UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

MÁSTER EN INGENIERÍA INFORMÁTICA

APLICACIÓN DE TÉCNICAS DE SUPER-RESOLUCIÓN Y DATA AUGMENTATION PARA MEJORAR LA DETECCIÓN DEL CÁNCER DE PIEL

APPLICATION OF SUPER-RESOLUTION AND DATA AUGMENTATION TECHNIQUES TO IMPROVE SKIN CANCER DETECTION

Realizado por
Nicolás Quintana Marín

Tutorizado por
Enrique Domínguez Merino

Departamento
LENGUAJES Y CIENCIAS DE LA COMPUTACIÓN
UNIVERSIDAD DE MÁLAGA

MÁLAGA, JUNIO DE 2024

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA
MÁSTER EN INGENIERÍA INFORMÁTICA

Aplicación de técnicas de super-resolución y data augmentation para mejorar la detección del cáncer de piel

Application of super-resolution and data augmentation techniques to improve skin cancer detection

Realizado por
Nicolás Quintana Marín

Tutorizado por
Enrique Domínguez Merino

Departamento
Lenguajes y Ciencias de la Computación

UNIVERSIDAD DE MÁLAGA

MÁLAGA, Junio de 2024

Resumen

El cáncer de piel supone un grave problema de salud pública, ya que se trata de uno de los más comunes y, además, su incidencia no para de crecer. De todos los tipos posibles el más mortal es el melanoma, el cual si es detectado de forma precoz, tiene una gran probabilidad de supervivencia.

Esa detección se suele llevar a cabo mediante dermatoscopias, exploraciones no invasivas del tumor que dependen en gran medida de la experiencia del profesional que la lleve a cabo, por lo que es posible apoyarse en sistemas CAD, que hacen uso de algoritmos inteligentes para ayudar al profesional a tomar una decisión.

Este Trabajo Fin de Máster (TFM) tiene como objetivo intentar solventar dos problemas comunes a la hora de entrenar estos algoritmos inteligentes, la baja resolución que ofrecen algunos dermatoscopios y la falta de conjuntos de datos sobre los que entrenar. Para ello, se propone una revisión del estado del arte donde se exploran distintos métodos de súper-resolución y generación de imágenes sintéticas, así como conjuntos de datos que puedan ser utilizados para esas tareas, seguido de un estudio que muestre los resultados de aplicar dichas técnicas a la detección del melanoma mediante algoritmos inteligentes.

En este estudio, además, se documenta la evaluación de las distintas técnicas estudiadas usando métricas adecuadas para medir su rendimiento bajo los distintos conjuntos de datos seleccionados, de forma que ESRGAN y StyleGAN han demostrado ser superiores al resto de los modelos en el estado del arte. Tras aplicar las técnicas a la clasificación a través de ResNet se ha logrado obtener un 12 % más de aciertos al añadir imágenes sintéticas a un conjunto de datos pequeños y un 3 % aplicando súper-resolución a imágenes de un tamaño de 32x32.

Palabras claves: Melanoma, GAN, Super-resolución, Imágenes sintéticas, IA Generativa.

Abstract

Skin cancer is a serious public health problem, as it is one of the most common cancers and its incidence is constantly increasing. Of all the possible types, the most deadly is melanoma, which, if detected early, has a high probability of survival.

This detection is usually carried out by means of dermatoscopies, non-invasive explorations of the tumor that depend to a great extent on the experience of the professional, so it is possible to rely on CAD, which makes use of intelligent algorithms to help the professional to make a decision.

This Master's Thesis aims to try to solve two common problems when training these intelligent algorithms, the low resolution offered by some dermatoscopes and the lack of datasets on which to train. To this end, a review of the state of the art is proposed where different methods of super-resolution and synthetic image generation are explored, as well as datasets that can be used for these tasks, followed by a study showing the results of applying these techniques to melanoma detection using intelligent algorithms.

This study also documents the evaluation of the different techniques studied using appropriate metrics to measure their performance under the different datasets selected, so that ESRGAN and StyleGAN have been shown to be superior to the rest of the models in the state of the art. After applying the techniques to classification through ResNet, 12 % more hits have been obtained by adding synthetic images to a small dataset and 3 % more hits by applying super-resolution to 32x32 images.

Keywords: Melanoma, GAN, Super-resolution, Synthetic images, Generative AI.

Índice de contenidos

1. Introducción **1**
1.1. Motivación y descripción del problema . . . 1
1.1.1. El cáncer de piel . . . 1
1.1.2. Problemas en la detección del melanoma . . . 2
1.1.3. Auge de la Inteligencia Artificial Generativa . . . 3
1.2. Objetivos . . . 4
1.3. Estructura de la memoria . . . 5

2. Marco teórico **7**
2.1. Inteligencia Artificial y Machine Learning . . . 7
2.1.1. Árboles de decisión . . . 8
2.1.2. Máquinas de vectores de soporte . . . 9
2.1.3. K-Nearest Neighbors . . . 10
2.2. Aprendizaje Profundo . . . 11
2.2.1. Neurona artificial y perceptrón multicapa . . . 11
2.2.2. Entrenamiento de la red . . . 13
2.3. Redes neuronales convolucionales . . . 16
2.3.1. Arquitectura y funcionamiento . . . 16
2.3.2. Transfer Learning . . . 18
2.4. Redes Generativas Antagónicas . . . 21
2.4.1. Generador y Discriminador . . . 21
2.4.2. Aprendizaje en las GANs . . . 22
2.4.3. cGAN . . . 24
2.5. Súper-resolución . . . 25
2.5.1. Métodos basados en interpolación . . . 25
2.5.2. Métodos basados en aprendizaje profundo . . . 26

3. Materiales y metodología **29**
3.1. Metodología y fases de trabajo . . . 29
3.2. Recursos hardware . . . 31
3.3. Conjuntos de datos . . . 32
3.3.1. HAM10000 . . . 32
3.3.2. BCN20000 . . . 36
3.3.3. Conjunto de 10000 imágenes de melanoma . . . 37
3.4. Modelos de súper-resolución . . . 38
3.4.1. EDSR . . . 38
3.4.2. SRGAN . . . 39

3.4.3. ESRGAN 40
3.5. Modelos de generación de datos sintéticos 42
3.5.1. DCGAN 42
3.5.2. StyleGAN 43

4. Resultados experimentales **47**
4.1. Evaluación de la súper-resolución 47
4.1.1. Métricas 47
4.1.2. Proceso de evaluación y resultados 48
4.2. Evaluación de la generación de datos sintéticos 55
4.2.1. Métricas 55
4.2.2. Proceso de evaluación y resultados 56
4.3. Evaluación del impacto en la clasificación 63
4.3.1. Métricas 63
4.3.2. Proceso de evaluación y resultados 64

5. Conclusiones y lineas de trabajo futuras **75**
5.1. Conclusiones 75
5.2. Lineas de trabajo futuras 76

Bibliografía **79**

Apéndice A. Imágenes usadas en la memoria **85**

CAPÍTULO 1

Introducción

En esta primera sección, se llevará a cabo un planteamiento de la motivación y descripción del problema que se pretende resolver con el presente trabajo, los objetivos planteados en el mismo y la organización y estructura de este documento, donde se definirá brevemente las distintas partes de las que se compone.

1.1. Motivación y descripción del problema

1.1.1. El cáncer de piel

El cáncer de piel supone un problema de salud de alta prioridad a nivel global ya que, durante muchos años, ha sido, de forma simultánea, el tipo de cáncer más común y de los más mortíferos en todo el mundo, llegando a estimarse en Estados Unidos más de 1,6 millones de casos nuevos y 12190 muertes tan solo en 2012 [1].

De entre todos los tipos de cáncer de piel, destaca en especial uno, el melanoma, ya que a pesar de su baja incidencia, siendo este tan solo el 4 % de los casos, representa el 65 % de las muertes derivadas del cáncer de piel, convirtiéndose en la forma más mortífera conocida de este tipo de cáncer.

Lo curioso de este cáncer es que es fácilmente tratable en las fases tempranas de la enfermedad, donde con una simple cirugía de extracción, se obtienen unos índices de supervivencia muy altos, sin embargo, una vez se produce la metástasis, estos índices se desploman y, tristemente, la supervivencia empieza a ser lo menos común [2], a pesar de ello, gracias a terapias recientes aplicables a melanomas metastásicos como BRAF, CTLA4 y PD1, se ha logrado reducir esos índices, sin embargo, debido a la naturaleza de la enfermedad, se ha comprobado que en aproximadamente la mitad de los casos, se desarrolla una resistencia a dicho tratamiento en poco tiempo, por lo que la detección temprana resulta algo vital.

En cuanto a la evolución de la incidencia de la enfermedad, tal y como se puede observar en la figura 1.1, a pesar de que tal y como hemos comentado, la mortalidad del mismo se ha logrado reducir, su incidencia parece seguir una tendencia a la alza, en ambos sexos, destacando sobre todo en el caso de los hombres [3], por lo que no parece que sea una enfermedad que vaya a desaparecer o reducirse en el corto o medio plazo.

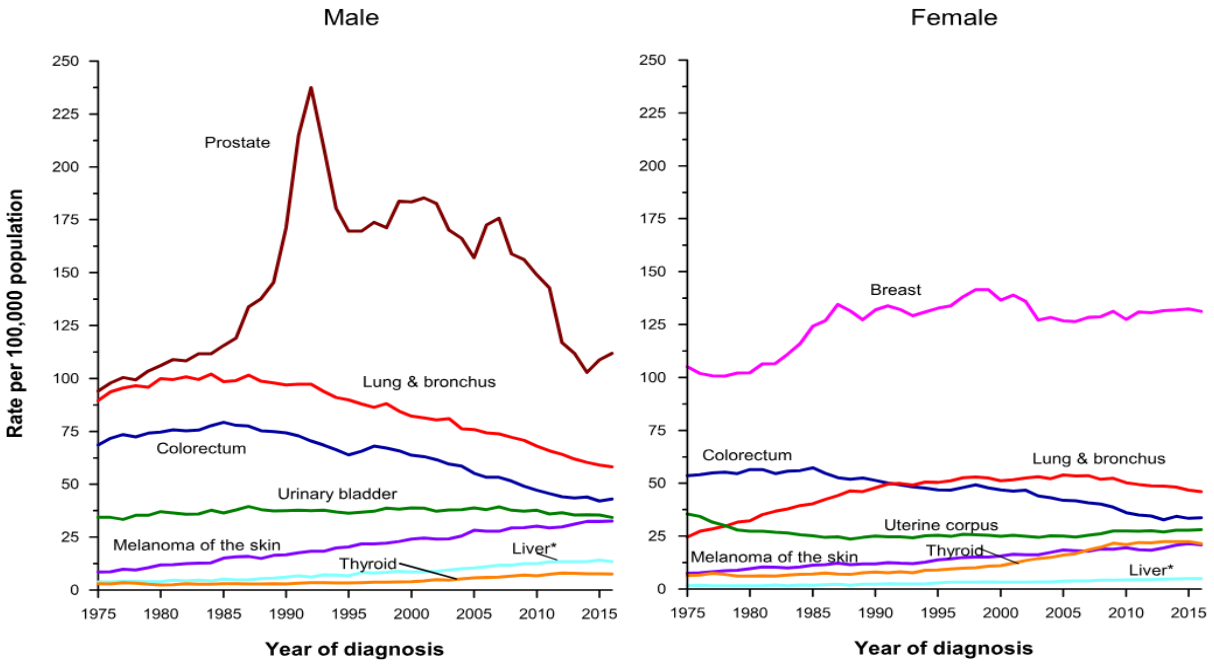


Figura 1.1: Evolución en la incidencia de los tipos de cáncer más comunes. [3]

Por tanto, si unimos el hecho de que se trata de un cáncer en el que la detección temprana es algo crucial para la supervivencia del paciente, con que cada vez la incidencia mayor, es evidente que se deben invertir recursos en la investigación para crear nuevos métodos de detección o mejorar los ya existentes mediante nuevas técnicas.

1.1.2. Problemas en la detección del melanoma

Existen diversas técnicas adoptadas por los profesionales para la detección y diagnosis del melanoma como, por ejemplo, la presencia de ciertos biomarcadores [2], pero una de las más comunes debido a su bajo coste y a la poca invasión en el paciente es la dermatoscopia [4].

Esta técnica consiste en usar una herramienta, el dermatoscopio, para iluminar y ampliar la visión de la zona de interés, de forma que el especialista pueda tener una referencia con mucho más detalle sobre la que realizar su análisis. Una vez se tiene dicha referencia, existen diferentes reglas que los especialistas aplican:

- **Regla ABCDE**: se analiza la asimetría (A), la irregularidad de los bordes (B), la heterogeneidad del color (C), el diámetro (D) y la evolución de todo lo anterior a lo largo del tiempo (E).
- **Lista de control de 7 puntos**: se comprueban las 7 características básicas del melanoma y se les asigna una puntuación a cada una.
- **Método Menzies**: se comprueban si están presentes o ausentes 11 características concretas.

Como se puede comprobar, son técnicas meramente subjetivas, pues no se basan en la existencia de unos indicadores claro, al final es una tarea compleja y que depende, en gran parte, de la experiencia del profesional que la realice, de forma que los más novatos pueden tender a dar diagnósticos menos certeros.

Es por ello por lo que el desarrollo de sistemas de diagnóstico asistido por ordenador (CAD) de calidad resulta imprescindible para ayudar a los profesionales a tomar una decisión sobre si es necesario realizar una biopsia y, de la misma forma, permitirles atender a una mayor cantidad de pacientes, pudiendo mitigar así la tendencia alcista de esta enfermedad [5].

La creación de estos sistemas se basa, principalmente, en algoritmos inteligentes, que han de ser entrenados con grandes y diversos conjuntos de datos, sobre todo en este caso, debido a la gran variedad de tipos concretos de melanoma y en la apariencia de los mismos, además de las distintas tonalidades de la piel en las diversas etnias [6], es aquí donde reside uno de los grandes problemas a la hora de implementar un CAD funcional, la falta de grandes conjuntos de datos diversos que permitan representar de forma adecuada el conjunto real de la población. A pesar de que se han realizado grandes esfuerzos para la elaboración de conjuntos de datos públicos, al final los investigadores suelen acabar utilizando conjuntos de datos demasiado pequeños, la mayoría de veces desbalanceados, necesitando recurrir a la aplicación de técnicas de *data augmentation* tradicionales.

Uno de los principales motivos por los que no existen grandes conjuntos de datos de este tipo es la delicadeza de los mismos, según el Reglamento General de Protección de Datos (RGPD) [7] están considerados como categoría especial de datos al estar relacionado con la salud, prohibiendo su tratamiento salvo en el caso el que paciente exprese su consentimiento y se utilice con un fin de interés para la salud pública. Esto es algo crítico, pues la tenencia y asociación de esta información con la de un identificador personal, como puede ser el nombre de una persona, puede suponer una brecha en la privacidad del paciente, derivándose su condición médica e, incluso, información acerca de su origen étnico, algo que podrían utilizar terceras partes para su propio interés.

Por desgracia, las brechas de datos y los ataques cibernéticos son cada vez más comunes y el ámbito de la medicina no se excluye, pero no solo eso, pues también existen diversos ataques que tienen como objetivo esos algoritmos inteligentes, de forma que es posible extraer información sensible de esos modelos [8], por lo que la escasez de datos es lo más común en este ámbito.

Otro de los problemas surge de la propia naturaleza de la técnica, esta reside en la identificación y reconocimiento de ciertos patrones que han de ser totalmente observables en la dermatoscopia, esto no siempre es posible, pues los instrumentos en ocasiones no tienen suficiente capacidad de aumento [9], obteniéndose imágenes con poca resolución y detalle, algo que puede repercutir tanto a la decisión independiente del profesional, como a la del CAD, por lo que mejorar esos aspectos resulta algo crucial para mejorar el diagnóstico.

1.1.3. Auge de la Inteligencia Artificial Generativa

Estamos viviendo una época revolucionaria en lo que a Inteligencia Artificial (IA) se refiere, el surgimiento y la publicación de nuevos modelos generativos, en concreto, las *Generative Adversarial Networks* (GAN), capaces de generar nuevo contenido como imágenes, música y texto con un alto grado de realismo [10], lo que no ha dejado indiferente a nadie, pues amenaza con destruir multitud de puestos de empleo.

Por un lado, estos modelos parecen poder llegar a sustituir a muchos artistas, pues con tan solo una descripción permiten generar contenido en cuestión de segundos, a lo sumo,

minutos. Además, el entrenamiento de estos modelos es cuestionado pues existe preocupación de que se realice con materiales con derechos de autor [11], además de que pueden tener sesgos, problemas de transparencia, de forma que se filtre información confidencial e incluso proveer información errónea.

Pero más allá de estos problemas, que se centran más en el uso "artístico" de estas herramientas, aparecen una multitud de oportunidades, y el campo de la dermatología no se iba a quedar atrás. Con este tipo de modelos es posible generar nuevas imágenes realistas para ampliar conjuntos de datos pequeños, pero no solo eso, algunos investigadores han logrado crear modelos que, a partir de una imagen con piel sana, convertirla en piel con eccemas y viceversa [12], lo que es realmente curioso pues puede permitir generar nuevos conjuntos de datos realmente interesantes.

Además de la generación de datos, también tienen otras utilidades, como la súper-resolución [13], que permite a partir de una imagen en baja resolución, generar la misma imagen pero en una resolución mayor, recuperando detalles de grano fino y mejorándolos de una forma notablemente mejor que los métodos clásicos de aumento de resolución.

1.2. Objetivos

Una vez conocida la motivación y la descripción del problema planteado, el principal objetivo de este Trabajo Fin de Máster es realizar un estudio sobre distintas técnicas de generación de datos sintéticos y súper-resolución mediante GANs, para comprobar sus beneficios y comprobar si es posible mejorar los resultados en la detección del cáncer de piel mediante algoritmos inteligentes.

Para ello, en primer lugar, se realizará una revisión del estado del arte, abordando la problemática del cáncer de piel y su detección, conjuntos de datos de carácter público, modelos de clasificación utilizados, y GANs utilizadas para la generación de datos sintéticos y súper-resolución, donde se estudiarán la base teórica de los modelos, su funcionamiento y las métricas correspondientes al problema al que están asociado. Tras la revisión, se documentará el marco teórico, el cual tratará todas las tecnologías utilizadas en este trabajo.

Una vez realizado el estudio, se evaluará el rendimiento de aplicar dichas técnicas mediante el ajuste de los modelos estudiados a la problemática de la detección del cáncer de piel, para ello se evaluarán los distintos modelos, individualmente, en base a su utilidad (clasificación, generación de datos y súper-resolución) comparando sus métricas, y se evaluará como afecta aplicar los métodos de mejora propuestos a la clasificación, es decir, a la detección de la enfermedad.

Al final, otro objetivo de este trabajo es aplicar el conocimiento adquirido durante el desarrollo del máster y, más en concreto, de la asignatura "Aprendizaje Profundo" y darle un enfoque más práctico aplicado a un caso real y de alta relevancia como es la detección precoz del melanoma. También pretende expandir el conocimiento ofrecido por la titulación y aprender conceptos y técnicas de relevancia en el mercado laboral que puedan ser de utilidad de cara a encontrar oportunidades de empleo en el sector de la IA.

1.3. Estructura de la memoria

La estructura de la presente memoria se compone de:

1. **Introducción**: esta misma sección, donde se introduce las distintas motivaciones que han llevado a realizar este trabajo, así como la descripción del problema que se quiere resolver, los objetivos del trabajo y la estructura de la memoria.
2. **Marco teórico**: esta sección cubre el origen la inteligencia artificial y su relación con la detección del melanoma, analizando la progresión y evolución hasta la concepción de modelos complejos como los de generación de modelos y súper-resolución, además de comprender las bases teóricas de los mismos.
3. **Materiales y metodología**: en esta sección se discute sobre la metodología adoptada durante el proyecto, así como de la organización en fases de la misma. Por otro lado, se detallan los recursos hardware utilizado, así como los distintos conjuntos de datos utilizados para cada tarea y los modelos elegidos para ser evaluados, donde se detalla más su arquitectura y principales ventajas.
4. **Resultados experimentales**: esta sección expone como se han llevado a cabo los distintos experimentos, así como las métricas utilizadas para su evaluación, los problemas encontrados y las soluciones adoptadas. Además, evidentemente, se muestran los resultados y las conclusiones extraídas de los mismos.
5. **Conclusiones y lineas de trabajo futuras**: por último, se realiza una reflexión sobre el proyecto y todo lo aprendido durante su desarrollo, se presentan las conclusiones obtenidas a partir de los resultados experimentales y se plantean posibles líneas de trabajo futuro para continuar avanzando en el tema.

Tras las conclusiones se encuentra la bibliografía consultada así como un apéndice con las imágenes usadas en la memoria para que puedan ser consultadas con mayor detalle.

CAPÍTULO 2

Marco teórico

Esta sección se centrará en el contexto y aspectos teóricos de los mencionados algoritmos inteligentes utilizados en el estado del arte para la investigación y desarrollo de CAD. Para ello, se ahondará en el origen de la inteligencia artificial y su relación con el campo de la detección del melanoma, además de cubrir los avances en la materia y la evolución hacia modelos más complejos, como los de generación de datos y súper-resolución.

2.1. Inteligencia Artificial y Machine Learning

La IA puede definirse como la capacidad de una máquina para realizar ciertas tareas propias de la inteligencia humana [14]. Esta definición tiene su origen en el Test de Turing, propuesto por Alan Turing en 1950 justamente para darle una definición concreta al término. Este test se consideraba exitoso si un interrogador haciendo preguntas a una máquina y una persona real no es capaz de distinguir la máquina de la persona, por lo que debía cumplir con las siguientes habilidades:

- **Procesamiento del lenguaje natural**: capacidad de comunicarse de forma fluida en algún idioma, en el caso del Test de Turing, en inglés.
- **Representación del conocimiento**: almacenar de alguna forma el conocimiento o la información que recibe.
- **Razonamiento automático**: usar el conocimiento para responder preguntas o generar conclusiones.
- **Machine Learning**: capacidad de adaptación y extracción de patrones.

Además, existe una versión del test, el Test Total de Turing, en el que se incluye una señal de vídeo del interrogador y la capacidad de pasarle objetos a través de un agujero, por lo que aparecían dos habilidades más:

- **Visión por computador**: capacidad de percibir objetos.
- **Robótica**: capacidad de manipular objetos.

A pesar de que han pasado más de 70 años desde la descripción de esta prueba, estos 6 conceptos se corresponden con 6 de las disciplinas más relevantes del sector de la IA en

la actualidad.

Si prestamos atención a la definición de estas disciplinas e intentamos relacionarlos con nuestro problema, al final, el proceso de detectar el melanoma a través de una dermatoscopia, si lo reducimos a lo absurdo, consiste en adquirir un conocimiento básico a través de la extracción de reglas y patrones sobre cuando es un caso positivo o negativo, a raíz de percibir muestras de dichos ejemplos y, de alguna forma, almacenar esa experiencia, de forma que cuanto a mayor cantidad y variedad de muestras se vean, mejor decisiones se tomarán. Es por ello por lo que los conceptos de *Machine Learning* y Visión por computador toman un papel crucial en la problemática que abordamos.

Centrándonos en el primer concepto mencionado, el término *Machine Learning* es toda una rama de la IA, en concreto, hace referencia al conjunto de técnicas y métodos que pueden detectar patrones de forma totalmente automática en los datos y, a raíz de estos, intentar predecir el futuro o, lo que nos interesa en nuestro caso, tomar una decisión [15]. Este conjunto de técnicas están sustentadas una gran variedad de modelos estadísticos y probabilísticos, de forma que no existe el modelo ideal o perfecto, en función del tipo y la naturaleza de nuestros datos habrá modelos que se adapte mejor o peor a los mismos.

Es común hablar de 3 tipos de *Machine Learning*:

- **Aprendizaje supervisado**: el objetivo es asociar unas entradas x a una salida y dado un conjunto de entrenamiento $D = \{(x_i, y_i)\}_{i=1}^{N}$ compuesto de pares entrada y salida.
- **Aprendizaje no supervisado**: en este caso, tan solo tenemos un conjunto de entradas $D = \{x_i\}_{i=1}^{N}$ y el objetivo es encontrar patrones en los datos, por ejemplo, de forma que se agrupen en función de ciertas características.
- **Aprendizaje por refuerzo**: es el menos común de los 3, en este caso el sistema aprende a través de la interacción con el entorno, que en función de la acción que tome en un determinado momento le dará una recompensa o una penalización.

Nos centraremos en este caso en al aprendizaje supervisado, pues el problema de la detección del melanoma es un claro ejemplo de esta categoría, ya que dada unas entradas, como puede ser la imagen del dermatoscopio o una serie de características extraídas por un profesional, queremos asociarla con una salida, positivo o negativo de melanoma.

Existen infinidad de modelos de aprendizaje supervisado, pero nos centraremos en los mas utilizados en nuestra problemática [4] y veremos una ligera pincelada sobre ellos.

2.1.1. Árboles de decisión

Los árboles de decisión son modelos basando en dividir recursivamente el espacio de entradas y definiendo un modelo en cada región resultante, de forma que cada hoja del árbol sea una región, es un concepto que parece realmente complicado, pero se entiende mucho mejor con una imagen.

Como podemos comprobar en la figura 2.1a, el primer nodo consulta si la variable x_1 supera un umbral t_1, en caso positivo, se pasa a otro nodo donde se consulta si la variable x_2 supera un umbral t_2, de forma que se llegue a una región R_1, también conocida como hoja, donde se usa un modelo probabilístico simple donde se deduce la clase en función de las muestras conocidas correspondientes a dicha región.

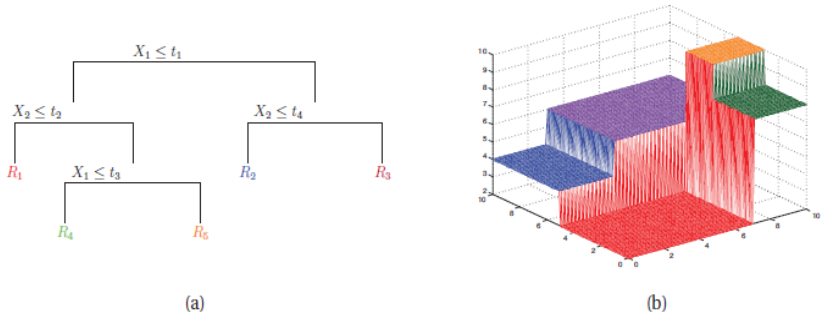


Figura 2.1: Árboles de decisión. [15]

Pueden realizar tanto tareas de clasificación como de regresión y, de forma general, ofrecen unos excelentes resultados para grandes conjuntos de datos, a pesar de su complejidad, pues tienen sentido tanto para variables categóricas como numéricas.

También existe el concepto de bosque aleatorio, que no son más que un conjunto de arboles de decisión, caracterizados por tener una varianza alta, el cual es uno de los modelos más robustos y ampliamente utilizados, ya que ofrece una alta capacidad de generalización.

2.1.2. Máquinas de vectores de soporte

Las máquinas de vectores de soporte (SVM) son unos de los modelos más simples y eficientes conocidos, se basa en el principio de minimización del riesgo estructural [16]. El caso más simple se puede observar en la figura 2.2 donde se trata de dividir el espacio en dos planos, correspondiente a dos clases, definiendo una superficie de separación óptima H, donde H_1 y H_2 son los puntos más cercanos a H de cada clase. En función de la dimensionalidad de los datos, la separación será a través de simplemente planos o, en su defecto, hiperplanos.

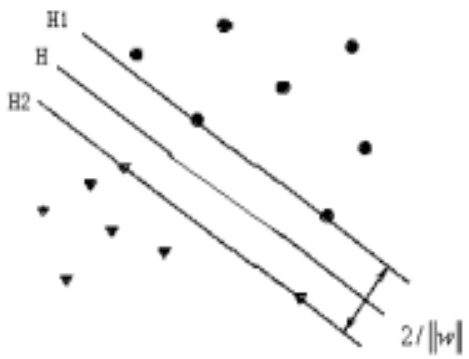


Figura 2.2: Superficie de separación óptima de un SVM. [16]

En función de la dimensionalidad de los datos y la propia distribución de los mismos existen dos casos posibles para un SVM, el caso más básico, en el que el problema es linealmente separable y puede usarse un clasificador lineal, simplemente una recta en el caso de un problema en dos dimensiones. En el caso de que los datos no sean linealmente separables, será necesario el uso de un kernel, como una función cúbica, cuadrática, Gaussiana, etc.

Entre las principales ventajas de este modelo se encuentra que reduce mucho el efecto de la maldición de la dimensionalidad. Este concepto se refiere a los problemas que surgen con algunos modelos cuando se someten a conjuntos de datos con una gran cantidad de variables independientes, entre estos problemas se encuentra la expansión del espacio de búsqueda, el aumento en la dispersión de los datos y el sobreajuste, además de problemas de memoria.

2.1.3. K-Nearest Neighbors

Por último, *K-Nearest Neighbors* (KNN) es otro modelo realmente simple cuyo funcionamiento se basa en la distancia con los puntos vecinos ya clasificados [4]. Esta distancia es, típicamente, la distancia euclídea (2.1) o la distancia manhattan (2.2), aplicada a cada una de sus dimensiones.

$$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \quad \text{(Distancia Euclídea)} \tag{2.1}$$

$$\sum_{i=1}^{n}|x_i - y_i| \quad \text{(Distancia Manhattan)} \tag{2.2}$$

El procedimiento es bastante fácil de comprender, se obtienen los K vecinos, normalmente, entre 3 y 10, haciendo uso de alguna de las distancias mencionadas y se realiza un voto por mayoría, de esta forma, se le asigna la etiqueta que corresponda al mayor número de votos. Se ilustra realmente bien en la figura 2.3, donde podemos ver las muestras etiquetadas y, a su lado, la probabilidad de pertenecer a la clase 1 en función de la posición de la muestra a clasificar, cuanto más cerca de las muestras de esa clase y más separado del resto está, mayor es dicha probabilidad.

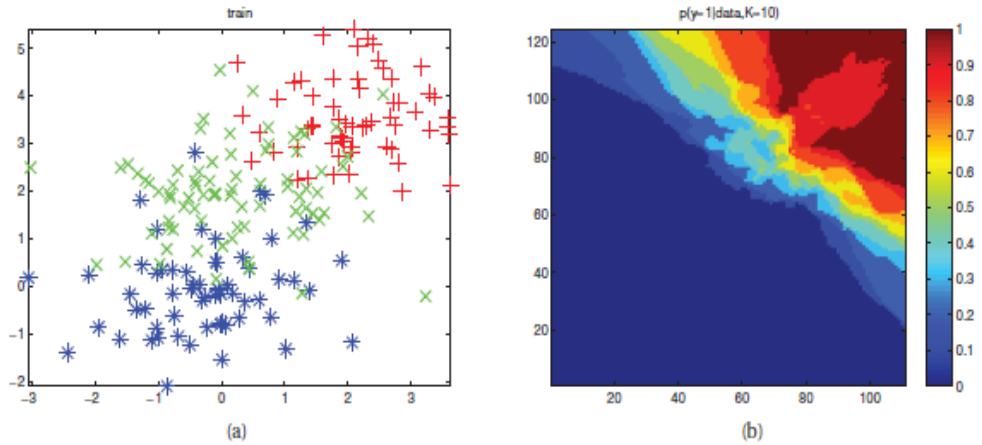


Figura 2.3: (a) Datos de entrenamiento. (b) Probabilidad de pertenecer a la clase 1. [15]

Pero claro, el hecho de que su funcionamiento sea tan simple y, además, ofrezca grandes resultados para algunos conjuntos de datos no resulta gratis, también sufre de muchos problemas [15], en concreto, debido a que este algoritmo realmente no tiene un proceso de aprendizaje, tan solo almacena las muestras etiquetadas para, después, consultarlas, esto

sumado a que también sufre mucho de la dimensión de la dimensionalidad, lo convierte en un modelo que ofrece un bajo rendimiento y con alta tendencia al sobreajuste con conjuntos de datos grandes y complejos.

2.2. Aprendizaje Profundo

El *Machine Learning* revolucionó la forma en que abordamos problemas complejos mediante el uso de la informática para aprender y mejorar a partir de datos en lugar de las intuiciones, sin embargo, a medida que fuimos avanzando y empezamos a aplicarla en el mundo real, nos encontramos con problemas más complejos para los que no logramos resultados óptimos con esta aproximación.

Una de las principales dificultades en muchas aplicaciones de inteligencia artificial surge cuando los factores de variación influyen en cada dato observable [17], por ejemplo, en una imagen de un coche rojo, los píxeles pueden variar enormemente dependiendo de factores como la iluminación o el ángulo de visión. La tarea de las aplicaciones de inteligencia artificial es analizar estos factores de variación y diferenciar los que son relevantes para la tarea en cuestión.

Es entonces cuando el aprendizaje profundo surge como la respuesta a este desafío a través de extraer representaciones significativas y abstractas de datos complejos a raíz de representaciones en términos de otras más simples. Un ejemplo puede ser el del coche rojo que mencionábamos antes, podemos entender un coche rojo como la unión de distintas partes como la puerta y la rueda, cada una de estas partes, puede entenderse como una unión de contornos, y cada uno de estos contornos como un conjunto de bordes, así sucesivamente.

2.2.1. Neurona artificial y perceptrón multicapa

Este aprendizaje profundo tiene su base en la neurona artificial, la cual a la vez es una representación simplificada de la neurona biológica, que representa nuestra unidad de procesamiento más básica. En nuestro cerebro, las neuronas están conectadas entre sí a través de sinapsis, reciben unas entradas, realizan alguna combinación de estas en el soma y emiten la salida a través del axón [18].

Este mismo concepto es el que sigue la neurona artificial que se puede observar en la figura 2.4, una unidad de procesamiento básica cuya función (2.3) se compone de los siguientes elementos:

$$y_i = f\left(\sum_{j=1}^{n} w_{ij}x_j - b\right) \quad \text{(Ecuación de una neurona artificial)} \tag{2.3}$$

- **Entradas** (x): simplemente son los valores que toma como entrada la neurona, veremos en el futuro que la salida de una neurona puede ser la entrada de otras.
- **Pesos sinápticos** (w): cada peso esta asociado a una entrada, determina la influencia de la misma en la neurona.

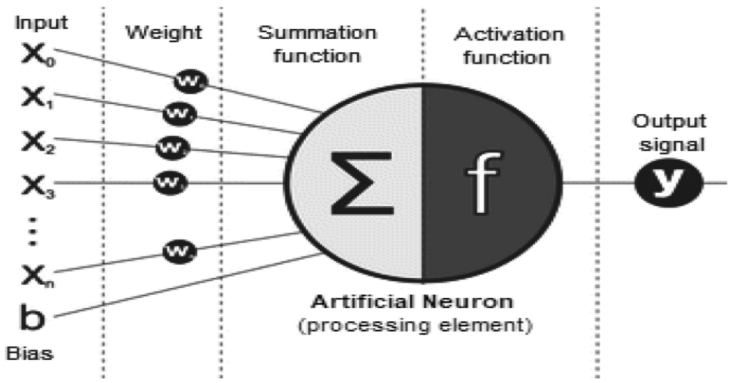


Figura 2.4: Neurona artificial. [19]

- **Función de activación** (f): es una regla matemática que dado el sumatorio de las entradas con los pesos y el sesgo, le da el valor de salida final a la neurona, lo veremos con más detalle.
- **Sesgo** (b): es un valor que se resta al resultado del sumatorio antes de aplicar la función de activación, permitiendo ajustar el umbral de activación de la neurona.

Esta función de activación será decisiva en función del objetivo de la neurona, pues indicará si la neurona tiene salida discreta o continua, y la regla matemática que se aplicará, que puede ser lineal, sigmoide, softmax, etc.

Con una simple neurona resulta trivial la implementación de, por ejemplo, las puertas lógicas básicas, pero, tiene el problema de restricción que también tenia el SVM más simple, solo puede dividir conjuntos linealmente separables. Mientras que en SVM, como hemos comentado, se usan kernels, en el caso de la neurona artificial, se combinan, de la misma forma que pasa en nuestro cerebro, dando lugar a lo que se conoce como perceptrón multicapa.

El perceptrón multicapa no es más que una forma de agrupación de las neuronas artificiales y es la base de prácticamente todos los modelos de aprendizaje automático. Las neuronas ahora se agrupan en filas, de forma que la primera capa es denominada entrada, las salidas de esta primera capa están conectadas a otra fila, siendo su entrada, denominada capa oculta, que realmente puede ser una o más, y por último, se conectan a la capa de salida, que en función del problema puede ser una (clasificación binaria) o varias (clasificación multi-clase, en la mayoría de los casos.

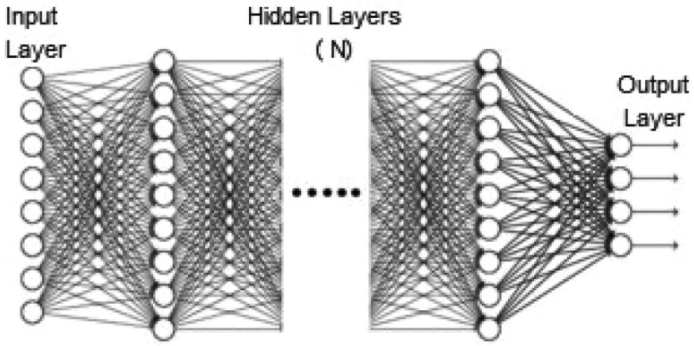


Figura 2.5: Red neuronal profunda. [19]

De ese hecho de acumular muchas capas ocultas es donde surge el concepto de red neuronal profunda, que podemos observar en la figura 2.5 y es la base teórica del aprendizaje profundo, pero hasta el momento, no se ha aprendido nada, tan solo tenemos un gran conjunto de pesos sinápticos que no aportan más que una salida arbitraria.

2.2.2. Entrenamiento de la red

Una vez definimos la arquitectura de la red neuronal, tan solo existe un parámetro modificable en las redes neuronales, los pesos sinápticos son los que representan el conocimiento y los patrones adquiridos por la red, de forma que serán esos valores los que se deban ajustar a la hora de realizar el entrenamiento.

El aprendizaje en este caso se basa en reducir el error entre la salida estimada y la real a través de la regla de actualización de pesos sinápticos (2.4), en este caso, para una sola neurona [17]. El procedimiento es simple, calcular el error, obtener el cambio proporcional multiplicando por la entrada de dicho peso y, por último se le aplica la tasa de aprendizaje η, la cual ajusta como de brusco es el cambio, pues este valor que se obtiene no es el nuevo peso sináptico, es la variación que sufre.

$$\Delta w_j(k) = \eta[z(k) - y(k)]x_j(k) \quad \text{(Regla de actualización)} \tag{2.4}$$

Pero claro, este es el caso más simple en el que tan solo tenemos una neurona, cuando se trata con redes profundas donde existen multitud de capas intermedias el proceso se complica. El principio sigue siendo el de minimizar el error, pero se produce una retro-propagación progresiva en todas las capas de la red.

La retro-propagación implica calcular el error en la salida de la red y luego propagarlo hacia atrás a través de todas las capas, ajustando los pesos en cada capa en función de cómo contribuyeron al error, esto se hace utilizando la regla de la cadena de cálculo para obtener el gradiente del error con respecto a los pesos en cada capa [17].

Existen multitud algoritmos de retro-propagación que varían en cómo calculan y aplican estos ajustes de peso. Algunos de los algoritmos más populares incluyen:

- **Retro-propagación estándar**: es el algoritmo clásico de retro-propagación que calcula los gradientes del error con respecto a los pesos en cada capa de la red y luego los utiliza para ajustar los pesos mediante algún método de optimización, como el descenso del gradiente estocástico.
- **Retro-propagación con momento**: introduce un término de momento que acumula las actualizaciones de los pesos en cada iteración, lo que puede ayudar a acelerar el entrenamiento y mejorar la convergencia.
- **Adam** (Adaptive Moment Estimation): combina las ideas del descenso de gradiente con momento y el método de estimación adaptativa del aprendizaje de la tasa de aprendizaje. Utiliza estimaciones de primer y segundo orden de los momentos del gradiente para adaptar la tasa de aprendizaje para cada parámetro de la red de manera adaptativa.
- **RMSprop** (Root Mean Square Propagation): RMSprop es una variante del descenso de gradiente estocástico que normaliza la tasa de aprendizaje utilizando un promedio

móvil del cuadrado de los gradientes anteriores, esto ayuda a mitigar el problema de la divergencia en el descenso del gradiente cuando se trabaja con tasas de aprendizaje variables.

Cuando se realiza el procesamiento de entrenamiento de una red neuronal, no se la estima el error y se retro-propaga dato a dato, pues ese proceso de retro-propagación es muy costoso computacionalmente hablando, se suele hacer uso de lotes de datos, de forma que se procesan de forma simultanea antes de actualizar los pesos, además de aprovechar la arquitectura de hardware especializado, como son las unidades de procesamiento gráfico (GPU), motivo por el que se utilizan para el entrenamiento de estas redes.

Asociado a este concepto de lote aparece el de época, que corresponde a una pasada completa a través de todo el conjunto de datos, y vuelve a surgir el de tasa de aprendizaje, el cual es un hiperparámetro que controla como de bruscos son los ajustes de peso realizados en cada iteración del algoritmo de optimización, de forma que una tasa de aprendizaje más alta acelerará el entrenamiento, pero puede provocar problemas como la divergencia del modelo, por otro lado, una tasa de aprendizaje más baja puede hacer que el proceso sea más lento pero más estable, por lo que la elección de un tamaño de lote y tasa de aprendizaje adecuadas, junto al número de épocas, será esencial para el entrenamiento.

Además de estos parámetros, otro aspecto crucial para el entrenamiento es el de función objetivo, el cual cuantifica la variación entre las predicciones de la red y las etiquetas reales de los datos, al final, el el objetivo del entrenamiento es minimizar el valor de esta función. Existen multitud de funciones:

- **Error Cuadrático Medio** (MSE): calcula la diferencia cuadrada entre las predicciones de la red y las etiquetas reales, y luego promedia estos valores. Es comúnmente utilizada en problemas de regresión, donde se busca predecir un valor numérico.
- **Entropía Cruzada** (Cross-Entropy): penaliza de manera más severa las predicciones incorrectas, lo que la hace especialmente efectiva en problemas de clasificación binaria y multiclase.
- **Entropía Cruzada Categórica** (Categorical Cross-Entropy): similar a la entropía cruzada, pero diseñada específicamente para problemas de clasificación multiclase donde las etiquetas están codificadas en forma de vectores one-hot.
- **Log-Verosimilitud Negativa** (Negative Log-Likelihood): es comúnmente utilizada en problemas de clasificación, especialmente en el contexto de redes neuronales con funciones de activación softmax en la capa de salida. Penaliza las predicciones incorrectas de manera proporcional a la probabilidad asignada a la etiqueta correcta por la red.

Una vez se concluye el entrenamiento se pueden producir, a priori, 3 posibles escenarios respecto al resultado del mismo y a su relación con los datos, los podemos observar en la figura 2.6.

En el primer caso, hablamos de *overfitting* o sobreajuste cuando el modelo aprende demasiado bien el conjunto de datos con el que se entrena, incluido el ruido, hasta el punto de que no es capaz de generalizar correctamente cuando se enfrenta a nuevos datos, el modelo se adapta demasiado bien a los datos con los que se ha entrenado y pierde la capacidad de capturar patrones generales que se apliquen a los nuevos datos desconocidos.

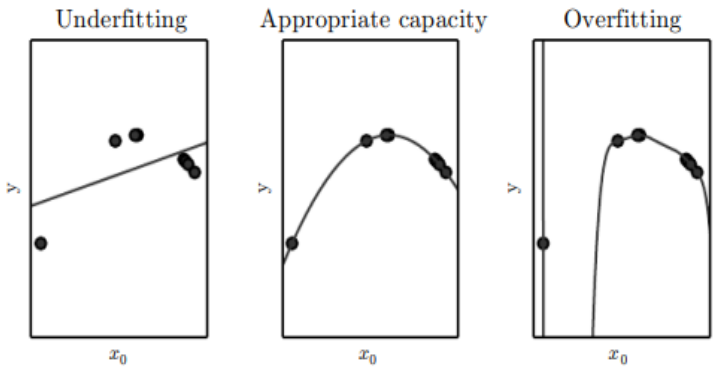


Figura 2.6: Posibles casos de entrenamiento. [17]

Por otro lado, el *underfitting* o subajuste ocurre cuando el modelo es demasiado simple para capturar los patrones y los detalles de los datos con los que se realiza el entrenamiento, el modelo no logra un ajuste lo suficiente bueno a los datos y, como resultado, no puede capturar patrones relevantes, lo que resulta en un rendimiento realmente malo incluso en el entrenamiento.

Por tanto, podemos definir el objetivo de un entrenamiento como un equilibrio entre sobreajuste y subajuste, lo que quiere decir un buen rendimiento tanto con los datos con los que se ha entrenado tanto con datos desconocidos y que en consecuencia el modelo ha logrado generalizar y extraer patrones de interés.

De entre estos dos problemas, el más común es el del sobreajuste, por lo que existen diversas técnicas para mitigar su aparición:

- **Regularización L1/L2**: consiste en penalizar los pesos grandes al agregar términos adicionales a la función de pérdida durante el entrenamiento, la regularización L1 agrega la suma de los valores absolutos de los pesos, mientras que la regularización L2 agrega la suma de los cuadrados de los pesos, lo que ayuda a evitar que los pesos se vuelvan demasiado grandes y limita la complejidad del modelo.

- **Dropout**: consiste en desactivar aleatoriamente un porcentaje de las neuronas durante el entrenamiento, lo que evita que las neuronas dependan demasiado de otras para la predicción.

- **Reducir el modelo**: existen casos en los que el modelo utilizado es demasiado complejo para los datos con los que se esta entrenando, lo que hace que sea propenso a sobreajustarse, por lo que la simplificación de este o la elección de uno de menor complejidad puede mejorar los resultados.

- **Data Augmentation**: aumentar el tamaño del conjunto de datos mediante técnicas como la rotación, el sesgo, el cambio de escala y el recorte aleatorio de imágenes puede ayudar a proporcionar una mayor variabilidad en los datos. Además de estas técnicas tradicionales, como hemos comentado y es el objetivo de este trabajo, es posible utilizar GANs para generar nuevas imágenes a partir de un conjunto reducido.

En cuanto al subajuste, lo más común es seleccionar un modelo más complejo, realizar ingeniería de características u optimizar hiperparámetros.

2.3. Redes neuronales convolucionales

Mientras que las redes neuronales, tal y como han sido descritas hasta ahora, son excepcionales para datos que no tienen localidad espacial, como pueden ser registros de temperatura, perfiles de usuarios, etc., no aprovechan dicha localidad, que tiene sentido en multitud de tipo de datos, como las series temporales y las imágenes, ya que, de la misma forma que nuestro cerebro relaciona los píxeles que están alrededor de un punto de referencia para detectar formas, las redes neuronales deben ser capaces de extraer esa información de una forma similar.

Intentando aprovechar este principio es donde surgen las Redes Neuronales Convolucionales (CNN), un tipo de red que se caracteriza por aprovechar la información espacial de los datos, basándose en una arquitectura inspirada en diversos estudios sobre la corteza visual y, tal como hace nuestro cerebro, se basa en el reconocimiento a múltiples niveles, de forma que en primer lugar se reconocen píxeles individuales, uniendo estos se reconocen formas, figuras geométricas y elementos sofisticados [17].

2.3.1. Arquitectura y funcionamiento

La principal diferencia a nivel de arquitectura entre una red neuronal tradicional y una convolucional es que en esta última, las capas de neuronas estan organizadas en 3 dimensiones, ancho, altura y profundidad. Ancho y altura representan la propia estructura de datos en rejilla de una imagen, mientras que la profundidad es comúnmente utilizado para canales, como pueden ser los colores y el canal alfa.

La composición tradicional de estas redes es de 3 tipos de capas u operaciones que se le aplican a los datos, convolución, *pooling* y totalmente conectadas, siendo estas últimas las capas típicas de las redes neuronales tradicionales.

En primer lugar, la convolución es una operación lineal cuyo objetivo es la extracción de características, donde un conjunto reducido de números, denominado kernel es operado junto con la entrada, denominada tensor [20]. En ella se calcula el producto elemento a elemento entre el kernel y el tensor en cada zona del mismo y se suma para calcular lo que se denomina el mapa de características.

Los tres principales hiperparámetros que definen la operación de convolución son el tamaño del filtro, el número de kernels y el *stride*. El primero es típicamente 3×3, pero a veces es 5×5 o 7×7, en función de la capa a la que pertenezca. El segundo determina la profundidad de los mapas de características de salida, es decir, el número de kernels o filtros que se aplican. El último especifica el solapamiento con el que se aplica el filtro, de forma que un *stride* de 1 significa deslizar el kernel pixel a pixel, en el caso de 2 se deslizan 2 píxeles en cada paso y así sucesivamente. Por ejemplo, en la figura 2.7 se esta aplicando un kernel de 3x3 con un *stride* de 1 y, en principio, un único filtro.

Una consecuencia de aplicar esta técnica es que se reduce el ancho y largo del mapa de características, por lo que se pierde control sobre el tamaño espacial de salida. Una solución a esto es aplicar *zero-padding*, lo cual consiste en rellenar el tensor de entrada con ceros alrededor del borde, con esto se soluciona este problema y se puede controlar con éxito el tamaño de salida.

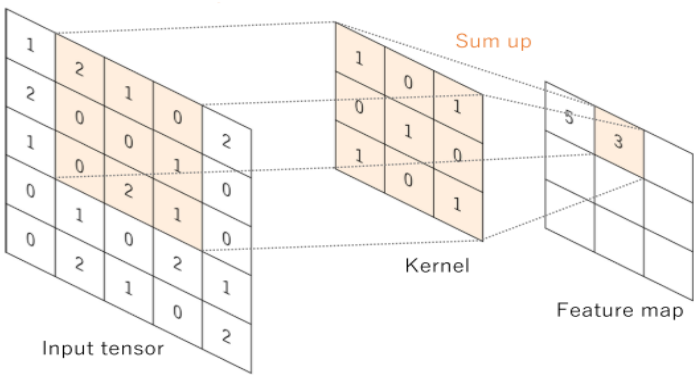


Figura 2.7: Ejemplo de convolución. [20]

La característica clave de esta operación es el hecho de que se compartan los pesos, los kernels se comparten en todas las posiciones de la imagen, de forma que los patrones de características extraídos por los kernels sean invariantes a las traslaciones, se aprendan jerarquías espaciales de patrones de características mediante el submuestreo, y se reduce el número de parámetros a aprender en comparación con las capas completamente conectadas, lo que permite aumentar la eficiencia del modelo .

Por otro lado, es típico introducir capas de *pooling* entre convoluciones sucesivas, esta operación consiste en realizar un submuestreo reduciendo progresivamente el tamaño espacial de salida y, en consecuencia, disminuyendo la cantidad de parametros ajustables en las capas siguientes, de forma que se reduce el riesgo de sobreajuste.

Existen multitud de formas de aplicar *pooling*, las más destacadas son:

- **Max Pooling**: extrae submuestras de los mapas de características de entrada, de forma que se devuelve el valor máximo en cada muestra y se descartan todos los demás valores. Es el más utilizado en la práctica, con un tamaño de kernel de 2×2 y un *stride* de 2, de forma que se reducen el 75 % de las activaciones. Un ejemplo se puede observar en la figura 2.8.
- **Global Average Pooling**: es exactamente igual que el anterior pero, en este caso, en lugar de obtener el valor máximo de cada parche se realiza un promedio y ese es el valor que se utiliza.

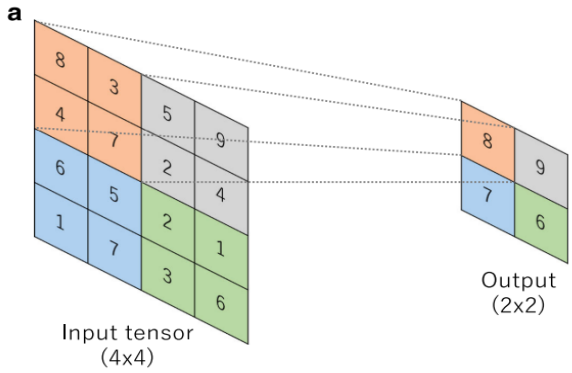


Figura 2.8: Ejemplo de *Max Pooling*. [20]

Por último, se convierte ese conjunto de características extraído a una única dimensión y es donde se realiza la tarea que se requiere, por ejemplo, la clasificación, mediante el uso

de capas totalmente conectadas.

Respecto a la disposición de las capas, lo más común es apilar varias capas de convolución y aplicar la función *Rectified Linear Units* (ReLU), que mantiene intactos los valores positivos y, los negativos, los convierte a 0. Tras esto se aplica *pooling* para reducir el tamaño espacial, esta combinación se repite de forma sucesiva hasta que se decide aplanar y usar capas totalmente conectadas para la salida de la red.

2.3.2. Transfer Learning

A la hora de entrenar una CNN es necesaria una gran cantidad de imágenes lo suficientemente variadas como para que el modelo sea capaz de extraer de forma efectiva las figuras geométricas y demás relacionas, además de generalizar correctamente. Además, aunque se tenga esa inmensa cantidad de datos, los entrenamientos pueden ser realmente largos y necesitar de una gran cantidad de recursos, ya que los modelos que se usan en la práctica son realmente complejos y estan formados por millones de parámetros.

Es a raíz de este motivo cuando surge el concepto de *Transfer Learning*, se basa en la liberación y publicación por parte, principalmente, de grupos de investigación, de arquitecturas CNN ya preentrenadas con grandes conjuntos de datos. El conjunto más popular, Imagenet [21] se compone de 1,2 millones de imágenes de un total de 1000 clases diferentes, lo que lo convierte, además, en un banco de pruebas excelente para este tipo de modelos.

Por tanto, en la práctica, lo más común es adoptar una red ya preentrenada con este u otro conjunto de datos similar, se sustituyen las últimas capas para la función que se quiera adaptar y se realiza un entrenamiento más ligero con el conjunto de datos específico con el que se quiera ajustar el modelo, de forma que las primeras capas, que contienen características más genéricas, serán de utilidad y se ajustarán ligeramente mientras que las últimas capas serán las específicas y las que más cambiarán.

Existen multitud de arquitecturas publicadas y preentrenadas, entre las más populares se encuentran AlexNet, VGGNet, Inception y ResNet.

AlexNet

AlexNet fue el modelo que resultó ganador por diferencia durante la competición *ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) 2012*, de forma que supuso un cambio de ruta en el paradigma de la visión por computador.

Entre los puntos más destacados del modelo se encuentra [22]:

- **Uso de ReLU**: se demostró que utilizando la función de activación ReLU en lugar de tanh o sigmoid permitía a la red aprender más rápido sin sacrificar la precisión.
- **Entrenamiento en múltiples GPUs**: se dividió el entrenamiento entre dos GPUs con 3GB de memoria cada una. Realizando esto, se logró reducir la tasa de errores del top-1 y top-5 en un 1,7 % y 1,2 % respectivamente respecto al entrenamiento con una única GPU.
- **Normalización**: una de las ventajas de ReLU es que no requiere de normalización, pero aun así se decidió aplicar y ofreció una mejora en las tasas mencionadas

anteriormente de un 1,4 % y 1,2 %.

- **Pooling con solapamiento**: se utiliza pooling con solapamiento en lugar de pooling convencional para reducir el sobreajuste, lo que también disminuye los errores top-1 y top-5.

En total, la red consta de 8 capas, donde las 5 primeras son convolucionales y las restantes totalmente conectadas, sumando un total de aproximadamente 62,3 millones de parámetros.

VGGNet

VGG [23] fue la red que quedó en segundo lugar en la competición *ILSVRC 2014* respecto a tareas de clasificación, su arquitectura se basa en:

- **Entradas**: su entrada está restringida a 224x224 para mantener un tamaño constante, por lo que se recortaban las imágenes de forma centrada a ese tamaño.
- **Capas convolucionales**: se usa el tamaño de filtro más pequeño posible, 3x3 con una gran cantidad de filtros, además de seguir la linea de AlexNet y utilizar la función ReLU.
- **Capas ocultas**: de nuevo, todas las capas usan ReLU, pero en este caso, el usar normalización aumentaba el uso de memoria y el tiempo de entrenamiento, por lo que se rechazó su uso.
- **Capas totalmente conectadas**: contiene tres de estas capas, donde las dos primeras tienen cada una 4096 neuronas, mientras que la tercera tiene 1000, una para cada clase.

Existen variantes de esta arquitectura en función de la cantidad de capas que tenga, dando lugar a las más populares, VGG-16 y VGG-19. Todas las configuraciones se pueden ver en la figura 2.9.

Inception

Esta fue la red que le robó el primer puesto a VGG, destaca principalmente por introducir el concepto de *inception* [24].

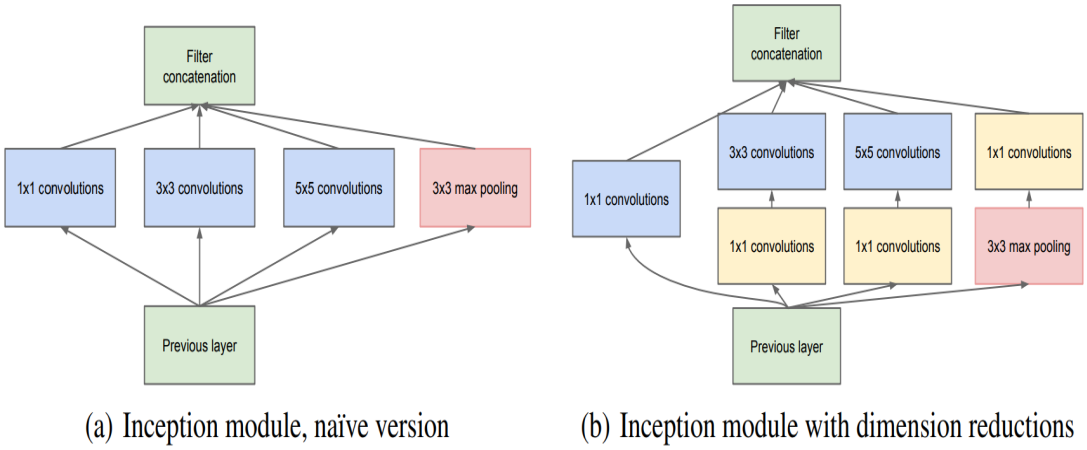
Hasta el momento, todas las redes simplemente apilaban capas convolucionales, con el objetivo de mejorar el rendimiento, lo cual era computacionalmente caro y aumentaba la tendencia al sobreajuste. Fue entonces cuando se percibió que las partes relevantes de una imagen puede ser de distinto tamaño y, por tanto, la elección del tamaño del kernel era algo crucial pero, a la vez, complicado de llevar a cabo.

Por tanto, la solución planteada fue aplicar tres tamaños de kernel diferente, 1x1, 3x3 y 5x5, para después concatenar las salidas, además, se le aplica también *max pooling* y convoluciones 1x1 para reducir el número de parámetros, tal y como se ve en la figura 2.10.

Esto logró solucionar la limitación que suponía tan solo apilar capas y capas de convolución con la esperanza de que mejoraran los resultados, pasando a tener 22 capas y 4 millones de parámetros.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figura 2.9: Configuraciones de VGG. [23]



(a) Inception module, naïve version (b) Inception module with dimension reductions

Figura 2.10: Distintos módulos de *inception*. [24]

ResNet

Por último, ResNet [25] surge de la misma observación de la que surge la anterior, los modelos se comportan peor a medida que se añaden capas, debido a lo que es conocido como desvanecimiento del gradiente, es decir, esto hace que el gradiente sea cero o demasiado grande.

Para mitigarlo, los investigadores introducen el concepto de *skip connections*, que permiten que la información de entrada se sume directamente a las salidas del siguiente bloque, lo que crea una ruta de flujo de información adicional que reduce la pérdida de información y el desvanecimiento del gradiente.

Por tanto, la estrategia detrás de esta red es dejar que la red se ajuste al mapeo residual en lugar de que las capas aprendan únicamente de la capa anterior, de la forma descrita en la figura 2.11. De esta forma y, al reducir el desvanecimiento del gradiente, es posible utilizar

un mayor número de capas, dándose versiones como ResNet50, Resnet101 y Resnet152, que disponen de 50, 101 y 152 capas respectivamente.

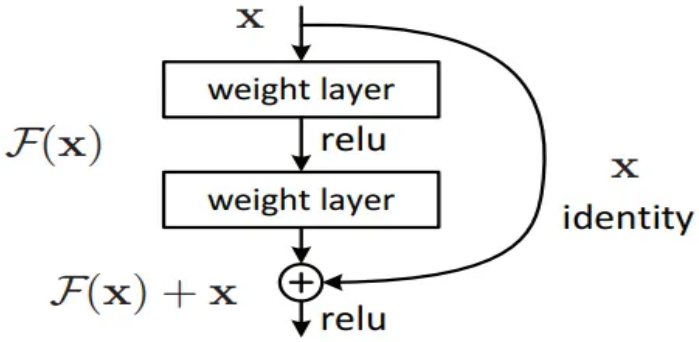


Figura 2.11: Ejemplo de *skip connection*. [25]

2.4. Redes Generativas Antagónicas

Hasta este momento, las redes neuronales resultan excepcionales para analizar y realizar predicciones o plantear sugerencias, pero tal y como se comentó, la inteligencia artificial persigue imitar la inteligencia humana y, nuestro cerebro, además de la labor de identificación, es capaz de imaginar y crear nuevas ideas y conceptos.

Es en esa búsqueda un sistema inteligente que pueda crear cosas nuevas donde surgen las Redes Generativas Antagónicas (GAN) [26]. Esta arquitectura de redes tiene como objetivo generar datos a raíz de una entrada arbitraria, para ello, se basan en un aprendizaje no supervisado, donde el modelo no aprende a etiquetar o a predecir una salida, si que su objetivo es descifrar la distribución subyacente a los datos y generar nuevas muestras que puedan pertenecer a dicha distribución. Las GAN tienen como base la coexistencia de dos modelos, un generador y un discriminador.

2.4.1. Generador y Discriminador

En primer lugar, el generador, tal y como su nombre indica, es el objetivo principal de la arquitectura, pues es el modelo que generará nuevos datos, para ello, toma como entrada una vector aleatorio de ruido, normalmente siguiendo una distribución gaussiana y devuelve una muestra.

Este vector de ruido es una semilla de aleatoriedad para el generador y, tras el entrenamiento, los modelos GAN aprenden a asignarle un significado a puntos específicos en el espacio latente, de manera que se pueden usar nuevos puntos de este espacio como entrada al modelo generador para producir ejemplos nuevos y diferentes. Este espacio latente es una representación abstracta de las características que no se pueden observar a simple vista de los datos de entrada.

Por tanto, la relación entre el ruido y el espacio latente es clave en este proceso de generación ya que el ruido inicialmente aleatorio se convierte, a través del proceso de entrenamiento, en un espacio latente semánticamente rico que el generador puede utilizar para producir muestras nuevas y significativas.

Por otro lado, el discriminador es algo mucho más simple a lo que ya estamos acostumbrados, toma como entrada muestras y decide si son reales o generadas, las reales provienen del conjunto de datos de entrenamiento y las generadas del generador. Por tanto, el discriminador no es más que un modelo de clasificación normal y corriente, por lo que se podría utilizar cualquiera de los estudiados en la sección de CNN.

2.4.2. Aprendizaje en las GANs

Una vez definimos los modelos que forman parte de esta arquitectura, es indispensable conocer como se realiza su entrenamiento y, por tanto, su aprendizaje. Aunque se ha comentado que este aprendizaje es no supervisado, su entrenamiento esta catalogado como supervisado, ya que, como veremos, el discriminador se entrenará con muestras etiquetadas.

Ambos modelos son entrenados a la vez, de forma que el generador genera un lote de muestras sintéticas, que, junto con otro lote de muestras reales es utilizado para entrenar al discriminador y actualizar para, ahora entrenar al generador en función de como logré engañar al discriminador.

De esta forma, hay una competición continua entre ambos modelos por superar constantemente uno el otro, lo que, según teoría de juegos, ambos son adversarios y están en una situación de juego de suma cero, lo que significa que, mientras el discriminador lo haga bien, sus pesos no se verán muy afectados, pero los del generador si que cambiarán, y el caso contrario, el penalizado resulta ser el discriminador.

Por tanto, para que el generador aprenda la distribución p_g sobre los datos x, se define una distribución previa en las variables de ruido de entrada $p_z(z)$, luego se representa un mapeo al espacio de datos como $G(z;\theta_g)$, donde G es una función diferenciable representada por un perceptrón multicapa con parámetros θ_g. También se define un segundo perceptrón $D(x;\theta_d)$ que, en este caso, produce un solo escalar. $D(x)$ representa la probabilidad de que x provenga de los datos en lugar de p_g, así que se entrena D para maximizar la probabilidad de asignar la etiqueta correcta tanto a ejemplos de entrenamiento como a muestras de G, mientras, a la vez, se entrena G para minimizar $\log(1 - D(G(z)))$ [26].

Es decir, se plantea como un juego minimax donde la función de valor es la siguiente:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))] \tag{2.5}$$

Una vez definido, el proceso de aprendizaje tiene lugar a través del algoritmo de descenso de gradiente alterno. Básicamente, consiste en fijar los parámetros del generador y realizar una única iteración de ascenso de gradiente sobre el discriminador utilizando las imágenes reales y las generadas, tras esto, se fija el discriminador y se entrena el generador para otra única iteración. De esta forma, se entrenan ambas redes en pasos alternos hasta que el generador produce imágenes de buena calidad, el pseudocódigo se puede ver en el Algoritmo 1.

A pesar de que se pueden usar para datos tabulares, su uso principal se encuentra para la generación de imágenes y, por tanto, su combinación con las CNN. Esto es debido, principalmente, a que cuando se definió esta nueva arquitectura las CNN estaban en plena expansión, lo que, sumado a la facilidad para los usuarios de evaluar como de buenas son

Algoritmo 1 Entrenamiento de una Red Generativa Antagónica (GAN)

1: **Entrada:** Conjunto de datos reales X, número de iteraciones N, tamaño del lote B, tasa de aprendizaje α.
2: Inicializar los pesos w_d y w_g de la red discriminatoria D y la red generadora G respectivamente.
3: **for** i desde 1 hasta N **do**
4: **for** j desde 1 hasta B **do**
5: Tomar una muestra de datos reales $\{x^{(1)}, x^{(2)}, ..., x^{(B)}\}$ de X.
6: Generar un lote de datos sintéticos $\{z^{(1)}, z^{(2)}, ..., z^{(B)}\}$ muestreando el espacio latente.
7: Calcular la pérdida del discriminador L_D utilizando los datos reales y sintéticos:

$$L_D = -\frac{1}{B}\sum_{k=1}^{B}\log(D(x^{(k)})) - \frac{1}{B}\sum_{k=1}^{B}\log(1 - D(G(z^{(k)})))$$

8: Actualizar los pesos w_d utilizando el descenso de gradiente:

$$w_d = w_d - \alpha\nabla_{w_d}L_D$$

9: **end for**
10: Generar un nuevo lote de datos sintéticos $\{z^{(1)}, z^{(2)}, ..., z^{(B)}\}$.
11: Calcular la pérdida del generador L_G utilizando los datos sintéticos:

$$L_G = -\frac{1}{B}\sum_{k=1}^{B}\log(D(G(z^{(k)})))$$

12: Actualizar los pesos w_g utilizando el descenso de gradiente:

$$w_g = w_g - \alpha\nabla_{w_g}L_G$$

13: **end for**

las muestras generadas, ya que al final son imágenes, es lo que lo convierte en el principal campo de uso para este tipo de arquitectura por encima de otro tipo de tareas.

2.4.3. cGAN

Además de esto, otra de las grandes ventajas es su versatilidad, destacando sobre todo la alternativa conocida como Redes Generativas Antagónicas Condicionales (cGAN) [27]. Esta arquitectura añade información adicional a ese vector de ruido que recibe al generador, de forma que esa información adicional puede hacer referencia a la clase de la imagen que se quiere generar, por lo que abre la posibilidad a, por ejemplo, a la hora de generar caras humanas, poder elegir entre hombre y mujer.

De la misma forma, el discriminador también recibe esta información, como se ve en la figura 2.12, por lo que no solo aprende a distinguir imágenes reales de generadas, es capaz de reconocer las diferencias entre las distintas clases con las que se entrene y, por tanto, transmitir ese conocimiento al generador.

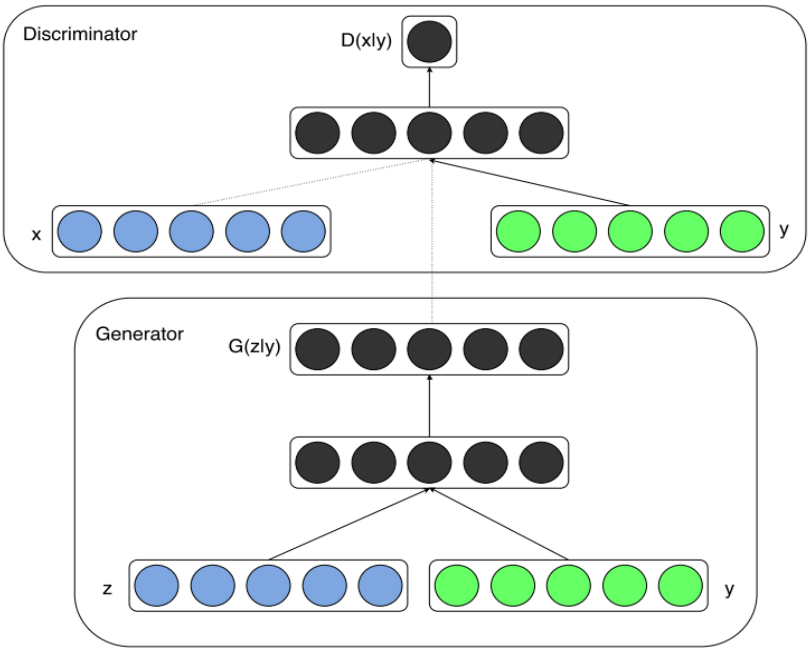


Figura 2.12: cGAN básica. [27]

Además de una simple etiqueta indicando la clase, se puede añadir mucha más información, dando lugar a modelos mucho más complejos como los vistos en la actualidad, donde destacan la creación de nuevas imágenes artísticas, predicción del siguiente fotograma de un vídeo, tareas de transformación de texto a imagen e imagen a imagen, siendo esta última, por ejemplo, convertir una imagen de un perro en un gato.

Pero claro, no todo iban a ser ventajas, estos modelos introducen a la problemática del aprendizaje profundo multitud de problemas, destacando sobre todo la inestabilidad en su entrenamiento, el alto coste computaciones que tienen, tanto de recursos como de tiempo, y el riesgo de sobreajuste, por lo que será necesario prestar atención a estos factores a la hora de elegir una GAN como solución al problema.

2.5. Súper-resolución

Otro de los problemas mencionados era que, en ocasiones, los dermatoscopios no disponen de la suficiente capacidad de aumento lo que resulta en imágenes en baja resolución y, por tanto, perdida de detalles esenciales que pueden mejorar tanto el diagnóstico de un profesional como la decisión de un clasificador.

Es en este problema en el que cobra interés el término de súper-resolución [28], una técnica que tiene como propósito recuperar una imagen en alta resolución a través de una observación de la misma en baja resolución. Esta técnica es considerada como un importante campo de interés durante las últimas décadas en el campo del procesamiento de imágenes, debido a su gran utilidad en entornos como la videovigilancia, imágenes por satélite y, la que es de interés en este trabajo, las imágenes médicas.

Podemos definirlo entonces como el siguiente proceso de degradación:

$$I_{lr} = DFBI_{hr} + N \tag{2.6}$$

Donde I_{hr} es la reconstrucción en alta resolución de I_{lr}, la imagen en baja resolución, D representa el *downsampling*, F la deformación geométrica, B el desenfoque y N el ruido. Por tanto, el objetivo de la súper-resolución es revertir este proceso de degradación y obtener una estimación precisa de la imagen en alta resolución a partir de la imagen en baja resolución.

2.5.1. Métodos basados en interpolación

Debido al auge de la visión por computador, se han desarrollado multitud de métodos diferentes de súper-resolución, donde en primer lugar aparecen los basados en interpolación, debido, principalmente, a que son los más rápidos y simples, ya que se basan únicamente en la comparación de píxeles vecinos.

Interpolación nearest-neighbor

La interpolación nearest-neighbor es el método más sencillo de interpolación, consiste en asignar el valor del píxel más cercano al píxel que se quiere interpolar. Esto significa que no se realiza ninguna operación de suavizado, lo que puede resultar en una calidad de imagen inferior, especialmente en imágenes con patrones finos o curvas suaves. La fórmula para este método es:

$$I(x, y) = I(\lfloor x \rfloor, \lfloor y \rfloor) \tag{2.7}$$

Donde $I(x, y)$ es el valor del píxel en la posición interpolada (x, y) y $\lfloor x \rfloor$ y $\lfloor y \rfloor$ son las coordenadas enteras más cercanas a x e y respectivamente.

Interpolación bilineal

La interpolación bilineal es un método que utiliza los valores de los cuatro píxeles más cercanos para calcular el nuevo valor. Este método realiza una interpolación lineal en

dos dimensiones, primero en una y luego en la otra, de forma que no afecta el orden, su fórmula es:

$$I(x,y) = (1-u)(1-v)I(x_1,y_1) + u(1-v)I(x_2,y_1) + (1-u)vI(x_1,y_2) + uvI(x_2,y_2)$$

Donde $I(x,y)$ es el valor del píxel en la posición interpolada (x,y), $I(x_1,y_1)$, $I(x_2,y_1)$, $I(x_1,y_2)$ y $I(x_2,y_2)$ son los valores de los píxeles vecinos más cercanos y u y v son las fracciones de la distancia entre los puntos x e y respectivamente.

Interpolación bicúbica

La interpolación bicúbica es un método más avanzado que la interpolación bilineal. Utiliza los valores de los píxeles vecinos más cercanos y sus gradientes para calcular el nuevo valor, lo que produce una imagen más suavizada y con menos artefactos que la interpolación bilineal. La fórmula para la interpolación bicúbica es más compleja y se basa en una función cúbica:

$$I(x,y) = \sum_{i=0}^{3}\sum_{j=0}^{3} a_{ij}x^i y^j$$

Donde $I(x,y)$ es el valor del píxel en la posición interpolada (x,y) y a_{ij} son los coeficientes de la función cúbica que se calculan utilizando los valores de los píxeles vecinos y sus gradientes.

2.5.2. Métodos basados en aprendizaje profundo

Dado que las CNN resultaron ser excelentes para tareas como la clasificación y la detección de objetos, se propuso usar las propias capas convolucionales y sus mapas de características para lograr generar una imagen en alta resolución, dada como entrada una en baja resolución. Con este principio, los investigadores han logrado desarrollar una gran variedad de tipos de arquitecturas para abordar esta problemática.

Redes ligeras

Las redes ligeras, están diseñadas específicamente para la súper-resolución, tienen una estructura más simple y menos parámetros que las CNN profundas tradicionales, por lo que utilizan menos capas y filtros, lo que las hace más eficientes computacionalmente, sin embargo, pueden tener un rendimiento inferior en lo que respecta a la calidad de imagen en comparación con las redes más complejas.

Debido a la inmensa variedad de métodos desarrollados por los investigadores, no se abordará en detalle todos los modelos disponibles, un resumen de estos se encuentran en la tabla 2.1, por lo que aquellos finalmente utilizados durante el trabajo si serán analizados con más detalle en secciones posteriores, para evitar un exceso de información en la memoria.

Tabla 2.1: Resumen de métodos de súper-resolución basados en redes ligeras

Método	**Concepto**
SRCNN	Super-Resolution Convolutional Neural Network
VDSR	Very Deep Super-Resolution
DRRN	Deep Recursive Residual Network
DRCN	Deeply-Recursive Convolutional Network
FSRCNN	Fast Super-Resolution Convolutional Neural Network
IMDN	Information Multi-Distillation Network
RFDN	Residual Feature Distillation Network
CARN	Cascading mechanism on A Residual Network
CBPN	Compact Back-Projection Network
MSRN	Multi-Scale residual Network for image Super-Resolution
MCSN	Multi-Scale Channel attention Network

Mecanismos de atención

Los mecanismos de atención se han integrado en las arquitecturas de súper-resolución para permitir que la red neuronal haga más hincapié en ciertas zonas específicas de la imagen que se toma como entrada durante la labor de generación de la imagen de alta resolución, lo que es especialmente útil cuando hay características importantes en la imagen de baja resolución que deben preservarse, lo que permite a la red enfocarse en estas características y evitar la generación de artefactos no deseados.

Un resumen de estos métodos se encuentra en la tabla 2.2.

Tabla 2.2: Resumen de métodos de súper-resolución basados en mecanismos de atención

Método	**Concepto**
RCAN	Residual channel attention network
SENet	Squeeze-and-Excitation Network
SESR	Squeeze and Excitation Super-Resolution
ScSE	Spatial and channel Squeeze and Excitation
SAN	Second-order Attention Neural network
SelNet	Deep Network with Selection units
DRLN	Densely Residual Laplacian Network
RNAN	Residual Non-Local Attention Networks
CSFM	Channel-wise and Spatial Feature Modulation
RAM	Residual Attention Module
RFANet	Residual Feature Aggregation Network
UAN	Upsampling attention network

Aprendizaje residual y denso

El aprendizaje residual y denso son técnicas utilizadas para profundizar las arquitecturas de las CNN y mejorar su capacidad para aprender representaciones más complejas de las imágenes. Estas técnicas son útiles en la súper-resolución porque permiten que el modelo capture características y patrones de diferentes escalas y niveles de abstracción. El aprendizaje residual se basa en el principio de que las diferencias entre la versión de

baja resolución y la versión de alta resolución de una imagen se pueden representar más fácilmente que la imagen de alta resolución completa. El aprendizaje denso, por otro lado, conecta todas las capas de la red entre sí, lo que permite que los mapas de características se transmitan más libremente por toda la red y, de esta forma, aumentar la capacidad para capturar detalles finos en la imagen de alta resolución.

Un resumen de estos métodos se encuentra en la tabla 2.3.

Tabla 2.3: Resumen de métodos de súper-resolución basados en aprendizaje residual y denso

Método	**Concepto**
EDSR/MDSR	Enhanced Deep Super-Resolution/Multi-scale Deep Super-Resolution
SRMDNF	Super-Resolution Multiple Degradation Noise-Free
RDN	Residual Dense Network
SR-DenseNet	Super-Resolution Dense Network
MixNet	Mixed Link Network
ESPCNN	Efficient Sub-Pixel Convolutional Neural Network
KARN	Kernel-Attended Residual Network
D-DBPN	Dense-Deep Back-Projection Networks

Redes Generativas Antagónicas

Las GANs, ya comentadas anteriormente, se han utilizado con éxito en la súper-resolución de imágenes. En este enfoque, el generador toma como entrada una imagen en baja resolución y crea su versión en alta resolución, mientras que el discriminador intenta diferenciar entre que imágenes han sido generadas y cuales son reales, evidentemente, de alta resolución. Este enfoque de entrenamiento ha demostrado ser de los más efectivos para mejorar la calidad general de las imágenes obtenidas en las tareas de súper-resolución.

Un resumen de estos métodos se encuentra en la tabla 2.4.

Tabla 2.4: Resumen de métodos de súper-resolución basados en GANs

Método	**Concepto**
SRGAN	Super-Resolution Generative Adversarial Network
ZSSR	Zero-Shot Super-Resolution
ESRGAN	Enhanced SRGAN
RaGAN	Relativistic average GAN
Rank-SRGAN	Super-Resolution Generative Adversarial Networks with Ranker
Unpaired GAN	Unpaired SR using a Generative Adversarial Network

Como se puede observar, existe una gran variedad de aproximaciones y modelos que pueden ser utilizados para las tareas de súper-resolución, por lo que investigar cuales son los mejores y los más utilizados en el sector de la detección del melanoma será crucial para obtener buenos resultados.

CAPÍTULO 3

Materiales y metodología

Esta sección contendrá información acerca de la metodología de trabajo seguida durante el desarrollo de este trabajo, además de las fases de las que se ha compuesto. De la misma forma, contendrá todo el material utilizado durante el mismo, como conjuntos de datos y modelos de aprendizaje profundo.

3.1. Metodología y fases de trabajo

Aplicar una metodología de trabajo adecuada es algo esencial para el desarrollo de un proyecto, sobre todo en el sector de la ingeniería del software, donde una decisión como esta puede tener una gran repercusión en la probabilidad de éxito del proyecto. Entonces, es por esta razón por lo que merece la pena investigar las distintas metodologías que se aplican en el entorno profesional y valorar cual puede tener más sentido para cada proyecto, atendiendo a los objetivos del mismo claro.

En el caso de este trabajo se va a utilizar el procedimiento ampliamente utilizado el sector de la minería y ciencia de datos, la metodología *CRoss Industry Standard Process for Data Mining* (CRISP-DM) [29], eso sí, evidentemente, orientada a un proyecto de investigación en solitario.

Esta metodología ofrece una visión general del ciclo de vida de un proyecto de ciencia de datos, la cual se divide en las siguientes fases:

1. **Entendimiento del negocio**: esta primera fase, el objetivo es comprender completamente los objetivos y requisitos del negocio, en su defecto, del ámbito de aplicación.
2. **Entendimiento de los datos**: se recopilan los datos y se explora, de forma que se realiza una exploración de los datos para conocerlos.
3. **Preparación de los datos**: tras conocer los datos, se realizan los métodos de limpieza, transformación y preprocesamiento necesarios para abordar el problema.
4. **Modelado**: en esta fase se seleccionan y aplican distintos modelos de aprendizaje que se adecuen a los datos y, por tanto, permitan resolver el problema.
5. **Evaluación**: tras generar los modelos, se someten a procesos de evaluación con

distintas métricas para comparar cuales de los modelos obtenidos obtienen un rendimiento mayor.

La metodología original contiene una fase adicional de despliegue la cual no tiene sentido para este proyecto, motivo por el que se omite dicha fase.

Como cualquier metodología de desarrollo software, se trata de un proceso iterativo y cíclico, pues las fases pueden no llevarse a cabo de forma lineal y repetirse varias veces a lo largo del proyecto en función de las necesidades de este.

En cuanto a la distribución de trabajo, se ha dedicado un total de 14 semanas de trabajo para completar este trabajo, de forma que cada semana consta de 25 horas de dedicación exclusiva al proyecto, lo que suma un total de 350 horas, por lo que se cubren con éxito las horas de dedicación estipuladas para el mismo.

Respecto a las fases, cada una de las que se consideran prácticas representa un ciclo de vida de CRISP-DM, ya que suponen afrontar un nuevo conjunto de datos, nuevos modelos y nuevas métricas de evaluación. La división de las mismas y su temporización es la siguiente:

1. **Revisión de estado del arte** (2 semanas): en esta primera fase se realizó una exhaustiva revisión del estado del arte, que se corresponde con lo resumido en la sección anterior, donde se han consultado además los distintos conjuntos de datos disponibles, así como trabajo relacionado que puede ser de interés.

2. **Experimentación con la súper-resolución** (3 semanas): esta fase se corresponde con la selección de un conjunto de datos que tenga sentido para poder entrenar un modelo de súper-resolución, la búsqueda de los repositorios de los modelos elegidos para evaluar y su instalación, así como su entrenamiento y posterior evaluación.

3. **Experimentación con la generación de datos sintéticos** (3 semanas): durante esta fase, se ha realizado lo mismo que la anterior pero para el caso de la generación de datos sintéticos. Además, se ha experimentado con la implementación propia de un modelo siguiendo una arquitectura concreta, DCGAN.

4. **Experimentación con la aplicación de técnicas a la clasificación** (4 semanas): se corresponde con la última fase de experimentación, en ella se han aplicado las anteriores técnicas a otro conjunto de datos para ver si mejoran los resultados de un clasificador. Esto ha supuesto aplicar mucha fase de prueba y error y consulta de otros investigaciones relacionadas hasta obtener unos resultados interesantes que logren algún tipo de aportación relevante.

5. **Redacción de la memoria y preparación de la presentación** (2 semanas): por último, se ha documentado todo el trabajo realizado en la presente memoria y se ha elaborado la presentación que se llevará a cabo como defensa del proyecto.

A pesar de que se diferencian claramente las 5 fases, algunas de ellas han estado solapadas, ya que, por ejemplo, durante la fase de experimentación con la aplicación de técnicas a la clasificación, ha surgido la necesidad de volver a entrenar los modelos de generación de datos, así como experimentar con distintas resoluciones, por lo que no están totalmente separadas unas de otras, algo de esperar en un proyecto software. Además de esto, se han llevado a cabo reuniones mensuales con el tutor para comentar los progresos y dificultades encontrados, de forma que se planteaban posibles soluciones o mejoras.

3.2. Recursos hardware

Una vez está definida toda la base teórica, así como la forma de trabajar, es momento de iniciar el trabajo de experimentación y, por tanto, entrenamiento y evaluación de modelos. Tal y como se comentó en la sección de marco teórico, las redes neuronales aprovechan la arquitectura de las GPU de forma que, en la actualidad, para entrenar un modelo medianamente complejo como los que se van a utilizar en este trabajo, es necesario tener una de estar tarjetas gráficas.

En concreto, aprovechan la tecnología CUDA de NVIDIA, que permite realizar cálculos de manera eficiente en paralelo en las GPU. Esta tecnología es fundamental para acelerar el proceso de entrenamiento de modelos de redes neuronales, ya que permite distribuir las tareas de cálculo entre los miles de núcleos y, así, mientras que el entrenamiento de un modelo complejo podría llevar días o semanas en una CPU convencional, con el uso de GPU y CUDA, este tiempo se reduce significativamente, permitiendo experimentar con diferentes arquitecturas de red, funciones de pérdida y conjuntos de datos de manera más ágil.

Esto, sumado a que NVIDIA ofrece un amplio soporte para las principales librerías de aprendizaje profundo como Tensorflow, Pytorch y cuDNN y a que ofrecen una amplia gama de las mismas con distintas prestaciones y presupuestos, lo convierten en la opción más popular en el ámbito de la inteligencia artificial.

En concreto, para este proyecto se ha utilizado un servidor destinado para estudiantes, propiedad del grupo de investigación de Inteligencia Computacional y Análisis de Imágenes (ICAI), equipado con los siguientes recursos:

Tabla 3.1: Especificaciones del hardware utilizado

Componente	**Especificación**
Procesador	Intel® Core™ i5-7400 @ 3.00 GHz
GPU	NVIDIA GeForce RTX 3080 10 GB
RAM	8 GB
Disco Duro	100 GB

Con estos recursos ha sido posible posible llevar a cabo el entrenamiento y la evaluación de los distintos modelos que se presentarán más adelante, además, es importante destacar que se tratan de recursos compartidos por varios alumnos, a excepción del disco duro, para el que cada uno dispone una partición de 100 GB, por lo que, en algunos casos, los tiempos de entrenamiento han sido algo mayores de lo esperado debido a esa compartición de recursos y la saturación de la memoria RAM.

Respecto a la GPU, con 10 GB de memoria gráfica ha sido suficiente para entrenar todos los modelos, para lograrlo, ha sido necesario modificar el tamaño de lote de algunos de los modelos para que la GPU pueda procesarlo de manera efectiva, pero se entrará más en detalle en cada modelo.

3.3. Conjuntos de datos

Una de las labores esenciales para llevar a cabo un proyecto de ciencia de datos es la recopilación y selección de uno o varios conjuntos de datos adecuados, ya que estos son la materia prima sobre la que van a aprender los modelos, por lo que la calidad y la relevancia de los mismos va a influir directamente en el resultado del aprendizaje.

En el ámbito del cáncer de piel y el melanoma, la principal fuente de datos públicos es la *International Skin Imaging Collaboration* (ISIC), una organización dedicada a la recopilación y distribución de imágenes dermatoscópicas de lesiones cutáneas, teniendo una de las base de datos más grandes y completas en este campo, con miles de imágenes de alta resolución que abarcan una amplia variedad de casos clínicos.

Esta organización dispone de varios conjuntos de datos que han sido utilizados como banco de pruebas y, también, como competiciones que cubren tanto la clasificación de lesiones cutáneas como la detección y localización de las mismas a través de modelos especializados.

Dado que se pretenden realizar 3 tareas, las cuales tienen objetivos diferentes, se han seleccionado 3 conjuntos de datos de forma que cada uno de los mismos esta orientado a cubrir una serie de requisitos específicos para cada tarea y así poder obtener resultados.

Además, es importante que no se crucen, por ejemplo, no tiene sentido evaluar la mejora en la clasificación usando súper-resolución habiendo entrenado este último con las mismas imágenes que se usan para clasificar, o comparar la mejora de añadir muestras sintéticas habiendo entrenado una GAN con imágenes nuevas para el modelo de clasificación.

3.3.1. HAM10000

El conjunto de datos *Human Against Machine with 10000 training images* (HAM10000) [30] es una colección de 10015 imágenes dermatoscópicas recopiladas durante más de 20 años por el departamento de dermatología de la Universidad de Medicina de Viena, en Austria, y el profesor Cliff Rosendahl, en Queensland, Australia.

Algunas de las imágenes estaban almacenadas como diapositivas tradicionales, pues provenían de una época anterior a la aparición de las cámaras digitales, por lo que fue necesario escanearlas y digitalizarlas. Además, se les aplicó una serie de preprocesamientos, como centrar las imágenes, recortarlas y aplicar una corrección manual del histograma de la imagen para mejorar el contraste y el color.

Además de esto, como se trata de un conjunto de datos etiquetado, se le realizó una validación técnica con el objetivo de mantener la certeza de que las etiquetas de los datos eran correctas. Esta validación se llevó a cabo a través de 4 tipos de etiquetado, hispatologia, seguimiento, microscopía confocal de reflectancia y consenso entre expertos.

Este conjunto de datos no se corresponde únicamente con imágenes de melanoma, esta es tan solo una de sus clases, la cual se distribuye tal y como se muestra en la tabla 3.2. Como existen multitud de clases que se desconocen, se van a comentar brevemente cada una de ellas y se mostrará un ejemplo de las mismas, para que se pueda comprobar la similitud y complejidad de distinción entre varias de estas.

Tabla 3.2: Distribución de clases del dataset HAM10000

Clase	Cantidad
akiec	327 (3.3 %)
bcc	514 (5.1 %)
bkl	1099 (11 %)
df	115 (1.1 %)
mel	1113 (11.1 %)
nv	6705 (67 %)
vasc	142 (1.4 %)

Queratosis actínica y Carcinoma intraepitelial (akiec)

Son variantes comunes del carcinoma de células escamosas que pueden tratarse localmente sin cirugía. Algunos autores las consideran precursoras del carcinoma de células escamosas y no como carcinomas reales, sin embargo, hay acuerdo en que estas lesiones pueden progresar al caso invasivo, que generalmente no está pigmentado. Ambos tipos de neoplasias muestran comúnmente descamación en la superficie y carecen de pigmento. Un ejemplo se puede ver en la figura 3.1.

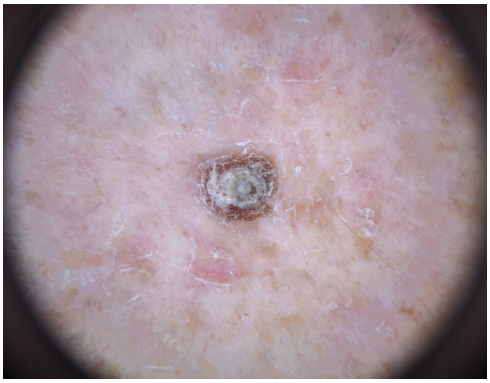


Figura 3.1: Muestra de akiec. [30]

Carcinoma basocelular (bcc)

Es una variante común del cáncer de piel epitelial que rara vez hace metástasis pero crece destructivamente si no se trata, además aparece en diferentes variantes morfológicas (planas, nodulares, pigmentadas, quísticas). Se ve en la figura 3.2

Queratosis benigna (bkl)

Es una clase genérica que incluye queratosis seborreicas ("verruga senil"), lentigos solares y queratosis liquenoides (LPLK). Los tres subgrupos pueden verse diferentes dermatoscópicamente, pero se agrupan juntos porque son similares biológicamente y se informan a menudo bajo el mismo término genérico histopatológicamente. La figura 3.3 muestra un ejemplo.

Dermatofibroma (df)

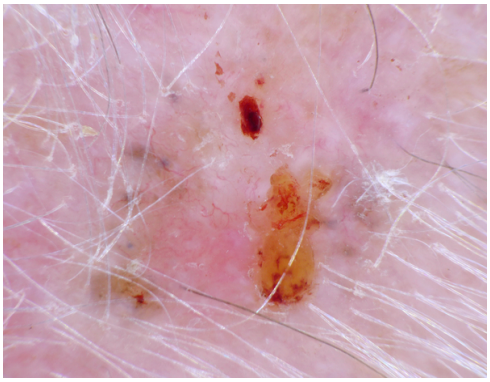


Figura 3.2: Muestra de bcc. [30]

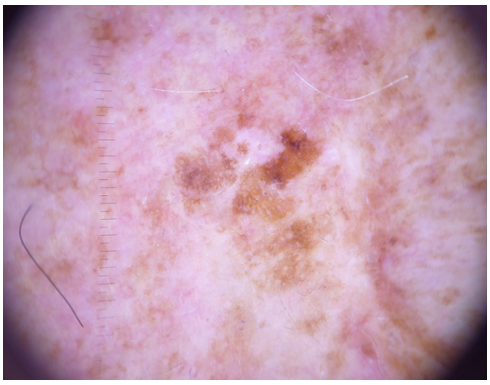


Figura 3.3: Muestra de bkl. [30]

Es una lesión cutánea benigna considerada como una proliferación benigna o una reacción inflamatoria al trauma mínimo. La presentación dermatoscópica más común es líneas reticulares en la periferia con una mancha blanca central que denota fibrosis, tal como se ve en la figura 3.4.

Figura 3.4: Muestra de df. [30]

Nevus melanocíticos (nv)

Son neoplasias benignas de melanocitos, coloquialmente conocidos como lunares, y aparecen en multitud de variantes, que están incluidas en el conjunto de datos. A diferencia del melanoma, generalmente son simétricos con respecto a la distribución de color y estructura, véase la figura 3.5.

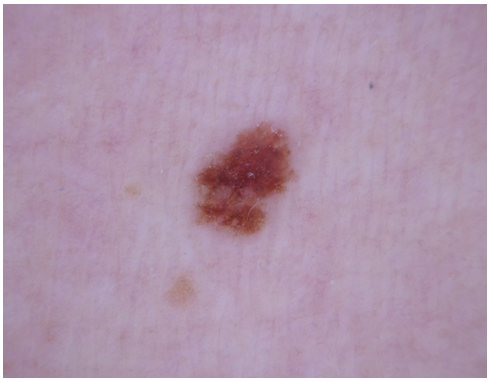


Figura 3.5: Muestra de nv. [30]

Melanoma (mel)

Es una neoplasia maligna derivada de melanocitos que puede aparecer en diferentes variantes, siendo un ejemplo el de la figura 3.6. Tal y como se comentó, si se extirpa en una etapa temprana, puede curarse mediante una simple cirugía. El conjunto de datos incluye todas las variantes de melanoma, incluido el melanoma in situ, pero excluye el melanoma no pigmentado, subungueal, ocular o mucoso.

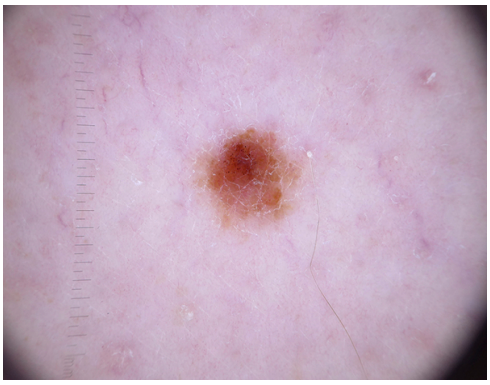


Figura 3.6: Muestra de mel. [30]

Lesiones vasculares (vasc)

Se corresponden con simples lesiones vasculares, en el conjunto de datos se recogen desde angiomas hasta angiokeratomas y granulomas piógenos, además del sangrado. Los angiomas se caracterizan dermatoscópicamente por su color rojo o púrpura y estructuras sólidas y bien circunscritas conocidas como coágulos o lagunas rojas, como se presenta en la figura 3.7.

Como se puede comprobar, este conjunto de datos ofrece una amplia variedad de muestras de distintas clases, con una resolución de 600x450, estas dos han sido las principales razones por las que ha sido utilizado para la evaluación de los modelos de generación de datos sintéticos.

En primer lugar, al tener una gran cantidad de clases, algunas desbalanceadas, lo hace un conjunto de datos, a priori, complejo para clasificación, alejado del modelo de clasificación binaria (melanoma/benigno) que se contemplaba al inicio del proyecto, pues habría que desarrollar un marco de trabajo alrededor de esto, de forma que se equilibrasen, además

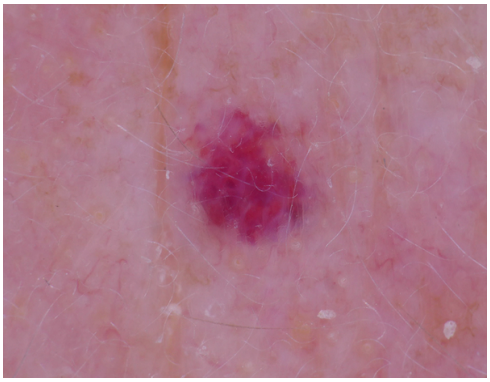


Figura 3.7: Muestra de vasc. [30]

de agruparlas para convertirlo en un modelo binario. Si bien es cierto que se podrían haber entrenado GANs para solventar este problema, es algo que se planteó, pero se escapa del alcance del proyecto, pues el objetivo del mismo se centra en el caso de los melanomas, y esto supondría un número de entrenamientos mucho superior que se escaparía de los límites del mismo.

Por otro lado, la resolución es adecuada para capturar detalles finos y sutiles, pero existen conjuntos de datos con mayor resolución. Estos dos descartes, fueron los principales motivos por los que se escogió este modelo para la evaluación de las GANs, además de ser uno de los conjuntos de datos de referencia en el estado del arte [4].

3.3.2. BCN20000

El conjunto de datos BCN20000 [31] es una colección de 19424 imágenes de alta resolución recopiladas por el Hospital Clínico de Barcelona, correspondientes a 5583 lesiones diferentes tomadas entre 2010 y 2016.

En este conjunto de datos contiene las mismas clases que el HAM10000, a excepción del caso de la queratosis benigna, que la trata en dos subclases, queratosis seborreica y actínica, ambas benignas. De nuevo, se ha realizado una verificación, en este caso mediante hispatologia, microscopía confocal de reflectancia y consenso. Tanto la distribución de los datos como muestras de las mismas pueden verse en las figuras 3.8 y 3.9

En este caso, las principales diferencias con el HAM10000 es que ahora las imágenes tienen una resolución de 1024x1024 y el número de estas es casi el doble, por lo que ha sido seleccionado como conjunto de datos para evaluar la súper-resolución.

Lo más evidente, al tener una resolución superior, la cantidad de detalles sutiles y de grano fino de las imágenes es bastante superior a las de HAM10000, lo que, obviamente, lo hace un conjunto de datos más interesante para la súper-resolución.

Además, el modelo de súper-resolución es el único que no tiene que volverse a entrenar para la fase de clasificación, es decir, en el caso de las GANs, una vez se evalué, el modelo puede descartarse ya que no tiene sentido usar datos que no ha visto el modelo de clasificación en una GAN, pues estaría introduciendo imágenes totalmente nuevas y no serían únicamente sintéticas. Esto hace que este modelo sea idóneo para entrenarlo con un gran número de imágenes, ya que, aunque el entrenamiento sea algo más largo, realmente se aprovecharán los resultados obtenidos.

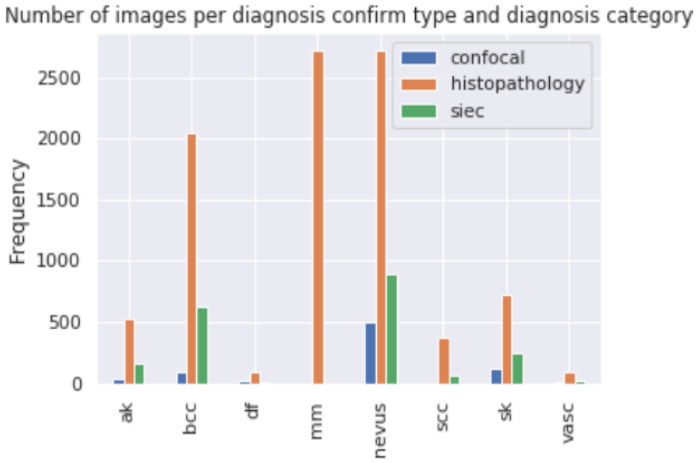


Figura 3.8: Distribución de clases de BCN20000. [31]

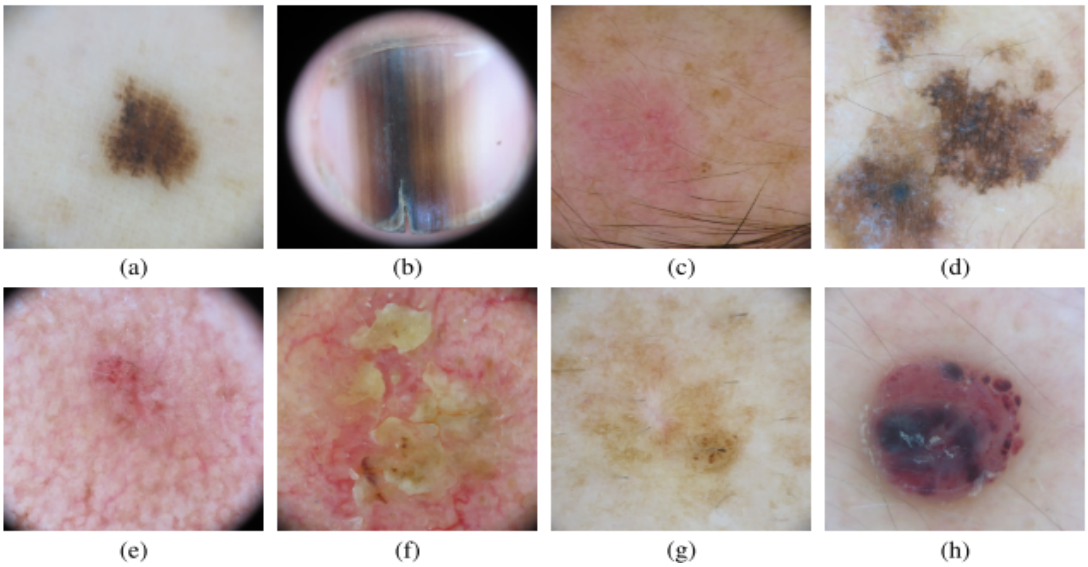


Figura 3.9: Muestras del conjunto de datos BCN20000 correspondientes a (a) nevus, (b) melanoma, (c) carcinoma de células basales, (d) queratosis seborreica, (e) queratosis actínica, (f) carcinoma de células escamosas, (g) dermatofibroma y (h) lesión vascular. [31]

3.3.3. Conjunto de 10000 imágenes de melanoma

Por último, el conjunto de 10000 imágenes de melanoma [32] es un conjunto de datos que ha sido recopilado de los desafíos del ISIC de 2018, 2019 y HAM10000, de forma que está balanceado y esta agrupado en dos clases, melanoma o benigno [33]. Además, esta dividido por el propio creador en entrenamiento y test, lo que lo hace perfecto para evaluar la clasificación.

En este caso, las imágenes han sido recortadas a la resolución de 300x300 y se componen de un total de 10600, 9500 para entrenamiento y 1100 para test, de las que 5000 pertenecen a la clase benigna y 4600 a melanoma, por lo que está prácticamente balanceado.

Todo este conjunto de características lo hace un conjunto de datos perfecto para evaluar el rendimiento de la clasificación. Uno de los problemas que se podrían dar es que contiene algunas imágenes del HAM10000, y se estarían cruzando entre los distintos conjuntos, pero esto realmente no supone ningún problema real, ya que la GAN va a ser entrenada

de nuevo con este último conjunto, de forma que no se va a reutilizar su entrenamiento, por lo que puede ser utilizado sin problema.

3.4. Modelos de súper-resolución

Una vez seleccionados los distintos conjuntos de datos, una labor importante del proyecto es elegir que modelos evaluar, por lo que se consultó el estado del arte y se concluyó que los 3 modelos más usados en el campo de las imágenes de enfermedades en la piel son EDSR, SRGAN y ESRGAN [34], por lo que en la presente sección se discutirá sobre los mismos y sus detalles más concretos.

3.4.1. EDSR

Enhanced Deep Residual Networks for Single Image Super-Resolution (EDSR) [35] es un modelo de súper-resolución presentado en 2017 y ganador del desafío de NTIRE del mismo año, demostrando un rendimiento superior a los modelos del estado del arte hasta el momento.

Este modelo, tal y como su nombre indica, se basa en el concepto de redes residuales, ya comentado con anterioridad. Más en concreto, se basa en intentar mejorar otro modelo propuesto anteriormente, SRResNet, el cual presentaba graves problemas de rendimiento, de forma que ahorrando aproximadamente un 40 % de memoria se obtuvieron mejores resultados.

Tanto SRResNet como EDSR siguen el mismo principio básico, adaptar la arquitectura conocida de ResNet, usada en clasificación, para ser usada en súper-resolución. Para ello, se realizaron las modificaciones que se pueden observar en la figura 3.10. Básicamente, se eliminaron las capas de normalización, lo que limita la flexibilidad de rango de los valores de las neuronas, esto fue demostrado experimentalmente que mejoraba los resultados obtenidos, además, se eliminó la función ReLU tras la suma del residuo.

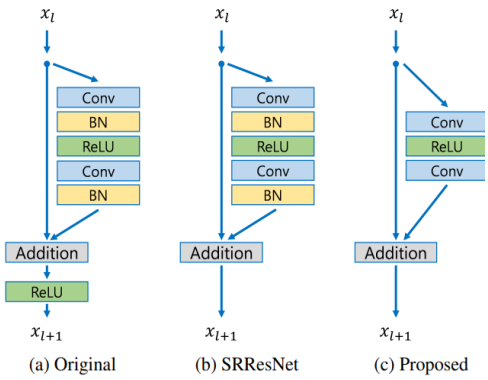


Figura 3.10: Comparación de los bloques residuales de ResNet, SRResNet y EDSR. [35]

Por tanto, teniendo en cuenta que el rendimiento era mayor y la carga computacional, menor, era posible hacer más complejo el modelo, de forma que era posible añadir más bloques residuales. Tanto que mientras SRResNet tenía 16 bloques, 64 filtros y 1,5 millones de parámetros, con EDSR es posible acumular 32 bloques, 256 filtros y 43 millones de parámetros, de forma que su arquitectura final resulta en la mostrada en la figura 3.11.

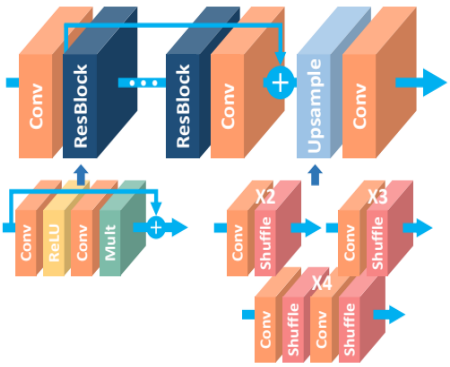


Figura 3.11: Arquitectura de EDSR. [35]

Esta arquitectura se trata de la versión a una única escala, ya que los investigadores también propusieron un diseño, muy similar, en el que se generan varias salidas en escalas diferentes, pero no ha sido utilizada para el proyecto. Además, como se verá en la siguiente sección, el entrenamiento de este modelo también puede realizarse haciendo uso de una arquitectura GAN.

3.4.2. SRGAN

Super-Resolution Generative Adversarial Network (SRGAN) [13] es otra propuesta de modelo de súper-resolución, también publicada en 2017, que fue, de las existentes en el estado del arte hasta el momento, la que mejor puntuación de opinión media (MOS) obtuvo, llegando a estar muy cerca de las imágenes reales en alta resolución.

Tal y como pasaba con el modelo anterior, este modelo se basa en los bloques residuales, de hecho, también es un intento de mejorar SRResNet, pues lo que hace realmente es darle una arquitectura de tipo GAN a este modelo concreto, por lo que implica la creación de un discriminador.

Durante el entrenamiento, las imágenes en baja resolución I_{lr} se obtienen aplicando un filtro gaussiano a la imagen en alta resolución I_{hr}, tras la que se le realiza un *downsampling* con factor r, por lo que se puede definir I_{lr} como un tensor $W \times H \times C$ donde W representa el ancho, H la altura y C como el número de canales o colores, y I_{hr} como $rW \times rH \times C$. Por tanto, como nos encontramos ante una arquitectura GAN, definimos el generador G_{θ_G} como un conjunto de pesos θ_G cuyo objetivo es calcular:

$$\theta_G = argmin_{\theta_G} \frac{1}{N} \sum_{n=1}^{N} \mathcal{L}^{SR}(G_{\theta_G}(I_{LR_n}), I_{HR_n}) \tag{3.1}$$

Donde $\mathcal{L}^{SR}$ es una función de pérdida perceptual definida por:

$$\mathcal{L}^{SR} = \mathcal{L}_X^{SR} + 10^{-3}\mathcal{L}_{Gen}^{SR} \tag{3.2}$$

En esta formula se definen por un lado, la perdida de contenido, $\mathcal{L}_X^{SR}$, que se define como el error cuadrático medio por píxel, y la perdida adversaria, $\mathcal{L}_{Gen}^{SR}$, la cual no es más que la perdida del generador tal y como se definió en el algoritmo 1 de la sección de la

arquitectura GAN. Además, tal y como se vio, es posible definirlo como un juego minimax cuya función de valor es muy similar a la 2.5:

$$min_{\theta_G} max_{\theta_D} E_{I_{HR} \sim p_{train}(I_{HR})}[logD_{\theta_D}(I_{HR})] + E_{I_{LR} \sim p_G(I_{LR})}[log(1 - D_{\theta_D}(G_{\theta_G}(I_{LR})))] \quad (3.3)$$

Para lograrlo, se propuso la arquitectura mostrada en la figura 3.12, la cual se compone del discriminador, el cual es básicamente SRResNet, y un discriminador, que destaca por la aparición de la función de activación Leaky ReLU, la cual es una modificación donde los valores negativos no son siempre cero, es decir, su pendiente no es horizontal como con ReLU, pero es realmente pequeña.

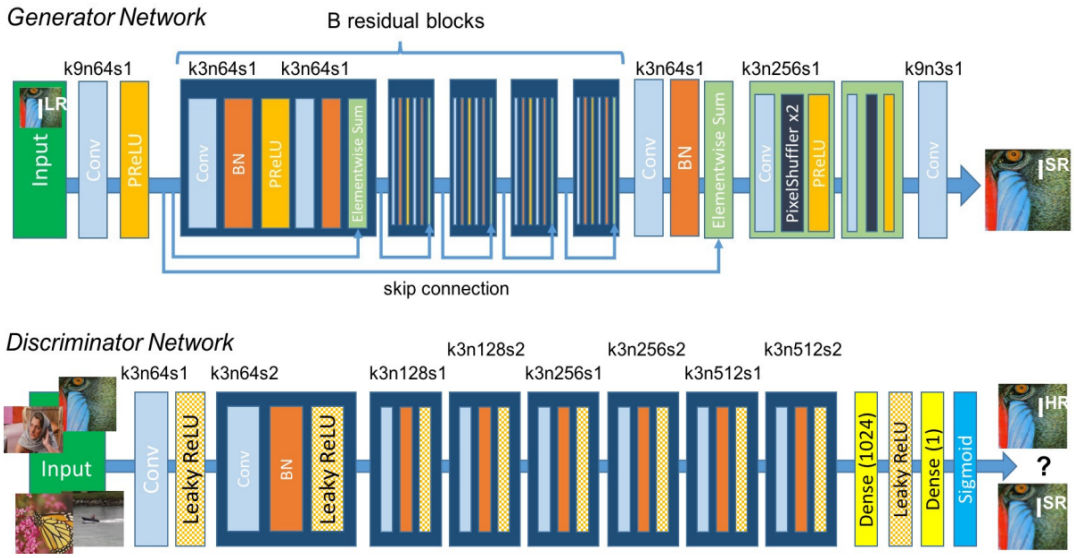


Figura 3.12: Arquitectura de SRGAN donde k hace referencia al tamaño del kernel, n el número de filtros y s el *stride*. [13]

3.4.3. ESRGAN

Enhanced Super-Resolution Generative Adversarial Network (ESRGAN) [36] es el último de los modelos de súper-resolución seleccionados, publicado en 2019 y obteniendo el primer lugar en el desafío PIRM2018-SR, ofreciendo un rendimiento espectacular nunca visto hasta el momento.

De la misma forma que los dos modelos vistos hasta el momento, surgen de la idea de mejorar otro, en este caso, como es evidente, SRGAN. Para lograrlo, introduce 3 cambios principales a la hora de diseñar la red, la creación de un nuevo bloque de capas, el uso de un discriminador relativista y la actualización de la función de perdida perceptual.

Residual-in-Residual Dense Block (RRDB)

El primer cambio, tal y como se ha comentado, consiste en modificar el bloque residual visto en SRGAN y convertirlo en el de la figura 3.13, al que llaman *Residual-in-Residual Dense Block*.

En primer lugar, tal y como sucedía en el caso de EDSR, se eliminan las capas de normalización para mejorar el rendimiento y, además, según han podido comprobar también reduce la aparición de artefactos en los resultados. Por otro lado, entra en juego el *residual-in-residual*, que consiste en aplicar el aprendizaje por residuo en distintos niveles, incluso dentro de los propios bloques.

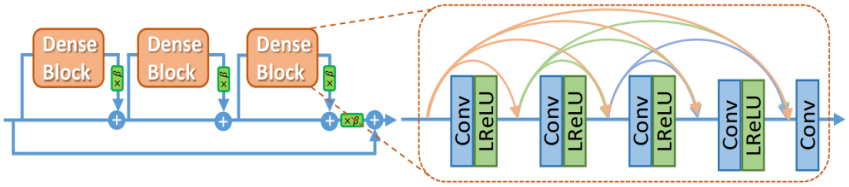


Figura 3.13: Diseño del RRBB. [36]

Discriminador relativista

Dado que se mejoró el generador, los investigadores también pensaron en mejorar el discriminador, por lo que introdujeron el concepto de discriminador relativista. Mientras que el discriminador tradicional estima la probabilidad de que la imagen sea real o generada, el relativista estima la probabilidad de que una imagen real sea relativamente más realista que una falsa.

El principio detrás de este es que, mientras SRGAN tan solo se tiene en cuenta el gradiente de la información generada, en ESRGAN entran en juego ambos, los del generado y los reales. Para ello, se define el discriminador relativista RaD de la siguiente manera:

$$D_{Ra}(x_r, x_f) = \sigma(C(x_r) - \mathbb{E}_{x_f}[C(x_f)]) \tag{3.4}$$

Donde σ representa la función sigmoide, C_x es la salida sin transformar del discriminador y $\mathbb{E}_{x_f}[.]$ la operación de calcular la media de todos los datos generados del lote. Esto supone cambios tanto en la perdida del discriminador como la del generador, donde se incluyen el nuevo discriminador y la función $\mathbb{E}_{x_f}[.]$.

A nivel práctico, los investigadores han logrado verificar que este cambio ayuda a aprender las texturas con más detalle y mejorar los bordes de forma significativa.

Función de perdida perceptual

Se desarrolló una pérdida perceptual más efectiva al considerar características antes de la activación en lugar de después, superando así limitaciones del enfoque convencional y resolviendo los dos principales problemas de SRGAN. La primera se trata de que las activaciones de las características están muy dispersas, lo que hace se traduce en una supervisión más débil y, por tanto, en un rendimiento menor. El otro surge cuando al calcular la función tras la activación causa que la reconstrucción de la iluminación se realice de forma incorrecta.

De esta forma, la función de perdida del generador queda así:

$$L_G = L_{\text{percep}} + \lambda L_{\text{Ra}}G + \eta L_1 \tag{3.5}$$

Donde $L_1 = \mathbb{E}_{x_i}[|G(x_i) - y|]$ es la pérdida de contenido que evalúa la distancia de norma 1 entre la imagen recuperada $G(x_i)$ y la verdadera y. λ y η son coeficientes para equilibrar la función.

Real-ESRGAN

A pesar de esto, esta no ha sido la versión del modelo utilizada, más en concreto, se ha usado una versión actualizada, Real-ESRGAN [37], publicada en 2021, la cual permite regenerar mejor degradaciones en las imágenes gracias a otra actualización del discriminador.

Mientras que la original usa un discriminador basado en la arquitectura VGG, en Real-ESRGAN se usa un discriminador con arquitectura U-Net acompañado de normalización espectral como el mostrado en la figura 3.14. El diseño de la arquitectura U-Net proporciona al generador una retroalimentación píxel a píxel mucho más detallada, lo que mejora los resultados, pero causa un problema, desestabiliza el entrenamiento. Es por esto por lo que se le aplica la regularización mediante normalización espectral, con lo que se obtiene un buen equilibrio entre mejorar los detalles y evitar artefactos en la imagen.

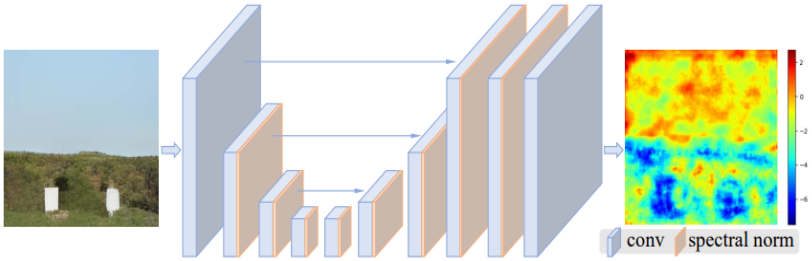


Figura 3.14: Arquitectura del discriminador U-Net con normalización espectral. [37]

3.5. Modelos de generación de datos sintéticos

Con el caso de la generación de datos sintéticos se ha seguido el mismo proceso, se ha revisado el estado del arte y se ha concluido que, en este caso, las arquitecturas más popularizadas son DCGAN y StyleGAN.

3.5.1. DCGAN

La arquitectura *Deep Convolutional Generative Adversarial Network* (GAN) [38] es una fusión entre redes convolucionales y GANs con el objetivo de lograr aprender la representación de imágenes, es decir, generar nuevas imágenes realistas a partir de datos de entrada, como ruido.

Muchos autores han intentado escalar las GANs para el uso de redes convolucionales más profundas, pero la mayoría han fracasado, lo que motivó a los principales investigadores a realizar una exploración con multitud de modelos disponibles en el estado del arte. Esta exploración tuvo como resultado una familia de arquitecturas, DCGAN, la cual demostró tener un entrenamiento más estable con mayores resoluciones y modelos más profundos. Esta arquitectura define una guia de aspectos a tener en cuenta a la hora de desarrollarse:

- **No pooling**: reemplazar las funciones de *pooling* por convoluciones con *stride* permite aprender a la red su propio *upsampling*.
- **Normalización**: aplicar normalización por lotes ayuda a solventar problemas que se pueden dar por una incorrecta inicialización de los pesos y ayudar al flujo del gradiente cuando se traten de modelos más profundos.
- **Eliminar capas totalmente conectadas**: han logrado demostrar que eliminar estas capas al inicio del generador y al final del discriminador (entre la convolución y la entrada/salida) logra mejorar el rendimiento comparado con el uso de las mismas.
- **Uso de ReLU en el generador**: se usa ReLU en todas las capas del generador, excepto en la salida, que se usa tanh. Esto se traduce en un aprendizaje más rápido y una mejor cobertura del espacio de color de la distribución de los datos de entrenamiento.
- **Uso de Leaky ReLU en el discriminador**: se propone usar Leaky ReLU en todas las capas del discriminador, ya que se ha comprobado que ofrece mejores resultados cuando se usan imágenes de mayor resolución.

Como primera aproximación, los autores definieron la arquitectura mostrada en la figura 3.15, cuya resolución de salida es de 64x64, la cual se entrenó con el optimizador Adam. La entrada z representa el ruido, es decir, el espacio latente, tal y como se comentó en la sección de GANs.

En principio, no es necesario aplicar ningún preprocesamiento a las imágenes, más allá de recortarlas a la resolución necesaria y escalarlas al rango $[-1, 1]$, para que tenga sentido aplicar la función tanh, algo importante ya que si no se realiza puede afectar gravemente a los resultados del entrenamiento.

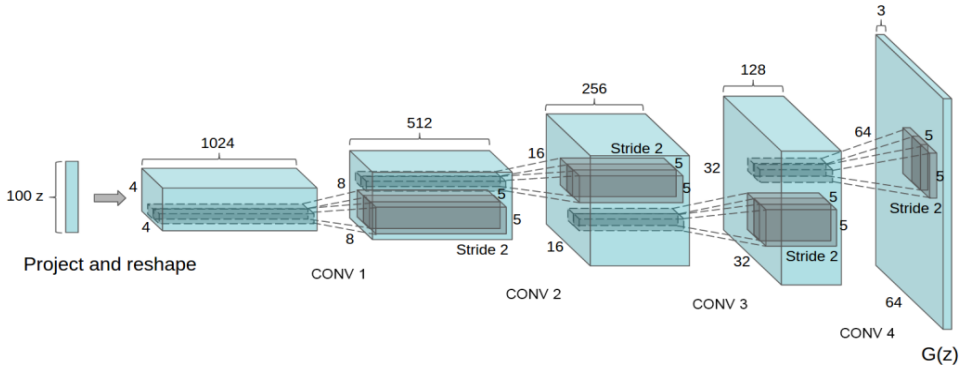


Figura 3.15: Arquitectura DCGAN. [38]

Por tanto, esta arquitectura será utilizada y adaptada a las necesidades del proyecto, principalmente será necesario aumentar la resolución de salida y solventar algún problema típico durante el diseño de GANs para la generación de datos sintéticos, pero la adaptación como las distintas soluciones adoptadas se comentarán en la sección correspondiente.

3.5.2. StyleGAN

Style-based GAN (StyleGAN) [39] es una arquitectura desarrollada por investigadores de NVIDIA con el objetivo de tener un mayor control sobre el proceso de síntesis de

las imágenes, publicada en el 2019, que no solo permitió lo anterior, también mejoró la calidad de las imágenes generadas hasta el momento por otros modelos en el estado del arte.

Esta nueva arquitectura propuesta se basa en la literatura de transferencia de estilos, de forma que modificando el generador es posible ajustar el estilo en cada capa convolucional, es decir, controlar las características de la imagen, lo que combinado con la inserción de ruido, permite separar las de alto nivel, como pueden ser la postura y el sexo en el caso de la generación de caras humanas, de las estocásticas, pecas y pelo en el mismo ejemplo, de forma que puedan ser combinadas e interpoladas de forma más intuitiva.

Para ello, introduce el concepto de generador basado en el estilo, mientras que en las GANs tradicionales el ruido es directamente introducido en la red, en StyleGAN se inicia con una constante, mientras que el ruido z perteneciente al espacio latente $\mathcal{Z}$, a través de un red de mapeo no lineal, es decir, una sucesión de capas totalmente conectadas $f : \mathcal{Z} \longrightarrow \mathcal{W}$ produce $w \in \mathcal{W}$, en resumen, se mapea el ruido de entrada a otro espacio latente secundario, que será el encargado de controlar los estilos, tal como se ve en la figura 3.16.

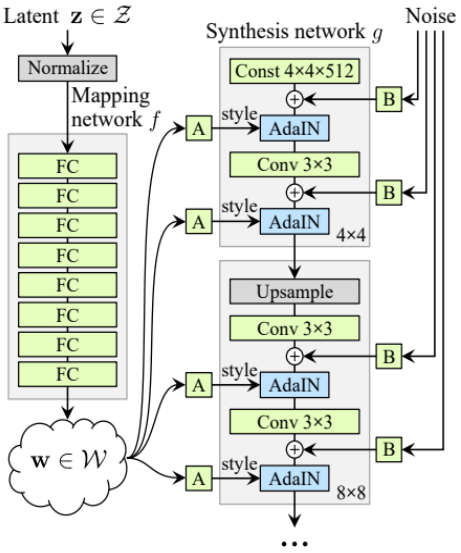


Figura 3.16: Arquitectura StyleGAN. [39]

El objetivo de este mapeo es que si, por ejemplo, la mayoría de imágenes usadas para el entrenamiento tienen el pelo negro, de nuevo, para el caso de caras humanas, una mayor cantidad de valores diferentes de entrada se mapearán a este estilo, lo que se traducirá posiblemente en que el modelo no pueda mapear correctamente algunas características del ruido de entrada a estilos como el color del pelo, la forma de los ojos, etc., un fenómeno conocido como enredo de características. Sin embargo, usando otra red neuronal el modelo puede generar un vector que no tiene que seguir la distribución de datos del entrenamiento y reducir la correlación entre las características de entrada.

Después de esto, se aplican transformaciones afines para transformar w en estilos $y = (y_s, y_b)$ que controlan otra nueva operación, la normalización de instancias adaptativa (AdaIN) definida por la formula 3.6, que permite normalizar los valores de las capas convolucionales pero, en lugar de usar los parámetros aprendidos, usa los estilos y calculados, lo que permite ajustar el estilo en cada capa convolucional.

$$\text{AdaIN}(x_i, y) = y_{s,i}\frac{x_i - \mu(x_i)}{\sigma(x_i)} + y_{b,i} \tag{3.6}$$

Además de esto, un ruido gaussiano es añadido tras cada capa convolucional, de forma que en función de la capa que sea se le aplica un factor de escala especializado. El motivo detrás de la inserción de este ruido adicional es el que comentábamos al principio de la sección, añadir este ruido permite diferenciar características más de alto nivel de otras que son estocásticas, de forma que estas últimas sean controladas por el propio ruido, algo que no tendría sentido para conceptos como la postura, ya que generaría resultados poco realistas.

A pesar de los buenos resultados de este modelo, este no ha sido el definitivo, pues se realizó una revisión en 2020 [40] ya que se detectó de forma generalizada la aparición de ciertos artefactos en las imágenes generadas, por lo que se trabajó para detectar las causas y proveer soluciones que permitan mejorar la calidad de los resultados.

El primero de los problemas era la aparición de artefactos similares a una mancha, el cual se relacionó directamente con la función AdaIN, que tal y como se ve en la formula 3.6 normaliza la media y la varianza de cada mapa de características, lo que potencialmente destruye información de características que se encuentran relativamente cerca unas a otras. Finalmente se confirmó, ya que al eliminar la función del modelo, los artefactos desaparecían.

Esto llevó a realizar una revisión en la arquitectura, que finalmente sufrió una serie de modificaciones que básicamente consistieron en sacar fuera del bloque de estilo el ruido y el sesgo, lo que ofrece resultados más predecibles, ya que trabajan sobre datos normalizados. Con este cambio además no es necesario normalizar la media, por lo que también puede eliminarse esa función, tal y como se ve en la figura 3.17c.

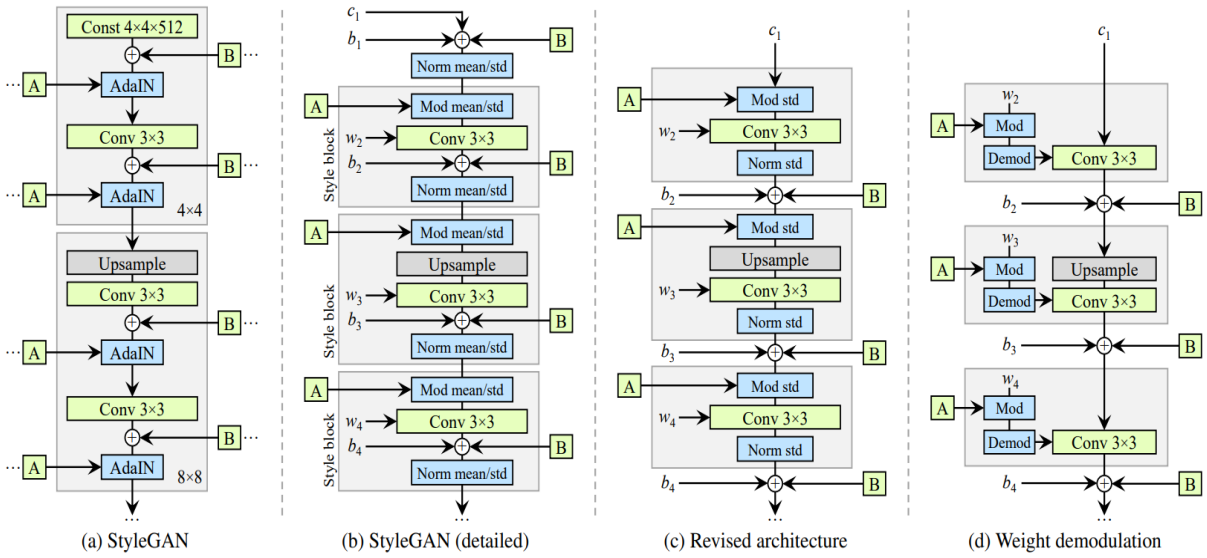


Figura 3.17: Revisión de la arquitectura StyleGAN. [40]

Una vez revisada la arquitectura, se revisó también la función de normalización, que era el principal objetivo, se propuso una técnica de demodulación que ajusta los pesos de la convolución en función del estilo, permitiendo una normalización más directa de las

características de salida, que se puede comprobar en la figura 3.17. Esta técnica, basada en análisis estadísticos de la señal, demostró ser eficaz en la mejora de la calidad de las imágenes generadas, contribuyendo a la reducción de los artefactos.

Por otro lado, también se detectaron artefactos relacionados con el crecimiento progresivo, una técnica que consiste en comenzar el entrenamiento con imágenes de baja resolución y luego aumentar progresivamente la resolución durante el proceso de entrenamiento, lo que permite que la red aprenda detalles de alto nivel en etapas iniciales y luego refinarse a medida que avanza el entrenamiento.

Para solventarlo, se exploraron multitud de arquitecturas alternativas que permitieran prescindir de este procedimiento, lo que concluyó finalmente con que utilizando *skip connections* en el generador y redes residuales en el discriminador, dos conceptos ya conocidos, mejoraron los resultados pudiendo prescindir por tanto del crecimiento progresivo y eliminando los artefactos relacionados sin sacrificar el rendimiento.

Además, dado que una parte importante de este proyecto se centra en el entrenamiento con pocos datos, se usa también una técnica de *data augmentation* denominada *adaptive discriminator augmentation* (ADA) [41] que se centra en reducir la posibilidad de *overfitting* en el discriminador cuando se entrena con pocas imágenes, algo realmente común.

Esta técnica busca evitar el ajuste manual de la intensidad de cada aumento, para ello se proponen dos heurísticas para determinar el sobreajuste del discriminador. La primera heurística relaciona las predicciones del discriminador para el conjunto de validación en relación con el conjunto de entrenamiento y las imágenes generadas. La segunda heurística la porción del conjunto de entrenamiento que obtiene salidas positivas del discriminador. En la práctica, el valor de p se inicializa con 0, y cada par de lotes se calculan las dos heurísticas, y p se ajusta agresivamente hacia arriba o hacia abajo para contrarrestar el sobreajuste. Ambas heurísticas son las siguientes:

$$rv = \frac{E[D_{\text{train}}] - E[D_{\text{validation}}]}{E[D_{\text{train}}] - E[D_{\text{generated}}]} \quad rt = E[\text{sign}(D_{\text{train}})] \tag{3.7}$$

Donde para ambas $r = 0$ implica nulo sobreajuste y $r = 1$ un sobreajuste total. Los investigadores han aplicado esta técnica con distintos conjuntos de tamaño reducido demostrando que es posible obtener mejores resultados aplicando esta metodología de *data augmentation*.

CAPÍTULO 4

Resultados experimentales

Una vez se ha descrito todo el marco teórico y los distintos recursos utilizados, se procede a documentar los resultados de los distintos experimentos, así como las observaciones y problemas enfrentados durante el desarrollo de los mismos.

4.1. Evaluación de la súper-resolución

En primer lugar, se procede a detallar como se han evaluado los distintos modelos de súper-resolución.

4.1.1. Métricas

El primer paso para evaluar una tarea de tarea de aprendizaje profundo es definir métricas de calidad en función del objetivo de dicha tarea con las que poder comparar distintos modelos de una forma objetiva. En el caso de la clasificación es muy trivial, pues al final es acertar o fallar, y a partir de ahí elaborar métricas. En el caso de la súper-resolución, se definen dos principales métricas [28], una basada estadísticas de los píxeles de la imagen, y otra en la información estructural.

Peak Signal-to-Noise Ratio

El *Peak Signal-to-Noise Ratio* (PSNR) es una métrica basada en la estadística de los píxeles de la imagen, se basa en el cálculo del error cuadrático medio píxel a píxel, definido en la fórmula 4.1. Por otro lado, el cálculo del PSRN se define en la fórmula 4.2.

$$\text{MSE} = \frac{1}{M \times N} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} |(F'(i,j) - F(i,j))^2| \tag{4.1}$$

$$\text{PSNR} = 10 \times \log \left(\frac{L \times L}{MSE} \right) \tag{4.2}$$

Donde $F'(i,j)$ y $F(i,j)$ son los valores de los píxeles de la posición (i,j) en la reconstrucción y la imagen original y L es el mayor valor de píxel en el rango 0-255. El PSNR es medido en decibelios (dB).

Tanto el PSNR como el MSE miden la calidad de la imagen calculando el tamaño del error de píxel entre la imagen a evaluar y la imagen de referencia, de forma que un valor de PSNR más alto resultará, de forma general, en una mayor calidad de imagen y menos distorsión en las imágenes evaluadas y de referencia. Sin embargo, este tipo de algoritmo se basa en las estadísticas globales de los valores de píxeles de la imagen y no considera los factores visuales locales del ojo humano, por lo que es imposible captar la calidad local de la imagen, por lo que un valor superior de PSNR no implica directamente un resultado mejor.

Índice de similitud estructural

Por otro lado el índice de similitud estructural (SSIM) es otra métrica habitual para evaluar la súper-resolución basada, tal y como su nombre indica, en la similutud estructural. A diferencia del MSE y PSNR, el SSIM utiliza la media y la varianza para medir la similitud de la estructura de la imagen de prueba y la imagen de referencia bajo características estadísticas globales. Su fórmula es la siguiente:

$$\text{SSIM} = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{4.3}$$

Donde μ_x y μ_y representa la media de las imágenes x e y, σ_x y σ_y la varianza y C_1 y C_2 son constantes para evitar que el denominador sea 0.

El SSIM construye la similitud estructural entre la imagen de referencia y la imagen evaluada según la correlación de los píxeles de las imágenes, de forma que cuanto mayor sea el valor de SSIM, mejor será la calidad de la imagen. De nuevo, este índice puede ayudar a evaluar que reconstrucción está mejor lograda, pero no ofrece la verdad absoluta.

Tal y como se comentan en ambas métricas, no suponen una verdad absoluta, por lo general, a mayor valor de ambas, mejores serán los resultados, pero al final la súper-resolución es una tarea compleja de evaluar ya que la calidad de la imagen no solo se basa en la precisión de los píxeles, sino también en otros aspectos como la percepción visual humana y la coherencia estructural. Por tanto, es un requisito indispensable acompañar de estas métricas de pruebas subjetivas, como una simple evaluación perceptual.

4.1.2. Proceso de evaluación y resultados

Una vez definidas las métricas, el proceso de evaluación comienza generando un conjunto de datos de entrenamiento y test, de forma que pueda evaluarse de forma independiente. Para el caso de la súper-resolución es algo diferente, ya que es necesario, por un lado, el conjunto de imágenes en baja resolución y por otro, en alta resolución.

Para ello, se ha realizado una división estratificada, es decir, respetando la distribución de clases del conjunto de datos original BCN20000, disponible en la figura 3.8, de forma que un 80 % se usará para entrenar, y un 20 % para evaluar [42], es decir, 9930 imágenes para entrenar y 2843 para evaluar. Además de esto, ha sido necesario generar las muestras en

baja resolución tal y como se comentaba anteriormente. Dado que las imágenes originales tienen una resolución original de 1024x1024 y se pretende realizar un aumento x4, se han reducido a 256x256 utilizando interpolación bicúbica [43].

De esta forma, los datos quedan repartidos en las siguiente estructura de ficheros:

```
/
├── BCN20000/
│   ├── meta_info/
│   ├── hr/
│   │   ├── ISIC_0053454.JPG
│   │   └── ...
│   └── lr/
│       ├── ISIC_0053454.JPG
│       └── ...
└── BCN20000_test/
    ├── hr/
    │   ├── ISIC_0053467.JPG
    │   └── ...
    └── lr/
        ├── ISIC_0053467.JPG
        └── ...
```

Con los datos ya preparados, no es necesario pero si recomendable empezar con modelos pre-entrenados y evaluar sus resultados, ya que son modelos que han sido entrenados con grandes conjuntos de datos y van a acelerar mucho el proceso de entrenamiento, además de que es posible que se obtengan buenos resultados sin ni siquiera realizar un entrenamiento.

Por ello, se utilizaron los 3 modelos ya pre-entrenados para generar las imágenes en alta resolución del conjunto de test y obtener las siguientes métricas:

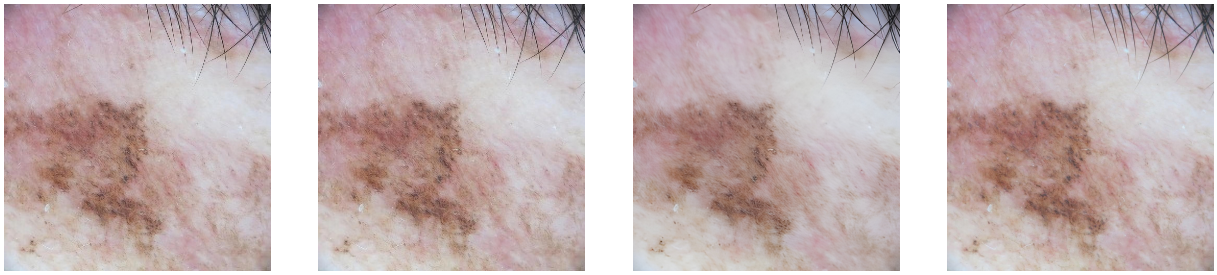
Tabla 4.1: Comparación de métricas entre modelos pre-entrenados.

Modelo pre-entrenado	**PSNR**	**SSIM**
EDSR	**37.84922753**	**0.93977308**
SRGAN	37.64416414	0.93623688
ESRGAN	36.57643168	0.93318650

(a) Imagen 1 (b) Imagen 2 (c) Imagen 3

Figura 4.1: Imágenes de referencia en baja resolución.

Como se puede comprobar, usando únicamente los pesos de los modelos pre-entrenados, el modelo que ha obtenido los mejores resultados ha sido EDSR, por lo que a priori, debería ser el mejor, a pesar de esto, tal y como comentamos, es indispensable un análisis visual de los resultados, mostrado en las figuras 4.2, 4.3 y 4.4. Las imágenes de referencia en baja resolución se muestran en la figura 4.1.



(a) EDSR (b) SRGAN (c) ESRGAN (d) Original

Figura 4.2: Comparación de imágenes de los modelos de súper-resolución pre-entrenados.

En esta primera comparación, donde se recomienda ampliar el presente documento para ver los resultados (se incluirán las imágenes en mayor tamaño en su sección correspondiente en el apéndice), podemos ver como tanto la imagen generada por EDSR como SRGAN es muy similar, ambas introducen bastante ruido pero la textura resultante de la herida es bastante similar a la original. Sin embargo, si prestamos atención a la de ESRGAN, podemos ver como el modelo le ha dado una textura más parecida a la de un pelaje animal, no obstante tiene mucho menos ruido y los detalles son mucho más claros, esto fue un signo claro de que este modelo era el que tenía mayor potencia de ofrecer mejores resultados si se le realiza un ajuste, ya que parece que gran parte del motivo por el que ha ofrecido peores resultados en las métricas parece ser principalmente este.

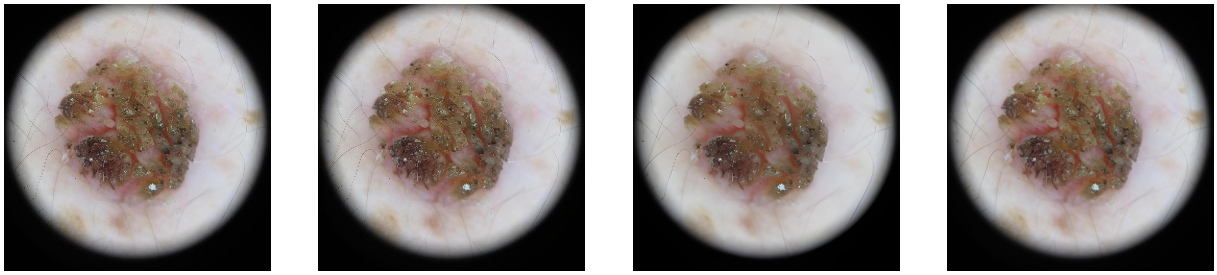
(a) EDSR (b) SRGAN (c) ESRGAN (d) Original

Figura 4.3: Comparación de imágenes de los modelos de súper-resolución pre-entrenados.

En la segunda imagen, en la que aparece una gran cantidad de pelo, podemos ver algo muy similar, de nuevo tanto EDSR como SRGAN ofrecen resultados muy similares, ESRGAN ofrece una imagen mucho más nítida pero la textura de la herida difiere más que los otros modelos. Además, podemos ver como el pelo está mucho logrado, pero de nuevo, esto no es más que otro indicio de que ESRGAN puede mejorar mucho más.

Por último, se vuelve a repetir lo mencionado anteriormente, mejor detalle en ESRGAN pero menor similitud con la textura de la herida, además, podemos ver como no se distinguen claramente las marcas del dermatoscopio.

Por tanto, una vez comprobado que ninguno de los resultados obtenidos es suficiente satisfactorio como para ser utilizado para mejorar la clasificación, se procedió a realizar el



(a) EDSR (b) SRGAN (c) ESRGAN (d) Original

Figura 4.4: Comparación de imágenes de los modelos de súper-resolución pre-entrenados.

entrenamiento, en este caso, ajuste fino, de cada uno de los modelos para ver si es posible lograr mejoras notables en dichos modelos y, de esta forma, poder volver a evaluarlos ya una vez ajustados.

Para ello, evidentemente, se utilizó el conjunto de datos de entrenamiento y se realizó el entrenamiento con la siguiente configuración:

Tabla 4.2: Configuración de entrenamiento de súper-resolución.

Número de Iteraciones	400k
Tamaño del Lote	3
Algoritmo de Optimización	Adam
Tasa de Aprendizaje	1×10^{-4}

En este caso, el número de iteraciones hace referencia al número de lotes que es mostrado al discriminador, mientras el resto de valores son bastante estándar. Puede destacar el tamaño del lote de 3, el cual es necesario que sea reducido ya que estos modelos ocupan bastante memoria gráfica y la GPU disponible aceptaba como máximo este tamaño.

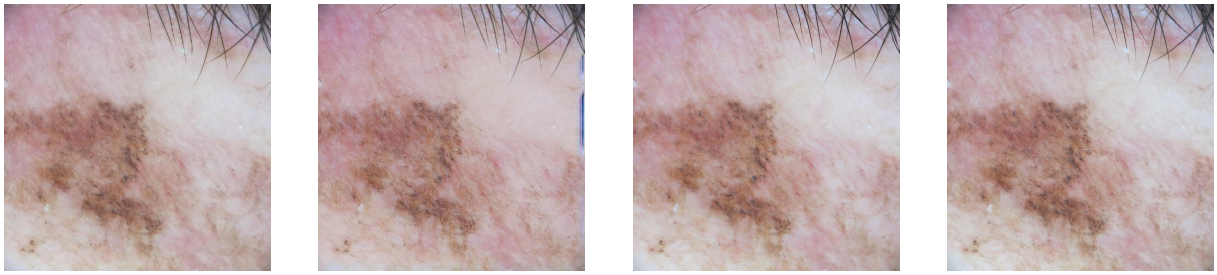
Respecto a la duración del entrenamiento, para el caso de EDSR y SRGAN tuvieron una duración aproximada de 14 y 18 horas respectivamente, mientras que para el caso de ESRGAN fueron 32 horas, es decir, 1 día y 8 horas, prácticamente el doble de tiempo de entrenamiento, por lo que será importante valorar en los resultados si merece la pena a pesar del coste temporal que tiene.

Una vez finalizado el entrenamiento de los modelos, los resultados obtenidos por los mismos sobre el conjunto de test son los siguientes:

Tabla 4.3: Comparación de métricas entre modelos entrenados con BCN20000.

Modelo entrenado	**PSNR**	**SSIM**
EDSR	36.56321984	0.94457680
SRGAN	30.85615089	0.94138455
ESRGAN	**38.92414366**	**0.95564024**

En este caso, el que ha obtenido mejores resultados, tal y como esperábamos, ha sido ESRGAN, pero sucede algo curioso. Mientras que el SSIM mejora en todos los modelos, el PSNR sufre un ligero descenso en el caso de EDSR y uno bastante pronunciado en el caso de SRGAN. De nuevo, es necesario un análisis visual, por lo que aprovecharemos para intentar explicar el por qué de estos resultados visualizando las figuras 4.5, 4.6 y 4.7.



(a) EDSR (b) SRGAN (c) ESRGAN (d) Original

Figura 4.5: Comparación de imágenes de los modelos de súper-resolución entrenados con BCN20000.

En la primera imagen podemos comprobar que la calidad de las 3 imágenes es bastante similar, resultaría complejo tener que elegir entre cualquiera de las 3, pero rápidamente podemos detectar algo extraño. En el caso de SRGAN, vemos como aparece un extraño artefacto en el borde derecho de la imagen, lo que nos llevó rápidamente a comprobar si era algo ocasional o realmente se había generalizado. Esto se confirmó revisando otras imágenes y, efectivamente, sucedía en multitud de ellas, por lo que este es el principal motivo de la baja puntuación de PSNR en el SRGAN, debido, seguramente, a un sobreajuste debido a que no se ha entrenado con suficiente variedad de datos.

Por otro lado, si se revisa con detalle la generada por EDSR, podemos comprobar también la existencia de un artefacto, en este caso, cercano al borde izquierdo, aparece una extraña franja ligeramente más oscura, que, de nuevo, se repite en varias imágenes, lo que seguramente se trate de un ligero sobreajuste que hace que el PSNR se reduzca. Esto tiene sentido, ya que, mientras que el índice de similitud mejora en todos los modelos, indicando que, en general, la calidad de la imagen ha generado, el PSNR ha empeorado a causa de los artefactos.

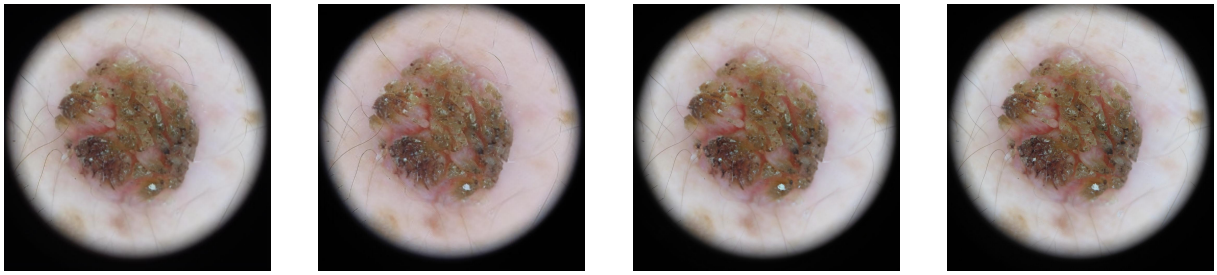
El único caso que mejora ambos y, además, superando a los demás, es ESRGAN, esto se traduce en una nitidez ligeramente superior, que se puede observar en los detalles más finos y en los propios pelos de la imagen.

(a) EDSR (b) SRGAN (c) ESRGAN (d) Original

Figura 4.6: Comparación de imágenes de los modelos de súper-resolución entrenados con BCN20000.

En la segunda imagen, si que se puede apreciar algo más la diferencia entre los modelos, más en concreto, destaca bastante ESRGAN ofreciendo una nitidez y un detalle bastante superior al resto, sin embargo, en esta imagen no aparecen artefactos en EDSR y SRGAN.

Por último, se vuelve a repetir, ESRGAN se ve bastante más nítido y detallado, además,



(a) EDSR (b) SRGAN (c) ESRGAN (d) Original

Figura 4.7: Comparación de imágenes de los modelos de súper-resolución entrenados con BCN20000.

los detalles del dermatoscopio están mucho más logrados y los pelos pequeños del fondo también se pueden ver mucho mejor.

Una vez concluidos los experimentos, podemos afirmar que el ajuste fino de los modelos pre-entrenados ha demostrado ser una estrategia efectiva para mejorar la calidad de las imágenes generadas en el proceso de súper-resolución. Aunque los modelos pre-entrenados ofrecen resultados algo aceptables, el refinamiento adicional a través del entrenamiento llevó a mejoras significativas en términos de métricas de calidad de imagen, así como percepción visual de las mismas en prácticamente todos los modelos.

Este análisis visual de las imágenes generadas reveló diferencias sutiles pero importantes entre los modelos. Mientras que todos los modelos pre-entrenados mostraron artefactos o imperfecciones en las imágenes generadas, el modelo ESRGAN, después del ajuste, logró producir imágenes más nítidas y detalladas, con ningún artefacto visible, lo que indica que fue capaz de capturar mejor las características específicas del conjunto de datos de entrenamiento y, por tanto, ha sido el modelo escogido para evaluar su mejora en la clasificación.

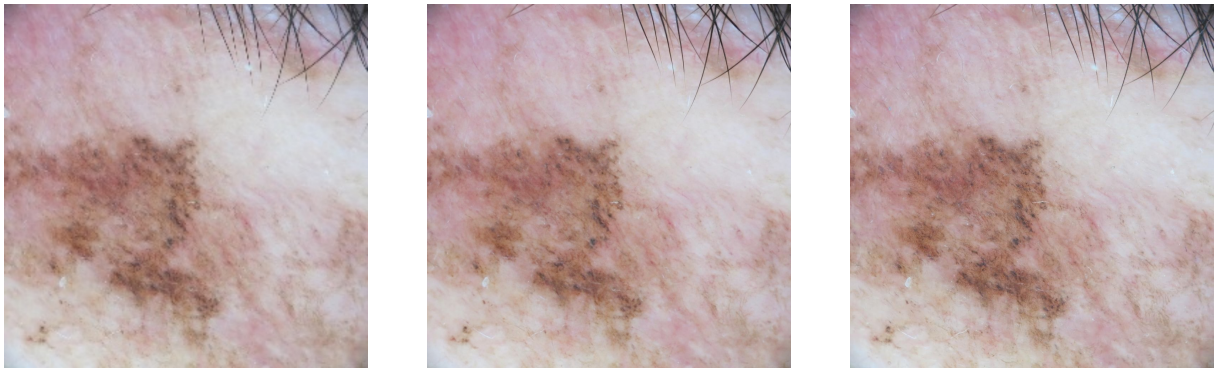
Por último, resta comprobar la variación de aplicar esta técnica a una tradicional como, por ejemplo, interpolación bicúbica. Los resultados se pueden comparar en las figuras 4.8, 4.9 y 4.10 y los valores obtenidos de las métricas se pueden comprobar en la siguiente tabla:

Tabla 4.4: Comparación de métricas entre ESRGAN e interpolación bicúbica.

Método	**PSNR**	**SSIM**
Bicúbica	**39.82045192**	**0.95919754**
ESRGAN	38.92414366	0.95564024

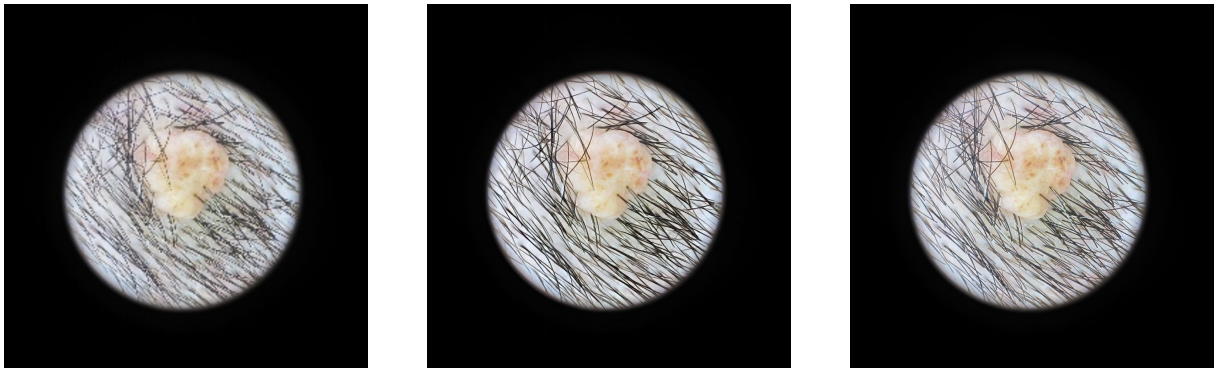
En este caso pasa algo realmente curioso, las métricas obtenidos colocan a la interpolación bicúbica como superior frente a ESRGAN y, por tanto, el resto de modelos. Sin embargo, si observamos las imágenes podemos comprobar que el resultado esta lejos de la realidad, pues las imágenes obtenidas con ESRGAN son muchísimo más nítidas y detalladas que las obtenidas con interpolación.

Si bien no pasa siempre, este fenómeno puede ocurrir a veces ya que aunque las métricas como PSNR y SSIM son útiles para evaluar la calidad de la imagen de manera objetiva, no siempre capturan completamente la percepción visual humana y se basan en estadística, por lo que podríamos decir que las imágenes obtenidas por interpolación son



(a) Interpolación bicúbica (b) ESRGAN (c) Original

Figura 4.8: Comparación de interpolación bicúbica con ESRGAN.



(a) Interpolación bicúbica (b) ESRGAN (c) Original

Figura 4.9: Comparación de interpolación bicúbica con ESRGAN.

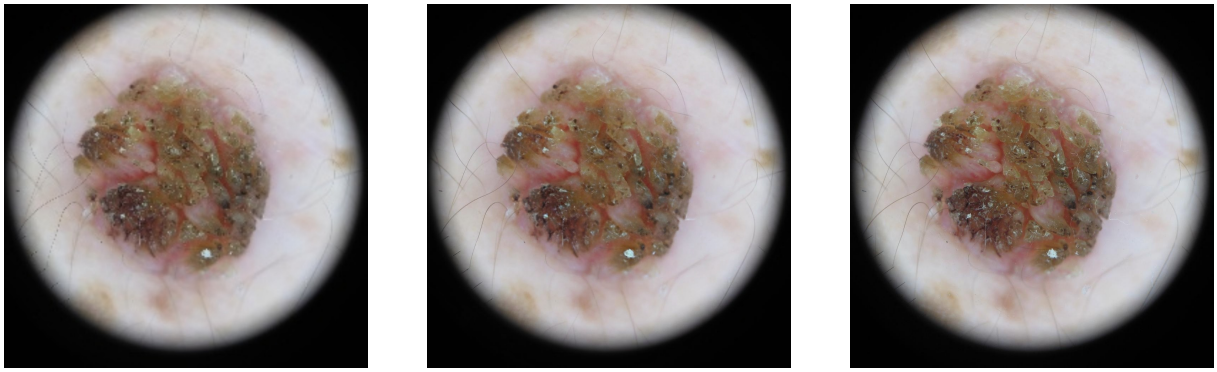
matemáticamente más fieles que las de ESRGAN.

Pero a nivel visual, las obtenidas a través de ESRGAN son realmente mejores, demostrando que puede producir imágenes visualmente superiores al enfocarse en aspectos perceptuales y detalles finos más allá de valores estadísticos, ofreciendo detalles que son importantes para la experiencia visual y pueden llegar a ser vitales para el reconocimiento de patrones por parte de un profesional. Será labor nuestra averiguar si también es importante este tipo de detalles para un modelo de clasificación.

A forma de resumen, se incluye una tabla comparativa de todas las configuraciones evaluadas y sus métricas en la tabla 4.5.

Configuración	**PSNR**	**SSIM**
EDSR	37.84922753	0.93977308
EDSR ajustado	36.56321984	0.94457680
SRGAN	37.64416414	0.93623688
SRGAN ajustado	30.85615089	0.94138455
ESRGAN	36.57643168	0.93318650
ESRGAN ajustado	38.92414366	0.95564024
Interpolación bicúbica	**39.82045192**	**0.95919754**

Tabla 4.5: Comparación de resultados de diferentes métodos de súper-resolución.



(a) Interpolación bicúbica (b) ESRGAN (c) Original

Figura 4.10: Comparación de interpolación bicúbica con ESRGAN.

4.2. Evaluación de la generación de datos sintéticos

El siguiente paso consiste en documentar el proceso de evaluar las distintas arquitecturas para la generación de datos sintéticos.

4.2.1. Métricas

Si evaluar la súper-resolución ha sido una labor difícil ya que depende mucho en la percepción visual de la imagen, evaluar la generación de nuevas imágenes no iba a ser tarea fácil, ya que mientras en súper-resolución tenemos una salida de referencia con la que poder comparar, en la generación de nuevas imágenes no hay una salida clara con la que comparar, tan solo el conjunto de datos de entrenamiento y su distribución de datos.

Es por este motivo por el que definir métricas de calidad y que logren evaluar correctamente este proceso se convierte en algo realmente complejo, ya que es necesario comparar con todo el conjunto, de forma que sea similar, pero no una réplica, además se tienen que tener aspectos en cuenta como el ruido, la coherencia visual, la diversidad y la calidad perceptual.

Para ello, se usa como principal métrica para la evaluación de estos modelos la *Fréchet Inception Distance* (FID) [44], la cual es una evolución de la *Inception Distance*, una métrica que evalúa la calidad y la diversidad de las imágenes generadas midiendo la calidad de clasificación de una red neuronal pre-entrenada Inception en las imágenes generadas.

La FID mejora esta métrica, ya que calcula la distancia Fréchet entre las distribuciones de características de las imágenes generadas y las del conjunto de datos de referencia, de nuevo, generadas por una red pre-entrenada Inception, lo que tiene en cuenta tanto la calidad de las imágenes generadas como su variabilidad, evitando así que el modelo obtenga una puntuación alta simplemente generando imágenes muy similares entre sí. Además, al utilizar la representación de características aprendida por una red neuronal convolucional pre-entrenada, la FID puede capturar aspectos más complejos de la calidad de las imágenes, como la coherencia visual y la diversidad estructural.

Por tanto, llamamos a la distancia Fréchet $d(.,.)$ entre la distribución gaussiana con media (m, C) obtenida de la probabilidad de que los datos sean generados $p(.)$ y la gaussiana con media (m_w, C_w) obtenida de la probabilidad de que los datos sean reales $p_w(.)$, de la

siguiente forma:

$$d^2\left((m, C), (m_w, C_w)\right) = \|m - m_w\|_2^2 + \mathrm{Tr}\left(C + C_w - 2(CC_w)^{\frac{1}{2}}\right) \tag{4.4}$$

De esta forma, en lugar de comparar píxel a píxel como hacen otras métricas, FID compara la media y la desviación, de la capa más profunda de un modelo, lo que corresponde a la detección de objetos más complejos que los de las primeras capas.

Esta métrica además se correlaciona bastante bien con el juicio humano sobre la calidad de las imágenes pero, de nuevo, acompañar esta métrica con una evaluación perceptual sobre las imágenes generadas será algo vital para interpretar con éxito los resultados obtenidos.

4.2.2. Proceso de evaluación y resultados

Con las métricas ya definidas, de nuevo, es hora de documentar el proceso de evaluación que se ha llevado a cabo en esta parte, la cual es bastante diferente a la de súper-resolución. Tal y comentamos en la sección de métricas, en el caso de la generación de datos, a diferencia de la súper-resolución, no tenemos una imagen o una referencia como salida.

Es por esto por lo que tan solo se ha utilizado un único conjunto de datos para este proceso, de forma que, se entrenará con este conjunto y se procederá a realizar el calculo del FID respecto al mismo, lo que permitirá ver como de bueno es el modelo generando los datos.

Respecto al conjunto de datos, tal y como se comentó, se utilizará HAM10000, pero como se desea someter el entrenamiento a una situación en la que se tienen pocos datos, se usarán tan solo los datos de melanoma, correspondientes a 1113 imágenes, como se puede observar en la tabla 3.2, que no son estrictamente pocas, pero ayudará a determinar mejor como se adapta el modelo a un conjunto algo más reducido.

Como bien ya sabemos, la resolución original de las imágenes de este conjunto de datos es de 600x450, un tamaño bastante malo para trabajar con el, ya que no es cuadrado y, además, bastante grande, lo que haría más complejo el entrenamiento de una red que, de por sí, ya es compleja. Por tanto, se le realizó un ligero pre-procesamiento a los datos en los que se recortó horizontalmente y de forma centrada hasta igualar la imagen en 450x450 [43], de forma que no se pierda información ya que, esta centrada y además se lograba eliminar algunos marcos negros, por lo que era doblemente beneficioso. Una vez centradas, se reducían a 256x256, uno de los tamaños recomendados [41], usando interpolación bicúbica. Además, como se ha mencionado, se tomaron tan solo las imágenes pertenecientes a melanomas [45].

Una vez preparados los datos, en primer lugar se probó con la arquitectura DCGAN ya mencionada y que se puede ver en la figura 3.15, para ello se implementó en Tensorflow [46], una de las principales bibliotecas de referencia para el aprendizaje profundo. Para ello, en primer lugar, se procedió a adaptar el modelo a la resolución deseada, es decir a 256x256.

El modelo de referencia tiene una salida de 64x64, por lo que para adaptarlo se añadieron dos bloques más, uno que escala la resolución a 128 y otro a 256, además, se añadieron más filtros a las capas ya que al añadir más bloques no tenía sentido no añadir más filtros

a las capas, de esta forma el primer modelo fue el mostrado en las figuras 4.11 y 4.12 y cuya implementación en Tensorflow es la siguiente:

```
def generator_model():
    model = tf.keras.Sequential()
    model.add(layers.Reshape(target_shape = [1, 1, NOISE_DIM], input_shape = [NOISE_DIM]))

    model.add(layers.Conv2DTranspose(filters = 512, kernel_size = 4, strides = 2))
    model.add(layers.BatchNormalization(momentum = 0.7))
    model.add(layers.Activation('relu'))

    model.add(layers.Conv2DTranspose(filters = 256, kernel_size = 4, strides = 2, padding = 'same'))
    model.add(layers.BatchNormalization(momentum = 0.7))
    model.add(layers.Activation('relu'))

    model.add(layers.Conv2DTranspose(filters = 128, kernel_size = 4, strides = 2, padding = 'same'))
    model.add(layers.BatchNormalization(momentum = 0.7))
    model.add(layers.Activation('relu'))

    model.add(layers.Conv2DTranspose(filters = 64, kernel_size = 4, strides = 2, padding = 'same'))
    model.add(layers.BatchNormalization(momentum = 0.7))
    model.add(layers.Activation('relu'))

    model.add(layers.Conv2DTranspose(filters = 32, kernel_size = 4, strides = 2, padding = 'same'))
    model.add(layers.BatchNormalization(momentum = 0.7))
    model.add(layers.Activation('relu'))

    model.add(layers.Conv2DTranspose(filters = 16, kernel_size = 4, strides = 2, padding = 'same'))
    model.add(layers.BatchNormalization(momentum = 0.7))
    model.add(layers.Activation('relu'))

    model.add(layers.Conv2DTranspose(filters = 3, kernel_size = 4, strides = 2, padding = 'same'))
    model.add(layers.Activation('tanh'))

    return model

def discriminator_model():
    model = tf.keras.Sequential()

    model.add(layers.Conv2D(filters = 16, kernel_size = 4, strides = 2, padding = 'same'))
    model.add(layers.LeakyReLU(0.2))

    model.add(layers.Conv2D(filters = 32, kernel_size = 4, strides = 2, padding = 'same'))
    model.add(layers.BatchNormalization(momentum = 0.7))
    model.add(layers.LeakyReLU(0.2))

    model.add(layers.Conv2D(filters = 64, kernel_size = 4, strides = 2, padding = 'same'))
    model.add(layers.BatchNormalization(momentum = 0.7))
    model.add(layers.LeakyReLU(0.2))

    model.add(layers.Conv2D(filters = 128, kernel_size = 4, strides = 2, padding = 'same'))
    model.add(layers.BatchNormalization(momentum = 0.7))
    model.add(layers.LeakyReLU(0.2))

    model.add(layers.Conv2D(filters = 256, kernel_size = 4, strides = 2, padding = 'same'))
    model.add(layers.BatchNormalization(momentum = 0.7))
    model.add(layers.LeakyReLU(0.2))

    model.add(layers.Conv2D(filters = 512, kernel_size = 4, strides = 2, padding = 'same'))
    model.add(layers.BatchNormalization(momentum = 0.7))
    model.add(layers.LeakyReLU(0.2))

    model.add(layers.Flatten())

    model.add(layers.Dense(256))
    model.add(layers.LeakyReLU(0.2))

    model.add(layers.Dense(1, activation = 'sigmoid'))

    return model
```

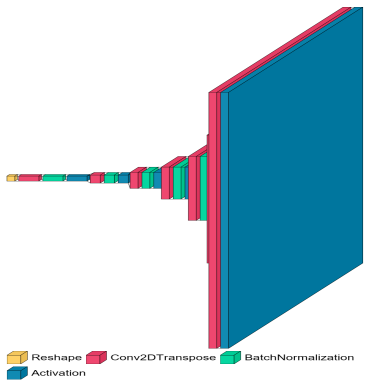


Figura 4.11: Arquitectura del generador.

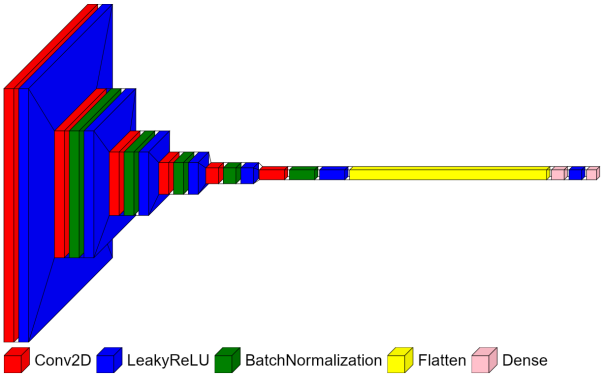


Figura 4.12: Arquitectura del discriminador.

Además, se definieron las funciones de perdida y el proceso de entrenamiento, el cual se puede consultar a continuación:

```
def discriminator_loss(real_output, fake_output):
    real_loss = cross_entropy(tf.ones_like(real_output), real_output)
    fake_loss = cross_entropy(tf.zeros_like(fake_output), fake_output)
    total_loss = real_loss + fake_loss
    return total_loss


def generator_loss(fake_output):
    return cross_entropy(tf.ones_like(fake_output), fake_output)


@tf.function
def train_step(images):
    noise = tf.random.normal([BATCH_SIZE, NOISE_DIM])

    with tf.GradientTape() as gen_tape, tf.GradientTape() as disc_tape:
        generated_images = generator(noise, training=True)

        real_output = discriminator(images, training=True)
        fake_output = discriminator(generated_images, training=True)

        gen_loss = generator_loss(fake_output)
        disc_loss = discriminator_loss(real_output, fake_output)

    gradients_of_generator = gen_tape.gradient(gen_loss, generator.trainable_variables)

    gradients_of_discriminator = disc_tape.gradient(
        disc_loss, discriminator.trainable_variables
    )

    generator_optimizer.apply_gradients(
        zip(gradients_of_generator, generator.trainable_variables)
    )

    discriminator_optimizer.apply_gradients(
        zip(gradients_of_discriminator, discriminator.trainable_variables)
    )

    return gen_loss, disc_loss


def train(dataset, start_epoch, epochs):
    for epoch in range(start_epoch, start_epoch + epochs):
        start = time.time()

        batches = 0
        for image_batch in dataset:
            print(".", end="")
            gen_loss, disc_loss = train_step(image_batch[0])
            batches += 1

            if batches >= len(dataset):
                break

        if (epoch) % 10 == 0:
            checkpoint.save(file_prefix=checkpoint_prefix)
            generate_save_images(generator, epoch, SEED)

        print(
            "Generator Loss: {:.4f}, Discriminator Loss: {:.4f}, Time for epoch {} is {} sec".format(
                gen_loss, disc_loss, epoch, time.time() - start
            )
        )
```

Respecto a la configuración del entrenamiento, fue la mostrada en la tabla 4.6.

Tabla 4.6: Configuración de entrenamiento de generación de datos sintéticos.

Número de imágenes	1 millón
Tamaño del Lote	8
Algoritmo de Optimización	Adam
Tasa de Aprendizaje	0.0025

En este caso el tamaño del lote ha podido ser aumentado hasta 8, lo que permitía la GPU, y se ha aumentado ligeramente la tasa de aprendizaje, tal y como indicaban los autores de StyleGAN [39]. Por otro lado, no se usa el número de iteraciones, se tiene en cuenta el número de imágenes que se muestran al discriminador, en este caso, 1 millón.

El entrenamiento tuvo una duración aproximada de 8 horas, una muestra de las imágenes que permite generar el modelo es la mostrada en la figura 4.13.

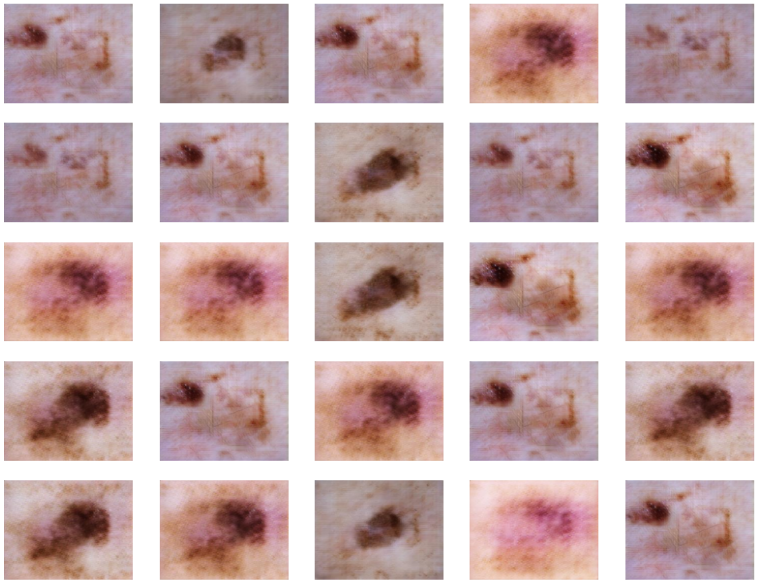


Figura 4.13: Imágenes generadas por DCGAN.

Se obtuvo un FID de 306,45, lo cual es un resultado bastante malo, algo que tiene sentido pues las imágenes generadas se ven poco realistas, pero lo que más destaca es que son muy similares entre si, lo que no es nada bueno.

Tras investigar acerca de esto, se concluyó que el modelo sufría de *mode collapse* [47], un fenómeno que ocurre cuando el generador logra desarrollar una estrategia en la que logra engañar al discriminador generando un pequeño subconjunto de imágenes difíciles de distinguir entre sí.

La principal solución que se plantea ante esta problemática es el uso de regularización, de forma que se evite el sobrejuste en el discriminador. Por tanto, se modificó el discriminador y pasó a tener la forma mostrada en la figura 4.14 y su implementación es la mostrada a continuación, de nuevo, en Tensorflow:

```
def discriminator_model():
    model = tf.keras.Sequential()

    model.add(layers.GaussianNoise(0.2, input_shape = [256, 256, 3]))

    model.add(layers.Conv2D(filters = 16, kernel_size = 4, strides = 2, padding = 'same'))
    model.add(layers.LeakyReLU(0.2))
    model.add(layers.Dropout(0.25))

    model.add(layers.Conv2D(filters = 32, kernel_size = 4, strides = 2, padding = 'same'))
    model.add(layers.BatchNormalization(momentum = 0.7))
    model.add(layers.LeakyReLU(0.2))
    model.add(layers.Dropout(0.25))

    model.add(layers.Conv2D(filters = 64, kernel_size = 4, strides = 2, padding = 'same'))
    model.add(layers.BatchNormalization(momentum = 0.7))
    model.add(layers.LeakyReLU(0.2))
    model.add(layers.Dropout(0.25))

    model.add(layers.Conv2D(filters = 128, kernel_size = 4, strides = 2, padding = 'same'))
    model.add(layers.BatchNormalization(momentum = 0.7))
    model.add(layers.LeakyReLU(0.2))
    model.add(layers.Dropout(0.25))

    model.add(layers.Conv2D(filters = 256, kernel_size = 4, strides = 2, padding = 'same'))
    model.add(layers.BatchNormalization(momentum = 0.7))
    model.add(layers.LeakyReLU(0.2))
    model.add(layers.Dropout(0.25))

    model.add(layers.Conv2D(filters = 512, kernel_size = 4, strides = 2, padding = 'same'))
    model.add(layers.BatchNormalization(momentum = 0.7))
    model.add(layers.LeakyReLU(0.2))
    model.add(layers.Dropout(0.25))

    model.add(layers.Flatten())

    model.add(layers.Dense(256))
    model.add(layers.LeakyReLU(0.2))

    model.add(layers.Dense(1, activation = 'sigmoid'))

    return model
```

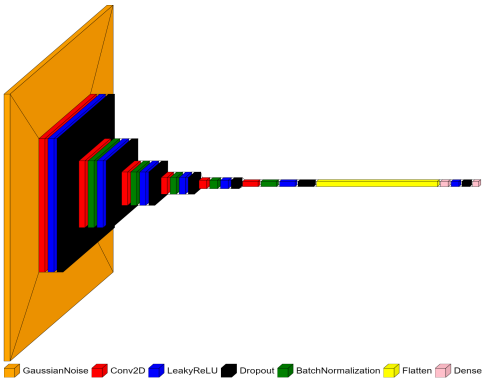


Figura 4.14: Arquitectura de la revisión del discriminador.

Básicamente se le añadió un filtro de ruido gaussiano a la entrada y se añadieron capas de *dropout*, de esta forma se lograría reducir el sobreajuste y, por tanto, el *mode collapse*. Tras el entrenamiento, que tuvo una duración similar al anterior, una muestra de las imágenes generadas son las mostradas en la figura 4.15.

Tras esta revisión se consiguió una mejora ya que el FID bajó a 193,4, sin embargo, la calidad de las imágenes no varió mucho, de hecho algunas se ven bastante más pixeladas pero sin embargo, las imágenes son algo más variadas entre sí. Este es el motivo principal detrás de la mejora del FID, lo que nos demuestra que aplicar técnicas de regularización logra reducir el *mode collapse*.

A pesar de esto, las imágenes obtenidas seguían siendo bastante similares entre sí y la calidad no parecía mejorar, estas dos cosas fueron lo que nos hicieron concluir que la arquitectura DCGAN era demasiado simple y podría llegar a tener sentido para resoluciones más pequeñas y problemas más sencillos, pero cuando se aumenta la resolución los

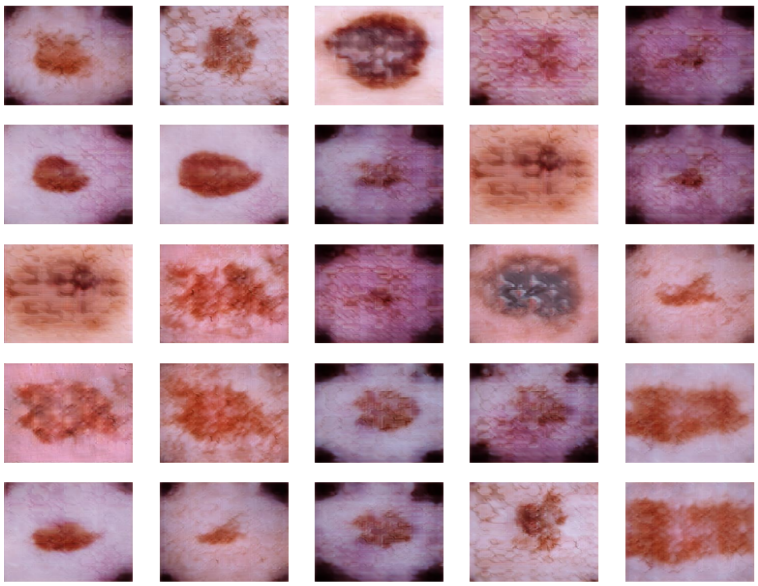


Figura 4.15: Imágenes generadas por la revisión de DCGAN.

resultados son bastante malos, tal y como se ha podido comprobar.

Dado que los resultados eran bastante pobres y no estaban a la altura de lo que realmente se podía obtener con el conocimiento disponible en el estado del arte, fue en este momento en el que se decidió adoptar otra solución, en concreto, utilizar un modelo más complejo y más moderno como StyleGAN, ya documentado en su sección.

Por tanto, se llevó a cabo el entrenamiento de este modelo, el cual, demostró ser más complejo sobre todo en su tiempo entrenamiento, que con la misma configuración mostrada en la tabla 4.6 tuvo una duración aproximada de 4 días, muy superior al anterior modelo.

Eso sí, es importante destacar que mientras que la arquitectura DCGAN empezaba el entrenamiento desde cero, para StyleGAN se ha realizado un ajuste del modelo, ya que estaba previamente entrenado con un conjunto de datos de imágenes de rostros faciales, algo que, sin duda, ayudará en el entrenamiento. Tras terminar, la muestra asociada a este modelo es la de la figura 4.16.

Con este nuevo modelo el FID se situó en 59,93, lo cual supone una muy buena mejora que, además, se ve reflejada en la calidad de las imágenes. Por un lado, son mucho más variadas, mostrando diferentes morfologías de melanoma muy diferentes entre sí. Por el otro, el realismo de las imágenes se ha visto muy incrementado, mientras que con el modelo anterior era realmente sencillo distinguir una imagen real de una generada, ahora están muy bien logradas y resulta bastante complicado, llegando a generar con éxito hasta el pelo, que es algo realmente complejo.

Pasa algo curioso con el pelo y es que el modelo lo interpreta a veces como grietas en la piel, por lo que en algunas imágenes aparecen multitud de pequeñas grietas que tendrían más sentido que fueran pelo, pero aun así, obviando esto, el modelo logra generar con

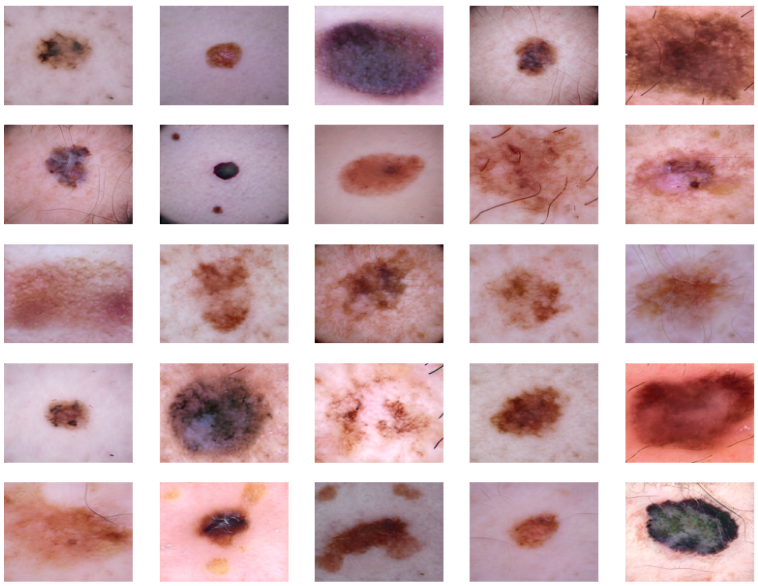


Figura 4.16: Imágenes generadas por StyleGAN.

éxito muchos detalles finos que se pueden observar en las distintas heridas.

En este caso la elección ha resultado ser mucho más sencilla, la arquitectura DCGAN ha demostrado lograr generar imágenes no suficientemente buenas para una resolución de 256x256, además de ser muy susceptible a sufrir de *mode collapse*, por lo que no es la elección adecuada para este trabajo.

Por otro lado, StyleGAN ha logrado generar imágenes de muy alta calidad, capturando detalles de grano fino y llegando a lograr generar hasta pelo con éxito, además de ofrecer una gran variedad en las imágenes, no sufriendo *mode collapse* de una forma tan brusca como lo hace DCGAN. Además del análisis visual, la FID lo da también como ganador, por lo que ha sido por diferencia el modelo elegido para ser evaluado junto a la clasificación.

De nuevo, se incluye la tabla 4.7 a modo de resumen con los distintos resultados obtenidos para las arquitecturas propuestas.

Configuración	**FID**
DCGAN	306.45
DCGAN con regularización	193.4
StyleGAN	**59.93**

Tabla 4.7: Comparación de resultados de diferentes métodos de generación de imágenes.

4.3. Evaluación del impacto en la clasificación

Con los modelos escogidos para súper-resolución y generación de datos sintéticos, ahora se plantean las métricas y el proceso llevado a cabo para evaluar si ofrecen mejora o no sobre la detección del melanoma mediante un CNN.

4.3.1. Métricas

Respecto a las métricas utilizadas para evaluar, mientras que para súper-resolución y la generación de datos es realmente complicado cuantificar como de buenos están siendo los resultados, para la clasificación es bastante trivial, para ello se definen las siguientes métricas [48].

Matriz de confusión

Una matriz de confusión es una herramienta que permite rápidamente visualizar como de bueno es un algoritmo de clasificación, las filas representan las clases reales, mientras que las columnas representan las clases predichas. Un ejemplo de matriz de confusión es el siguiente:

Clase Real	Clase Predicha	
	Positivo	Negativo
Positivo	TP	FN
Negativo	FP	TN

Donde:

- TP (True Positive): casos positivos que fueron correctamente identificados.
- TN (True Negative): casos negativos que fueron correctamente identificados.
- FP (False Positive): casos negativos que fueron incorrectamente identificados como positivos (Error Tipo I).
- FN (False Negative): casos positivos que fueron incorrectamente identificados como negativos (Error Tipo II).

Exactitud

Es la proporción de predicciones correctas entre el total de predicciones realizadas. Se calcula mediante la fórmula:

$$\text{Exactitud} = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.5}$$

Sensibilidad

Es la proporción de casos positivos que fueron correctamente identificados por el modelo, en el ámbito de la medicina es la métrica más importante. Se calcula mediante la fórmula:

$$\text{Sensibilidad} = \frac{TP}{TP + FN} \tag{4.6}$$

Especificidad

Es la proporción de casos negativos que han sido correctamente identificados por el modelo. Se calcula mediante la fórmula:

$$\text{Especificidad} = \frac{TN}{TN + FP} \tag{4.7}$$

Precisión

Es la proporción de casos positivos predichos correctamente respecto al total de casos positivos predichos por el modelo, tiene bastante importancia en los casos en los que un falso positivo puede tener consecuencias negativas en el paciente. Se calcula mediante la fórmula:

$$\text{Precisión} = \frac{TP}{TP + FP} \tag{4.8}$$

Puntuación F1

Es la media armónica entre precisión y sensibilidad, de forma que penaliza valores extremos en ambos. Se calcula mediante la fórmula:

$$\text{F1} = \frac{2 \times \text{Precisión} \times \text{Sensibilidad}}{\text{Precisión} + \text{Sensibilidad}} \tag{4.9}$$

Todas las métricas comentadas comparten el mismo rango de valores, $[0, 1]$, de forma que un valor de 0 indica el peor resultado y un valor de 1 el mejor resultado posible. Usando estos valores podremos decidir que técnicas ayudan a mejorar la labor de clasificación y de que forma lo hacen.

4.3.2. Proceso de evaluación y resultados

Una vez definimos las métricas, de nuevo, se define a continuación el proceso de evaluación llevado a cabo para medir como de efectivas son las técnicas estudiadas. En primer lugar, se escogió un modelo de clasificación sobre el que realizar un ajuste con los datos pertinentes, dicho modelo fue ResNet [25], ya que es de las redes más utilizadas en este campo de investigación, de 238 artículos consultados de los últimos 12 años aparece en el 36,1 % de los mismos [5], siendo la más común y, además, de la que mejores resultados ofrece.

En cuanto a los datos, ya se comentó que el conjunto utilizado sería el de las 10000 imágenes de melanoma, siendo este una recopilación de varios conjuntos de ISIC, basado en 2 clases, melanoma y benigno, equilibrado y que no interfiere con los datos usados para el modelo de súper-resolución, por lo que no supone ningún riesgo a la hora de alterar los

resultados. Respecto a la división en entrenamiento y test, ya se comentó que el propio autor ofrece la división ya realizada.

La resolución original de las imágenes es de 300x300, por lo que se redujeron a 256x256 usando interpolación bicúbica, para mantener un formato estándar de resolución. El modelo utilizado tiene la siguiente forma:

```
base_model = ResNet50(
    include_top=False,
    weights='imagenet',
    input_shape=(256, 256, 3)
)

x = base_model.output
x = tf.keras.layers.GlobalAveragePooling2D()(x)
x = tf.keras.layers.Dropout(0.5)(x)
x = tf.keras.layers.Dense(1024, activation='relu')(x)
x = tf.keras.layers.Dropout(0.5)(x)
predictions = tf.keras.layers.Dense(1, activation='sigmoid')(x)

model = tf.keras.Model(inputs=base_model.input, outputs=predictions)
```

Básicamente, se utiliza la versión de ResNet con 50 capas, ya que es la más simple y la que nos permitiría realizar los experimentos con más fluidez, además de que el objetivo del trabajo no es obtener la máxima clasificación posible, sino ver que las técnicas ofrecen mejoras. Además, seguramente sería necesario optar por hardware más potente para usar versiones con más capas.

Además, se usa el modelo base, sin las capas convolucionales, al que se le añade un *global average pooling*, junto a capas *dropout* y totalmente conectadas, finalmente se sitúa la capa de salida que indicará la clasificación binaria. Se ha decidido no congelar capas para el entrenamiento, ya que las imágenes dermatoscópicas son muy diferentes a las de ImageNet, por lo que en los casos en los que se congelaban las primeras capas de la red el modelo no lograba aprender y siempre mostraba los mismos resultados.

En cuento al entrenamiento, la configuración fue la mostrada en la tabla 4.8.

Tabla 4.8: Configuración de entrenamiento de clasificación.

Número de épocas	20
Tamaño del Lote	8
Algoritmo de Optimización	Adam
Tasa de Aprendizaje	1×10^{-5}

Se han realizado 20 épocas, es decir, el modelo ve todo el conjunto de imágenes 20 veces, usando un tamaño de lote de 8 y el algoritmo de optimización Adam. Respecto a la tasa de aprendizaje, se ha usado una tasa muy pequeña, 1×10^{-5}, ya que estamos ajustando un modelo ya pre-entrenado sin congelar ninguna capa, lo que ayuda a garantizar que los ajustes no perturben demasiado los pesos ya aprendidos durante el pre-entrenamiento, permitiendo que el modelo retenga la información útil aprendida previamente mientras se adapta a las nuevas características del conjunto de datos dermatoscópico.

Además, se realizó *data augmentation* tradicional para reducir aún más el sobreajuste, dicho proceso consta de las modificaciones mostradas en la tabla 4.9 a las imágenes cada lote de forma aleatoria.

En resumen, se voltean tanto horizontalmente como verticalmente las imágenes, ya que la orientación de la herida ha de ser indiferente, además se le aplican ligeras rotaciones, desplazamientos y aumentos con el objetivo de variar la posición de la herida. Por último, se re-escalan los valores de las imágenes para que estén en el rango $[0, 1]$.

Tabla 4.9: Configuración del generador de imágenes para el entrenamiento con *data augmentation*

Parámetro	**Valor**
Re-escalado	1./255
Volteo Horizontal	Sí
Volteo Vertical	Sí
Rango de Rotación	20 grados
Rango de Desplazamiento Horizontal	0.1
Rango de Desplazamiento Vertical	0.1
Rango de Zoom	0.1
Modo de Relleno	'nearest'

Con todos los datos, el modelo y el *data augmentation* definido, se procedió en primer lugar a realizar un entrenamiento base con todos los datos sin aplicar ninguna técnica de las mencionada, los resultados fueron los siguientes:

Clase Real	Clase Predicha	
	Positivo	Negativo
Positivo	436	64
Negativo	8	492

Métrica	**Valor**
Exactitud	0.928
Sensibilidad	0.872
Especificidad	0.984
Precisión	0.982
Puntuación F1	0.923

Tabla 4.10: Resultados de la clasificación 256x256.

Estos fueron los mejores resultados obtenidos usando tan solo el conjunto de datos, podemos ver que el modelo presenta bastantes buenos resultados a pesar de no haber realizado un ajuste de hiperparámetros ni haber preprocesado las imágenes, lo cual abre la posibilidad a una mejora mayor usando estas técnicas. Si atendemos más a los detalles, podemos ver que es bastante bueno detectando los casos benignos pero algo peor con los melanomas, dado que su especificidad es de 0,984 y su sensibilidad es de 0,872.

Aplicación de súper-resolución

Tras esto, se convirtieron los dos conjuntos de imágenes de entrenamiento y de test en súper-resolución usando el modelo entrenado Real-ESRGAN, por lo que pasaron a tener un tamaño de 1024x1024, tras esto, se volvió a entrenar el modelo y los resultados fueron los siguientes:

Clase Real	Clase Predicha	
	Positivo	Negativo
Positivo	442	58
Negativo	16	484

Configuración	**Exactitud**	**Sensibilidad**	**Especificidad**	**Precisión**	**F1**
256x256	**0.928**	0.872	**0.984**	**0.982**	**0.923**
1024x1024SR	0.926	**0.884**	0.968	0.965	**0.923**

Tabla 4.11: Comparación de resultados entre aplicar súper-resolución a imágenes de 256x256.

Lo cual no fue satisfactorio, pues los resultados fueron muy parecidos a los originales, por un lado, la exactitud se vio ligeramente reducida, lo que quiere decir que, en total, menos muestras fueron catalogadas correctamente, sin embargo la detección de melanomas mejoró ligeramente como se puede ver en la sensibilidad, eso sí, a costa de la especificidad, lo que se traduce en una puntuación F1 similar.

Por tanto, viendo estos resultados no se puede concluir que aplicar súper-resolución aporte mejoras en la clasificación cuando se aplica a estas resoluciones. Esto se puede deber en parte a que al tratarse de altas resoluciones, como 1024x1024, el modelo introduzca ruido que realmente no aporte nada más al modelo que lo que hagan las imágenes con resolución 256x256.

A raíz de esto, se consultaron otras investigaciones en las que se hiciera algún trabajo con la clasificación [49], y no se utilizaban estas resoluciones, si no más cercanas a 128x128, 64x64, 32x32, etc. Fue por este motivo por el que se decidió adaptarse a esta metodología para comprobar si aplicar súper-resolución en estas dimensiones si que aportaba algo de beneficio.

En primer lugar, se redujeron las imágenes a 64x64 y se entrenó con este tamaño, los resultados obtenidos fueron los siguientes:

Clase Real	Clase Predicha	
	Positivo	Negativo
Positivo	433	67
Negativo	28	472

En este caso, al reducir la resolución si podemos ver como empeoran todas las métricas de forma generalizada, lo cual tiene sentido pues se pierde detalle, por lo que ahora, estas imágenes se escalan a 256x256 usando Real-ESRGAN para comprobar si ofrece alguna mejora en estos tamaños. Los resultados son los siguientes:

Clase Real	Clase Predicha	
	Positivo	Negativo
Positivo	441	59
Negativo	22	478

Configuración	**Exactitud**	**Sensibilidad**	**Especificidad**	**Precisión**	**F1**
64x64	0.905	0.866	0.944	0.939	0.901
256x256SR	0.919	**0.882**	0.956	0.952	0.916
256x256	**0.928**	0.872	**0.984**	**0.982**	**0.923**

Tabla 4.12: Comparación de resultados entre aplicar súper-resolución a imágenes de 64x64.

En la tabla 4.12 podemos comprobar una mejora notable, en todas las métricas respecto a la clasificación con resolución 64x64, destacando tanto la exactitud como la puntuación F1 que mejoran en un 1,4 % y 1,5 % respectivamente. Sin embargo, si lo comparamos con la clasificación con las imágenes base en 256x256 vemos como no se alcanza los mismos resultados, con lo que podemos deducir que el modelo no de súper-resolución no es infalible y no logra reconstruir la imagen con total detalle, algo esperable al fin y al cabo.

Sin embargo sucede algo curioso, de la misma forma que pasaba cuando se aplicaba súper-resolución a un tamaño de 1024x1024, la sensibilidad subía, en este caso también mejora respecto a las imágenes originales en 256x256, lo que puede ofrecernos un indicio de que el modelo de súper-resolución este captando algunos detalles relevantes de los melanomas y dándoles más importancia.

A pesar de esto, se buscó otro ejemplo en el que poder verificar esta mejora, para ello, se redujeron las imágenes a 32x32 y se repitió el proceso, los resultados obtenidos son:

Clase Real	Clase Predicha	
	Positivo	Negativo
Positivo	417	83
Negativo	24	476

De nuevo, al volver a bajar la resolución empeoran los resultados y, sobre todo, la sensibilidad, lo cual es realmente malo. Tras esto, se redujeron las originales a 128x128, estos fueron los resultados:

Clase Real	Clase Predicha	
	Positivo	Negativo
Positivo	431	69
Negativo	11	489

En este caso hay una mejora de un casi 3 % tanto en la exactitud como en la puntuación F1, por lo que puede que sea un caso interesante para comprobar los beneficios reales de la súper-resolución. Por tanto, se procede a escalar las imágenes de 32x32 a 128x128 usando Real-ESRGAN:

Clase Real	Clase Predicha	
	Positivo	Negativo
Positivo	440	60
Negativo	24	476

Configuración	**Exactitud**	**Sensibilidad**	**Especificidad**	**Precisión**	**F1**
32x32	0.893	0.834	0.952	0.945	0.886
128x128SR	0.916	**0.88**	0.952	0.948	0.913
128x128	**0.92**	0.862	**0.978**	**0.975**	**0.915**

Tabla 4.13: Comparación de resultados entre aplicar súper-resolución a imágenes de 32x32.

Como podemos ver en la tabla 4.13 mejora bastante, de forma que casi iguala los resultados obtenidos por las imágenes originales, lo cual es bastante bueno, pues indica que

trabajando con estas resoluciones el modelo de súper-resolución logra reconstruir mejor los detalles, lo que se traduce en una mejor labor de clasificación. De nuevo, pasa lo mismo con la sensibilidad, es mayor que la obtenida usando las imágenes originales, por lo que parece que la hipótesis que planteábamos podría tener sentido.

Vistos estos resultados, podemos concluir que aplicar súper-resolución a imágenes de resoluciones más bajas permite mejorar los resultados de clasificación, pero realmente queremos compararlo con los métodos tradicionales para ver si realmente aporta algo más de valor. Para ello, se escalaron las imágenes usando interpolación bicúbica y se volvieron a obtener resultados:

Clase Real	Clase Predicha	
	Positivo	Negativo
Positivo	434	66
Negativo	26	479

Configuración	**Exactitud**	**Sensibilidad**	**Especificidad**	**Precisión**	**F1**
128x128	**0.92**	0.862	**0.978**	**0.975**	**0.915**
128x128IN	0.913	0.868	0.958	0.953	0.908
128x128SR	0.916	**0.88**	0.952	0.948	0.913

Tabla 4.14: Comparación de resultados entre ESRGAN e interpolación bicúbica.

En este caso, los resultados, visibles en la tabla 4.14, son ligeramente peores que usando súper-resolución, destacando especialmente que la métrica que empeora es la sensibilidad, la cual es posiblemente la que menos queremos que empeore, además haciéndolo por un 2 %. A pesar de esto, es verdad que no aporta una mejora crucial a la clasificación respecto a aplicar interpolación, el cambio más notable es el de un 0,5 % en la puntuación F1 y la mejora de la sensibilidad, ambas métricas importantes para la detección del melanoma.

Parece que la aplicación de súper-resolución a imágenes de resoluciones más bajas puede mejorar los resultados de clasificación en la detección de melanomas, algo que no sucede cuando se aplica a resoluciones altas, esto se evidencia especialmente en la sensibilidad, que mejora con el uso de súper-resolución en comparación con métodos de interpolación más tradicionales. Sin embargo, no se observa una mejora drástica en todas las métricas, y en algunos casos, como en la resolución de 128x128, los resultados usando interpolación bicúbica no están muy lejos de los obtenidos con súper-resolución, a pesar de mejorarlos.

A pesar no aportar una mejora diferencial a la clasificación, la calidad final de las imágenes es mucho mejor que con métodos tradicionales, por lo que puede ser de especial interés para los profesionales, ya que pueden mejorar la calidad de sus imágenes y tener una base con más detalle sobre la que tomar una decisión profesional.

En resumen, aunque la súper-resolución puede ser beneficiosa en la mejora de la clasificación de melanomas, su efectividad puede variar según la resolución inicial de las imágenes y la calidad del modelo de súper-resolución utilizado, además, es importante considerar el costo computacional de aplicar súper-resolución, especialmente en imágenes de alta resolución, tal y como podemos deducir si vemos la tabla 4.15 que muestra todos los resultados de esta sección.

Configuración	Exactitud	Sensibilidad	Especificidad	Precisión	F1
256x256	**0.928**	0.872	**0.984**	**0.982**	**0.923**
1024x1024SR	0.926	**0.884**	0.968	0.965	0.923
64x64	0.905	0.866	0.944	0.939	0.901
256x256SR	0.919	0.882	0.956	0.952	0.916
32x32	0.893	0.834	0.952	0.945	0.886
128x128	0.92	0.862	0.978	0.975	0.915
128x128SR	0.916	0.88	0.952	0.948	0.913
128x128IN	0.913	0.868	0.958	0.953	0.908

Tabla 4.15: Comparación de resultados de diferentes configuraciones de súper-resolución.

Aplicación de generación de datos sintéticos

Una vez concluida la experimentación con la súper-resolución, se procedió a realizar el mismo proceso con la generación de datos sintéticos, tomando de nuevo como partida el primer entrenamiento base con todos los datos, cuyos resultados se muestran en la tabla 4.10.

El experimento planteado consiste en duplicar el tamaño actual del conjunto de datos usando StyleGAN, de forma que se generen tanto imágenes de melanoma como benignas para comprobar si duplicando el número de imágenes se logra mejorar los resultados. Para ello, existían dos posibilidades, entrenar dos modelos, uno para imágenes benignas y otro para melanomas o aprovechar que StyleGAN tienen la posibilidad, tal y como vimos, de convertirse en una GAN condicional, de forma que se pueda entrenar el mismo modelo tanto para generar un tipo de imágenes como de otras.

Finalmente, la opción escogida fue entrenar un único modelo condicional, ya que permitiría aprovechar la capacidad de generalización del modelo para aprender características comunes y específicas de ambos tipos de imágenes, a parte de de simplificar el proceso de entrenamiento y reducir la complejidad computacional y de gestión de modelos. Además, al entrenar un único modelo, se podría aprovechar mejor el conjunto de datos completo, lo que podría resultar en una mejora de la calidad y diversidad de las imágenes generadas.

Para ello, tan solo es necesario etiquetar el conjunto de datos indicando a que clase pertenece cada imagen, en este caso, tan solo se usará para entrenar el conjunto de entrenamiento, pues usar el conjunto de test para entrenar un GAN y luego añadir las imágenes generadas al de entrenamiento estaría introduciendo información nueva de otras imágenes totalmente nuevas, algo que no tiene sentido, pues queremos asegurarnos de que el modelo aprenda únicamente de la información disponible en los datos de entrenamiento originales y no de información adicional que se pueda extraer del conjunto de test.

Tras realizar el entrenamiento de StyleGAN, se añadieron al conjunto de entrenamiento 5000 imágenes de cada clase generados con el propio modelo y se volvió a entrenar el modelo y los resultados obtenidos son los siguientes:

Clase Real	Clase Predicha	
	Positivo	Negativo
Positivo	440	60
Negativo	16	484

Configuración	**Exactitud**	**Sensibilidad**	**Especificidad**	**Precisión**	**F1**
Completo	**0.928**	0.872	**0.984**	**0.982**	**0.923**
Completo + Gen	0.924	**0.88**	0.968	0.964	0.92

Tabla 4.16: Comparación de resultados incluyendo imágenes generadas al conjunto completo.

Los resultados obtenidos, visibles en la tabla 4.16, fueron prácticamente los mismos, lo que nos muestra que usando un conjunto de datos tan grande el modelo no logra extraer información adicional de los datos obtenidos y ofrecer una mayor diversidad de imágenes, de hecho, parece que introduce incluso ruido a los datos pues el rendimiento de la clasificación se ve afectado siendo reducido ligeramente.

Tras asumir estos resultados, tal y como se hizo para la súper-resolución, se consultaron trabajos relacionados y se constató trabajo relacionado [50] donde se aplica generación de datos mediante GANs a conjuntos más pequeños y con una resolución menor, algo que tiene más sentido pues la adición de datos generados a conjuntos reducidos puede tener un impacto mayor y ayudar a la generalización, traduciéndose en un rendimiento mayor.

Por tanto, se realizó un muestreo del conjunto de datos de entrenamiento de forma que se escogieron 100 imágenes de cada clase que se redujeron a 64x64 usando interpolación bicúbica y ese pasó a ser el conjunto reducido con el que se entrenó el modelo de clasificación, el conjunto de test tan solo fue reducido a 64x64, no se eliminaron imágenes, ya que no afecta en nada al entrenamiento, solo a la evaluación. Los resultados del entrenamiento con 200 imágenes fueron los siguientes:

Clase Real	Clase Predicha	
	Positivo	Negativo
Positivo	339	161
Negativo	84	416

Métrica	**Valor**
Exactitud	0.755
Sensibilidad	0.678
Especificidad	0.832
Precisión	0.801
Puntuación F1	0.734

Tabla 4.17: Resultados de la clasificación 64x64 del conjunto reducido.

Como era de esperar, los resultados se vieron muy afectados, se pasó a tener un 75,5 % de exactitud y la métrica más afectada fue la sensibilidad, que mide la capacidad del modelo para detectar positivos verdaderos, obtuvo un valor de 67,8 %, lo que indica una reducción en la capacidad del modelo para detectar correctamente casos de melanoma, algo muy malo.

Además, podemos ver en la figura 4.17 como evoluciona la exactitud durante el entrenamiento, se puede comprobar como el modelo apenas logra aprender, llegando a sufrir incluso un ligero sobreajuste, no llegado a alcanzar ni siquiera el 80 %.

Tras esto, se volvió a entrenar otro StyleGAN, de nuevo, condicional, en este caso con las

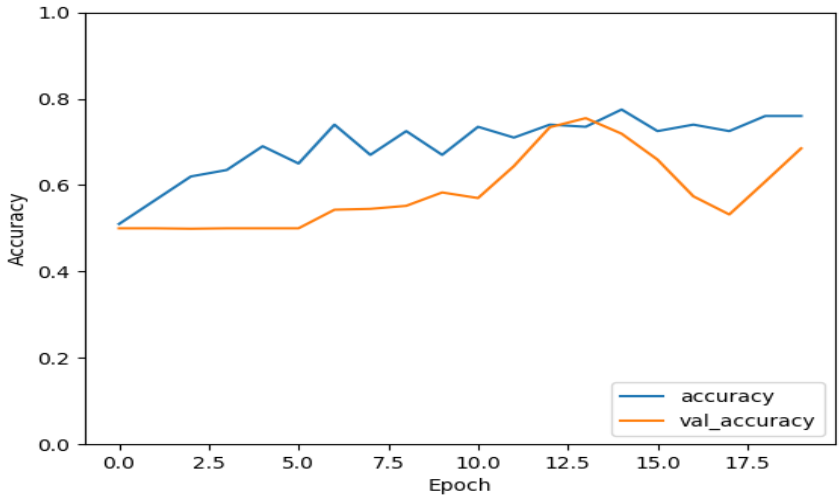


Figura 4.17: Evolución de la clasificación en el entrenamiento del conjunto reducido 64x64.

200 imágenes muestreadas y con resolución 64x64. Una vez acabado, se le añadieron al conjunto reducido 100 imágenes generadas de cada clase, es decir, se duplicó su tamaño y se volvió a entrenar obteniendo los siguientes resultados:

Clase Real	Clase Predicha	
	Positivo	Negativo
Positivo	433	67
Negativo	59	441

Configuración	**Exactitud**	**Sensibilidad**	**Especificidad**	**Precisión**	**F1**
Reducido	0.755	0.678	0.832	0.801	0.734
Reducido + Gen	**0.874**	**0.866**	**0.882**	**0.88**	**0.873**

Tabla 4.18: Comparación de resultados incluyendo imágenes generadas al conjunto reducido.

En este caso los resultados obtenidos en la tabla 4.18 son muy satisfactorios, encontramos una mejora clara en todas las métricas, llegando a alcanzar un 87,4 % de exactitud, una subida del 12 %, lo cual supone una mejora increíble en el rendimiento del modelo usando exactamente las mismas imágenes originales y enriqueciéndolas con la generación de datos sintéticos. La mejora más notable, además, es en la sensibilidad, con una subida del 18,8 %, llegando a alcanzar un 86,6 % y, por tanto, detectando con mucho más éxito los melanomas.

Respecto al entrenamiento, podemos ver en la figura 4.18 como la evolución de la exactitud tiene mucha mejor apariencia, generaliza mucho mejor y logra evitar el sobreajuste, obteniendo además mejores resultados, lo que nos permite comprobar las claras mejoras que ofrece esta técnica sobre el entrenamiento con conjuntos de datos pequeños.

En conclusión, los resultados obtenidos muestran que la generación de datos sintéticos es una estrategia efectiva para mejorar el rendimiento de los modelos de aprendizaje automático, especialmente cuando se aplican a conjuntos de datos más pequeños, teniendo impacto en la capacidad de generalización del modelo. A pesar de esto, en conjuntos de

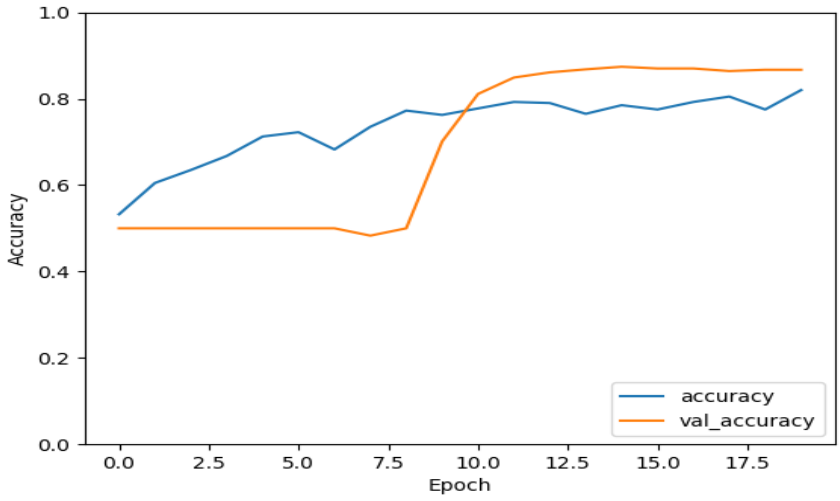


Figura 4.18: Evolución de la clasificación en el entrenamiento del conjunto reducido 64x64 con imágenes generadas.

datos grandes la adición de datos generados puede no ofrecer beneficios significativos e incluso introducir ruido.

Resumiendo, la combinación de datos reales con datos generados puede aumentar la diversidad y cantidad de muestras de entrenamiento, lo que a su vez puede mejorar la capacidad del modelo para detectar patrones y realizar predicciones precisas en nuevas instancias, permitiendo por tanto mejorar en labores tan importantes como la detección del melanoma.

Se muestra un resumen de todos los resultados en la tabla 4.19.

Configuración	**Exactitud**	**Sensibilidad**	**Especificidad**	**Precisión**	**F1**
Completo	**0.928**	0.872	**0.984**	**0.982**	**0.923**
Completo + Gen	0.924	**0.88**	0.968	0.964	0.92
Reducido	0.755	0.678	0.832	0.801	0.734
Reducido + Gen	0.874	0.866	0.882	0.88	0.873

Tabla 4.19: Comparación de resultados incluyendo imágenes generadas.

CAPÍTULO 5

Conclusiones y lineas de trabajo futuras

Una vez finalizado el proyecto, se exponen las conclusiones obtenidas del mismo y las posibles lineas de trabajo futuras que pueden ser abordadas como futuros proyectos.

5.1. Conclusiones

La realización de este Trabajo Fin de Máster ha permitido combinar todos los conocimientos adquiridos y desarrollados durante la realización del máster y el grado, además de permitir expandirlo mediante el estudio de nuevas disciplinas y enfocarlo a un caso con impacto real como es la detección del melanoma, de forma que estos conocimientos se han vuelto efectivos y han tenido un impacto real a la hora de aplicarse durante este proyecto.

Como resultado, se ha indagado en el estado del arte y se ha realizado un revisión enriquecedora que, sin duda, será de utilidad en el futuro profesional ya que ha permitido tratar con tecnología puntera como es la generación de datos mediante inteligencia artificial, algo de especial relevancia a día de hoy.

Además, se ha experimentado con la súper-resolución, evaluando los principales modelos usados en el estado del arte, aplicando métricas especializadas y demostrando que es posible obtener una mejora en la clasificación del 3 % aplicando esta técnica a imágenes con baja resolución, en concreto, 32x32, mientras que aplicarla a imágenes con resoluciones más altas puede tener sentido para que los profesionales puedan apreciar mejor los detalles y tomar decisiones sobre muestras de mayor calidad.

Por otro lado, también se ha experimentado con la generación de datos sintéticos, evaluando las principales arquitecturas en el estado del arte e intentando resolver un problema tan común como es el *mode collapse*. Aplicando esta técnica a la clasificación, se ha logrado demostrar que puede mejorar notablemente el rendimiento sobre conjuntos muy limitados, llegando a alcanzar un 12 % más de aciertos, mientras que aplicarla a conjuntos grandes no ha logrado aportar ningún beneficio más alla de introducir ruido a los datos.

Por tanto, se ha demostrado que aplicando técnicas novedosas de inteligencia artificial es posible optimizar el proceso de detección del melanoma, lo que tiene un impacto significativo en la precisión y eficiencia de este proceso crucial para la salud pública. Además, el

trabajo ha permitido no solo profundizar en el conocimiento teórico de diversas técnicas relevantes a nivel profesional, sino también adquirir habilidades prácticas en su implementación y evaluación, ya que estas han sido necesarias para el desarrollo del mismo.

Este proyecto ha resaltado la vital importancia de la interdisciplinariedad en la investigación y la aplicación de la inteligencia artificial en el ámbito médico, donde la colaboración entre expertos de distintos ámbitos, como dermatología, informática y estadística es fundamental para avanzar en el desarrollo de soluciones efectivas y confiables que mejoren la vida de los demás.

A nivel personal, este proyecto ha supuesto una experiencia realmente interesante que ha fortalecido mis habilidades analíticas, de resolución de problemas y de comunicación. Además, me ha permitido consolidar mi interés en la inteligencia artificial y en las novedosos modelos generativos, dándoles un uso profesional y de interés más allá del ocio, permitiendo llegar a mejorar la vida de las personas.

En resumen, este Trabajo Fin de Máster ha sido una oportunidad enriquecedora para aplicar y ampliar los conocimientos adquiridos, contribuyendo de manera significativa al campo de la detección del melanoma y sentando las bases para futuras investigaciones y aplicaciones prácticas en el área de la salud.

5.2. Lineas de trabajo futuras

Vistas las conclusiones obtenidas con el trabajo, a continuación se proponen una serie de lineas de trabajo futuras que pueden ser exploradas para avanzar en el campo de la detección del melanoma y la inteligencia artificial.

1. **Estudio del uso del discriminador como modelo de clasificación**: una línea futura de investigación interesante podría ser explorar el potencial del discriminador como modelo de clasificación en sí mismo. Este enfoque implica utilizar el discriminador entrenado en GAN para clasificar imágenes de melanomas como benignas o malignas de forma que sea posible investigar la eficacia de esta técnica en comparación con otros modelos de clasificación convencionales, así como su capacidad para generalizar a nuevos conjuntos de datos. Esta linea es realmente interesante pues le da otro posible uso a las GANs en el ámbito médico.

2. **Uso de GANs para generar conjuntos de datos sintéticos y mitigar problemas de privacidad**: una dirección prometedora para la investigación futura es la exploración de cómo las GANs pueden utilizarse para generar conjuntos de datos sintéticos que preserven la privacidad de los pacientes sin comprometer el rendimiento de los modelos de inteligencia artificial. Al entrenar una GAN en datos médicos reales, se puede generar un conjunto de datos sintéticos que capture las características clave de los datos originales sin contener información privada identificable. Estos datos sintéticos pueden utilizarse para entrenar modelos de detección de enfermedades, como el melanoma, sin riesgo de revelar información personal de los pacientes.

3. **Creación de un modelo de texto a imagen médica**: esta línea de investigación implica desarrollar un modelo capaz de generar imágenes médicas a partir de descripciones textuales, como informes clínicos o notas de médicos, por lo que se

centraría en la aplicación de técnicas de generación de imágenes, como redes neuronales generativas, modelos difusos, etc., para sintetizar imágenes médicas realistas y especialmente útiles a partir de texto descriptivo, pudiendo ser aplicado, por ejemplo, en la educación médica, la investigación clínica y el desarrollo de herramientas de entrenamiento para profesionales.

4. **Aplicación de inteligencia artificial en otros campos médicos**: se pueden explorar oportunidades similares en otros campos médicos, como la radiología, la patología o la oncología, donde el análisis de imágenes juega un papel crucial en el diagnóstico y tratamiento de enfermedades.

5. **Investigación en generación de datos sintéticos**: a pesar de los avances demostrados, el desafío del *mode collapse* en la generación de datos sintéticos sigue siendo una preocupación, por lo que se podría investigar para abordar este problema de manera más efectiva, desarrollando técnicas que mitiguen o eliminen este fenómeno y permitan una generación de datos más diversa y representativa.

6. **Validación clínica y despliegue práctico**: para que las soluciones desarrolladas tengan un impacto real en la práctica clínica, es necesario realizar estudios de validación clínica rigurosos y profesionales, lo que implica poner mayores esfuerzos en obtener mejores resultados y colaborar estrechamente con profesionales de la salud para evaluar la efectividad y la seguridad de las herramientas desarrolladas en entornos reales.

Todas estas áreas de investigación contribuyen significativamente al avance de la detección del melanoma y la aplicación de la inteligencia artificial en medicina, abriendo nuevas posibilidades para mejorar el diagnóstico y tratamiento de esta enfermedad y otras condiciones médicas, el principal objetivo de este proyecto.

Bibliografía

[1] R. Gordon, "Skin cancer: An overview of epidemiology and risk factors," *Seminars in Oncology Nursing*, vol. 29, 2013.

[2] L. E. Davis, S. C. Shalin, and A. J. Tackett, "Current state of melanoma diagnosis and treatment," 2019.

[3] R. L. Siegel, K. D. Miller, and A. Jemal, "Cancer statistics, 2020," *CA: A Cancer Journal for Clinicians*, vol. 70, 2020.

[4] F. Grignaffini, F. Barbuto, L. Piazzo, M. Troiano, P. Simeoni, F. Mangini, G. Pellacani, C. Cantisani, and F. Frezza, "Machine learning approaches for skin cancer classification from dermoscopic images: A systematic review," 2022.

[5] M. K. Hasan, M. A. Ahamad, C. H. Yap, and G. Yang, "A survey, review, and future trends of skin lesion segmentation and classification," 2023.

[6] K. Munir, H. Elahi, A. Ayub, F. Frezza, and A. Rizzi, "Cancer diagnosis using deep learning: A bibliographic review," *Cancers*, vol. 11, 2019.

[7] C. Europea and D. G. de Justicia y Consumidores, *RGPD : nuevas oportunidades, nuevas obligaciones : lo que las empresas tienen que saber sobre el Reglamento general de protección de datos de la Unión Europea*. Oficina de Publicaciones, 2018.

[8] E. J. Topol, "High-performance medicine: the convergence of human and artificial intelligence," 2019.

[9] A. Masood and A. A. Al-Jumaily, "Computer aided diagnostic support system for skin cancer: A review of techniques and algorithms," 2013.

[10] G. Harshvardhan, M. K. Gourisaria, M. Pandey, and S. S. Rautaray, "A comprehensive survey and analysis of generative models in machine learning," *Computer Science Review*, vol. 38, 2020.

[11] M. Sallam, "Chatgpt utility in healthcare education, research, and practice: Systematic review on the promising perspectives and valid concerns," 2023.

[12] F. Furger, L. Amruthalingam, A. Navarini, and M. Pouly, "Applications of generative adversarial networks to dermatologic imaging," vol. 12294 LNAI, 2020.

[13] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-realistic single image super-resolution using a generative adversarial network," vol. 2017-January, 2017.

[14] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*. Prentice Hall,

3 ed., 2010.

[15] K. P. Murphy, *Machine learning : a probabilistic perspective.* MIT Press, 2013.

[16] Y. Zhang, "Support vector machine classification algorithm and its application," vol. 308 CCIS, 2012.

[17] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning.* MIT Press, 2016.

[18] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The Bulletin of Mathematical Biophysics*, vol. 5, 1943.

[19] I. H. Sarker, "Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions," 2021.

[20] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," 2018.

[21] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," pp. 248–255, 2009.

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," vol. 2, 2012.

[23] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015.

[24] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," vol. 07-12-June-2015, 2015.

[25] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," vol. 2016-December, 2016.

[26] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," vol. 3, 2014.

[27] M. Mirza and S. Osindero, "Conditional generative adversarial nets," 2014.

[28] H. Al-Mekhlafi and S. Liu, "Single image super-resolution: a comprehensive review and recent insight," 2024.

[29] R. Wirth, "Crisp-dm : Towards a standard process model for data mining," *Proceedings of the Fourth International Conference on the Practical Application of Knowledge Discovery and Data Mining*, pp. 29–39, 2000.

[30] P. Tschandl, C. Rosendahl, and H. Kittler, "Data descriptor: The ham10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions," *Scientific Data*, vol. 5, 2018.

[31] M. Combalia, N. C. F. Codella, V. Rotemberg, B. Helba, V. Vilaplana, O. Reiter, C. Carrera, A. Barreiro, A. C. Halpern, S. Puig, and J. Malvehy, "Bcn20000: Dermoscopic lesions in the wild," 8 2019.

[32] M. H. Javid, "Melanoma skin cancer dataset of 10000 images," 2022.

[33] M. H. Javid, W. Jadoon, H. Ali, and M. D. Ali, "Design and analysis of an improved deep ensemble learning model for melanoma skin cancer classification," pp. 1–6, 2023.

[34] I. A. El-Fattah, A. M. Ali, W. El-Shafai, T. E. Taha, and F. E. A. El-Samie, "Deep-learning-based super-resolution and classification framework for skin disease detection applications," *Optical and Quantum Electronics*, vol. 55, 2023.

[35] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee, "Enhanced deep residual networks for single image super-resolution," vol. 2017-July, 2017.

[36] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and C. C. Loy, "Esrgan: Enhanced super-resolution generative adversarial networks," vol. 11133 LNCS, 2019.

[37] X. Wang, L. Xie, C. Dong, and Y. Shan, "Real-esrgan: Training real-world blind super-resolution with pure synthetic data," vol. 2021-October, 2021.

[38] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2016.

[39] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," vol. 2019-June, 2019.

[40] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," 2020.

[41] T. Karras, M. Aittala, J. Hellsten, S. Laine, J. Lehtinen, and T. Aila, "Training generative adversarial networks with limited data," vol. 2020-December, 2020.

[42] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[43] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[44] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," vol. 2017-December, 2017.

[45] T. pandas development team, "pandas-dev/pandas: Pandas," Feb. 2020.

[46] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: A system for large-scale machine learning," 2016.

[47] H. Thanh-Tung and T. Tran, "Catastrophic forgetting and mode collapse in gans," 2020.

[48] S. A. Hicks, I. Strümke, V. Thambawita, M. Hammou, M. A. Riegler, P. Halvorsen, and S. Parasa, "On evaluation metrics for medical applications of artificial intelligence," *Scientific Reports*, vol. 12, 2022.

[49] W. Ahmad, H. Ali, Z. Shah, and S. Azmat, "A new generative adversarial network for medical images super resolution," *Scientific Reports*, vol. 12, 2022.

[50] J. Röglin, K. Ziegeler, J. Kube, F. König, K. G. Hermann, and S. Ortmann, "Improving classification results on a small medical dataset using a gan; an outlook for dealing with rare disease datasets," *Frontiers in Computer Science*, vol. 4, 2022.

Apéndices

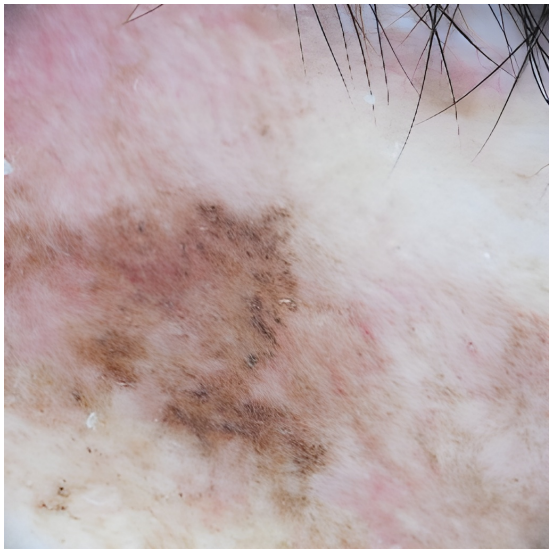
APÉNDICE A

Imágenes usadas en la memoria

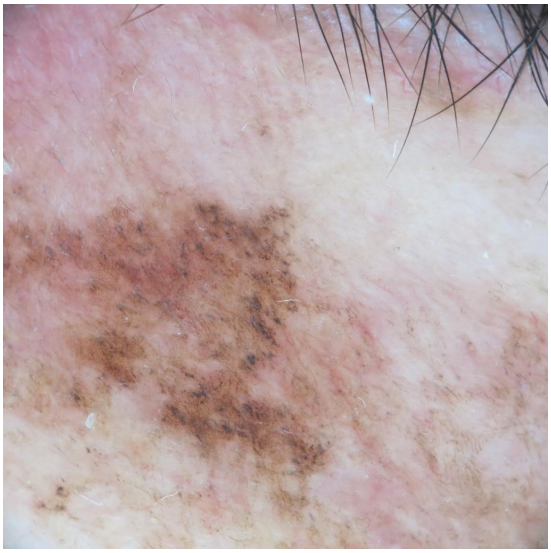
Figura A.1: Imágenes de referencia en baja resolución.

(a) EDSR

(b) SRGAN



(c) ESRGAN



(d) Original

Figura A.2: Comparación de imágenes de los modelos de súper-resolución pre-entrenados.

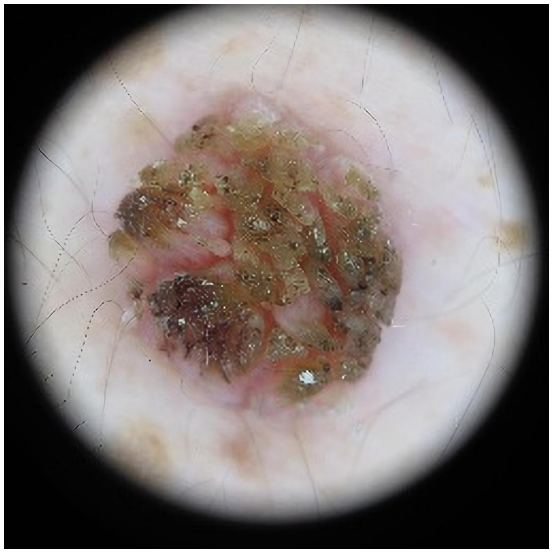
(a) EDSR

(b) SRGAN

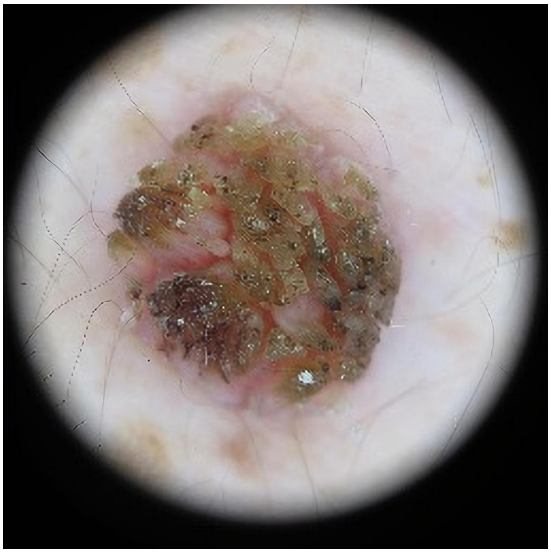
(c) ESRGAN

(d) Original

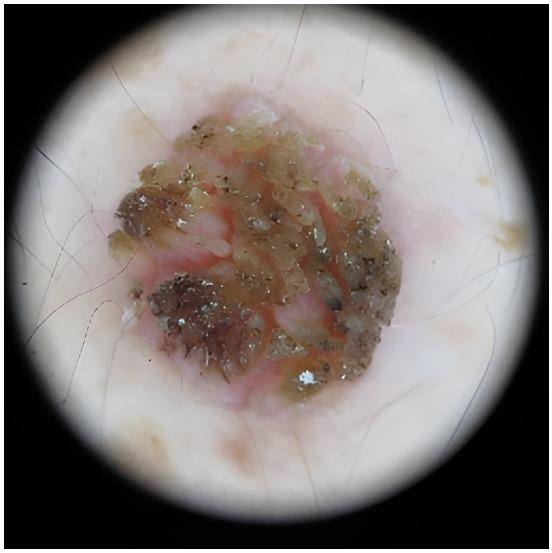
Figura A.3: Comparación de imágenes de los modelos de súper-resolución pre-entrenados.



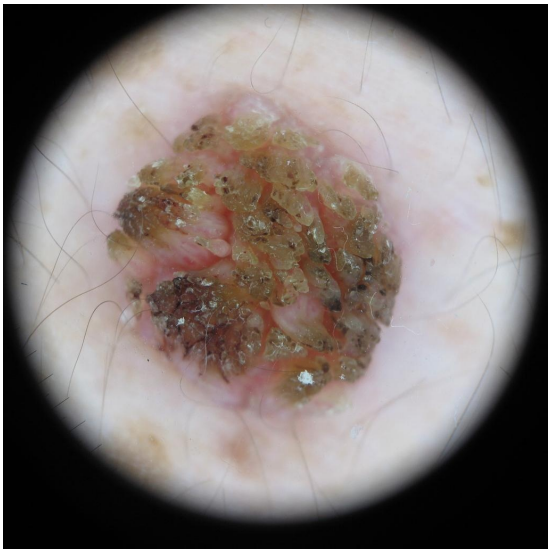
(a) EDSR



(b) SRGAN

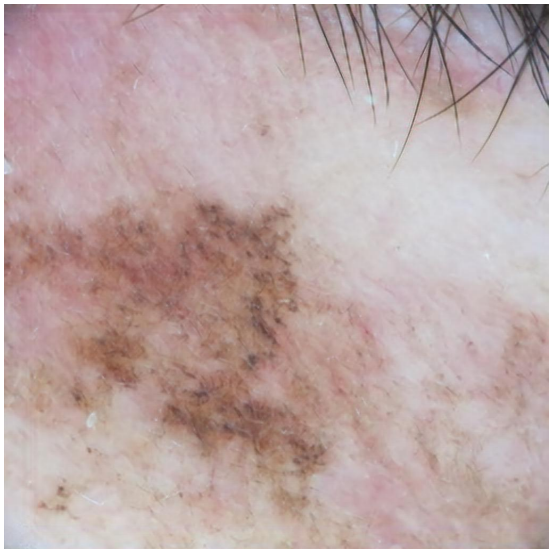


(c) ESRGAN

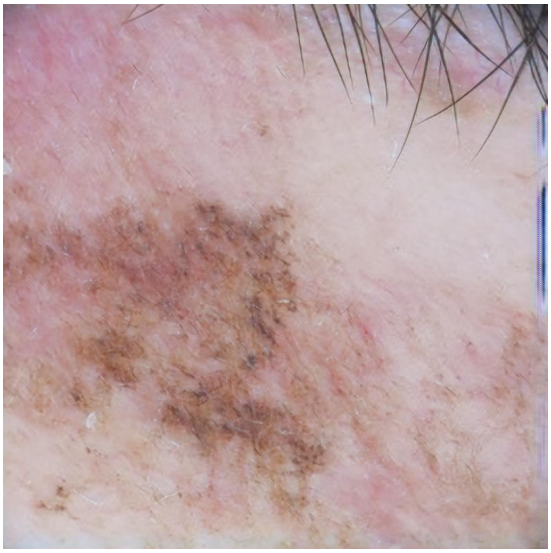


(d) Original

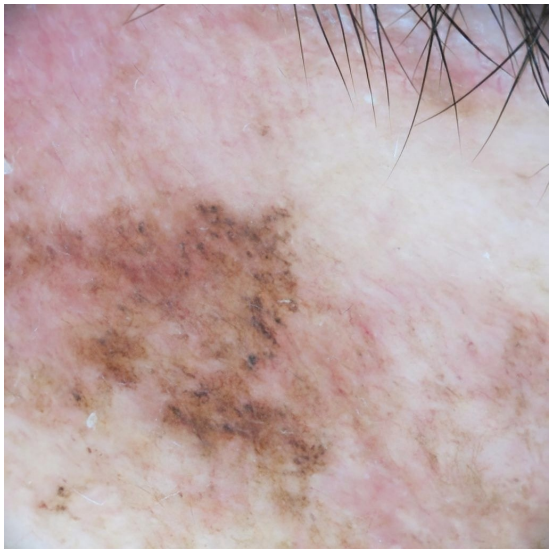
Figura A.4: Comparación de imágenes de los modelos de súper-resolución pre-entrenados.



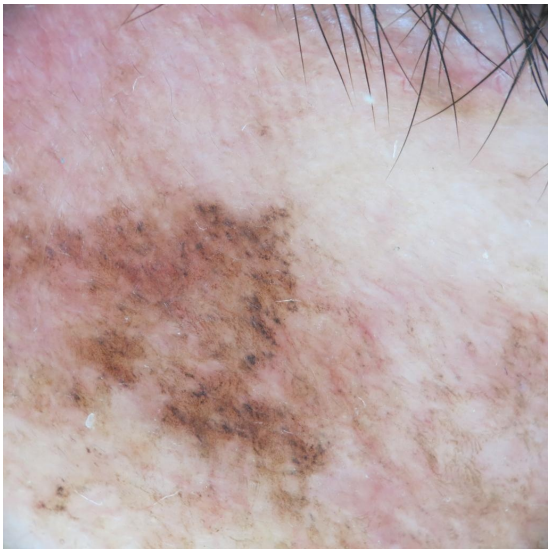
(a) EDSR



(b) SRGAN



(c) ESRGAN



(d) Original

Figura A.5: Comparación de imágenes de los modelos de súper-resolución entrenados con BCN20000.

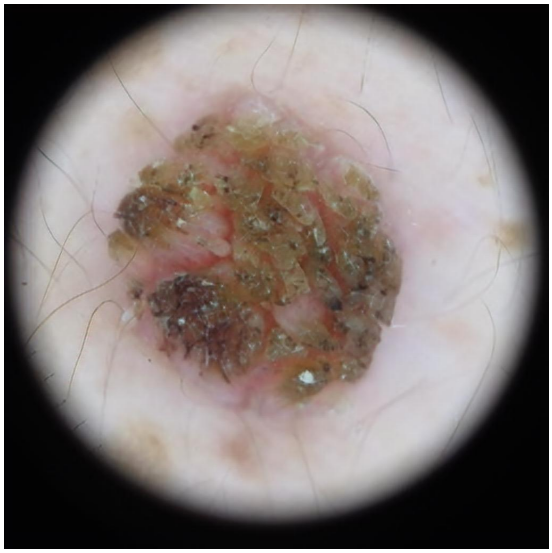
(a) EDSR

(b) SRGAN

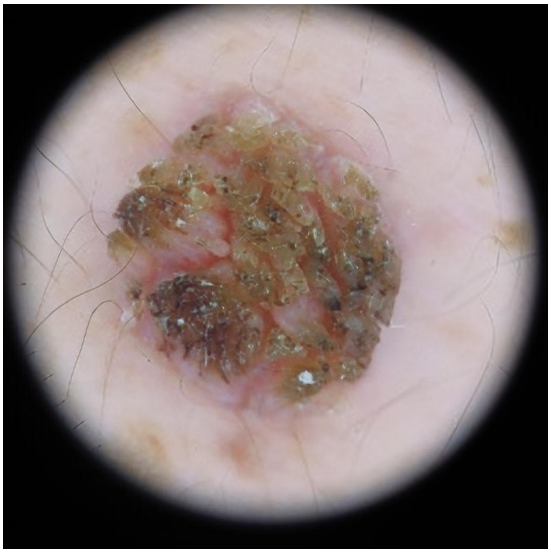
(c) ESRGAN

(d) Original

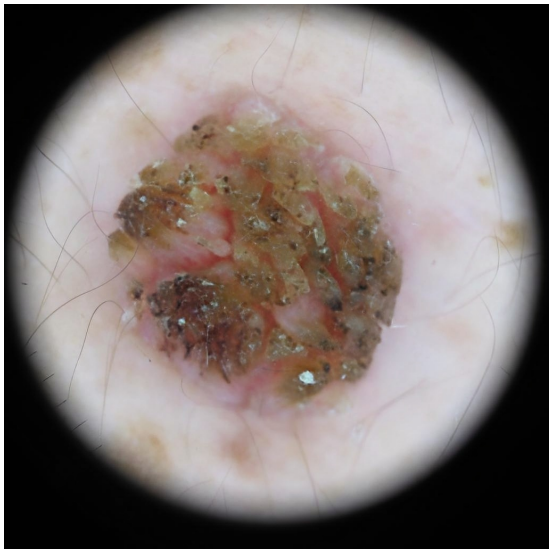
Figura A.6: Comparación de imágenes de los modelos de súper-resolución entrenados con BCN20000.



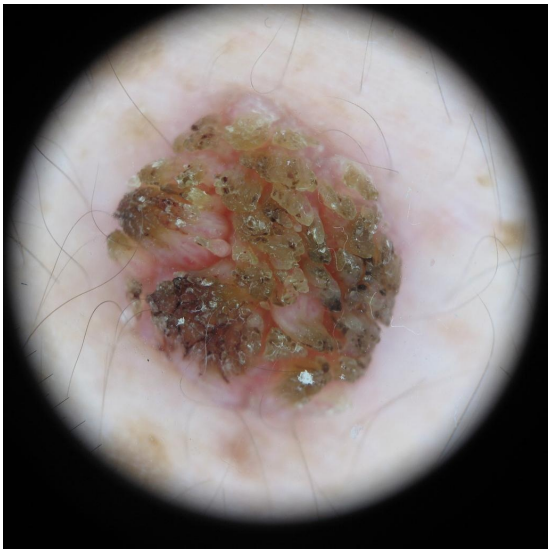
(a) EDSR



(b) SRGAN



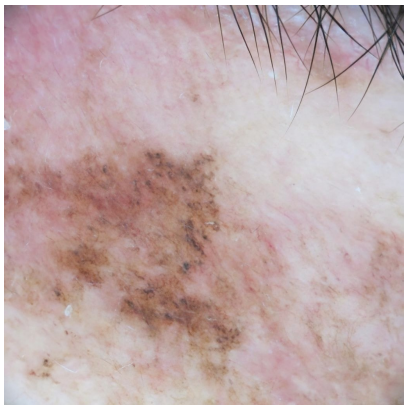
(c) ESRGAN



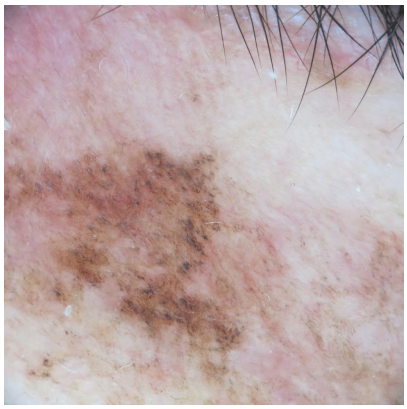
(d) Original

Figura A.7: Comparación de imágenes de los modelos de súper-resolución entrenados con BCN20000.

(a) Interpolación bicúbica



(b) ESRGAN



(c) Original

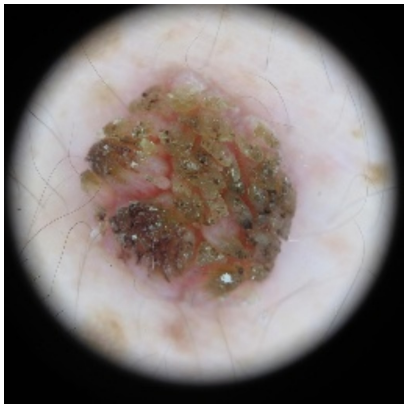
Figura A.8: Comparación de interpolación bicúbica con ESRGAN.

(a) Interpolación bicúbica

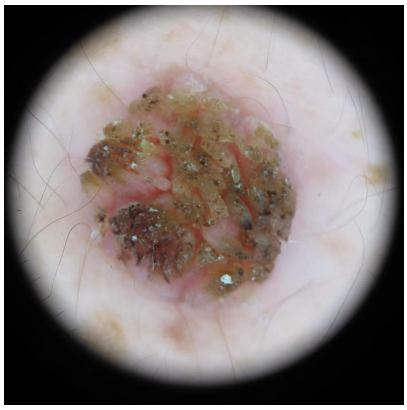
(b) ESRGAN

(c) Original

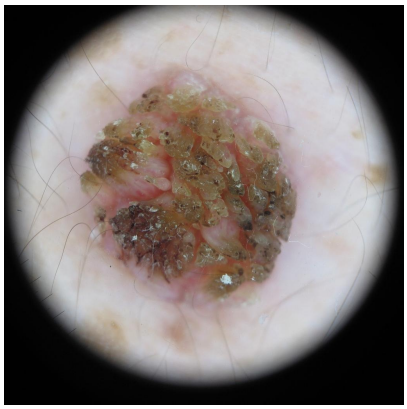
Figura A.9: Comparación de interpolación bicúbica con ESRGAN.



(a) Interpolación bicúbica



(b) ESRGAN



(c) Original

Figura A.10: Comparación de interpolación bicúbica con ESRGAN.

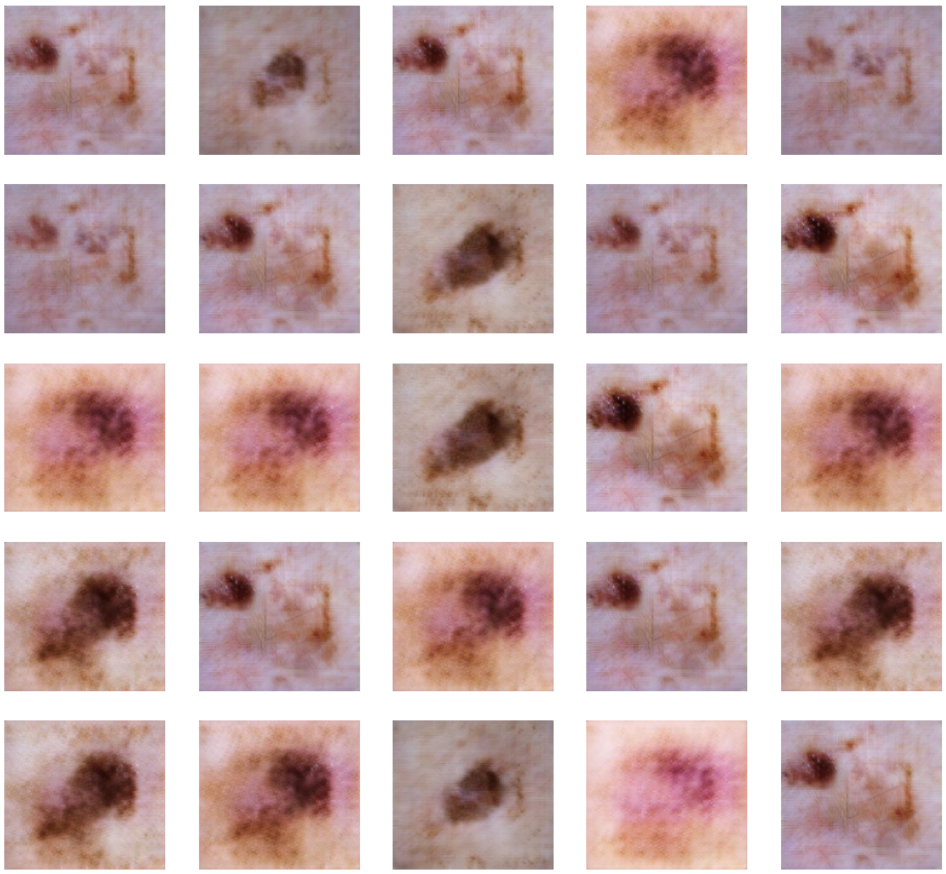


Figura A.11: Imágenes generadas por DCGAN.

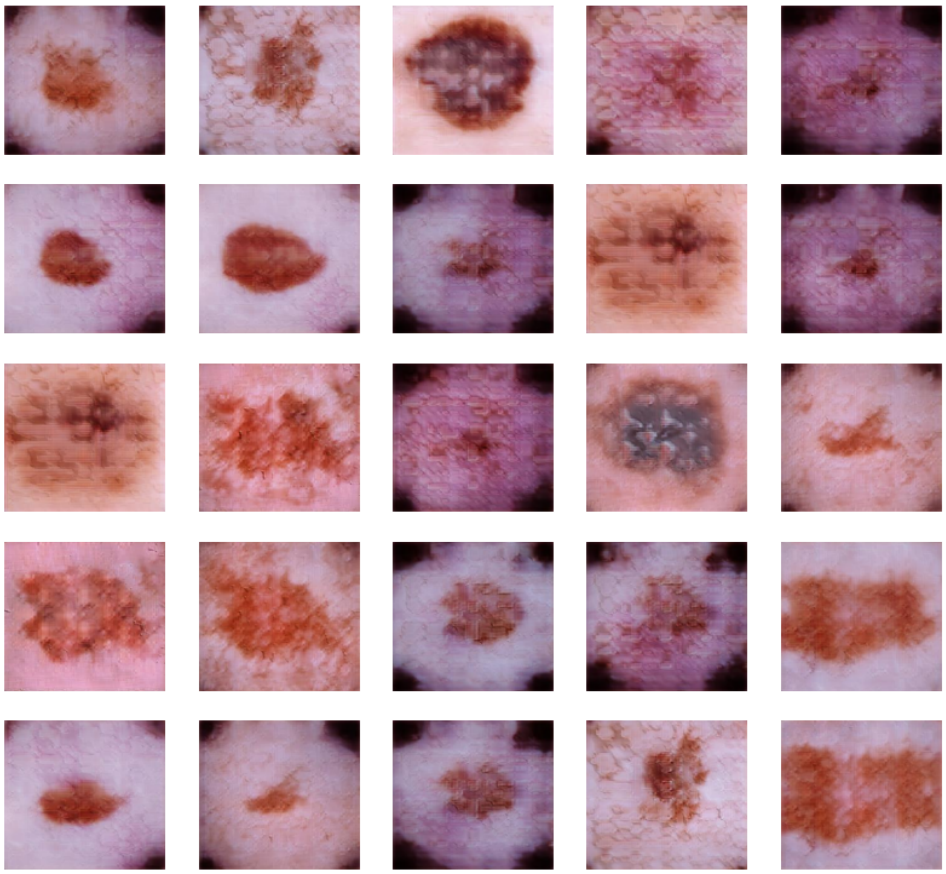


Figura A.12: Imágenes generadas por la revisión de DCGAN.

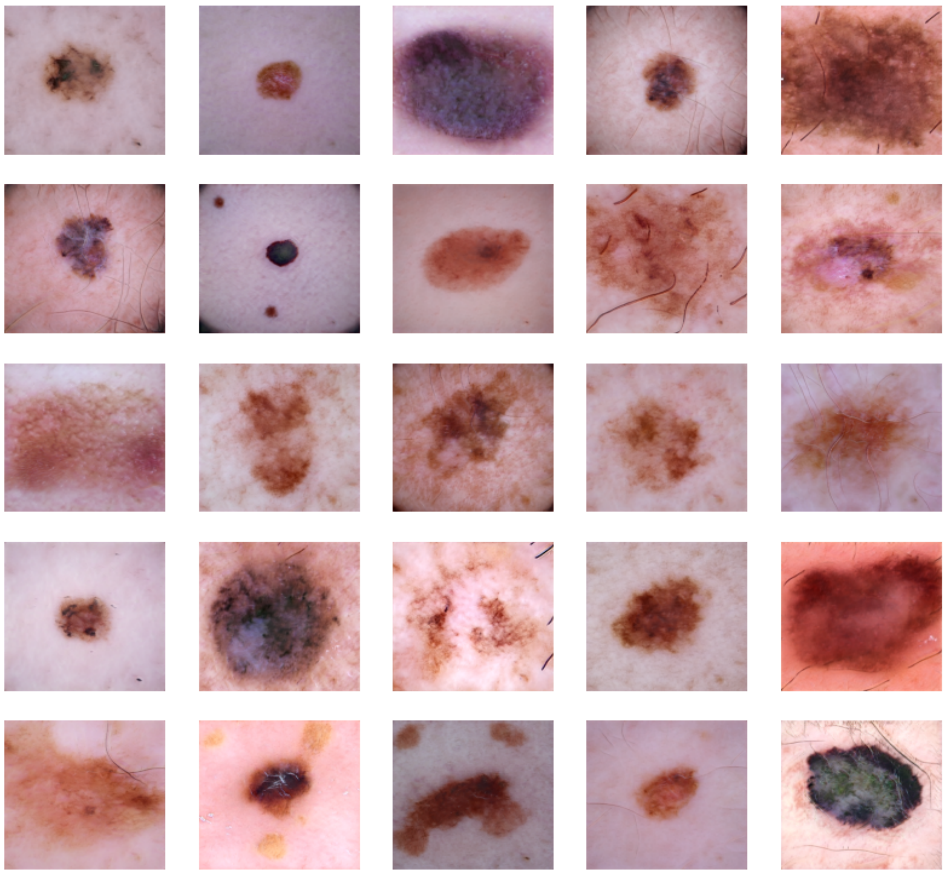


Figura A.13: Imágenes generadas por StyleGAN.

UNIVERSIDAD DE MÁLAGA | uma.es

E.T.S. DE INGENIERÍA INFORMÁTICA

E.T.S de Ingeniería Informática
Bulevar Louis Pasteur, 35
Campus de Teatinos
29071 Málaga