusión binaria.
Figura 5-1. Training loss y accuracy prueba 1.
Figura 5-2. Training loss y accuracy prueba 2 con 50 epochs.
Figura 5-3. Training loss y accuracy prueba 2 con 60 epochs.
Figura 5-4. Training loss y accuracy prueba 3.
Figura 5-5. Training loss y accuracy prueba 4.

Índice de Tablas

Tabla 4-1. Base de datos inicial.
Tabla 4-2. Tipos de patrones.
Tabla 4-3. Comparación Python y C++.
Tabla 4-4. Cálculo de métricas en matriz de confusión.
Tabla 4-5. Balanceo de casos.
Tabla 4-6. Train y test de pruebas propuestas.
Tabla 5-1. Matrices de confusión sin aumento de casos.
Tabla 5-2. Parámetros entrenamiento prueba 1.
Tabla 5-3. Parámetros sin aumento de casos.
Tabla 5-4. Métricas sin aumento de casos.
Tabla 5-5. Matrices de confusión reducción casos globular.
Tabla 5-6. Parámetros de entrenamiento prueba 2.
Tabla 5-7. Parámetros reducción casos globular 50 epochs.
Tabla 5-8. Métricas reducción casos globular 50 epochs.
Tabla 5-9. Parámetros reducción casos globular 60 epochs.
Tabla 5-10. Métricas reducción casos globular 60 epochs.
Tabla 5-11. Matrices de confusión aumento casos reticular.
Tabla 5-12. Parámetros de entrenamiento prueba 3.
Tabla 5-13. Parámetros aumento casos reticular.
Tabla 5-14. Métricas aumento casos reticular.
Tabla 5-15. Matrices de confusión reducción casos globular y aumento casos reticular.
Tabla 5-16. Parámetros de entrenamiento prueba 4.
Tabla 5-17. Parámetros reducción casos globular y aumento reticular.
Tabla 5-18. Métricas reducción casos globular y aumento reticular.