UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
PROGRAMA DE MAESTRÍA Y DOCTORADO EN INGENIERÍA
INGENIERÍA ELÉCTRICA - PROCESAMIENTO DIGITAL DE SEÑALES

Detección de Lesiones en Imágenes Dermatológicas con Técnicas de Aprendizaje Profundo

TESIS
QUE PARA OPTAR POR EL GRADO DE:
Maestro en Ingeniería

PRESENTA:
José Carlos Moreno Tagle

TUTOR
Dr. Boris Escalante Ramírez
PROGRAMA DE MAESTRÍA Y DOCTORADO EN INGENIERÍA

CIUDAD UNIVERSITARIA, CDMX, ENERO 2020

JURADO ASIGNADO:

Presidente:	Dr. Arámbula Cosío Fernando
Secretario:	Dr. Fuentes Pineda Gibran
1er. Vocal:	Dr. Escalante Ramírez Boris
2do. Vocal:	Dra. Olveres Montiel Jimena
3er. Vocal:	Dr. Padilla Castañeda Miguel Ángel

La tesis se realizó en el Laboratorio Avanzado de Procesamiento de Imágenes de la Facultad de Ingeniería de la UNAM.

TUTOR DE TESIS:

Dr. Boris Escalante Ramírez

_____

Dedicatoria

A mis amados padres, Carlos Héctor Moreno Flores y Patricia Tagle Balderas, porque la conclusión de esta etapa es más un logro de ustedes que mío, pues su vida la han dedicado a educar, cuidar y proveer de todo lo necesario a cada uno de sus hijos. Su vida es un ejemplo de superación, trabajo diario y dedicación pero sobre todo, son un ejemplo de amor incondicional a la familia. Sin el amor y cuidados que recibo de ustedes todos los días, difícilmente hubiese podido llegar a este punto en las condiciones que lo hago. Es la felicidad que me produce tenerlos presentes en mi vida lo que me motiva a seguir avanzando. Como siempre, este trabajo al igual que cada paso que doy, está dedicado a ustedes con amor $\rightarrow \infty$.

A mis hermanos Iván y Lilia por su amor y compañía siempre presentes, por ser la fuente de vida en casa junto a nuestros padres, por todos los recuerdos, las risas y bromas a lo largo de todos estos años.

A Carolina Castro Alarcon por el tiempo, la paciencia, la amistad y el cariño a lo largo de muchos años. Porque es imposible concebirme como soy hoy sin mencionar tu presencia, por lo mucho que influiste en mi concepción del mundo, por mostrarme a través de ti que es posible conservar la inocencia, la bondad, la honestidad, la sencillez, ... , aprendí sobre la vida misma.

A toda mi demás familia: mis amados abuelos, Lilia Balderas y Rafael Tagle; a mis amados ti@s, Adriana, Rafael, José, Memo, Enrique, Ricardo, Manuel y Francisco; a todos mis prim@s desde los más grandes a las más pequeñas. Por su amor, preocupación y apoyo siempre presentes, a quienes llevo en mi corazón junto a muchos de mis recuerdos más felices.

Y por supuesto a nuestro Freddy por regalarnos tanto amor y alegría ∞ *a través de su pequeño tamaño y gran presencia.*

Agradecimientos

A la Universidad Nacional Autónoma de México y a la Facultad de Ingeniería por haberme dado la oportunidad de realizar un posgrado, por darme los conocimientos y herramientas capaces de transformar positivamente mi entorno y acercarme aún más, a la materialización de tantas metas. Espero en el futuro poder regresarle a la Universidad algo de lo mucho que generosamente me ha dado a lo largo de todos estos años.

Al Dr. Boris Escalante Ramírez y a la Dra. Jimena Olveres Montiel por la oportunidad de colaborar como su tesista y darme un lugar dentro de su grupo de investigación. Por compartir su conocimiento en las sesiones del grupo y darme plena libertad para explorar aquellos temas de mi interés. Gracias por toda la amabilidad, paciencia y apoyo que han mostrado a lo largo de esta etapa, por empujarme a enfrentar y superar algunos de mis temores académicos.

A los brillantes profesores del Posgrado en Ingeniería y del IIMAS con quienes tuve la fortuna y el reto de tomar clase: Abel Pacheco, Berenice y Ricardo Montalvo, Boris Escalante, Gibrán Fuentes, Jesús Savage, Jimena Olveres, Larry Escobar, Miguel Moctezuma y Pablo Pérez. Por compartir generosamente sus conocimientos, experiencias, y mostrar día a día esa dedicación a su quehacer.

A mi compañero y amigo Edgar Silva Guzmán con quien aprendí, trabajé y reí durante nuestros días en este posgrado. Por todo el tiempo, las charlas y el apoyo que siempre me brindaste, gracias :).

A mis compañeros de generación con quienes compartí este viaje quienes siempre estuvieron dispuestos a ayudar y a hacer más ameno el posgrado: Edgar, Jesús, Laura, Luis y Michel.

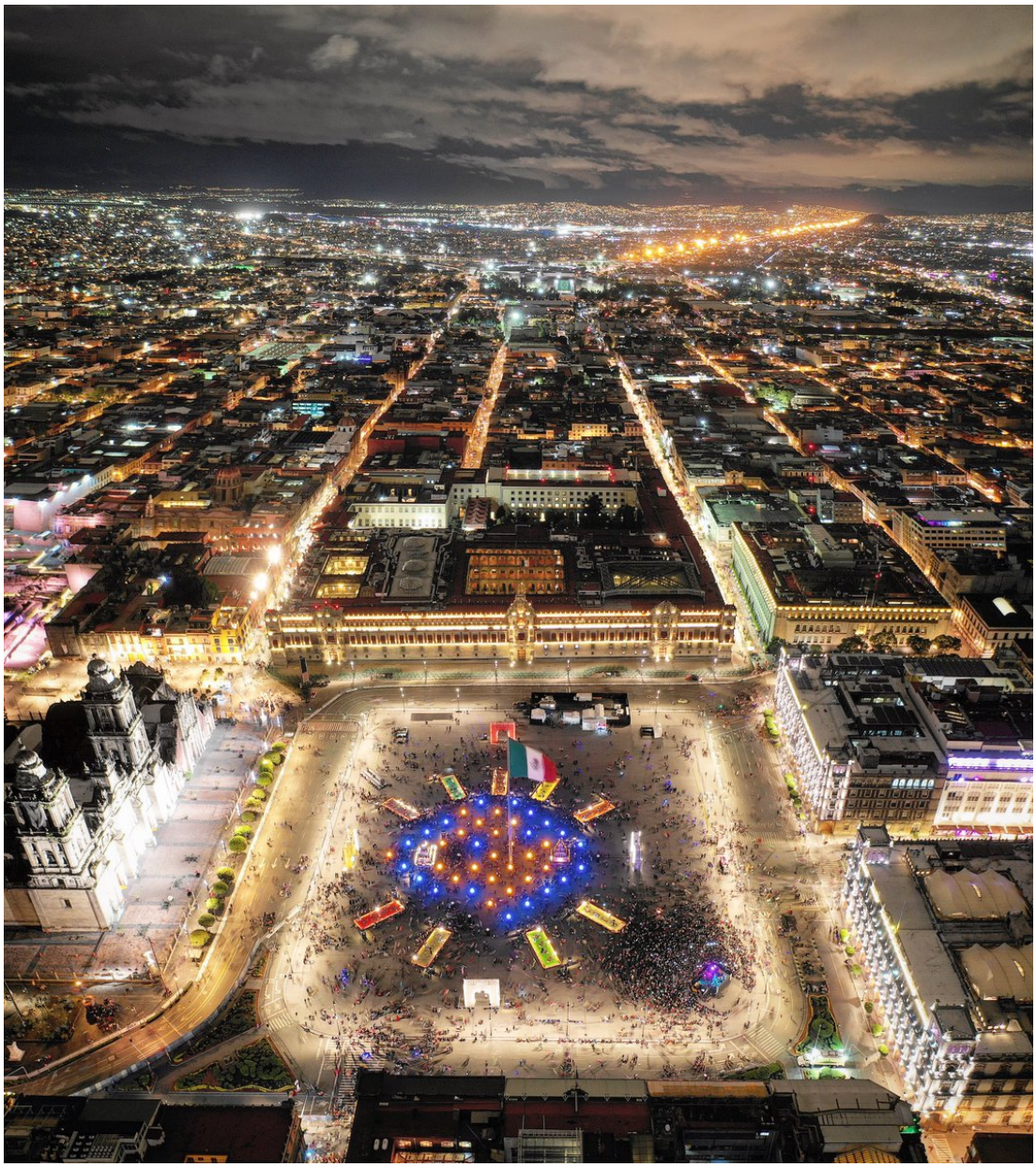
A los compañeros y amigos del LαPI que me tocó conocer: Alan, Erik, Erika, Fabián, Germán, Leonardo, Oscar, Rafael, Rodrigo, Steve y Vivian :). Gracias por hacerme sentir cómodo en mi tiempo como miembro del laboratorio, por ese compañerismo y amabilidad que siempre mostraron para conmigo.

A la diseñadora Marina López Hernández por su amistad, además de su generosa, desinteresada y muy oportuna ayuda en la creación de las figuras que acompañan este trabajo. Sin tu ayuda en los gráficos este trabajo no hubiese estado listo a tiempo.

Al Dr. Jesús Hernández Calderón por compartir su experiencia en dermatología y su disposición a seguir colaborando en este proyecto.

Y por supuesto, al M.I. Emilio Jiménez Madrigal por haberme introducido al apasionante campo del Procesamiento de Señales.

Finalmente, agradezco al Consejo Nacional de Ciencia y Tecnología (CONACyT) por el apoyo económico que me brindó (CVU: 857565) a lo largo de mis estudios de posgrado, apoyo que me permitió dedicarme de tiempo completo a mis estudios e investigación. Se agradece también el apoyo de los programas UNAM-PAPIIT IA103119 e IN11691.



La plancha del Zócalo de la Ciudad de México y sus alrededores en días previos al Día de Muertos. Fotografía de Santiago Arau (@Santiago_Arau), 27 de octubre del 2018.

"Esta gran ciudad de Tentixtitan está fundada en esta laguna salada, y desde la tierra firme hasta el cuerpo de la dicha ciudad, por cualquiera parte que quisieren entrar a ella, hay dos leguas. Tiene cuatro entradas, todas de calzada hecha a mano, tan ancha como dos lanzas jinetas. Es tan grande la ciudad como Sevilla y Córdoba. Son las calles de ella, digo las principales, muy anchas y muy derechas, y algunas de éstas y todas las demás son la mitad de tierra y por la otra mitad es agua, por la cual andan en sus canoas, y todas las calles de trecho a trecho están abiertas por do atraviesa el agua de las unas a las otras, y en todas estas aberturas, que algunas son muy anchas, hay sus puentes de muy anchas y muy grandes vigas, juntas y recias y bien labradas, y tales, que por muchas de ellas pueden pasar diez de a caballo juntos a la par"

– Hernán Cortés (1520), Segunda carta de relación al emperador Carlos V

"y cuando llegamos a la gran plaza, que se dice el Tatelulco, como no habíamos visto tal cosa, quedamos admirados de la multitud de gente y mercaderías que en ella había y del gran concierto y regimiento que en todo tenían [...] Y había entre los nuestros, soldados que habían estado en diferentes lugares del mundo, en Constantinopla, en toda Italia y en Roma, y dijeron que jamás habían visto un mercado tan ordenado y bien organizado, tan grande, tan lleno de gente."

– Bernal Díaz del Castillo (1568), Historia verdadera de la conquista de la Nueva España

Esta tesis también está dedicada a la Ciudad de México, a lo que representa, a quienes habitan, trabajan, comen y duermen en ella, a quienes la llenan de alegría y vida. Tan bella y difícil a la vez. Su belleza nace a raíz de su complejidad. Lugar donde he vivido más de un cuarto de mi vida y que sin importar a qué lugares me lleve la vida, o cuan lejos me encuentre de ella, siempre estará dentro de mí, de lo que soy, porque siempre seré mexicano.

Índice general

Índice de figuras

Capítulo 1

Introducción

La piel es el órgano más grande del cuerpo humano [27] y es responsable de muchas funciones vitales, nos brinda protección del mundo exterior pues protege nuestro cuerpo contra la luz solar dañina, las temperaturas extremas, los gérmenes y posibles sustancias dañinas [40]. La piel también se encarga de entre muchas otras cosas, de la sensación, la regulación del calor y la producción de vitamina D [8].

El cáncer de piel es la forma más común de cáncer en los Estados Unidos [1]. Según la Academia Americana de Dermatología (*American Academy of Dermatology*), se estima que uno de cada cinco estadounidenses desarrollará cáncer de piel en su vida. Al igual que con cualquier otro tipo de cáncer, cuanto antes se detecte, mejor será el pronóstico. Cuando el cáncer de piel se detecta temprano, es altamente tratable. Este tipo de cáncer se diagnostica principalmente mediante una inspección visual. El proceso de detección comienza con una evaluación clínica inicial [13]. Los estudios pueden seguir, según los resultados de la detección, en un análisis dermatoscópico, una biopsia y un examen histopatológico.

El melanoma es una forma rara de cáncer de piel que se desarrolla a partir de células de la piel llamadas melanocitos que se encuentran en la capa de células basales en la parte más profunda de la epidermis. También se encuentran en el iris, el oído interno, el sistema nervioso, el corazón y los folículos pilosos, entre otros tejidos. El melanoma representa menos del 5 % de todos los cánceres de piel en los Estados Unidos. Sin embargo, el melanoma también es la forma más mortal de cáncer de piel, es responsable de aproximadamente el 75 % de todas las muertes relacionadas con el cáncer de piel [44]. Algunos estudios han sugerido que la tasa de incidencia anual de melanoma ha aumentado entre la población caucásica de 4 a 8 % [16]. De acuerdo con la Sociedad Americana del Cáncer (*American Cancer Society*) en su artículo del 2019 de estadísticas sobre el cáncer [1], se estima que la cantidad de nuevos casos de melanoma diagnosticados en 2019 aumentará en 7.7 %.

Los dermatólogos utilizan diferentes métodos de inspección visual para determinar si una lesión de la piel bajo estudio pudiese ser un melanoma (lesión maligna). Algunos de estos métodos son la regla ABCDE, 7 *points checklist* y el método Menzies [26] solo por nombrar algunos. Todos estos métodos se basan en las características visuales que hacen que una lesión de melanoma maligno sea distinguible. Por ejemplo, el método ABCDE evalúa si está presente una o más de las siguientes características clínicas de un melanoma: asimetría(las lesiones de melanoma a menudo son asimétricas), irregularidad

del borde (bordes irregulares que son difíciles de definir), variación de color, diámetro (diámetro mayor que 6 [mm]), evolución (una lesión que cambia de color, forma o tamaño).

En los últimos años, ha habido un creciente interés en aplicar técnicas de aprendizaje profundo a problemas médicos [14]. Las redes neuronales profundas han mostrado resultados sorprendentes en las tareas de clasificación de lesiones cutáneas, algunos ejemplos sobresalientes de estos se pueden encontrar en [9, 13, 15]. La dermatología es una especialidad médica que puede beneficiarse enormemente de las poderosas capacidades de extracción de características de las redes neuronales profundas.

1.1. Antecedentes

Según la Fundación de Cáncer de Piel (*Skin Cancer Foundation*), el cáncer de piel se define como el crecimiento descontrolado de células anómalas de la piel. El cáncer ocurre cuando el daño en el ADN de las células de la piel, provocado principalmente por la radiación ultravioleta procedente de la luz solar o de las camas de bronceado, desencadena mutaciones o defectos genéticos que hacen que las células de la piel se multipliquen rápidamente dando lugar a tumores malignos.

1.1.1. La epidermis: tipos de células y neoplasias malignas

La Sociedad Americana del Cáncer (*American Cancer Society*) hace la siguiente descripción sobre la epidermis, la capa más externa de la piel, y sobre las células principales que ahí se encuentran, siendo la primera, la región afectada por la exposición a la radiación ultravioleta proveniente del Sol y por lo tanto, susceptible a la aparición de algún tipo de cáncer. También nos describe algunos de los tipos de cáncer más comunes.

Hay tres tipos principales de células en la capa superior de la piel llamada epidermis [2]:

- **Células escamosas**: son células planas en la parte externa de la epidermis que se desprenden constantemente a medida que las nuevas células se forman.
- **Células basales**: son células que están en la parte inferior de la epidermis, llamada capa de células basales. Estas células se dividen constantemente para reemplazar las células escamosas que se desprenden de la superficie de la piel. A medida que estas células se desplazan hacia la epidermis se vuelven más planas, y con el tiempo se convierten en células escamosas.

- **Melanocitos**: estas células producen el pigmento marrón llamado melanina que causa que la piel se broncee o se ponga morena. La melanina actúa como bloqueador solar natural del cuerpo que protege las capas más profundas de la piel contra algunos de los efectos nocivos del Sol. Para la mayoría de las personas, cuando la piel se expone al Sol, los melanocitos producen más pigmento, causando que la piel se torne bronceada o más oscura

La epidermis está separada de las capas más profundas de la piel por la membrana basal. Cuando un cáncer de piel se vuelve más avanzado, por lo general atraviesa esta barrera y las capas más profundas.

Existen diferentes tipos de cáncer de piel, a continuación se hace una breve descripción de los tipos más comunes así como del más mortífero, el melanoma [2].

- **Carcinoma de células escamosas**. Alrededor de dos de cada diez casos de cáncer de piel son carcinomas de células escamosas también llamados cánceres de células escamosas. Las células en estos cánceres lucen como versiones anormales de las células escamosas vistas en las capas externas de la piel.

  Estos cánceres comúnmente aparecen en las áreas del cuerpo expuestas al Sol tal como la cara, las orejas, los labios y el dorso de las manos. También puede surgir en cicatrices, llagas crónicas de la piel o en otras partes del cuerpo.

  Los cánceres de células escamosas tienen más probabilidad de crecer hacia las capas más profundas de la piel y propagarse hacia otras partes del cuerpo que los cánceres de células basales, este comportamiento resulta ser poco común.

- **Carcinoma de células basales**. El carcinoma de células basales, también llamado cáncer de células basales, es el tipo más común de cáncer de piel. Alrededor de ocho de cada diez casos de cáncer de piel son carcinomas de células basales. Cuando se observan con un microscopio, las células en estos cánceres lucen como células en la capa más inferior de la epidermis, llamada capa de células basales.

  Por lo general, estos cánceres surgen en las zonas expuestas al Sol, especialmente la cabeza y el cuello. Estos cánceres tienden a crecer lentamente. Es muy poco común que el cáncer de células basales se propague a otras partes del cuerpo. Sin embargo, si un cáncer de células basales se deja sin tratar, puede extenderse a las áreas cercanas y afectar los huesos, así como otros tejidos debajo de la piel.

  Si no se extrae completamente, el carcinoma de células basales puede reaparecer en el mismo lugar de la piel.

- **Melanoma**. Este tipo de cáncer se origina a partir de los melanocitos, las células de la piel que producen los pigmentos. Los melanocitos también pueden formar crecimientos benignos (no cancerosos), conocidos comúnmente como lunares. El melanoma es mucho menos común que los cánceres de piel de células basales o de células escamosas, pero es más propenso a crecer y propagarse si no se trata.

1.1.2. Inspección visual de lesiones en piel

Contar con un diagnóstico temprano resulta vital en el tratamiento de esta enfermedad. Por medio de un examen médico visual se puede emitir un primer diagnóstico al fijar la atención en las características presentes en las lesiones indicadas por ejemplo, por la regla **ABCDE** del cáncer de piel.

La regla **ABCDE** contempla las siguientes características para evaluar si una lesión se trata de un melanoma:

- **A**simetría
- **B**ordes irregulares
- **C**olor
- **D**iámetro
- **E**volución

Es necesario comentar que existen otros criterios de clasificación tales como el método de Menzies, 7 *Points Checklist*, *CASH* y *CHAOS & CLUES*.

1.1.3. El archivo de imágenes de la ISIC

La Colaboración Internacional de Imágenes de la Piel (*International Skin Imaging Collaboration*, **ISIC**) es un esfuerzo internacional para mejorar el diagnóstico de melanoma, esta colaboración es patrocinada por la Sociedad Internacional de Imágenes Digitales de la Piel (*International Society for Digital Imaging of the Skin*, **ISDIS**).

El archivo ISIC contiene la colección pública más grande de imágenes dermatológicas de alta calidad de lesiones cutáneas. En la actualidad, el archivo de la ISIC contiene más de 23,000 imágenes que se recopilaron en centros clínicos líderes a nivel internacional y se adquirieron de una diversidad de dispositivos dentro de cada centro.

Todas las imágenes candidatas a pertenecer al archivo de la ISIC se analizan para garantizar la privacidad y la calidad. La mayoría de las imágenes tienen metadatos clínicos asociados, que han sido examinados por expertos en melanoma reconocidos. Un subconjunto de las imágenes ha sido anotado y marcado por expertos reconocidos en cáncer de piel. Estas marcas incluyen características dermatoscópicas tales como elementos morfológicos globales y locales en la imagen que se sabe que discriminan entre los tipos de lesiones cutáneas.

La ISIC organiza anualmente un *challenge* donde invita a los interesados a participar en la detección de melanoma y otras afecciones de la piel empleando modelos computacionales sobre las imágenes de su archivo. En su edición 2017, el reto *"ISIC 2017:*

Skin Lesion Analysis Towards Melanoma Detection", en la categoría de clasificación, se centró en identificar tres afecciones de la piel:

- Melanoma
- Nevus
- Queratosis seborreica

El objetivo del reto se centraba en lograr que un modelo por computadora pudiese distinguir entre dos afecciones benignas (nevus y queratosis) de una afección maligna (melanoma). Esto resulta ser una tarea de gran dificultad inclusive para profesionales de la salud pues en el caso de nevus y melanoma son lesiones que en las primeras etapas presentan una enorme similitud visual. Por esto mismo, contar con un modelo computacional que pueda distinguir entre las lesiones con una alta confiabilidad es de gran interés para la comunidad médica.

1.2. Motivación

En Estados Unidos se tienen 5.4 millones de nuevos casos de cáncer de piel al año. De los diferentes tipos de cáncer de piel, el melanoma es un tipo de cáncer de piel raro pero a la vez el más letal.

El melanoma representa:

- **5 %** de todos los casos de cáncer de piel.
- **75 %** de todas las muertes relacionadas con cáncer de piel.

La tasa de supervivencia a 5 años es de:

- 99 % en primeras etapas
- 14 % en las últimas etapas.

En México, el melanoma es responsable del 80 % de las muertes por cáncer de piel; va en aumento en el mundo, más que cualquier otra neoplasia maligna. De acuerdo al último reporte del **Instituto Nacional de Cancerología (INCan)**, en el país aumentó el número de casos de 300 a 500 % en los últimos años.

El tratamiento del melanoma dependerá de la etapa y del lugar en el que se encuentre. La Sociedad Americana del Cáncer (*American Cancer Society*) agrupa el avance de un cáncer en 5 grandes etapas: 0, I, II, III y IV.

En la **etapa 0** los melanomas no han crecido más allá de la epidermis por lo que resultan fáciles de extraer a través de cirugía. El melanoma es un tipo de cáncer que puede hacer metástasis y que exhibe una alta tasa de mortalidad. Sin embargo, si se detecta a tiempo, una persona tiene una probabilidad de supervivencia de casi 99.9 %.

En contraste, los melanomas **etapa IV** son muy difíciles de tratar debido a la expansión de la metástasis. Algunos de los lugares donde pueden metastatizar son en el cerebro, el hígado y la piel [47].

1.3. Objetivos de la tesis

El propósito de este proyecto es implementar un sistema automático de diagnóstico basado en Aprendizaje Profundo (*Deep Learning*) que esté inspirado en el protocolo de inspección médico, -el cual se guía por las características visuales ABCDE del cáncer de piel- que logre detectar (localizar y clasificar) las lesiones presentes en imágenes de piel, contribuyendo al diagnóstico temprano de esta enfermedad.

1.3.1. Objetivos específicos

- Realizar una revisión bibliográfica referente a los modelos de detección en aprendizaje profundo para ser aplicados al diagnóstico de lesiones en piel.
- Realizar el análisis de datos de las imágenes dermatológicas del archivo ISIC y su preprocesamiento para el aumentado de datos.
- Implementar un algoritmo de segmentación para las lesiones en las imágenes dermatológicas.
- Adecuar e implementar modelos de detección basados en Aprendizaje Profundo.
- Evaluar el desempeño de las técnicas de preprocesamiento junto a los modelos seleccionados en la tarea de detección de lesiones bajo las métricas correspondientes.

1.4. Planteamiento del problema

Factores como los cambios en el estilo de vida de la población a nivel mundial y su exposición a la radiación ultravioleta han traído consigo un aumento en los casos de cáncer de piel en los últimos años. Según la Organización Mundial de la Salud (OMS), se estima que cada año se producen entre 2 y 3 millones de casos de cáncer de

piel en el mundo, de los cuales 132,000 corresponden a casos de melanoma y mueren aproximadamente 66,000 personas por causa de éste y otros tipos de cáncer de piel [49].

Existen diversos métodos de diagnóstico que facilitan la detección del cáncer de piel. Un ejemplo de ellos es la dermatoscopia digital, un método no invasivo y fácil de practicar a cualquier persona. Este método consigue que los dermatólogos puedan monitorear de forma muy precisa la evolución de los lunares y otras lesiones sospechosas, incluso antes de que evolucionen en un cáncer de piel.

La realización de esta prueba anualmente en personas con riesgo de desarrollar la enfermedad facilita un diagnóstico precoz y por lo tanto, el mejor de los pronósticos en caso de resultar positivo. Sin embargo, la mera inspección visual por dermatólogos expertos ofrece un diagnóstico con una exactitud de alrededor del 60 %, lo que significa que muchos melanomas potencialmente curables no se detectan hasta etapas más avanzadas [30].

A pesar de los esfuerzos que se han realizado en los últimos años para lograr la detección temprana de estas patologías, la detección es todavía un problema abierto desde el punto de vista tecnológico, pues existe la necesidad de desarrollar e implementar nuevos algoritmos que superen los resultados obtenidos hasta el momento. Uno de los objetivos es aumentar los índices de exactitud en los diagnósticos automatizados por computadora para dotar al médico de una herramienta valiosa, que sea capaz de brindar un primer veredicto confiable y le ayude a emitir un diagnóstico final con mucho mayor certidumbre que en el pasado.

Dada la popularidad y masificación de los teléfonos inteligentes en los últimos años, esta tecnología podría estar en manos del paciente para dotarlo de información sobre su condición actual de salud y emitirle una recomendación sobre si debe visitar a un médico para corroborar el resultado de un primer diagnóstico. Esto último ayudaría a hacer posible la detección temprana de cualquier posible anormalidad en el paciente.

1.5. Metodología

La metodología propuesta para el proyecto se basa en los pasos obligatorios para poder implementar, adaptar y evaluar un modelo de aprendizaje profundo en imágenes médicas. De forma general, los pasos necesarios para llevar a cabo un proyecto de esta naturaleza son los siguientes:

- Análisis de datos.
- Realizar el preprocesamiento de las imágenes.
- Estudiar y seleccionar una arquitectura de red neuronal profunda.
- Implementar el modelo.

- Realizar la transferencia de conocimiento y el ajuste fino en las capas convolucionales.
- Entrenar el modelo y realizar el ajuste de hiperparámetros.
- Evaluar el modelo con las métricas apropiadas para el problema.

Para este proyecto se hará uso de las imágenes del archivo ISIC, la más grande colección pública de imágenes dermatológicas. Se debe comenzar por descargar y preprocesar estas imágenes, este último aspecto es de vital importancia pues entre más imágenes (datos) se tengan, mejor aprenderá la red convolucional a identificar las diferentes clases y mostrará un mejor desempeño durante su validación. A este paso dentro del preprocesamiento se le llama aumentado de datos, pues se trata de aplicar diferentes transformaciones a las imágenes originales. Estas transformaciones son operaciones tales como rotaciones, traslaciones, *zooms*, esto se hace con la finalidad de contar con más ejemplares provenientes de una sola imagen. El aumentado de datos también debe buscar mitigar los efectos del desbalance de clases que es usualmente el caso de donde se parte.

Realizar una segmentación a las imágenes de la base de datos también puede ayudar al aumentado de datos pues dota a la red de un número adicional y significativo de ejemplos donde el ruido o el fondo pudiesen afectar el proceso de aprendizaje. En muchas ocasiones se prefiere segmentar primero toda la base de datos y sobre el resultado de este proceso se aplican las transformaciones a las imágenes para aumentar la base de datos. Trabajar con modelos de aprendizaje profundo suele ser un proceso bastante empírico por lo que sólo realizando pruebas partiendo de diferentes condiciones puede conocerse cuál es la mejor opción para el problema que se está abordando, teniendo en mente que lo que se desea es que la red neuronal logre el mejor desempeño.

La investigación en aprendizaje profundo ha sido intensa en los últimos años por lo que las publicaciones sobre nuevas arquitecturas prometedoras son numerosas. Identificar la arquitectura que pueda ser de utilidad para el problema que se desea abordar debe hacerse con cuidado pues las pruebas en este campo suelen consumir un tiempo de cómputo considerable. Para la implementación del modelo existen muchas opciones siendo las más populares aquellas construidas sobre las librerías *TensorFlow* y *Keras*.

Los modelos de redes neuronales profundas necesitan millones de ejemplos para que puedan aprender a generalizar la tarea para la cual se les está entrenando. La gran mayoría de estos modelos se encuentran pre-entrenados sobre el *dataset* masivo *ImageNet*.

Por medio de la técnica conocida como transferencia de conocimiento y el ajuste fino de capas convolucionales, el usuario puede adaptar un modelo pre-entrenado en *ImageNet* a su propio *dataset* de imágenes. Existen diferentes variantes de estas técnicas, el camino a seguir dependerá de que tan similares o diferentes sean las imágenes del *dataset* con el que sea desea reentrenar a la red, si son muy diferentes como es el

caso de un *dataset* de dermatología habrá que elegir hacer ajustes finos en los valores de los pesos presentes en los filtros de todas las capas.

Con base en lo anterior, se puede proceder a reentrenar el modelo. En un primer entrenamiento se debe elegir el método de optimización, el número de épocas, la tasa de aprendizaje, entre otros hiperparámetros. Dependiendo de los resultados obtenidos de esa primera prueba, se deberán ajustar nuevamente los hiperparámetros del modelo y proceder a iniciar un nuevo entrenamiento.

La evaluación del modelo vendrá dada en base a métricas como exactitud, sensibilidad y especificad para un problema puramente de clasificación.

1.6. Estado del arte

El aprendizaje profundo es un subcampo dentro del aprendizaje automático, se trata de diferentes conjuntos de modelos computacionales, cada uno especializado en procesar un tipo particular de señal o dato que al verse frente a un gran número de ejemplos de entrenamiento, son capaces de aprender a realizar una tarea específica. La idea es similar a como las personas cultivamos una nueva habilidad, por medio de un gran número de ejemplos y practicando sobre ellos.

El aprendizaje profundo está presente en muchos de los productos y servicios que tenemos disponibles actualmente, desde los asistentes personales por voz, algunos presentes en teléfonos y tabletas, las recomendaciones emitidas por los servicios de *streaming* de música o películas, cuyas sugerencias se basan en nuestro historial y lo que hemos indicado que nos gusta, así como en los vehículos autónomos que ya circulan en algunas partes del planeta, pues les permite distinguir los carriles por donde circulan así como su entorno, desde el estado de un semáforo hasta las personas, animales u otros carros que les rodean. La medicina es otro de los campos que más espera beneficiarse con herramientas y productos construidos en torno a esta tecnología, que tiene como una de sus finalidades apoyar al médico en su toma de decisiones. Recientemente, se ha popularizado abordar diferentes problemas científicos y tecnológicos a través del uso de modelos de aprendizaje profundo pues está consiguiendo resultados que antes no eran posibles.

Con ayuda de los *frameworks* actuales se puede tener acceso a todo un conjunto de métodos para la definición y construcción de arquitecturas, pre-procesamiento de datos, algoritmos de optimización, herramientas de visualización, métricas de evaluación de modelos, entre muchas otras funciones. Esto permite rápidamente definir y entrenar un modelo para realizar por ejemplo, tareas de clasificación directamente a partir de imágenes o archivos de voz. Estos modelos son arquitecturas de redes neuronales que constan de varias capas que se entrenan mediante conjuntos de datos masivos y etiquetados. Los modelos de aprendizaje profundo pueden llegar a lograr un desempeño que, en ocasiones, supera al del ser humano.

Muchas de las ideas sobre las que se sustenta el aprendizaje profundo se desarrollaron durante el siglo XX [43], sin embargo, existen dos motivos principales que han hecho posible el auge que goza en la actualidad:

- **Acceso a conjuntos de datos masivos y/o públicos**. Los modelos de aprendizaje profundo requieren de grandes cantidades de datos etiquetados para poder generalizar la tarea para la cual se les entrena. Actualmente se tiene acceso a conjuntos de datos masivos como ImageNet y COCO que permiten entrenar y evaluar el desempeño de alguna nueva arquitectura propuesta. A la par se han consolidado comunidades en sitios como *Kaggle* donde en los últimos años se han hecho públicos conjuntos de datos más especializados para alguna tarea particular como reconocimiento de rostro, detección de alguna patología médica en imágenes, corpus de archivos de voz para traducción de voz a texto, predicción de terremotos, clasificación de fuentes lumínicas astronómicas, entre muchos otros. Lo anterior sin mencionar que en nuestro tiempo, una persona con acceso y actividad en la internet genera por sí misma una cantidad de datos significativa, datos que muchas veces son aprovechados por compañías con la finalidad de mejorar un determinado servicio o producto cuyo funcionamiento haga uso del aprendizaje profundo.
- **Poder de cómputo**. Para poder entrenar un modelo de aprendizaje profundo se requiere de una gran capacidad de cómputo. Los GPU han resultado ser una solución muy popular dada su capacidad de realizar procesamiento en paralelo. Gracias al uso de este tipo de *hardware* o de infraestructura como el cómputo en la nube se ha podido reducir significativamente el tiempo para entrenar una red neuronal profunda.

En los últimos años ha surgido un fuerte interés por aplicar todo este nuevo conjunto de modelos y herramientas al área médica [14]. Un ejemplo notable de esto, es el uso de esta técnica de inteligencia artificial para detectar una enfermedad llamada retinopatía diabética. La retinopatía diabética es la causa de ceguera de más rápido crecimiento en todo el mundo con casi 415,000,000 de pacientes diabéticos en riesgo. Si se detecta a tiempo, la enfermedad puede ser tratada, en caso contrario puede desembocar en una ceguera irreversible.

Una de las formas más comunes de detectar esta enfermedad ocular, es hacer que un especialista examine las imágenes de la parte posterior del ojo y las evalúe según la presencia y gravedad de la enfermedad. Para la interpretación de estas imágenes se requieren de médicos especializados y en muchas regiones del mundo, sobre todo en poblaciones de escasos recursos, no hay suficientes especialistas calificados para evaluar a todas las personas que están en riesgo.

Contar con métodos automatizados de detección de retinopatía diabética que ofrezcan una alta exactitud en su diagnóstico, tiene el potencial de ayudar a los médicos a

evaluar a más pacientes y canalizar rápidamente a aquellos que lo requieran hacia un especialista [20].

En el área de dermatología, se han publicado diferentes trabajos que abordan principalmente el problema de clasificación de lesiones por medio de algoritmos de preprocesamiento en conjunto con modelos de aprendizaje de máquina.

En [6] y [45] se evalúan distintos métodos para la detección del contorno de las lesiones. En [35] y [37] se proponen varios algoritmos de segmentación y métodos para la eliminación de vello en la imagen. En [39] se utiliza la información del mapa de color de la imagen para la segmentación. Además, en ese mismo trabajo se propone un método de segmentación preciso y rápido capaz de ser ejecutado en un *smartphone*.

El color es uno de los descriptores más fuertes y significativos en este tipo de imágenes, la presencia de diferentes tonalidades en una lesión tiene relevancia para un diagnóstico. En [11], los autores utilizan el método de análisis de componentes principales (*Principal Component Analysis*, PCA) como ayuda para la segmentación de regiones de interés. Se trata de una técnica de segmentación por color que utiliza el método de PCA para obtener un histograma bidimensional de la información del color en la imagen, que es posteriormente procesado para segmentar las zonas con diferente tonalidad.

Existen una variedad de publicaciones recientes que hacen uso de las herramientas y modelos de aprendizaje profundo para abordar el problema de clasificación de lesiones de piel, el uso de estas técnicas ha logrado un mucho mejor desempeño en términos de métricas como exactitud, especificidad y sensibilidad en comparación con los resultados obtenidos con técnicas clásicas de aprendizaje de máquinas. Algunos ejemplos de ellos se describen a continuación.

En el artículo de [10] se propone una nueva red neuronal profunda totalmente convolucional para segmentar automáticamente el melanoma fuera de las imágenes de la piel mediante el aprendizaje de extremo a extremo con solo píxeles y etiquetas como datos de entrada. El método propuesto es capaz de utilizar información local y global para segmentar el melanoma mediante la adopción de capas de omisión.

En la publicación de [33] se implementa un método de clasificación completamente automático, que emplea redes neuronales convolucionales pre-entrenadas que aprenden a clasificar las lesiones de la piel, basándose en características morfológicas, textura, estructura y color. En este estudio se utilizaron 2,000 imágenes de lesiones cutáneas divididas en tres categorías principales: 374 imágenes de melanomas, 254 de queratosis seborreica y 1,372 imágenes de nevus. El clasificador entrenado se evaluó luego en 150 imágenes. Los resultados fueron de 84.8 % y 93.6 % de exactitud para el problema de clasificación binaria de melanoma contra el resto de lesiones y queratosis seborreica contra el resto de lesiones, respectivamente.

Uno de los artículos más relevantes sobre las aplicaciones y el potencial que puede llegar a ofrecer el uso del aprendizaje profundo dentro del campo de la dermatología fue el presentado por [13]. En este artículo, se propone:

- Utilizar la arquitectura de la red neuronal GoogleNet Inception v3.
- Recopilar y hacer uso de la mayor cantidad de imágenes dermatológicas hasta ese momento provenientes de diferentes bases de datos.
- Crear una taxonomía de enfermedades en la piel y un algoritmo para mapear enfermedades en clases de entrenamiento.
- Hacer transferencia de conocimiento con ajuste fino en todas las capas para re-entrenar la red con las imágenes dermatológicas.
- El sistema debe ser capaz de clasificar de manera correcta los tipos de lesiones en la piel.

Se prueba el desempeño de esta red frente a 21 dermatólogos certificados en dos casos críticos:

- Carcinomas vs queratosis seborreica (identificación del cáncer más común vs lesión benigna).
- Melanomas vs nevus (identificación del cáncer más mortal vs lesiones benignas visualmente similares).

Este artículo publicado en la revista *Nature* obtiene de su estudio los siguientes resultados:

- La exactitud general de la red neuronal convolucional (*convolutional neural network*, **CNN**) es de: **72.1 ± 0.9 %** (media ± d.e.) (representa el promedio de las cifras de exactitud individuales por clase resultado de la inferencia).
- Dos **dermatólogos** logran una precisión del **65.56 % y 66.0 %** en un subconjunto del set de validación.
- Durante la validación del algoritmo con una partición de nueve clases de enfermedades.
    - Precisión de la **CNN**: **55.4 ± 1.7 %** mientras que los mismos dos **dermatólogos** obtienen una precisión de **53.3 % y 55.0 %**

Con los resultados anteriores, los investigadores concluyen:

- La red neuronal convolucional logra un desempeño superior a la lograda por los expertos en dermatología.
- Se propone utilizar el sistema en dispositivos móviles para poder facilitar un diagnótico temprano.

A partir de los resultados mostrados por este estudio surgió un gran interés de parte de los diferentes grupos de investigación por explotar las herramientas de aprendizaje profundo con el objetivo de superar los resultados logrados. Desde entonces, muchos enfoques diferentes se han propuesto para abordar este problema y mitigar la falta de una base de datos tan rica y amplia como la utilizada en el artículo de *Nature*. Algunos de los trabajos más recientes con resultados importantes son los presentados por [9], [15], [34] y [23] todos ellos se enfocan estrictamente en resolver el problema de clasificación.

1.7. Contribución

La contribución principal de este trabajo es la de un sistema de diagnóstico en tiempo real de lesiones de la piel basado en aprendizaje profundo y técnicas de procesamiento de imágenes. Este sistema es capaz de distinguir entre tres diferentes tipos de lesiones de la piel, dos de ellas de naturaleza benigna, nevus y queratosis seborreica, y un tipo de lesión maligna o cancerígena, melanoma. El sistema muestra un buen desempeño acorde a las métricas de precisión y *recall* con posibilidad de mejoría al explorar otras técnicas de procesamiento de imágenes, la incorporación de nuevas imágenes o bien, la generación automática de la región de interés a partir de las imágenes ya segmentadas.

La literatura existente se ha centrado en resolver estrictamente el problema de clasificación logrando resultados sobresalientes sin embargo, esta es la tarea más simple que una red neuronal convolucional es capaz de resolver. El problema de detección (clasificación y localización) para la identificación de lesiones en la piel no se ha explorado al momento de escribir este trabajo. Existen un conjunto de nuevas arquitecturas de detección basadas en redes neuronales convoluciones que podrían emplearse como solución al problema que podrían dotar a un médico de una riqueza de información más amplía que una simple clasificación. En este trabajo se exploró, adaptó y puso en operación una de ellas con resultados satisfactorios.

1.8. Organización del trabajo

Este trabajo se compone de 5 capítulos que se describen a continuación.

En el capítulo 1 se presentó una introducción al contexto dermatológico, las definiciones y conceptos necesarios para poder ganar un entendimiento de la problemática desde la perspectiva de un médico especialista, se dio la definición de algunas de las lesiones de la piel incluyendo aquellas con las que trata este trabajo, se presentó el enfoque que sigue un médico especialista para diagnosticar este tipo de lesiones cuando acude un paciente a consulta. A continuación se describió el esfuerzo de la ISIC

por hacer público una base de datos de imágenes dermatológicas con más de 20,000 muestras que resulta ideal para ser utilizado para entrenamiento de modelos de aprendizaje profundo. Posteriormente, se realizó un resumen de los trabajo más recientes e importantes que han trabajado con la base de datos de la ISIC para abordar algún problema similar al de esta tesis. Finalmente, se comentó sobre la aportación que hace este trabajo como solución al problema de diagnóstico de lesiones en piel.

El capitulo 2 se centra en presentar los fundamentos sobre aprendizaje profundo necesarios para poder abordar un problema de este tipo. Se comienza por ubicar el lugar de esta disciplina dentro de la inteligencia artificial y como este se distingue de otros enfoques. Posteriormente se da una introducción a las redes neuronales, se presenta el modelo de la neurona artificial que intenta emular a grandes rasgos el funcionamiento de la neurona biológica. Se presenta la idea detrás de algoritmos fundamentales como propagación hacia atrás, descenso del gradiente y la necesidad de contar con una función de pérdida o error. Finalmente, se presenta las redes neuronales profundas, se trata el caso particular de las redes neuronales convolucionales, sus parámetros e hiperparámetros, funciones de activación, optimizadores, entre otros.

En el capitulo 3 se describen en la arquitectura de red neuronal así como los algoritmos de segmentación que fueron utilizados en este trabajo. Se comienza con una descripción de la arquitectura base de red neuronal convolucional sobre la que opera el sistema de detección YOLOv3. Se describe la idea fundamental detrás del algoritmo YOLO, sus bondades sobre otros enfoques, así como las técnicas que implementa en su última versión que le permiten distinguirse de los demás. A continuación, se introducen algunos conceptos y algoritmos previos como antecedente a la descripción de los métodos de segmentación. Finalmente, se presenta la aplicación de esos métodos a la segmentación de imágenes dermatológicas.

El capítulo 4 muestra los diferentes experimentos que se realizaron así como los resultados que se obtuvieron con cada uno de ellos, se evidencian los efectos de realizar un pre-procesamiento como el aumentado de datos y el balanceo de clases. Se describe la estrategia empleada para adaptar una base de datos pensada para clasificación al problema de detección. De igual manera, se comenta la estrategia seguida para lograr para realizar un etiquetado automático. En esta sección se presentan las gráficas de desempeño y su discusión.

El capitulo 5 presenta las conclusiones extraídas del análisis de los resultados obtenidos así como algunas de las futuras lineas de desarrollo inmediatas que puede tomar este trabajo. Se emiten algunas recomendaciones que podrían mejorar las capacidades actuales de este sistema así como algunas ideas que podrían explorarse con diferentes finalidades, desde la generación de un producto comercial a la generación de un conjunto de datos propio.

Capítulo 2

Fundamentos de Aprendizaje Profundo

2.1. Inteligencia Artificial y Aprendizaje de Máquina

El nacimiento de la inteligencia artificial puede situarse en la década de 1950, cuando un pequeño grupo de personas del naciente campo de las ciencias de la computación comenzó a preguntarse si podrían crearse computadoras capaces de "pensar". De acuerdo con [7], la inteligencia artificial se puede definir de forma sencilla como: "el esfuerzo por automatizar tareas intelectuales que normalmente son realizadas por humanos".

La inteligencia artificial es un campo mucho más amplio y general que incluye tanto al aprendizaje de máquina como al aprendizaje profundo así como otros enfoques que no implican ningún tipo aprendizaje. Por ejemplo, los primeros programas capaces de jugar ajedrez se diseñaron puramente en base a reglas, reglas que eran codificadas a mano por programadores. A este enfoque de definir a mano un conjunto amplio de reglas explícitas con la finalidad de manipular alguna base de conocimiento se le conoce como inteligencia artificial simbólica y fue el paradigma dominante de este campo desde la década de 1950 hasta finales de la década de 1980. Este enfoque alcanzó su máxima popularidad durante el auge de los sistemas expertos de los años ochenta.

En el paradigma simbólico, las reglas así como los datos que serán procesados de acuerdo con estas reglas son introducidos como entrada a través de un programa y lo que se obtiene es únicamente las respuestas al problema dadas las reglas definidas. La inteligencia artificial simbólica demostró ser adecuada para resolver problemas lógicos bien definidos, sin embargo, resultó ser poco efectiva para descubrir patrones o reglas al abordar problemas mucho más complejos como la clasificación de imágenes, el reconocimiento de voz o la traducción de idiomas. Estas limitaciones llevaron al surgimiento del aprendizaje de máquina.

En 1950 Alan Turing publica "*Computing Machinery and Intelligence*" donde se cuestiona si las computadoras de propósito general podrían ser capaces de aprender, de ser originales y concluye que estas podrían ser capaces de tales hazañas.

El aprendizaje de máquina está inspirado en los cuestionamientos de Turing, en este enfoque el desarrollador ingresa los datos así como las respuestas que espera obtener

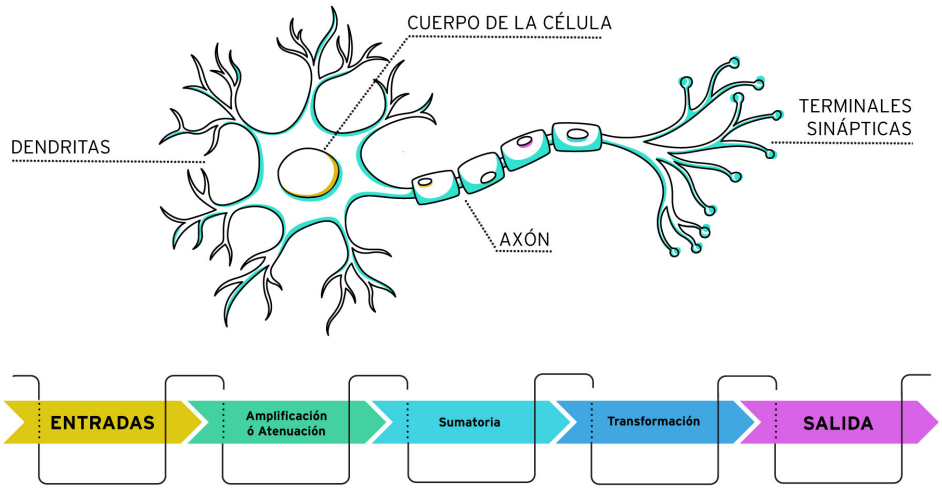


Figura 2.1: La neurona biológica y la abstracción del procesamiento de los estímulos.

de ellos y lo que obtiene es un modelo que permite a una computadora realizar la tarea para la cual fue entrenada. Estos modelos se pueden aplicar a nuevos datos para generar respuestas originales.

Podemos concluir entonces que la diferencia fundamental entre los enfoques presentados es que, una aplicación basada en aprendizaje de máquina se entrena en lugar de programarse explícitamente. Para ello, se le expone a un número de ejemplos significativos a una tarea y logra encontrar una relación estadística entre ellos, forma patrones que le permiten automatizar la tarea deseada.

2.2. Introducción a las Redes Neuronales

Los primeros modelos de redes neuronales tomaron inspiración de la neurociencia. En particular del modelo simplificado de la neurona biológica. Este modelo se ilustra en la figura 2.1.

De este modelo, se pueden identificar tres estructuras clave: el árbol dendrítico, el núcleo o cuerpo de la célula y el axón.

La parte distal del axón muestra una estructura en forma de botón a la que llamaremos engrosamiento presináptico. A continuación de este tendremos las dendritas correspondientes a alguna otra neurona. De igual forma, las dendritas de nuestra neurona bajo estudio serán estimuladas por el axón de alguna otra neurona.

Si hay suficiente estimulación al árbol dendrítico tendremos un pico eléctrico que viajará por el axón como si se tratase de una línea de transmisión. Después de este suceso, la neurona entrará en un estado de reposo al que se le conoce como periodo refractario.

Si analizamos con detalle la conexión entre el axón con el arból dendrítico. El axón tiene dentro de sí un conjunto de vesículas que protegen y almacenan a los neurotransmisores. Cuando el axon es estímulado, libera los neurotransmisores en el espacio intersináptico. El médico español Santiago Ramón y Cajal demostró que una neurona no es parte de la otra, sino que están separadas por una brecha sináptica.

El modelo biológico simplificado descrito anteriormente se puede modelar mediante un grafo computacional y es el que generalmente se encuentra en la literatura. Al modelo que representa el grafo también se le conoce como neurona artificial o perceptrón.

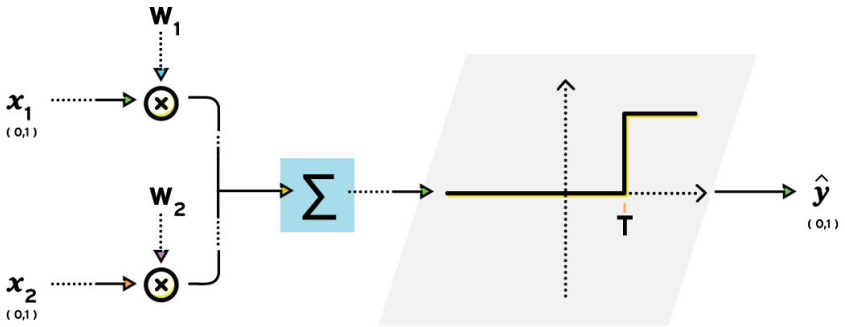


Figura 2.2: Modelo de la neurona artificial o perceptrón.

En el modelo de la figura 2.2 x_1 y x_2 modelan los estímulos a las dendritas, estos estímulos pueden ser amplificados o atenuados mediante dos parámetros que llamaremos pesos, w_1 y w_2. La manera de englobar el estímulo total proveniente de cada una de estas líneas es mediante un sumador. Para decidir si la influencia del conjunto de entradas es suficiente para hacer que la neurona dispare, la señal de salida del bloque de suma, será la entrada a un bloque de umbralización. En este bloque, se tendrá una señal a la que llamaremos función de activación.

En la figura 2.2 se utiliza como función de activación un escalón desfasado un valor o umbral T. Si el conjunto de estímulos provenientes del sumador son lo suficientemente fuertes, tal que su valor es mayor que el umbral T, entonces la neurona dispara. En otras palabras, la función de activación se valúa con el valor de la señal de salida del sumador. Dependiendo del valor que toma la función a partir de haber sido valuada, decimos que la neurona dispara o no, entrega un 1 o un 0.

Podemos ver que en el modelo de la neurona artificial se modelan los pesos sinápticos a partir de los bloques que multiplican a las entradas, el efecto total de los estímulos

se modela a partir de un bloque sumador y finalmente, el resultado se hace pasar por un bloque de umbralización o función de activación, tal que si el conjunto total de los estímulos a la neurona son lo suficientemente fuertes, se obtiene una respuesta.

Debemos dejar claro que el modelo de la neurona artificial es una sobresimplficación del comportamiento real de una neurona biológica. Los mecanismos de muchos de los procesos que ocurren en las neuronas biológicos se desconocen hoy día. Algunos de los fenómenos biológicos que este modelo computacional no contempla son: el periodo refractario, las bifurcaciones presentes en el axón que permiten que un pulso emitido por una neurona pueda viajar por una rama u otra, el tiempo de llegada de los pulsos provenientes de una neurona hacia las dendritas de otra y el efecto que tiene sobre la capacidad de reconocimiento de esta última.

Si deseamos representar la dinámica ideal de una red neuronal, podemos hacerlo si pensamos en esta como un bloque que recibe un conjunto de entradas $x_1, x_2, \ldots, x_m$ y entrega un conjunto de salidas ideales $y_1, y_2, \ldots, y_n$. Este bloque lo podemos escribir en forma vectorial de la siguiente manera:

$$\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{X}, \boldsymbol{W}, \boldsymbol{T}) \tag{2.1}$$

Durante el entrenamiento de una red neuronal lo que se busca hacer es ajustar los pesos y umbrales para obtener la salida que uno desea. Puede decirse entonces que una red neuronal es una manera de aproximar un modelo o función.

Una vez entrenada la red, podemos representarla de la siguiente manera:

$$\hat{\boldsymbol{y}} = \boldsymbol{d}(\boldsymbol{X}) \tag{2.2}$$

Donde la salida deseada o estimada $\hat{\boldsymbol{y}}$ es función únicamente de las entradas.

Para conocer el desempeño de una red neuronal en la ejecución de una tarea, se necesita plantear una medida de desempeño. Por simplicidad, una de las formas más ingenuas de hacerlo es comparar el valor real de las salidas contra el valor deseado u obtenido por la red neuronal. Esto se puede lograr mediante la siguiente expresión:

$$\mathcal{L}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \|\hat{\boldsymbol{y}} - \boldsymbol{y}\| \tag{2.3}$$

La expresión anterior cuantifica el desempeño de la red pues compara la distancia que existe entre los dos vectores y se le conoce como función de pérdida o error. Haciendo uso de técnicas de optimización, se busca minimizar la función de pérdida.

Sin embargo, la expresión (2.3) tiene el inconveniente de que describe una función muy brusca. Para subsanar esto, es conveniente tratarla como una función cuadrática. Si además se le añade un factor que ayude en el cómputo de la minimización, la expresión queda de la siguiente manera:

$$\mathcal{L}(\hat{y}, y) = \frac{1}{2} \|\hat{y} - y\|^2 \tag{2.4}$$

Si a esta expresión se le añade un signo $-$, adquiere el nombre de función de ganancia por lo que el objetivo cambia a maximizar la ganancia.

$$\mathcal{G}(\hat{y}, y) = -\frac{1}{2} \|\hat{y} - y\|^2 \tag{2.5}$$

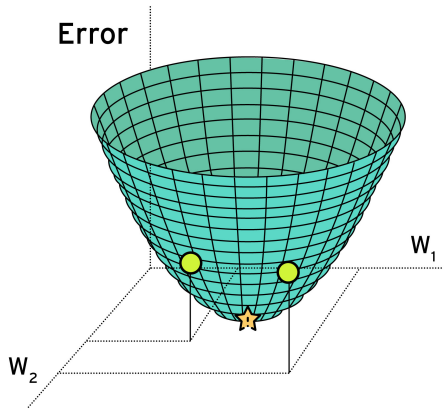


Figura 2.3: Gráfica de una función de pérdida o error en términos de dos pesos.

En la figura 2.4, se muestran las curvas de nivel de una función de pérdida que se encuentra únicamente en términos de los dos pesos que conforman la red, por simplicidad se omiten los sesgos también conocidos como *bias*. La función de pérdida se encuentra en términos de todos los parámetros de la red, los pesos y sesgos, por lo que en general, no es posible visualizarla en términos de todos sus parámetros.

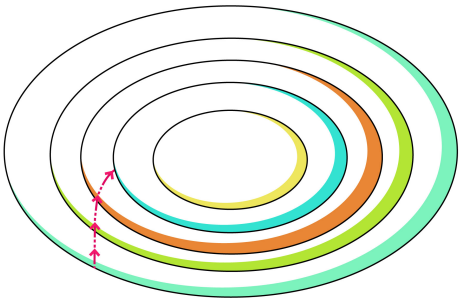


Figura 2.4: Curvas de nivel de una función de pérdida o error en términos de dos pesos.

Para poder atacar el problema de encontrar el mínimo de la función de pérdida, se emplea un método de optimización conocido como descenso del gradiente.

Las curvas de nivel 2.4 nos sirven para ejemplificar la idea central detrás de este método. Se trata de encontrar el mínimo de una función haciendo uso de su derivada, dando pequeños pasos en dirección del mínimo dictados por un hiperparámetro llamado α, también conocido como tasa de aprendizaje. Para plantear la expresión de descenso del gradiente, necesitamos calcular las derivadas parciales de la función de pérdida y considerar al hiperparámetro α. Cada paso de descenso del gradiente los pesos se actualizan de acuerdo a la expresión:

$$\boldsymbol{\Delta\omega} = -\alpha \left(\frac{\partial \mathcal{L}}{\partial \omega_1} i + \frac{\partial \mathcal{L}}{\partial \omega_2} j \right) \tag{2.6}$$

En la expresión (2.6) es posible ver que se requiere calcular el gradiente de la función de pérdida o error. Este cómputo se logra mediante otro algoritmo conocido como propagación hacia atrás o *backpropagation*.

Las derivadas parciales de la expresión (2.6) nos indican una medida de la mejora que estamos logrando al movernos en sus respectivas direcciones o bien, que tanto está cambiando el error al movernos en dichos ejes. La tasa de aprendizaje regula que tan grandes serán los pasos. En otras palabras, solo se están dando pasos siguiendo la dirección del gradiente.

Si deseamos utilizar descenso del gradiente, nos enfrentamos ante un primer problema al trabajar con el modelo 2.2 pues se tiene como función de activación a una función discontinua. Para poder aplicar descenso del gradiente, se requiere que la función sea continua y suave. Tras varios años de investigación fue hasta 1974 cuando Paul Werbos de la Universidad de Harvard mostró en su tesis doctoral el proceso de entrenamiento de una red neuronal artificial a través del método de propagación hacia atrás. La tesis, junto con algunos anexos, se puede encontrar en su libro, *The Roots of Backpropagation: From Ordered Derivatives to Neural Networks and Political Forecasting* [48].

Dos de las propuestas que hace Werbos en su trabajo doctoral son:

- Eliminar el umbral o desfase T presente en la función de activación, añadiendo a la red neuronal una entrada adicional igual a 1, que será multiplicada por un peso que llamaremos w_0, con $w_0 = T$. Con esto se logra que la función de activación esté centrada en 0.
- Cambiar la función de activación por una función continua y suave. Se propone el uso de una sigmoide. Podemos definir a la función sigmoide como:

$$\sigma(\beta) = \frac{1}{1 + e^{-\beta}} \tag{2.7}$$

  De esta expresión podemos notar lo siguiente. Si $\beta = 0$ entonces la función toma el valor de 0.5. Si β es un valor considerablemente grande entonces $e^{-\beta}$ es extremadamente pequeño, por lo que la función toma un valor asintótico de 1.

Finalmente, si β toma un valor extremadamente negativo entonces, $e^{-\beta}$ resulta en un valor positivo extremadamente grande por lo que la función sigmoide tiende asintóticamente a cero. En la figura 2.5 se muestra una gráfica de la función sigmoide descrita.

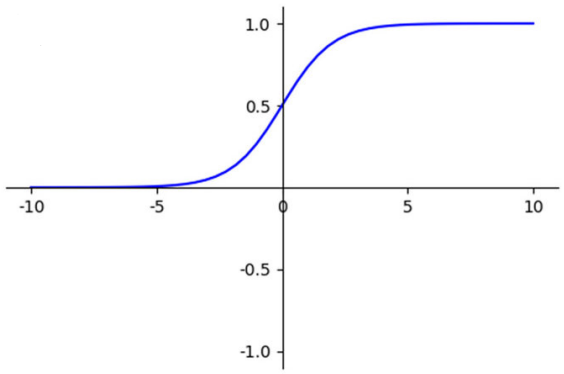


Figura 2.5: La función sigmoide como función de activación.

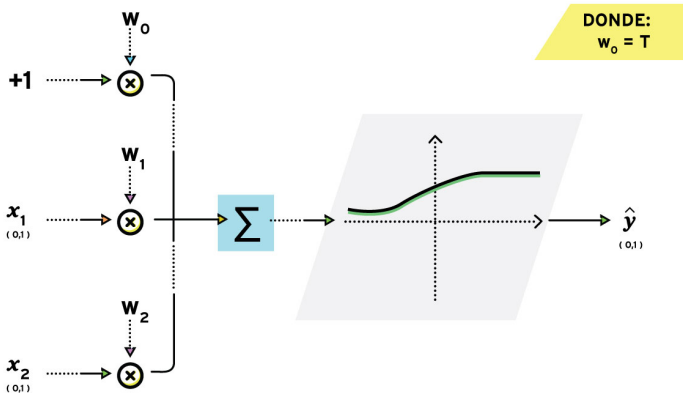


Figura 2.6: Neurona artificial con una función de activación suave y centrada en 0.

Con estas nuevas condiciones, finalmente se pueden calcular las derivadas parciales de la expresión (2.6) para poder plantear una manera de entrenar a la red neuronal.

Para continuar con el desarrollo, consideremos un modelo compuesto por dos neuronas basado en el originalmente propuesto. Este modelo se muestra en la figura 2.7. Debemos notar que es un modelo que consta de solo dos parámetros, ω_1 y ω_2.

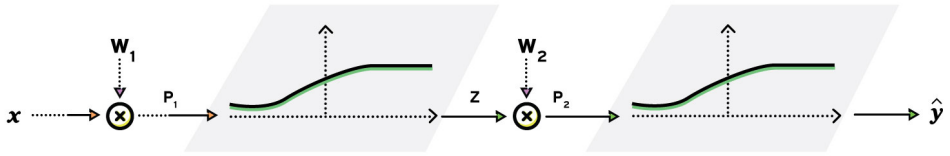


Figura 2.7: Red neuronal con un único atributo de entrada y una sola salida.

Lo que deseamos conocer es como cambia nuestra función de pérdida si movemos el valor del peso ω_2. Para ello, necesitamos calcular la derivada parcial de la pérdida con respecto a ω_2. Notamos que para llevar a cabo el cálculo, necesitamos conocer previamente como cambia la pérdida cuando cambia la salida de la red, además debemos conocer como cambia la pérdida cuando cambia el producto p_2. En otras palabras, para conocer como cambia una variable que se encuentra en un extremo con respecto a otra que se ubica en el otro extremo, que pasa por variables intermedias, simplemente debemos aplicar la regla de la cadena, a este proceso es a lo que se le conoce como propagación hacia atrás. Lo anterior puede escribirse como sigue:

$$\frac{\partial \mathcal{L}}{\partial \omega_2} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial \omega_2} \tag{2.8}$$

Podemos ver que el término de la derecha se puede expandir por medio de la regla de la cadena, de forma que:

$$\frac{\partial \hat{y}}{\partial \omega_2} = \frac{\partial \hat{y}}{\partial p_2} \cdot \frac{\partial p_2}{\partial \omega_2} \tag{2.9}$$

Sustituyendo la expresión (2.9) en (2.8), se tiene lo siguiente:

$$\frac{\partial \mathcal{L}}{\partial \omega_2} = \underbrace{\frac{\partial \mathcal{L}}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial p_2}} \cdot \frac{\partial p_2}{\partial \omega_2} \tag{2.10}$$

Para conocer como cambia la pérdida con respecto al otro peso aplicamos el mismo método:

$$\frac{\partial \mathcal{L}}{\partial \omega_1} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial z} \cdot \frac{\partial z}{\partial p_1} \cdot \frac{\partial p_1}{\partial \omega_1} \tag{2.11}$$

Igual que en el caso anterior, podemos expandir uno de los términos:

$$\frac{\partial \mathcal{L}}{\partial \omega_1} = \underbrace{\frac{\partial \mathcal{L}}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial p_2}} \cdot \frac{\partial p_2}{\partial z} \cdot \frac{\partial z}{\partial p_1} \cdot \frac{\partial p_1}{\partial \omega_1} \tag{2.12}$$

Con las expresiones (2.10) y (2.12) calculadas podemos ahora obtener algunas de las derivadas. Comenzamos por calcular el primer término (de izquierda a derecha) de la expresión (2.10):

$$\frac{\partial \mathcal{L}}{\partial \hat{y}} = (\hat{y} - y) \tag{2.13}$$

De la figura 2.7, podemos notar que $p_2 = \omega_2 \cdot z$ entonces:

$$\frac{\partial p_2}{\partial \omega_2} = z \tag{2.14}$$

Debemos notar que para calcular la derivada de la salida con respecto al producto p_2 es necesario conocer la derivada de la función sigmoide. Vale la pena realizar el desarrollo para llegar a una conclusión interesante.

$$\frac{d\sigma}{d\beta} = \frac{d}{d\beta}\left(\frac{1}{1+e^{-\beta}}\right) = (1+e^{-\beta})^{-2} \cdot e^{-\beta} = \frac{e^{-\beta}}{(1+e^{-\beta})} \cdot \frac{1}{(1+e^{-\beta})} \tag{2.15}$$

Si además hacemos lo siguiente en el numerador para simplificar la expresión (2.15) :

$$\begin{aligned} \frac{1+e^{-\beta}-1}{(1+e^{-\beta})} \cdot \frac{1}{(1+e^{-\beta})} &= \left(\frac{1+e^{-\beta}}{1+e^{-\beta}} - \frac{1}{1+e^{-\beta}}\right) \cdot \left(\frac{1}{1+e^{-\beta}}\right) \\ &= (1-\sigma) \cdot \sigma \end{aligned} \tag{2.16}$$

Del desarrollo anterior, podemos concluir que la derivada de una función sigmoide está dada en términos de la misma sigmoide. Aplicando esta noción podemos escribir que:

$$\frac{\partial \hat{y}}{\partial p_2} = \hat{y} \cdot (1 - \hat{y}) \tag{2.17}$$

Con este último paso, se tiene completo el cálculo de como varía la función de pérdida si cambia el valor del peso ω_2. Se calcularon todos los términos de la expresión (2.10).

De este ejemplo se pueden realizar las siguientes observaciones:

- Realizar el algoritmo de propagación hacia atrás requiere de una gran cantidad de cálculos. Esto fue evidente al mostrar un ejemplo que consta de solo dos neuronas, una a continuación de la otra, con un solo atributo de entrada.

- En las ecuaciones (2.10) y (2.12) se puede observar un comportamiento que aparece al aplicar el método de propagación hacia atrás. Varios de los términos que se requieren para calcular la derivada de la función de pérdida con respecto a uno de los pesos ya fueron calculados previamente durante el cálculo de la derivada con respecto a otro de los pesos. Aquellos términos que se repiten en la expresión (2.12) que ya fueron calculados en la expresión (2.10) son señalados con una llave.

  Este comportamiento permite reutilizar gran parte del cómputo realizado previamente. Esto también ocurre en otros algoritmos como en la transformada rápida de Fourier (FFT) donde se reutilizan los resultados parciales calculados previamente para hacer a este un algoritmo rápido y eficiente.

2.3. Redes Neuronales Profundas

En 2012 Alex Krizhevsky, Ilya Sutskever y Geoffrey Hinton de la Universidad de Toronto en Canadá sorprendieron al mundo con su trabajo [31] justo cuando en muchas partes se dudaba de la utilidad de las redes neuronales y se estaba por abandonar la investigación en ellas. Esta red neuronal profunda pasó a conocerse como AlexNet y consta de poco más de 60 millones de parámetros entrenables. El propósito era clasificar imágenes de entre más de mil categorías.

El aprendizaje profundo es un subcampo del aprendizaje de máquina. Se trata de un nuevo paradigma del aprendizaje de representaciones a partir de los datos, se basa en el aprendizaje de capas sucesivas de representaciones cada vez más significativas. De acuerdo con [7], la palabra *profundo* no hace referencia a que se logra algún tipo de "comprensión profunda" de los datos. En realidad, hace referencia a la idea de emplear capas sucesivas de representaciones donde el número de capas que contribuyen a un modelo es lo que se denomina profundidad del modelo.

El aprendizaje profundo comúnmente emplea decenas o incluso cientos de capas, todas ellas se aprenden automáticamente a partir de la exposición a datos de entrenamiento. Aquellos modelos que sólo aprenden una o dos capas se les suele llamar modelos de aprendizaje superficial.

Para poder hacer uso de un modelo de aprendizaje profundo o de aprendizaje de máquina es necesario contar con los siguientes tres elementos:

- **Datos de entrada**. Si la tarea que se quiere lograr es clasificar imágenes, los datos de entrada serán imágenes. Si en cambio, el objetivo es hacer reconocimiento de voz, estos datos serán archivos de audio de personas hablando.
- **Datos etiquetados con la salida deseada**. En una tarea de clasificación de imágenes, los resultados esperados corresponderían a etiquetas como "perro", "gato". En la tarea de reconocimiento de voz, las salidas del modelo podrían ser transcripciones de los archivos de audio.

- **Una función de pérdida o error**. Para saber si el modelo o algoritmo se está desempeñando apropiadamente, se requiere determinar la distancia entre la salida actual del algoritmo y la salida deseada, esto se cuantifica definiendo una función de pérdida. Los valores de la función de pérdida se utilizan como retroalimentación para ajustar la manera como se desempeña el modelo. Este proceso de ajustar los parámetros del modelo es lo que se conoce como aprendizaje.

El reto principal al que se enfrentan el aprendizaje de máquina y el aprendizaje profundo es lograr transformar de forma significativa los datos [7]. Un modelo de aprendizaje de máquina se encarga esencialmente de transformar los datos de entrada en salidas significativas, este proceso se "aprende" a partir de exponer al modelo a ejemplos conocidos de entradas y salidas. El aprendizaje profundo, es una disciplina práctica en la que las ideas se prueban más a menudo experimentalmente que teóricamente.

Gran parte del desempeño que pueden lograr los modelos de aprendizaje de máquina depende en gran medida de buscar la representación adecuada para los datos de entrada. El concepto de representación debe entenderse como la forma en la cual se observa, representa, o codifican los datos. Si se está trabajando con imágenes, una imagen a color se puede codificar en formato RGB (rojo, verde, azul) o en formato HSV (tono, saturación, valor). Estos formatos corresponden a dos diferentes tipos de representación de los mismos datos. Algunas tareas que pueden ser difíciles de lograr con una representación se pueden volver sencillas al pasar a otra.

Podemos distinguir entre cuatro diferentes arquitecturas de redes neuronales profundas.

1. **Redes neuronales *fully-connected***. Comúnmente se utilizan para procesar datos estructurados, es decir, datos organizados por instancias o vectores de características, donde cada elemento de la instancia es un atributo o característica.

2. **Redes neuronales convolucionales**. Se usan comúnmente en aplicaciones que involucran imágenes como entrada, estás aplicaciones pueden ser clasificación, detección, segmentación, entre otras.

3. **Redes neuronales recurrentes**. Este tipo de arquitecturas están diseñadas para manejar secuencias de datos como entrada. Encontramos este tipo de entradas cuando se trabaja por ejemplo con procesamiento de texto (procesamiento de lenguaje natural) o con procesamiento de voz.

4. **Redes neuronales no supervisadas**. En este grupo se encuentran los *autoencoders* y las redes adversarias generativas (GANs). Este tipo de arquitecturas no introducen nuevos componentes estructurales, como en los casos anteriores sino que usan la estructura más apropiada para el problema. Por ejemplo, una red adversaria o un *autoencoder* para imágenes hará uso de convoluciones. Las GANs por ejemplo, son modelos capaces de generar imágenes realistas. Para ello

utilizan dos redes, un generador y un discriminador. Donde el modelo generador tiene por objetivo producir la salida más auténtica posible que será introducida al discriminador que busca diferenciar si la imagen es real o proviene del generador.

Existen otras formas de agrupar los modelos de aprendizaje profundo, pudiendo ser también por el tipo de aprendizaje que realizan. Dada la naturaleza de esta tesis nos enfocaremos en describir exclusivamente a las redes neuronales convolucionales.

2.3.1. Redes Neuronales Convolucionales

Las redes neuronales convolucionales también conocidas como CNNs (*Convolutional Neural Networks*) son un tipo específico de modelo de red neuronal que usualmente en su forma más básica se compone de los siguientes tipos de capas en diferente número: capa de convolución, capa de *pooling* y una capa *fully-connected* [4].

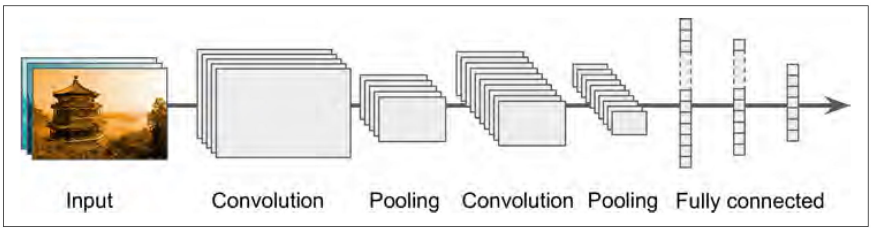


Figura 2.8: Arquitectura típica de una red neuronal convolucional. Figura tomada de [21].

Estos tipos de capas se explican a continuación:

- Capa de convolución (CONV). Esta capa emplea filtros que realizan la operación de convolución a lo largo de toda la entrada (I) con respecto a sus dimensiones. Los hiperparámetros de esta capa son el tamaño del filtro (f) y el *stride* o paso (s). La salida resultante (O) se denomina mapa de características o mapa de activación.
- Capa de *pooling* (POOL): Esta capa se encarga de realizar la operación de *down-sampling* o mejor dicho, en esta capa se realiza una disminución a la resolución con lo que se logra algo de invariancia espacial. Generalmente se aplica después de una capa de convolución. Dos tipos particulares de *pooling*, son *max* y *average pooling* donde se toma el valor máximo o el valor promedio de una vecindad de valores de una ventana de un tamaño previamente definido. La ventana se va deslizando por cada canal del mapa de características mientras aplica la operación de *pooling* por lo que el número de canales del mapa de características no cambia, sólo cambia su resolución.

  - **Observación**. *Max pooling* preserva las características detectadas y es el más comúnmente utilizado. *Average Pooling* no es tan común, sin embargo, fue empleado en la famosa arquitectura LeNet.

- Capa *fully-connected* (FC): La capa *fully-connected* (FC) opera con un vector como entrada, resultado de aplanar el último mapa de características obtenido por la red neuronal convolucional. Cada entrada está conectada a todas las neuronas. Si la capa está presente, las capas FC generalmente se encuentran al final de las arquitecturas CNN y se utilizan como clasificadores para dos o más clases posibles.

Como se comentó, en las capas convolucionales se encuentran los filtros que al aplicarlos, por ejemplo, a una imagen de entrada forman un primer mapa de características. Estos filtros poseen varios hiperparámetros que es importante conocer.

- Dimensiones de un filtro: un filtro de tamaño $f \times f$ aplicado a una entrada con c canales es en realidad un volumen de dimension $f \times f \times c$ que realiza la operación de convolución en una entrada de tamaño $I \times I \times c$ y genera un mapa de características de salida (también llamado mapa de activación) de tamaño $O \times O \times 1$.

  - **Observación**: Aplicar n filtros de tamaño $f \times f$ da como resultado un mapa de características de salida de tamaño $O \times O \times n$.

- *Stride*: El *stride* (s) indica el número de píxeles por los cuales la ventana se mueve después de realizar la operación correspondiente. Este hiperparámetro se aplica tanto para una ventana que realiza la operación de convolución como de *pooling*.

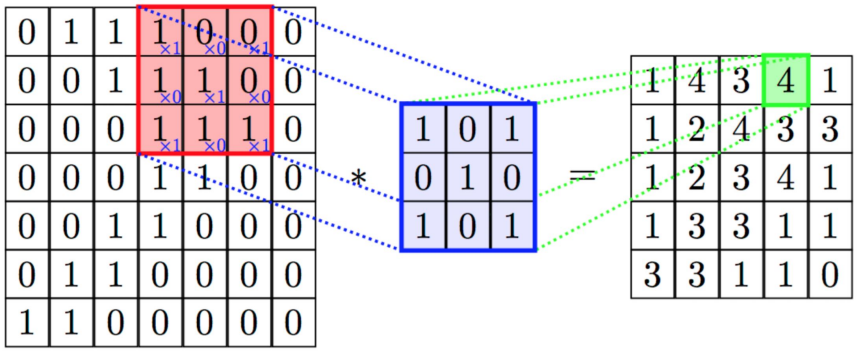


Figura 2.9: La convolución de un filtro de $3 \times 3 \times 1$ (*stride* $= 1$) con una imagen de $7 \times 7 \times 1$ (*padding* $= 0$) genera un mapa de características de $5 \times 5 \times 1$. Figura tomada de [36].

- *Zero-padding* (p): *Zero-padding* se refiere al proceso de añadir ceros a cada lado de los límites de la entrada. Este valor se puede especificar manualmente o bien de forma automática. Para este trabajo, $p = 0$ significa que no se hace uso de *padding* a la entrada, mientras que $p = 1$ significa que la entrada es rodeada por una sola frontera de ceros. Se puede añadir una frontera adicional, además de la anterior, a la entrada. Por ejemplo, con $p = 2$ indicaríamos una frontera adicional.

  - **Observación**: Al momento de realizar la operación de convolución con un filtro se puede especificar si a la entrada se le va realizar el proceso de *padding* o no, a través de los argumentos *valid* y *same*. Con el argumento *valid* se especifica que no se hará uso de *padding* en la entrada. Mientras que con el argumento *same*, indicamos que se realice el *padding* apropiado a la entrada de forma que la salida resultante del proceso de convolución tenga el mismo tamaño que la entrada.

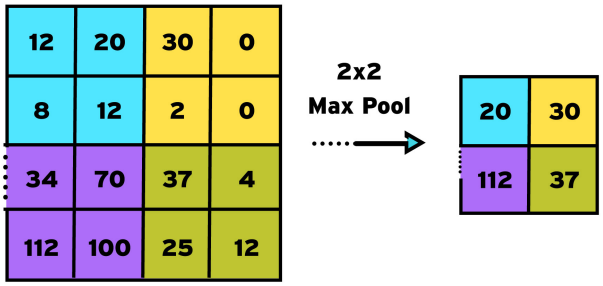


Figura 2.10: Mapa de características resultante de 2×2 despúes de la aplicación de una ventana de *max-pooling* de 2×2 con *stride* $= 2$ sobre el mapa de características original de 4×4.

Los hiperparámetros comentados anteriormente se relacionan de la siguiente forma en la capa convolucional. Si seguimos la notación anterior tenemos que si I es el tamaño de la entrada, f el tamaño de los filtros en una capa, p el uso de zero-padding, s el *stride*, entonces el tamaño de la salida del mapa de características O, está dado por:

$$O = \frac{I + p_{start} + p_{end} - f}{s} + 1 \tag{2.18}$$

Usualmente, se tiene que $p_{start} = p_{end} = p$, por lo que la expresión anterior cambia a:

$$O = \frac{I + 2p - f}{s} + 1 \tag{2.19}$$

Para una red neuronal convolucional una posible notación puede ser la siguiente:

- $a^{[\ell]}$: Activaciones de la capa ℓ donde $a^{[0]}$ representa las activaciones de la capa 0, en otras palabras la imagen de entrada.
- $z^{[\ell]}$: Logits
- $g(\cdot)$: Función de activación

Dicho la anterior, para una primera capa de convolución se tendría lo siguiente:

$$z^{[1]} = W^{[1]}a^{[0]} + b^{[1]} \tag{2.20}$$

$$a^{[1]} = g(z^{[1]}) \tag{2.21}$$

Estas ideas se ilustran en la figura 2.11:

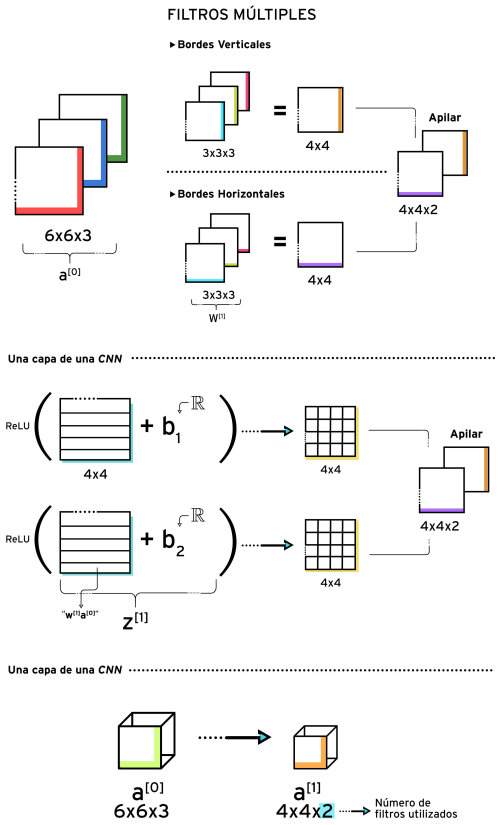


Figura 2.11: Primera capa convolucional de una *CNN*.

Al igual que los modelos de redes neuronales del tipo *fully-connected*, las redes neuronales convolucionales emplean diferentes funciones de transformación para obtener las activaciones de la siguiente capa. A continuación se describen ejemplos de algunas de las funciones más comunes que son empleadas en las capas intermedias de este tipo de modelos.

- **Sigmoide.** La función de activación sigmoide se utiliza en muchas redes neuronales en la última capa (capa de salida). Esta función entrega como salida un valor entre 0 y 1. Esto característica permite que las salidas puedan ser tratadas como probabilidades. Esta función se puede definir de la siguiente manera:

$$g(z) = sigmoid(z) = \frac{1}{1 + e^{-z}} \tag{2.22}$$

- **Tangente hiperbólica.** Hasta hace poco tiempo esta función era la función de activación más popular, por lo que todavía se puede encontrar en algunos modelos. Un inconveniente que presenta esta función es que para valores lejanos a cero presenta un comportamiento muy "plano". Esto trae como resultado un pequeño gradiente por lo que durante el entrenamiento la red puede tardar mucho tiempo en cambiar su comportamiento. A esta función la podemos definir de la siguiente manera:

$$g(z) = tanh(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}} \tag{2.23}$$

- **Softmax.** La función softmax se puede ver como una generalización de la función sigmoide. Se tiene como entrada un vector de *logits* o puntajes $\boldsymbol{z} \in \mathbb{R}^n$ que al pasar por la función softmax colocada al final de una arquitectura, se genera un vector de probabilidades de salida $\boldsymbol{p} \in \mathbb{R}^n$. El vector de probabilidades de salida se puede escribir como:

$$\boldsymbol{p} = \begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_n \end{bmatrix} \tag{2.24}$$

  donde cada uno de sus elementos se puede obtener a partir de la función softmax como:

$$p_i = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}} \tag{2.25}$$

- **Rectified Linear Unit (ReLU)**. ReLU es una función de activación $g(\cdot)$ que se aplica a todos los elementos de un volumen resultado de haber realizado la operación de convolución de una entrada con un número de filtros dado y sumar a ello un valor de *bias*. Utilizar esta función tiene por objetivo introducir no linealidades en la red neuronal. Esta función se puede definir como sigue:

$$g(z) = ReLU(z) = max(0, z) \tag{2.26}$$

Existen algunas variantes a la versión original y se muestran a continuación:

- **Leaky ReLU**. Esta función busca mitigar algunas de las deficiencias que se han observado al emplear la función ReLU clásica. Puede verse que para valores negativos, la función ReLU es siempre cero; esto no siempre es deseable. Para solucionar esto, puede emplearse leaky ReLU que puede definirse como sigue:

$$g(z) = max(\epsilon \cdot z, z) \text{ con: } \epsilon \ll 1 \tag{2.27}$$

- **Exponential Linear Unit (ELU).** La función ELU tiene la particularidad de ser diferenciable en todo su dominio. La función se puede definir como sigue:

$$g(z) = max(\ \epsilon\ (e^z - 1), z) \text{ con: } \epsilon \ll 1 \tag{2.28}$$

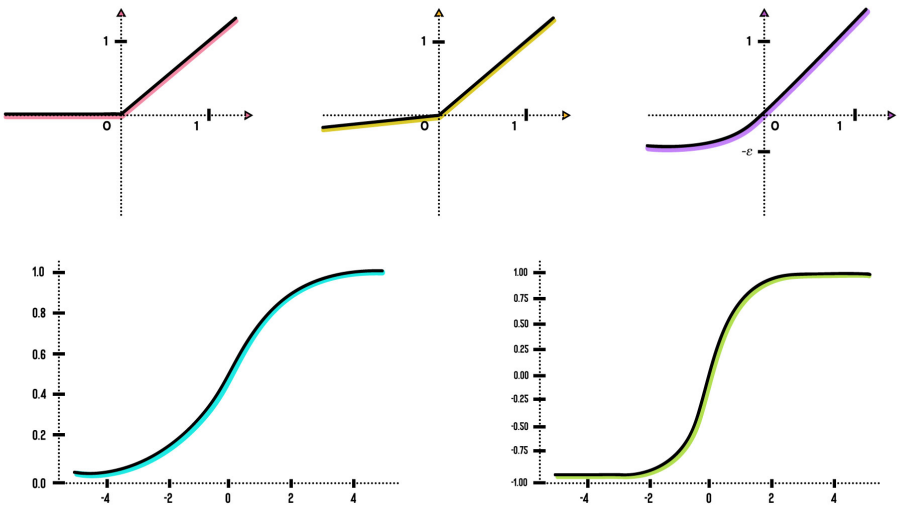


Figura 2.12: Funciones de activación. ReLU (roja), leaky ReLU (amarilla), ELU (púrpura), sigmoide (cian) y tangente hiperbólica (verde).

Para conocer el desempeño de nuestro modelo durante el entrenamiento necesitamos de una medida que nos indique que tanto difieren la salida del modelo con respecto a la salida real, para cuantificar esto se define una función de pérdida. Dependiendo de la naturaleza de la tarea que estemos intentando abordar, podemos definir tres principales funciones de pérdida.

- Clasificación Binaria y clasificación multi-etiqueta: Para este problema de clasificación se suele utilizar como función de pérdida a la entropía cruzada binaria $\mathcal{L}(\hat{y}, y)$ que se define de la siguiente manera:

$$\mathcal{L}(\hat{y}, y) = -[y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})] \tag{2.29}$$

  En nuestro conjunto de datos para una tarea de clasificación binaria las salidas de cada instancia podrían estar etiquetadas como 1 o 0 para cada clase. Entonces, si $y = 1$ la función anterior toma la forma siguiente:

$$\mathcal{L}(\hat{y}, y) = -y \log(\hat{y}) \tag{2.30}$$

  Para este caso se requiere que $y \log(\hat{y})$ y $\hat{y}$ sean lo más grande posible puesto que se busca que $\mathcal{L}$ sea mínimo.

  Para el segundo caso donde $y = 0$, la función de pérdida puede escribirse como:

$$\mathcal{L}(\hat{y}, y) = -\log(1 - \hat{y}) \tag{2.31}$$

  De igual manera para minimizar a $\mathcal{L}$, buscamos que $\log(1 - \hat{y})$ sea lo más grande posible por lo que $\hat{y}$ deberá ser un valor lo más pequeño posible.

  Si deseamos conocer la pérdida para todo el conjunto de datos de entrenamiento entonces podemos escribir la siguiente función de costo:

$$J = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(\hat{y}^{(i)}, y^{(i)}) \tag{2.32}$$

  Donde m es el número total de ejemplos en el conjunto de datos y los superíndices entre paréntesis i se utilizan para hacer referencia a cada uno de los elementos del conjunto.

- Clasificación Multiclase: Para este tipo de clasificación se utiliza como función de pérdida la entropía cruzada categórica, la cual no difiere de su versión binaria y se define como sigue:

$$\mathcal{L}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = -\sum_{i} y_i \log(\hat{y}_i) \tag{2.33}$$

  Donde $\hat{\boldsymbol{y}}$ son las predicciones del modelo después de haber pasado por una función softmax y $\boldsymbol{y}$ son las etiquetas reales expresadas en *one hot encoding*. El índice i se utiliza para hacer referencia a cada elemento del vector de predicciones y del

vector de salidas reales, estos vectores tienen tantos elementos como número de clases se tengan. Consideremos el siguiente ejemplo, supongamos que tenemos 3 clases diferentes y para una primer instancia la red neuronal arroja el siguiente vector de predicciones:

$$\hat{\boldsymbol{y}} = \begin{bmatrix} 0.7 \\ 0.2 \\ 0.1 \end{bmatrix} \ , \ \boldsymbol{y} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{2.34}$$

Entonces, si calculamos la pérdida:

$$\mathcal{L}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = -[1 \cdot \log(0.7) + 0 \cdot \log(0.2) + 0 \cdot \log(0.1)] = 0.15490195998 \tag{2.35}$$

Si ahora deseamos calcular la función de pérdida para todo el conjunto de datos entonces como en el caso anterior, podemos definir una función de costo:

$$J = \frac{1}{m} \sum_{j=1}^{m} \mathcal{L}(\hat{\boldsymbol{y}}^{(j)}, \boldsymbol{y}^{(j)}) \tag{2.36}$$

2.3.1.1. Preprocesamiento de datos

Muy frecuentemente para mejorar el desempeño de los modelos de redes convolucionales se suele realizar algún tipo de preprocesado a la imagen. Algunas de las técnicas más comunes se describen brevemente a continuación:

- **Aumento de datos**: Las redes convolucionales y en general los modelos de aprendizaje profundo necesitan de una enorme cantidad de datos para poder ser capaces de generalizar una tarea a partir de un proceso de entrenamiento. El efecto de sobreajuste u *overfitting* ocurre cuando se tienen muy pocos ejemplos de los que aprender, lo que impide entrenar un modelo capaz de generalizar a nuevos datos. Si tuviesemos datos infinitos, el modelo estaría expuesto a todos los aspectos posibles de la distribución de los datos y nunca se sobreajustaría el modelo.

  El aumento de datos consiste en generar más datos de entrenamiento por medio de transformaciones aleatorias a los existentes de manera que se obtengan imágenes de apariencia creíble. El objetivo es que durante el entrenamiento nuestro modelo nunca vea exactamente la misma imagen dos veces, esto ayuda a que el modelo se exponga a más aspectos de los datos y generalice mejor.

  Algunas de las transformaciones que se le pueden aplicar a una imagen son las siguientes:

  Existen otras transformaciones que se le pueden aplicar a una imagen como pueden ser:

- **Cambios en los canales de color de la imagen.** Ayuda a contemplar el ruido que pudiese ocurrir con la exposición a una fuente de luz. Para ello, se cambian ligeramente los valores de los canales RGB.
- **Adición de ruido.** Ayuda a incrementar la tolerancia del modelo frente a variaciones en la calidad de las imágenes de entrada.
- **Pérdida de información.** Se ignoran algunas partes de la imagen. Intenta contemplar posibles pérdidas de partes en la imagen.
- **Cambios en el contraste.** Se utiliza para contemplar cambios en la luminosidad. Ayuda a mitigar diferencias en la exposición debido al momento del día en que se pudo adquirir la imagen.

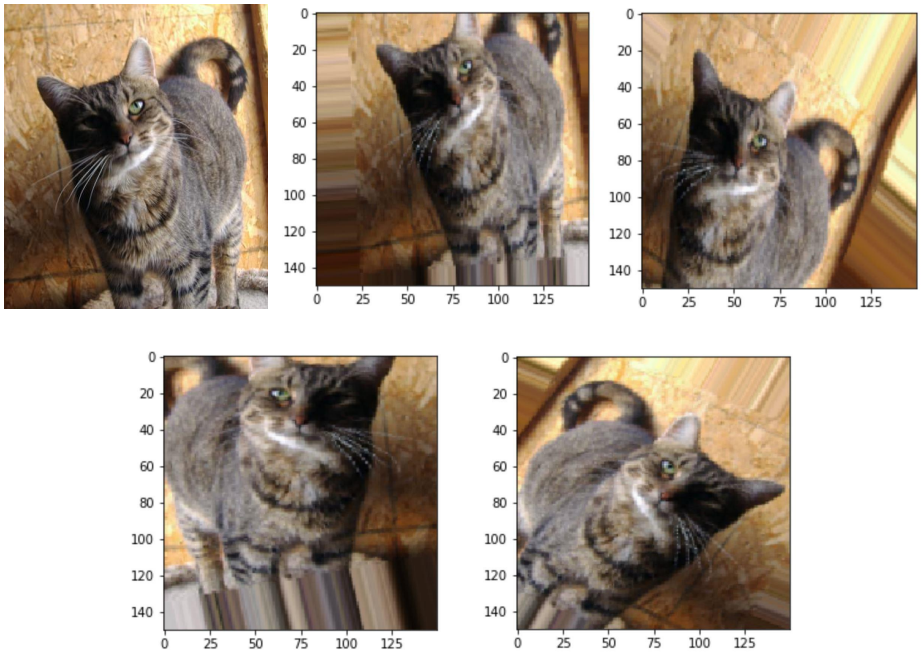


Figura 2.13: Imagen original (celda superior izquierda) a la que se aplicaron transformaciones aleatorias de rotación, giros horizontales, corrimientos a lo ancho/alto y acercamientos (*zooms*) como parte del aumento de datos. Imagen perteneciente al conjunto de datos *dogs vs cats*.

- ***Batch normalization:*** Este algoritmo hace posible entrenar redes muy profundas pues permite acelerar el aprendizaje de los parámetros en la red. Existe un debate en la comunidad sobre si esta técnica se debe aplicar a los *logits* o bien a las activaciones, en la practica es más frecuente encontrar que se aplique a los *logits*, es decir, antes de aplicar la no-linealidad.

Si denotamos μ_B y σ_B^2 como la media y varianza del *batch* y además denotamos a $z^{(1)}, z^{(2)}, \ldots, z^{(m)}$ como los *logits* de una cierta capa, entonces podemos plantear las ecuaciones de este método como sigue:

$$\mu_B = \frac{1}{m} \sum_i z^{(i)} \tag{2.37}$$

$$\sigma_B^2 = \frac{1}{m} \sum_i (z^{(i)} - \mu_B)^2 \tag{2.38}$$

$$z_{norm}^{(i)} = \frac{z^{(i)} - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \tag{2.39}$$

$$\tilde{z}^{(i)} = \gamma \, z_{norm}^{(i)} + \beta \tag{2.40}$$

Con la expresión (2.39) se logra que las compomentes de $z^{(i)}$ tengan una distribución con media 0 y varianza 1. El parámetro ϵ se añade para fines de estabilidad numérica pues evita que en algún cálculo el denominador tome un valor de 0.

Sin embargo, no siempre es deseable que las componentes de $z^{(i)}$ tenga dicha distribución. Por ello, en la expresión (2.40) se introducen dos hiperparámetros γ y β. Estos son dos hiperparámetros entrenables que pueden ser optimizados a través de algún algoritmo como descenso del gradiente o cualquier otro y su actualización se hace de la misma manera que los pesos.

El efecto de estos hiperparámetros es que permiten que los *logits* de las capas ocultas tengan una media y varianza cualquiera, por lo que los nuevos valores $\tilde{z}^{(i)}$ serían aquellos que se utilizarían en cualquier cálculo posterior. Por ejemplo, si $\gamma = \sqrt{\sigma_B^2 + \epsilon}$ y $\beta = \mu_B$, y sustituyendo en (2.40) se tiene que:

$$\tilde{z}^{(i)} = z^{(i)} \tag{2.41}$$

Por lo que en la expresión (2.41) se está calculando esencialmente una función identidad.

Finalmente, la ecuación (2.40) se puede reescribir de la siguiente manera, pues este método se suele aplicar tanto a las entradas como a los valores intermedios en una red neuronal. El superíndice $[\ell]$ hace referencia a los *logits* de una capa en específico mientras que (i) denota alguna unidad particular de la capa $[\ell]$.

$$x^{(i)} \longleftarrow \gamma \frac{x^{(i)} - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} + \beta \tag{2.42}$$

$$z^{[\ell](i)} \longleftarrow \gamma \frac{z^{[\ell](i)} - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} + \beta \tag{2.43}$$

2.3.1.2. Entrenamiento de una red neuronal profunda

Algunos conceptos que debemos tener presentes al momento de entrenar un modelo de aprendizaje profundo son los siguientes:

- **Época**. Durante el proceso de entrenamiento, el concepto de época se refiere a completar la iteración en la cual el modelo termina de ver todo el conjunto de datos de entrenamiento para actualizar sus pesos.
- **Descenso del gradiente por mini-batches (*Mini-batch gradient descent*)**. Durante la etapa de entrenamiento, la actualización de los pesos por lo general, no se realiza procesando todo el conjunto de datos de entrenamiento a la vez o en el extremo opuesto, un solo ejemplo a la vez. El paso de actualización de los pesos se realiza en *mini-batches* donde el número de muestras en un *batch* es un hiperparámetro que se puede ajustar.
- **Propagación hacia atrás (*Backpropagation*)**. En la sección anterior se introdujo el algoritmo de propagación hacia atrás para un sencillo ejemplo de red neuronal. Este algoritmo es una de las ideas más importantes pues hicieron posible entrenar a este tipo de modelos. A manera de síntesis podemos decir que se trata de un método para actualizar los pesos en la red neuronal teniendo en cuenta la salida real del conjunto de datos y la salida deseada proveniente del modelo. La derivada de la función de pérdida / costo con respeto a cada peso w se calcula utilizando la regla de la cadena.

  Este algoritmo es necesario para poder actualizar los pesos a través del método de descenso del gradiente dado por la siguiente expresión:

$$w \longleftarrow w - \alpha \nabla J(\boldsymbol{w}) \tag{2.44}$$

  Una manera alternativa igualmente común de escribir la expresión anterior es:

$$w_j \longleftarrow w_j - \alpha \frac{\partial J(\boldsymbol{w})}{\partial w_j} \tag{2.45}$$

  Recordemos que en una red neuronal los pesos se actualizan de acuerdo al siguiente procedimiento:

  1. Tomar un *batch* de datos de entrenamiento y realizar la propagación hacia adelante para calcular la función de pérdida.
  2. Propagar hacia atrás la función de pérdida para calcular el gradiente de la pérdida con respecto a cada peso.
  3. Actualizar los pesos de la red a partir del gradiente calculado en el paso anterior.

2.3.1.3. *Overfitting* y *underfitting*

Cuando un modelo se desempeña muy bien en el conjunto de datos de entrenamiento pero al ser expuesto a nuevos datos su desempeño es muy pobre, decimos que este modelo está sobreajustado (se presenta *overfitting*). En otras palabras, no generaliza bien a nuevos datos. Este fenómeno es común se obtuvo un modelo muy complejo que se ajusta muy bien a los datos de de entrenamiento pero no así a nuevas observaciones. Al proceso de combatir el fenómeno de *overfitting* se le llama regularización.

Por el contrario, cuando un modelo se desempeña muy pobremente tanto en el conjunto de datos de entrenamiento como en el de validación decimos que el modelo está subajustado, se está ante el fenómeno de *underfitting*. Esto es común cuando el modelo recién se está entrenando y sus parámetros se están ajustando, la red aún no ha modelado las características relevantes del conjunto de datos de entrenamiento.

La meta es que a través del proceso de optimización se pueda obtener un modelo que se desempeñe lo mejor posible en el conjunto de datos entrenamiento y que generalice bien a datos que nunca haya visto antes.

2.3.1.4. Ajustes de parámetros

- **Inicialización Aleatoria de los Pesos.** En una red neuronal profunda los pesos se inicializan comúnmente de forma aleatoria muestreando valores de la distribución normal estándar y multiplicando cada valor por un valor muy pequeño. Esta primera técnica de inicialización tiene como propósito mitigar dos problemas, el problema del desvanecimiento del gradiente y explosión del gradiente.

  El problema de explosión del gradiente ocurre cuando el valor inicial de los pesos es demasiado grande de forma que los valores que toman las activaciones crece de forma exponencial conforme se avanza en profundidad en la red. Cuando estas activaciones se usan en el proceso de propagación hacia atrás es cuando se presenta el problema de la explosión del gradiente. Es decir, los gradientes de la función de costo con respecto a los parámetros son demasiado grandes, esto hace que la función de costo oscile alrededor del valor mínimo.

  En contraste, el problema de desvanecimiento del gradiente ocurre cuando los pesos son inicializados con valores muy pequeños, esto produce que los valores de las activaciones decrezcan exponencialmente conforme se avanza en profundidad en la red. Cuando estas activaciones se utilizan en la propagación hacia atrás es cuando se presenta el problema de desvanecimiento del gradiente, es decir, los gradientes de la función de costo con respecto a los parámetros son demasiado pequeños, lo que lleva a que la función de costo converga antes de que alcance su valor mínimo.

  Podemos decir entonces que inicializar los pesos con valores inapropiados puede llevar a divergencias u ocasionar un entrenamiento mucho más lento. Para evitar

que los gradientes de las activaciones de la red se desvanezcan o exploten, se siguen las siguientes reglas:

- La media de las activaciones debe ser cero.
- La varianza de las activaciones debe permanecer igual a lo largo de todas las capas de la red.

En [28] se puede encontrar una descripción más a fondo de los conceptos antes presentados, el origen del problema a través de una arquitectura ejemplo, simulaciones interactivas que permiten ver el efecto de los diferentes métodos de inicialización así como una justificación formal para uno de estos.

- **Inicialización de Xavier e inicialización de He.** La inicialización de Xavier permite tener pesos iniciales que tienen en cuenta características que son únicas a la arquitectura. La inicialización de Xavier funciona cuando se utiliza *tanh* para generar las activaciones. Otra método común es la inicialización de He, la cual se aplica cuando se utilizan funciones ReLU para generar las activaciones. Con la inicialización de He, los pesos se inicializan multiplicando por un factor de 2 la varianza de la inicialización de Xavier.

  Ambos tipos de inicialización tienen como propósito cuidar que la varianza sea la misma a lo largo de todas las capas y que la media de las activaciones sea 0. Para esto los pesos se inicializan a partir de muestrear aleatoriamente valores de la distribución normal con media cero y la varianza particular para cada método. Las publicaciones respectivas a ambos métodos se pueden consultar en [17] y [24].

- **Transferencia de conocimiento (*Transfer Learning*).** Como ya se ha comentado ampliamente, entrenar un modelo de aprendizaje profundo requiere de una enorme cantidad de datos para que pueda verdaderamente aprender a generalizar una tarea dada, más aún, requiere de una gran cantidad de tiempo de computo. Esto hace que una estrategia común sea aprovechar los pesos de modelos pre-entrenados sobre grandes conjuntos de datos que obtuvieron otros grupos de investigación a partir de entrenamientos que tomaron días, semanas o incluso más. La idea es aprovechar estos modelos pre-entrenados y ajustarlos para la aplicación particular que estemos desarrollando. Dependiendo de la cantidad de datos que se tenga, existen diferentes formas de aplicar esta idea:

  - **Pocos datos**. Cuando se trata de este caso se congelan todas las capas convolucionales y sólo se entrenan los pesos correspondientes al clasificador.
  - **Mediana cantidad de datos.** En este caso se congelan casi todas las capas correspondientes a la parte convolucional excepto la última y la etapa del clasificador pues estas se vuelven a entrenar.
  - **Gran cantidad de datos.** Ninguna capa se congela, se vuelven a entrenar todas las capas de la parte convolucional así como la parte del clasificador, partiendo de los pesos pre-entrenados.

- **Tasas de aprendizaje adaptables.** En la sección previa de introducción a redes neuronales se presentó una descripción completa sobre el hiperparámetro conocido como tasa de aprendizaje en ocasiones denotado como α y otras como η. Se comentó que este hiperparámetro indica a qué tasa se actualizan los pesos en el proceso de optimización. Este valor puede ser fijo o permitir que cambie de forma adaptable. Por ejemplo, el método de optimización más popular lleva por nombre Adam e incorpora una tasa de aprendizaje adaptable.

  Permitir que la tasa de aprendizaje se adapte durante el entrenamiento del modelo hace que el tiempo de entrenamiento se reduzca además de mejorar el resultado del proceso de optimización. Además del optimizador de Adam existen muchos otros métodos que pudiesen entregar mejores resultados para una aplicación dada, entre ellos están Adadelta, Adagrad, RMSprop, descenso de gradiente estocástico (SGD, *Stochastic Gradient Descent*) y descenso del gradiente con momentum. Este último se comenta brevemente a continuación:

  - ***Gradient Descent with Momentum.*** Este método suaviza los pasos u oscilaciones que normalmente están presentes en el método estándar de descenso del gradiente pues se basa en la técnica estadística conocida como *exponentially weighted moving averages* [18, 32] que suaviza este tipo de datos a través de la suma de la muestra actual y las pasadas, ponderadas por coeficientes que decaen exponencialmente. Como consecuencia este método acelera la convergencia al mínimo. En otras palabras, el entrenamiento de un modelo se acelera pues permite el uso de valores más grandes para la tasa de aprendizaje.

    La presencia de oscilaciones en el método de descenso del gradiente alentan la convergencia al mínimo pues se realizan demasiados pasos hasta oscilar lentamente al mínimo de la función de costo. Este problema evita que se puedan emplear tasas de aprendizaje más grandes pues se podría presentar un sobrepaso y diverger. La implementación de este método se puede plantear como sigue [29].

    Para cada iteración t:

    Calcular dw, db para el *batch* o *mini-batch* actual y,

$$v_{dw} = \beta v_{dw} + (1 - \beta)dw \tag{2.46}$$

$$v_{db} = \beta v_{db} + (1 - \beta)db \tag{2.47}$$

$$w \longleftarrow w - \alpha v_{dw} \tag{2.48}$$

$$b \longleftarrow b - \alpha v_{db} \tag{2.49}$$

En este método se tienen dos hiperparámetros α y β, donde este último controla el número de las últimas iteraciones de gradiente a promediar, un valor típico para β suele ser 0.9. Por su parte, dw y db son los gradientes de la función de costo con respecto a los pesos y a los *biases*, respectivamente

- **Regularización**

  - ***Dropout.*** Para mitigar el problema de *overfitting* en los datos de entrenamiento se puede emplear la técnica conocida como *dropout*. Esta técnica se aplica a cada unidad de alguna capa en particular y busca apagar la unidad acorde a una cierta probabilidad p. En otras palabras, considerando la probabilidad de desechar una neurona, se hace un ensayo sobre cada unidad y dependiendo del resultado del ensayo se apaga o no. Por ejemplo, si definimos que $p = 0.2$, quiere decir que cada unidad de alguna capa en especifico tiene una probabilidad de 0.2 de ser desechada y 0.8 de permanecer en el modelo. Algunos *frameworks* manejan el parámetro *keep* tal que el valor que se especifica hace referencia a la probabilidad de conservar una neurona en específico.

    De esta forma lo que se entrena es un modelo de menor complejidad además de evitar que el modelo dependa de conjuntos particulares de características.
  - **Regularización de los pesos.** Otra de las técnicas para mitigar el *overfitting* es añadir un término de regularización a la función de costo a optimizar. Aplicar este tipo de técnicas tiene por objetivo penalizar el valor de los pesos del modelo es decir, que los pesos no sean demasiado grandes y que el modelo no se sobreajuste al conjunto de entrenamiento. A continuación se describe brevemente las dos más comunes:
    - *Regularización L_1 (Lasso).* Esté método añade un término de regularización a la función de costo. Este término adicional está formado por la norma L_1 de los pesos multiplicado por un factor de regularización llamado λ. Con este método se vuelven cero algunos de los pesos por lo que ayuda a la selección de características.
    - *Regularización L_2 (Ridge).* Esté método al igual que el anterior añade un término de regularización a la función de costo. Este término de regularización consiste en el factor λ multiplicado por la norma L_2 al cuadrado de los pesos. Esta penalización trae como consecuencia pesos más pequeños.
  - **Paro Temprano (*Early Stopping*)**. El paro temprano puede servir como una técnica para combatir el *overfitting*, puede considerarse entonces como un método de regularización. Consiste en detener el proceso de entrenamiento de un modelo tan pronto como la función de pérdida de validación se asiente sobre un valor o bien se presente un rebote y comience a aumentar.

2.3.1.5. Métricas para evaluación.

Para evaluar el desempeño de un modelo sobre un conjunto de datos podemos hacer uso de las diferentes métricas que nos brinden esta información. A continuación se presentan tres de las métricas más comunes para evaluar una tarea que involucre clasificación [19].

- **Exactitud (*Accuracy*)**. La exactitud es una métrica para evaluar las predicciones de un modelo que realiza tareas de clasificación. La exactitud puede entenderse como la fracción o porcentaje de las predicciones que el modelo realizó correctamente. Se puede definir de la siguiente manera:

$$Exactitud = \frac{\text{Número de predicciones correctas}}{\text{Número total de predicciones}} \tag{2.50}$$

$$Exactitud = \frac{VP + VN}{VP + VN + FP + FN} \tag{2.51}$$

  Donde VP = verdaderos positivos, VN = verdaderos negativos, FP = falsos positivos y FN = falsos negativos.

- **Precisión (*Precision*)**. La precisión nos indica que proporción de las predicciones identificadas como positivas fue correcta. Se define de la siguiente forma:

$$\text{Precisión} = \frac{VP}{VP + FP} \tag{2.52}$$

  Puede verse que cuando un modelo no genera falsos positivos tiene una precisión de 1.0.

- ***Recall***. El *recall* o sensibilidad nos indica la proporción de positivos verdaderos que se identificó correctamente. Se define de la siguiente manera:

$$Recall = \frac{VP}{VP + FN} \tag{2.53}$$

  Puede verse que un modelo que no genera falsos negativos tiene un *recall* de 1.0.

Capítulo 3

Aprendizaje Profundo para Visión y técnicas de Procesamiento de Imágenes

3.1. Redes Neuronales Profundas para detección

You **O**nly **L**ook **O**nce o YOLOv3 es el estado del arte en sistemas de detección de objetos en tiempo real, la primera versión fue desarrollada por Redmon et al. (2016) [41]. YOLOv3 es extremadamente rápida y precisa [42] y ofrece un excelente balance entre precisión y velocidad.

YOLO es diferente de los sistemas de detección tradicionales, que eran básicamente clasificadores/localizadores modificados, rediseñados para realizar tareas de detección. Estos sistemas de detección clásicos utilizan una ventana deslizante a través de una imagen para buscar objetos en diferentes ubicaciones y escalas. Naturalmente, esta es una operación computacionalmente costosa, por lo que el tamaño de la ventana es fijo. Aquellas regiones de la imagen –analizadas por la ventana deslizante– con la puntuación más alta son consideradas como detecciones.

Este sistema de detección del estado del arte se distingue de otras soluciones al aplicar una única red neuronal a la imagen completa [41] y para una imagen de 320 $\times$ 320 es capaz de realizar inferencia en 22 [ms] en una GPU Titan X. YOLO divide la imagen en regiones y predice cuadros delimitadores, conocidos en la literatura como *bounding boxes*, así como las probabilidades para cada región; los *bounding boxes* son ponderados por las probabilidades predichas [42]. Las detecciones son umbralizadas por un valor de confianza de 0.25, por lo que cualquier objeto de interés que posea un valor de confianza mayor o igual a 0.25 será detectado. De esta forma solo quedan las detecciones con el mayor valor de confianza.

YOLOv3 utiliza una nueva arquitectura de red neuronal convolucional profunda llamada Darknet-53. Esta nueva red está inspirada en los trabajos previos de Redmon y Farhadi presentados en YOLOv2 y Darknet-19, pero incorpora los nuevos conceptos

de redes residuales (ResNets). Darknet-53 tiene 53 capas convolucionales y utiliza principalmente filtros de 3×3 y 1×1 así como conexiones de salto, conocidas en la literatura de aprendizaje profundo como *shortcut connections / skip connections* [42]. Darknet-53 emplea como función de activación para las capas convolucionales a la función *leaky* ReLU.

Cada capa convolucional en YOLOv3 hace uso de la técnica conocida como *batch normalization*. Esta técnica permite que la red neuronal se entrene más rápido, reduce la varianza en las entradas y en las activaciones de las capas ocultas, permite que cada capa de la red aprenda de manera independiente a las otras y reduce el efecto de *overfitting* o sobreajuste porque introduce un efecto de regularización. La normalización de las características extraídas se logra al ajustar y escalar las activaciones de la capa de entrada [18].

Desde sus primeras versiones YOLO utiliza el concepto de *anchor boxes*. Los *anchor boxes* son, en palabras simples, *bounding boxes* a priori que fueron calculados a partir del conjunto de datos COCO utilizando el método de clusterización *k-means*. YOLOv3 busca predecir el ancho y la altura del *bounding box* como *offset* de los centroides de los *clusters*. Las coordenadas del centro del *bounding box* se calculan utilizando la función sigmoide.

YOLOv3 entrega las coordenadas predichas para el *bounding box*: t_x, t_y, t_w, t_h. El primer par de valores corresponde a las coordenadas del centro del objeto de interés detectado y el último par corresponde al ancho y alto del *bounding box*. Para la tarea de predicción del *bounding box*, se utiliza como función de pérdida a la suma de errores cuadráticos. YOLOv3 también predice un puntaje de objeto (*objectness score*) para cada *bounding box* mediante regresión logística. Este puntaje de objeto es 1 para el *bounding box* que mejor se traslapa con el *ground truth* del objeto, mejor que cualquier otro *bounding box* previo. Para la predicción de la clase a la que pertenece el objeto de interés, YOLOv3 no utiliza a la función softmax. En cambio, emplea clasificadores logísticos independientes junto con la entropía cruzada binaria como función de pérdida.

Para la predicción de los *bounding boxes*, YOLOv3 realiza análisis de imágenes en tres escalas diferentes. Las características se extraen de estas escalas de forma similar a las redes de pirámides de características (*Feature Pyramid Networks*, FPN). Al extractor de características base Darknet-53 se le agregan varias capas convolucionales adicionales, la última de estas capas predice el *bounding box*, el puntaje de objeto y la predicción para la clase a la que pertenece el objeto.

Es común que se presente una situación en la que YOLOv3 pudiese detectar el mismo objeto varias veces, donde los *bounding boxes* solo difiriesen sutilmente en tamaño y ubicación central. Para solucionar este problema, YOLOv3 aplica supresión de no-máximos para eliminar todos los *bounding boxes* traslapados redundantes con un puntaje de confianza más bajo. La salida de YOLOv3 está codificada en un tensor 3d de dimensiones $N\times N\times[3\times(4+1+80)]$ donde N es el tamaño de la escala de análisis, 3 es el número de *bounding boxes* por cada escala, 4 es el número de coordenadas del

bounding box predicho, 1 es para el puntaje de objeto (si el objeto está presente en la imagen) y 80 es el número de clases suponiendo que YOLOv3 fue entrenado en el conjunto de datos COCO.

Un primer mapa de características se toma de un par de capas anteriores a la última y se sobremuestrea por un factor de 2. Un segundo mapa de características se toma de las primeras capas de la red y se fusiona con el primero mediante concatenación. De acuerdo con [42], este método permite obtener información semántica más significativa de las características sobremuestreadas e información de grano fino del mapa de características primigenio. Para procesar el mapa de características combinado y predecir un tensor del doble del tamaño se incorporan capas convolucionales adicionales en el diseño de la red. El mapa resultante de las capas de sobremuestreo concatenado con el de las capas primigenias ayuda a preservar las características de grano fino lo que permite detectar objetos pequeños.

Las predicciones para la tercera y última escala se enriquecen de todas las características extraídas previamente, así como de las características de grano fino extraídas del principio de la red. Realizar detecciones a diferentes escalas ayuda a resolver el problema de detectar objetos de diferentes tamaños. Al final, YOLOv3 se compone de 106 capas, conformadas por bloques residuales pues está construida sobre Darknet-53 (sin la etapa de clasificación), capas convolucionales, capas de sobremuestreo y capas de detección. En el apéndice de este trabajo se puede encontrar la figura A.1 que ilustra la arquitectura YOLOv3.

3.2. Segmentación: ¿Qué es segmentación?

La segmentación de una imagen es un paso preliminar esencial en la mayoría de los problemas de procesamiento digital de imágenes y visión computacional.

La segmentación se puede definir como el proceso de particionar una imagen digital en múltiples segmentos o conjuntos de pixeles. El objetivo de una segmentación puede plantearse de muchas maneras, algunas de ellas son [22]:

- Simplificar y/o cambiar la representación de una imagen en algo que sea más significativo y más fácil de analizar.
- Ubicar objetos de interés y su frontera, dada por líneas y curvas.

El resultado de aplicar una segmentación es un conjunto de segmentos que cubren a toda la imagen o bien un conjunto de contornos extraídos de la imagen tal que los píxeles pertenecientes a una de las regiones segmentadas son similares entre sí con respecto a alguna propiedad calculada como el color, la intensidad o la textura.

Algunas de las técnicas actuales de segmentación de imágenes incluyen la segmentación basada en regiones, la segmentación por detección de bordes, la segmentación basada en agrupamientos, la segmentación por redes neuronales convolucionales [50].

La diversas técnicas de segmentación de imágenes son fuertemente explotadas para diferentes aplicaciones como procesamiento de imágenes médicas, reconocimiento de objetos, detección de señalamientos de tránsito y peatones, reconocimiento facial, entre muchas otras.

En este trabajo se utilizaron dos técnicas de segmentación: segmentación por el método Grabcut y cambio de espacio de color y segmentación basada en detección de bordes.

3.2.1. Segmentación basada en Grabcut en el espacio de color HSV

En el artículo [46] se propone el uso del método de Grabcut en el espacio de color HSV para segmentar automáticamente lesiones de piel. Antes de presentar el *pipeline* que se propone en dicho artículo, se describirá brevemente en qué consiste de forma general cada uno de los conceptos y métodos empleados.

3.2.1.1. Modelo de color RGB

El modelo de color RGB (*Red, Green, Blue*) corresponde a la composición del color en términos de la intensidad de los colores primarios de la luz. Se trata de un modelo de color aditivo esto quiere decir que la luz roja, verde y azul se suman de varias maneras para reproducir una amplia gama de colores.

Un color en el modelo RGB se representa indicando la cantidad presente de rojo, verde y azul; el color se expresa como un triplete formado por las componentes RGB: (r, g, b). Cada componente puede tomar un valor de cero a un valor máximo definido. Si todas las componentes están en cero, el color resultante es negro mientras que si todas las componentes se encuentran en el valor máximo, el resultado es el color blanco representable más brillante.

Se puede representar de diferentes maneras el rango de valores que pueden tomar las componentes de un color. Por ejemplo, algunas de las más comunes son:

- **Aritmética.** Toma valores de 0 a 1 con valores decimales intermedios. Esta representación se utiliza en sistemas que utilizan representaciones de punto flotante.
- **Porcentaje.** El valor de cada componente de color se escribe como un porcentaje, de 0 % a 100 %.

- **Cuantización de 8-bits por canal.** En esta representación, los valores de las componentes se expresan como números enteros en el rango de 0 a 255. En otras palabras, es el rango de valores que puede representarse con 8 bits.

3.2.1.2. Modelo de color HSV

El modelo de color HSV (*Hue, Saturation, Value*) es un modelo que se expresa en términos de sus componentes de matiz, saturación, valor/brillo. El matiz se representa por una región circular mientras que para la saturación y el valor del color se emplea una región triangular separada. El eje horizontal de esta región triangular corresponde a la saturación mientras que el eje vertical corresponde al valor o brillo del color.

Para elegir un color, primero se selecciona su matiz de la región circular y posteriormente, se selecciona la saturación y el valor de la región triangular.

La región circular del matiz toma valores angulares en grados que van del 0 al 360 [°], cada valor angular corresponde a un color. Sin embargo, en ocasiones estos valores se normalizan por lo que se expresan como porcentajes del 0 al 100 %.

Una forma intuitiva de asociar los colores expresados en RGB con la componente de matiz de este modelo es de la siguiente manera. Se tienen los 360 [°] grados de la región circular del matiz, si esta región se divide en tres para cada color primario del modelo RGB entonces la región queda segmentada en tres grandes regiones, es decir, tenemos regiones de 120 [°] por color. De esta manera, tenemos que el rojo está en 0 [°], el verde en 120 [°] y el azul en 240 [°]. Para las mezclas de colores se emplean los grados intermedios de tal forma que el amarillo que está entre el rojo y el verde, puede hallarse en 60 [°]. Al sumar 60 [°] podemos recorrer la rueda de matiz y pasar por los colores primarios así como por los colores intermedios formados por estos.

Por otro lado, se dice que la saturación es la distancia al eje blanco-negro. La componente de saturación puede tomar valores que van de 0 a 100 %. Mientras más pequeño sea el valor de saturación de un color más decolorado estará.

Finalmente, la componente de valor o brillo representa la altura en el eje blanco-negro. La componente de valor/brillo puede tomar valores que van de 0 a 100 %. Un valor de 0 corresponde a negro y dependiendo de la saturación, un valor de 100 % puede ser blanco o algún color con cierto valor de saturación.

3.2.1.3. Clusterización por *k-means*

El método de clusterización o agrupamiento por *k-means* es un algoritmo de aprendizaje no supervisado, es decir, trabaja con datos que no se encuentran etiquetados por lo que trata de encontrar la estructura intrínseca existente en los datos de entrada. La

siguiente descripción del algoritmo está basada en la publicación de Firdaouss Doukkali y se puede encontrar en [12].

Motivemos la utilidad del algoritmo con el siguiente ejemplo. Se tiene un conjunto de datos clínico (numérico) de tumores cancerosos en 4 etapas diferentes, de la 1 a la 4, y se quiere estudiar a los tumores en sus respectivas etapas. Sin embargo, se tiene el problema de que no se puede identificar a los tumores que se encuentran en la misma etapa porque no está etiquetado el conjunto de datos de características de los tumores.

Este problema puede ser resuelto por el algoritmo de *k-means* porque trabaja sobre datos numéricos sin etiquetar y buscará agrupar rápida y automáticamente los datos en 4 clústeres. Para el ejemplo, se elige un valor de $k = 4$ porque se tienen 4 diferentes etapas de tumores. Sin embargo, si se desean agrupar los tumores en términos de su estructura, velocidad de crecimiento o tipo de crecimiento entonces seguramente k será un valor diferente de 4.

Puede llegar a ser un problema desconocer el número de clústeres que se desea obtener porque *k-means* necesita como entrada un número específico de k grupos para poder hacer su trabajo. Como primera reflexión podemos decir que, es necesario conocer el conjunto de datos sobre el que se trabaja para poder tener una base y definir un criterio sobre el cual se desea agruparlos; conocer los datos ayuda a tener una intuición del número de clústeres que se necesitan.

El algoritmo de *k-means* se compone de tres pasos:

1. **Inicialización**. Lo primero que hace *k-means* es elegir de manera aleatoria k muestras o ejemplos del conjunto de datos como centroides iniciales. Esto se debe porque en este paso se desconoce dónde está el centro de cada clúster. **Observación**: Un centroide es el centro de un clúster.

2. **Asignación a un clúster**. Todos los puntos de datos que se encuentran cerca (guardan similitud) de un centroide, formarán un clúster. Si se utiliza la distancia euclidiana entre los puntos de datos y cada centroide, se traza una línea recta entre dos centroides, y a continuación se traza una bisectriz perpendicular (frontera) que divide la línea recta anterior para formar dos clústeres.

3. **Mover los centroides**. En este paso ya se tienen nuevos clústeres que necesitan de nuevos centroides. El nuevo valor de un centroide será la media de todos los ejemplos en un clúster. Los pasos 2 y 3 se repiten hasta que los centroides dejen de moverse, en otras palabras, hasta que el algoritmo de *k-means* haya convergido.

k-means es un método rápido y eficiente pues la complejidad de una iteración es $k \times n \times d$ donde k es el número de clústeres, n el número de ejemplos y d el tiempo de cálculo de la distancia euclidiana entre 2 puntos.

Existen métodos cuando se desconoce por completo el mejor número de clústeres para abordar un problema [12]. Si los centroides iniciales elegidos de manera aleatoria

Algoritmo 1 Algoritmo Básico de *k-means*

```
Elige aleatoriamente k ejemplos como centroides iniciales
while true do
    Crea k clústeres al asignar cada ejemplo al centroide más cercano
    Calcula k nuevos centroides al promediar los ejemplos de cada clúster
    if los centroides no cambian then
        break
```

resultan no ser los mejores, al algoritmo le llevará más tiempo converger o bien puede atorarse en un óptimo local lo que resultaría en una mala clusterización.

3.2.1.4. Ecualización adaptable del histograma

Consideremos una imagen donde los valores de los píxeles se ubican únicamente en un rango muy específico de valores. En el caso de una imagen brillante, por ejemplo, todos sus píxeles tendrán valores altos por lo que en un histograma todos los pixeles estarán distribuidos principalmente a la derecha del punto medio del mismo. Sin embargo, en una buena imagen se tendrán los píxeles distribuidos a lo largo de todo el rango de valores de la imagen. En palabras simples, la técnica de ecualización del histograma busca estirar el histograma a cualquiera de los extremos según sea necesario. Hacer uso de esta técnica ayuda a mejorar el contraste de la imagen.

Realizar la ecualización del histograma de una imagen trae buenos resultados cuando el histograma de la imagen se encuentra confinado a una región en particular. Sin embargo, la técnica no funciona tan bien en casos donde hay grandes variaciones de intensidad, aquellas donde el histograma abarca una región grande y donde están presentes tanto píxeles brillantes como oscuros.

El primer tipo de ecualización que se describió anteriormente se conoce como ecualización global del histograma pues considera el contraste global de la imagen. El artículo de [46] propone el uso de CLAHE (*Contrast Limited Adaptive Histogram Equalization*) para realizar la ecualización del histograma de las imágenes del *dataset* de lesiones en piel.

Esta técnica consiste en lo siguiente [38]. La imagen se divide en pequeños bloques o mosaicos (*tiles*) por defecto en OpenCV el tamaño del mosaico es de 8×8. A continuación se ecualiza el histograma de cada uno de estos bloques; con esto se logra que el histograma quede confinado a una región pequeña. Si se tiene la presencia de ruido, este se amplificará. Para subsanar esto se tendrá que limitar el contraste. Si cualquier *bin* del histograma se encuentra por encima del límite de contraste especificado, esos píxeles son recortados y distribuidos uniformemente a otros *bins* antes de aplicar ecualización

del histograma. Finalmente, después de realizar el proceso de ecualización, se aplica el método de interpolación bilineal para eliminar los posibles defectos en los bordes de los mosaicos.

3.2.1.5. Método de Grabcut

El algoritmo de GrabCut fue propuesto en 2004 por Carsten Rother, Vladimir Kolmogorov y Andrew Blake de *Microsoft Research* en el Reino Unido. Su trabajo se puede encontrar en el artículo titulado, *GrabCut: Interactive Foreground Extraction Using Iterated Graph Cuts*. El método nació a partir de la necesidad de un algoritmo para la extracción de objetos de interés en primer plano con una interacción mínima de parte del usuario.

La idea detrás del algoritmo es simple.

- El usuario dibuja un rectángulo alrededor del objeto de interés en primer plano, el objeto de interés debe estar completamente enmarcado dentro del rectángulo.
- El algoritmo segmenta iterativamente la región de interés hasta obtener el mejor resultado.

En algunos casos, la segmentación puede no ser perfecta. Puede haber casos donde por ejemplo, se pudo tratar alguna región de interés como fondo y viceversa. En ese caso, el usuario debe realizar ajustes finos. Para ello, basta con marcar aquellas regiones de la imagen donde se hayan presentado errores. Las marcas indican si una región que haya sido tratada como fondo debe en realidad ser tratada como parte del objeto en primer plano. En la siguiente iteración, el error se corrige por lo que se obtienen mejores resultados en la segmentación.

3.2.1.6. Método de Grabcut en el espacio de color HSV

Una vez presentados los conceptos y algoritmos centrales que se utilizan en el proceso de segmentar las lesiones de piel podemos entonces introducir el *pipeline* propuesto por [46] para realizar la segmentación.

Como puede verse en la figura 3.1, el flujo de trabajo consta de las siguientes etapas:

1. Se ejecuta el algoritmo de *k-means* con 8 centroides iniciales con la finalidad de cuantizar los colores en la imagen.
2. Una vez concluido el proceso de clusterización de colores, se hace la transformación de la imagen del espacio de color BGR al espacio HSV. Realizar este paso

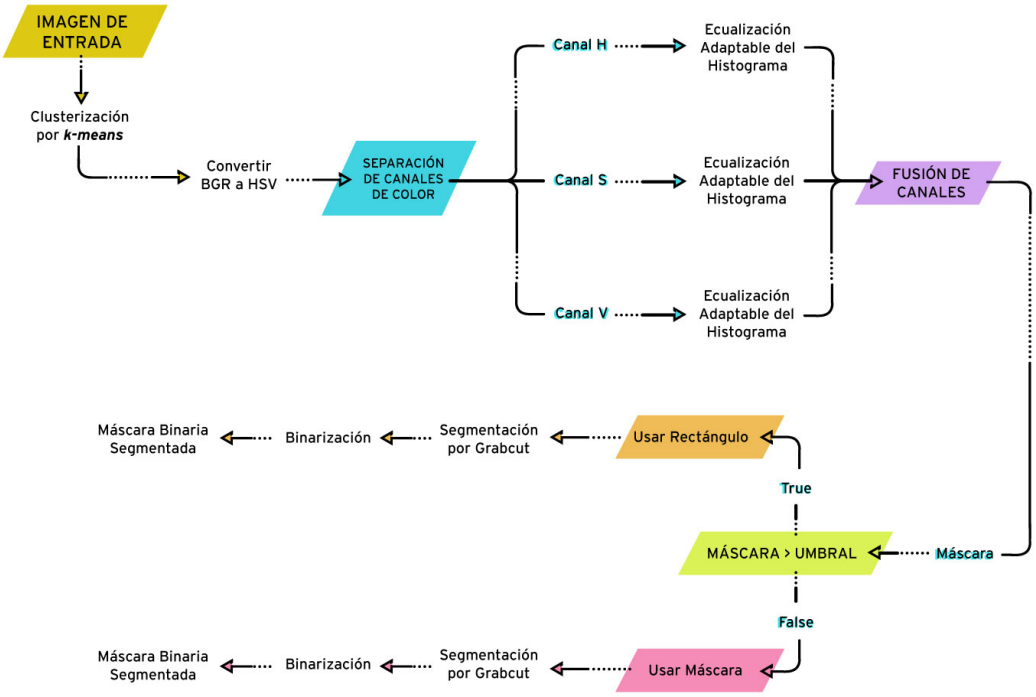


Figura 3.1: Segmentación de lesiones en piel en el espacio de color HSV por Grabcut [46].

tiene una ventaja muy singular, pues la gran mayoría de las lesiones de piel independientemente de su naturaleza benigna o maligna quedan delimitadas por un distintivo color verde, en ocasiones de diferentes tonalidades, incluso puede llegar a tener componentes de amarillo a naranja.

3. El siguiente paso es separar cada uno de los canales que forman esta nueva representación de la imagen en sus componentes de tono (H), saturación (S) y brillo (V).

4. A continuación se aplica a cada canal, por separado, el método de ecualización adaptable del histograma con contraste limitado con la finalidad de mejorar el contraste en cada canal de la imagen.

5. En este paso los canales recién procesados se vuelven a juntar.

6. En este paso se hace una umbralización del color verde con la finalidad de generar una máscara. En la región donde la máscara tiene un valor de 1 se le considera como la región de interés pues es aquella donde está la lesión. Aquellas regiones donde la máscara tenga un valor de 0 son consideradas como parte del fondo. Con estas ideas, se puede aplicar directamente el método de Grabcut sobre la máscara. El resultado se binariza y se obtiene finalmente una máscara binarizada, la cual se puede aplicar a la imagen original para obtener la segmentación final.

7. Si el tamaño de la mascara excede el 70 % de la imagen, la máscara se desecha. Entonces, se procede a generar un *bounding box* calculado a partir de las dimensiones de la imagen. Este *bounding box* servirá para que el método de Grabcut pueda segmentar la región de interés, siendo esta aquella que se encuentra dentro de él. Por su parte, se considera como fondo todo aquello que se encuentra fuera del *bounding box*.

8. Entonces, teniendo el *bounding box*, podemos aplicar el método de Grabcut. El resultado se binariza y obtenemos finalmente la máscara binarizada que podemos aplicar a la imagen para segmentarla.

3.2.2. Segmentación basada en detección de bordes

El algoritmo de detección de bordes de Canny es uno de los algoritmos más populares en visión computacional. Fue desarrollado por John F. Canny en 1986 y puede encontrarse en su famoso trabajo, *A Computational Approach to Edge Detection* [5]. A continuación se explica cada uno de los pasos que constituyen al también llamado Filtro de Canny.

1. **Reducción de Ruido.** La detección de bordes es sensible al ruido presente en una imagen por ello, el primer paso es eliminar el ruido en la imagen con un filtro gaussiano de 5×5.

2. **Cálculo del gradiente de la intensidad de la imagen.** A la imagen suavizada se le aplica un filtro Sobel en dirección horizontal y vertical con esto podemos conocer la primera derivada tanto en dirección horizontal (G_x) como en vertical (G_y). Con las dos imágenes resultantes, podemos encontrar el gradiente de borde y la dirección para cada píxel como sigue:

$$\nabla G = \sqrt{(G_x)^2 + (G_y)^2} \tag{3.1}$$

$$\theta = tan^{-1}\left(\frac{G_y}{G_x}\right) \tag{3.2}$$

   La dirección del gradiente es siempre perpendicular a los bordes. Esta, se redondea a uno de cuatro posibles ángulos que representan las direcciones vertical, horizontal y dos diagonales.

3. **Supresión de no-máximos.** Después de obtener la magnitud y la dirección del gradiente, se realiza un análisis de la imagen completa para eliminar aquellos píxeles no deseados que pueden no ser parte de un borde. Para cada píxel se verifica si es un máximo local en su vecindario en la dirección del gradiente.

   En resumen, el resultado que se obtiene es una imagen binaria con bordes delgados.

4. **Umbralización con histéresis.** En este bloque se analiza cuáles de todos los bordes detectados son realmente bordes y cuáles no. Para esto se requieren dos valores de umbral, un valor de umbral mínimo y un umbral máximo. Cualquier candidato a borde con un gradiente de intensidad mayor que el valor de umbral máximo se considera como borde mientras que aquellos por debajo del umbral mínimo no se consideran bordes, por lo que son descartados. Aquellos que se encuentren entre estos dos umbrales se decide si son bordes o no según su conectividad. Si están conectados a píxeles que pertenecen a un borde, se les considera parte de este. En caso contrario, también se descartan.

   En resumen:

   - Si el valor del gradiente de un píxel es más alto que el umbral superior, el píxel se acepta como un borde.
   - Si el valor del gradiente de un píxel está por debajo del umbral inferior, entonces se rechaza.
   - Si el valor del gradiente de un píxel está entre los dos umbrales, solo se acepta si está conectado a un píxel que está por encima del umbral superior.

   Canny recomienda una relación de 2:1 y 3:1 para el valor de los umbrales superior e inferior.

Este algoritmo de detección de bordes se utilizó para realizar la segmentación de las lesiones en piel. Una de los inconvenientes que presenta segmentar las imágenes por el método anterior es que, en ocasiones, para algunas imágenes el resultado de la segmentación es inaceptable pues se observa que el proceso de segmentación considera parte de la lesión de interés como piel por lo que el resultado es inaceptable. Dado que el método presentado anteriormente es automático, el mismo flujo de trabajo se aplica a todas las imágenes. Entonces para aquellos casos en donde la imagen no fue segmentada correctamente se decidió abordar el problema por este enfoque basado en detección de bordes.

El flujo de trabajo propuesto es el siguiente y corresponde con una de las estrategias más versátiles y populares en visión computacional:

1. Redimensionar la imagen a 608×608.
2. Transformar la imagen a escala de grises.
3. Aplicar un filtro Gaussiano de 5×5. Este paso tiene la finalidad de suavizar la imagen y remover algo de ruido en caso de estar presente.
4. Se aplica el filtro de Canny partiendo de umbrales separados por un amplio rango como los valores iniciales. Como este algoritmo se aplica sólo a unas cuantas

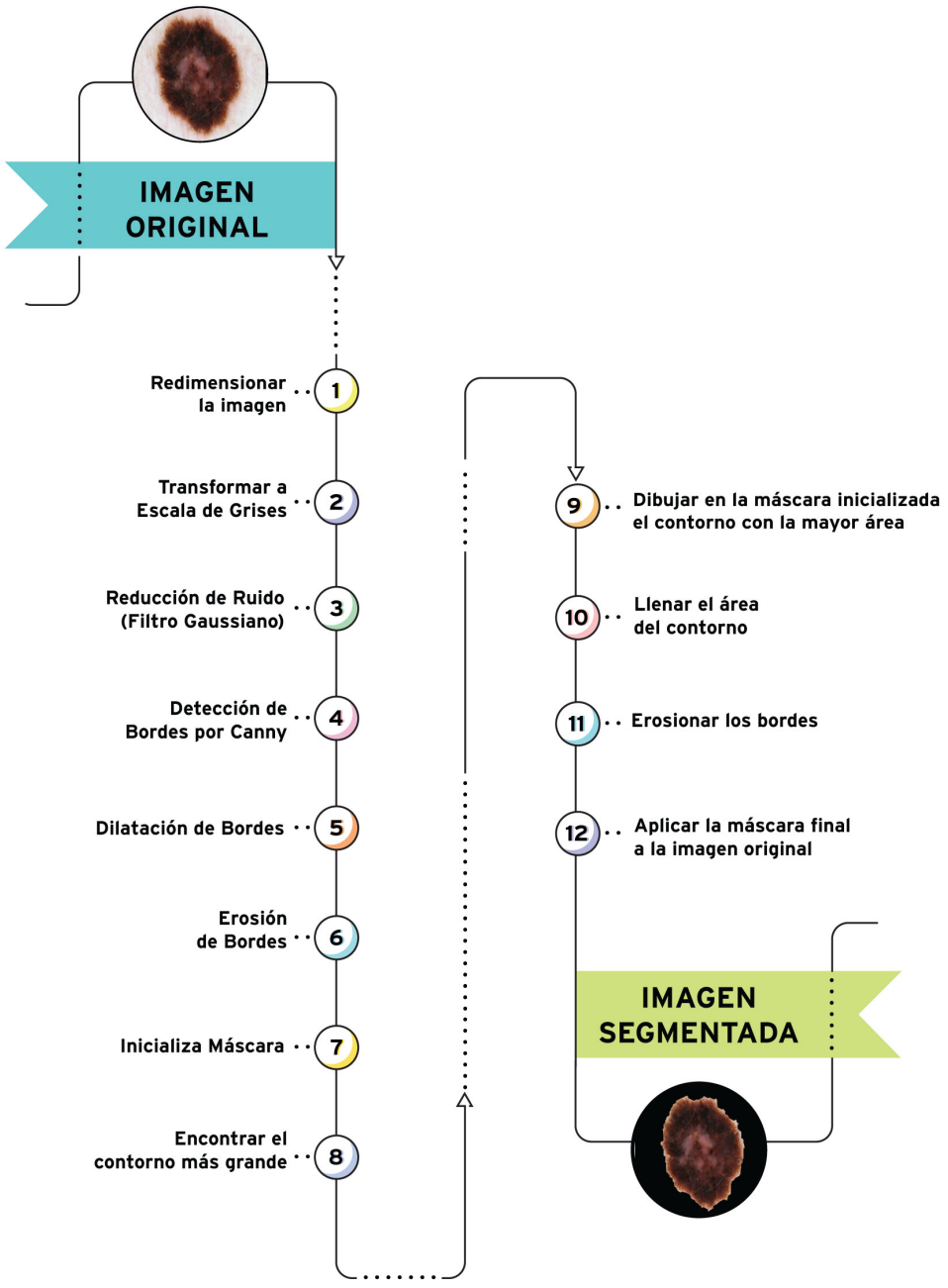


Figura 3.2: Segmentación de lesiones en piel por detección de bordes.

imágenes es factible seleccionar el valor adecuado de los umbrales de forma manual a través de un *slider* hasta obtener el mejor resultado. En caso de que se necesite aplicar a un mayor número de imágenes, existen métodos estadísticos para calcular el mejor valor para los umbrales dado un *batch* de imágenes.

5. Aplicar un operador morfológico de dilatación con una elipse de 5×5 por 10 iteraciones a los bordes detectados.
6. Aplicar un operador morfológico de erosión por 5 iteraciones con el mismo elemento estructural.
7. Crear una máscara inicial (negra) de las mismas dimensiones y canales que la imagen del bloque anterior.
8. Encontrar los contornos de la imagen resultante de la aplicación de los operadores morfológicos.
9. Dibujar en la máscara inicializada aquel contorno con la mayor área.
10. Llenar el área interior del contorno encontrado.
11. Aplicar una última operación de erosión por 5 iteraciones con el mismo elemento estructural.
12. Aplicar la máscara final a la imagen original para lograr la segmentación.

Capítulo 4

Experimentos y resultados

En este capítulo se presentan los principales experimentos desarrollados así como el análisis de resultados de cada uno de ellos.

4.1. Base de datos

El archivo *International Skin Imaging Collaboration* (archivo ISIC) es una base de datos de imágenes dermatológicas de alta calidad, las imágenes se obtuvieron de centros clínicos líderes a nivel internacional y se adquirieron de una variedad de dispositivos dentro de cada centro [25]. Las imágenes que se pueden obtener directamente del archivo ISIC proporcionan información clínica en extenso, como la edad y el sexo del paciente, la localización de la lesión, el diagnóstico y el método de confirmación de dicho diagnóstico. Este tipo de información permite, por ejemplo, organizar fácilmente las imágenes en dos clases, benignas y malignas. Al momento de preparar este trabajo, el conjunto de datos consistía en 19,330 lesiones benignas y 2,286 lesiones malignas.

Para la primer prueba se abordo el problema de detectar únicamente las lesiones como benignas o malignas. Por medio de un *script* y aprovechando que cada imagen tiene asociado un ID único, se descargaron del sitio ISIC aquellas imágenes con dichas etiquetas asignadas. El conjunto de datos descargados está fuertemente desbalanceado pues consta de 19,330 lesiones benignas, 2,286 lesiones malignas y posee un tamaño de aproximadamente 49.1 [GB]. En la sección correspondiente al primer experimento se detalla como se abordó esta dificultad.

En particular, con excepción de la primer prueba, los experimentos fueron realizados con el conjunto de datos ISIC 2017, las imágenes de este conjunto de datos son un subconjunto seleccionado de imágenes del archivo ISIC, destinado a ser utilizado en su desafío anual. El conjunto de datos consta de imágenes de tres clases: 521 de melanoma, 386 de queratosis seborreica y 1,843 de nevus.

El conjunto de datos ISIC 2017 está particionado de origen en datos para entrenamiento, validación y prueba. Los modelos de aprendizaje profundo logran generalizar mejor una tarea cuando se les entrena con la mayor cantidad de datos (imágenes) posible [18]. Con base en esta idea, decidimos unir los conjuntos de datos en uno y dividirlos en 90 % para entrenamiento y 10 % para validación.

4.2. Requerimientos de la arquitectura

Para entrenar YOLOv3 con el conjunto de datos ISIC para la identificación y localización de tres diferentes tipos de lesiones en piel, el primer paso es etiquetar todas las imágenes de acuerdo con el formato requerido por YOLOv3. El proceso es sencillo, cada imagen en el conjunto de datos debe emparejarse con un archivo *.txt*. Este archivo debe crearse con el mismo nombre de la imagen correspondiente y con la siguiente información contenida dentro de este: *clase* c_x c_y w h

La primera etiqueta corresponde a la clase de objeto donde las clases se enumeran a partir de 0 para la primer clase, c_x y c_y denotan el centro del objeto de interés a detectar y el par (w, h) denota las dimensiones del *bounding box*. Las coordenadas y dimensiones que se indiquen deben estar normalizadas, es decir, divididas por el ancho y alto de la imagen.

La mayoría de las lesiones cutáneas se encuentran en el centro de la imagen, por lo que aprovechando esta característica se le asignó a c_x y c_y un valor fijo de 0.5. Para los pocos casos en que la lesión está considerablemente lejos del centro de la imagen, el archivo correspondiente se modificó manualmente, este proceso de edición de los archivos puede llevar bastante tiempo.

En los primeros experimentos se notó que si el *bounding box* inicial no está bien dibujado, YOLOv3 tiende a enfocarse en el fondo de la imagen, lo que resulta en predicciones incorrectas. Este problema se abordó con la siguiente estrategia. Se definió un pequeño *bounding box* inicial con un ancho y alto de 0.15, a continuación se aumentó su tamaño sumando 0.15 a esas dimensiones iniciales. Este proceso de adición se realizó dos veces en total. Al final se obtienen tres *bounding boxes* con una diferencia de tamaño de 0.15 para una sola lesión cutánea. La idea es enfocar la atención de la red solo en las regiones donde está la lesión, evitando la mayor cantidad posible de piel circundante (fondo). En otras palabras, se buscó dar a la red solo regiones correspondientes a lesiones como conocimiento previo.

Previamente a reentrenar la red neuronal, se hace uso del proceso conocido como transferencia de conocimiento a través de los pesos obtenidos de entrenar YOLOv3 sobre el conjunto de datos de imágenes ImageNet. Esto permite que durante el proceso de entrenamiento se realice un ajuste fino (*fine tuning*) en las todas las capas convolucionales, lo que hace que el detector se pueda ajustar a realizar una nueva tarea completamente diferente, identificar y localizar diferentes clases de lesiones en piel.

4.3. Detección de lesiones como malignas o benignas

Para el primer experimento se decidió abordar el problema de detectar las lesiones como benignas o malignas. Como ya se comentó, el conjunto de datos que se utilizó

está fuertemente desbalanceado pues un 89.4 % de las imágenes son benignas y tan sólo un 10.6 % son malignas.

Para mitigar este problema se aplicó un submuestreo (*under-sampling*) a la clase sobre-representada por lo que se tomaron las siguientes acciones:

- Borrar la gran mayoría de las instancias del conjunto de datos SONIC (*Study of Nevi in Children*). El archivo ISIC es en realidad una colección de diferentes conjuntos de datos dermatológicos, SONIC es uno de ellos. Este conjunto se compone de imágenes de nevus en infantes y la gran mayoría de estas posee pequeños parches de colores que pudiesen afectar el entrenamiento de la red neuronal. Los dos enfoques posibles que emplean otros trabajos son borrar las instancias o bien, recortar el área de interés. Por practicidad, se preservan sólo aquellas imágenes donde el parche no ocluya la lesión y donde el *bounding box* no se vea afectado.
- Borrar aquellas imágenes donde la lesión no se logre apreciar completamente. Ya sea por oclusiones por vello o por un tamaño o color que hacen difícil distinguirla de la piel que les rodea.

Al final de este proceso, el conjunto de datos final constó de: 2,286 lesiones benignas y 2,286 lesiones malignas. Este conjunto de datos final se particionó en 90 % para entrenamiento y 10 % para validación.

Como un paso más de preprocesamiento, se realiza el aumento de datos sobre el conjunto de entrenamiento. Añadir más datos para entrenamiento a los modelos de aprendizaje profundo tiende a aumentar su desempeño. En el caso particular de las imágenes dermatológicas, se puede aprovechar su naturaleza para aplicar diferentes transformaciones a cada imagen. Para este tipo de imágenes, el desempeño de la red neuronal no se ve afectado si la imagen se ve borrosa, fue volteada horizontal o verticalmente o incluso si fue rotada una cierta cantidad de grados.

Las técnicas seleccionadas para el aumento de datos fueron giros horizontales, rotaciones un valor aleatorio de entre -90 y 90 grados así como transformaciones en los colores de las imágenes originales a través de los valores de saturación, exposición y tono. En particular, los valores de tales parámetros se establecieron en 1.5, 1.5 y 0.1 respectivamente. Las transformaciones anteriores se aplican a cada imagen del conjunto de entrenamiento, generando así una nueva muestra para la red.

A un subconjunto de las imágenes de entrenamiento que se usa en una iteración le llamamos tamaño del *batch*. El tamaño del *batch* para las imágenes de entrenamiento se configuró a 24 mientras que el hiperparámetro de subdivisiones a 8. Configurar el tamaño del *batch* a 24 significa que se utilizan 24 imágenes hasta completar una iteración y actualizar los parámetros de la red neuronal. Aunque es posible usar un tamaño de *batch* más grande para entrenar a la red, la memoria de la GPU es una limitante para ese propósito. El parámetro llamado subdivisiones permite procesar una fracción del tamaño del *batch* a la vez en la GPU; la GPU terminará procesando el tamaño del *batch*

/ subdivisiones de imágenes durante la etapa de entrenamiento. Sin embargo, el *batch* completo o la iteración se completan una vez que se hayan procesado las 24 imágenes.

Para este experimento, YOLOv3 se entrenó por 364 épocas y durante más de 200,000 iteraciones usando descenso de gradiente con momentum con los siguientes hiperparámetros: *batch* = 24, subdivisiones = 8, momento = 0.9, disminución de la tasa de aprendizaje = 0.0005 y tasa de aprendizaje = 0.001. Todas las imágenes durante esta etapa se redimensionaron a 608 × 608.

Los diferentes modelos obtenidos de la etapa de entrenamiento se evaluaron sobre el conjunto de datos de validación. Las gráficas muestran el desempeño de cada uno. Puede verse que los mejores resultados se obtuvieron con los pesos correspondientes a la iteración 80,000 con los que se logra una precisión y un *recall* de 74 % y 86 %, respectivamente.

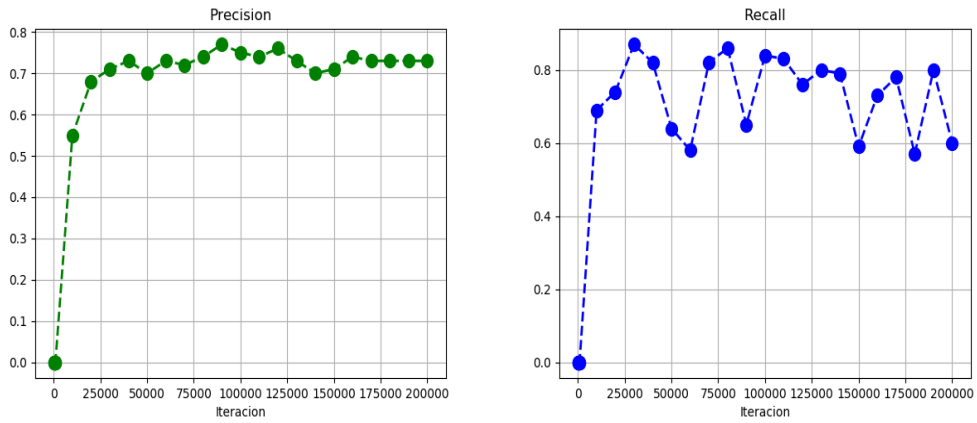


Figura 4.1: Desempeño sobre el conjunto de validación.

En la figura 4.2 se muestran algunos resultados visuales del conjunto de prueba, ninguna de estas imágenes fue vista previamente por la red neuronal. Todas las imágenes del conjunto de prueba fueron verificadas por un dermatólogo.

4.4. Detección de tres diferentes lesiones en piel

4.4.1. Entrenamiento sin aumento de datos

Para este experimento se abordó el problema de detectar los tres diferentes tipos de lesiones presentes en el conjunto de datos ISIC 2017: melanoma, nevus y queratosis seborreica. El conjunto de datos está distribuido de la siguiente forma:

Figura 4.2: Predicciones para imágenes del conjunto de prueba. La imagen de la izquierda corresponde a un nevus (lesión benigna), la imagen de la derecha corresponde a un melanoma (lesión maligna). El *bounding box* muestra la etiqueta de la clase predicha (benigna = 95 %, maligna = 99 %).

	Número de imágenes en cada partición		
Clase	Entrenamiento	Validación	Prueba
Melanoma	374	30	117
Nevus	1,372	78	393
Queratosis	254	42	90
Tamaño	5.38 [GB]	829 [MB]	5.16 [GB]

Puede verse que se trata de un conjunto de datos desbalanceado. Por lo que se aplicaron técnicas de submuestreo (*under-sampling*) y sobremuestreo (*over-sampling*) a cada partición original, la distribución final quedó de la siguiente manera:

	Número de imágenes en cada partición		
Clase	Entrenamiento	Validación	Prueba
Melanoma	374	42	117
Nevus	374	42	117
Queratosis	374	42	117

Este conjunto de datos final se unió en uno solo y se particionó en 90 % para entrenamiento, 10 % para validación. A diferencia del primer experimento, no se hace aumento de datos de ningún tipo, pues se desea ver el impacto de esta técnica para este tipo de imágenes.

La arquitectura se entrenó por 606 épocas y durante 120,000 iteraciones usando descenso de gradiente con momentum con los siguientes hiperparámetros: momento = 0.9, disminución de la tasa de aprendizaje = 0.0005 y tasa de aprendizaje = 0.001. Todas las imágenes durante esta etapa se redimensionaron a 608 × 608.

El desempeño de los modelos se evaluó sobre el conjunto de datos de validación. Puede verse en las gráficas que el mejor resultado corresponde a los pesos de la iteración 70,000 donde se obtuvo una precisión del 67 % y un *recall* del 77 %.

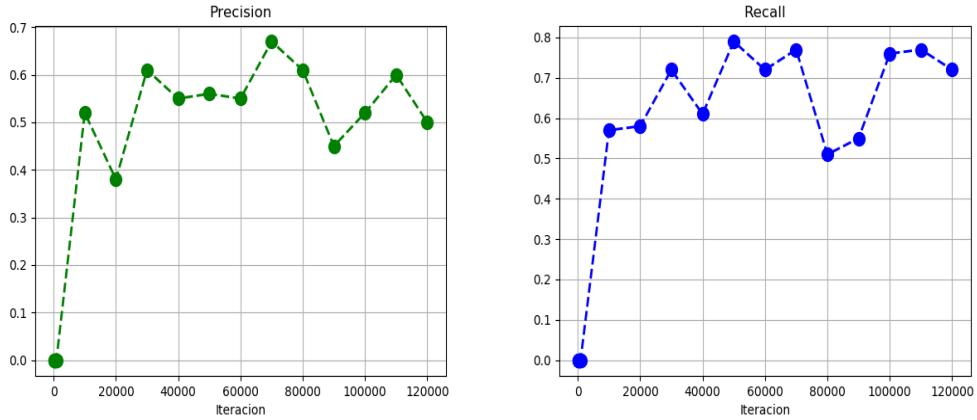


Figura 4.3: Desempeño sobre el conjunto de validación.

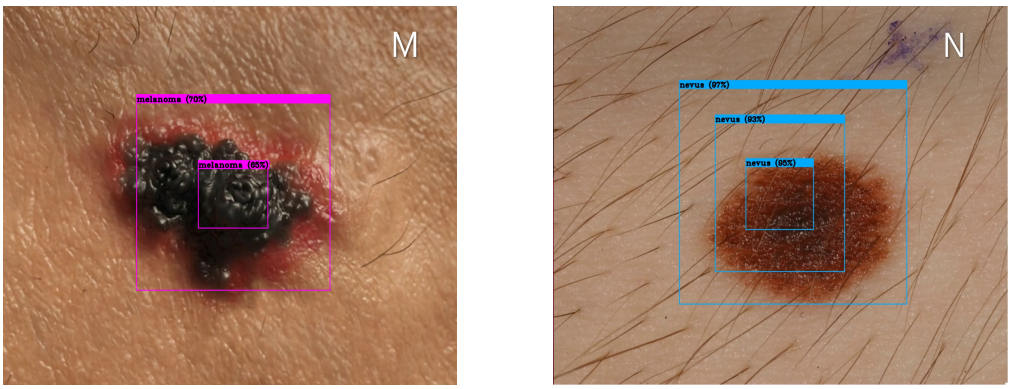


Figura 4.4: Predicciones para imágenes del conjunto de prueba con la probabilidad de la clase predicha. Etiqueta real (letra blanca) melanoma (izquierda), detectada como melanoma (70 %). Etiqueta real (letra blanca) nevus (derecha), detectada como nevus (97 %).

4.4.2. Entrenamiento con aumento de datos

Para este experimento se abordó el mismo problema de detectar los tres diferentes tipos de lesiones presentes en el conjunto de datos ISIC 2017: melanoma, nevus y queratosis seborreica. Esta prueba se realizó con el mismo conjunto de datos balanceado del segundo experimento.

Este conjunto de datos final se unió en un solo y se particionó en 90 % para entrenamiento, 10 % para validación. Al igual que en el primer experimento se realiza aumento de datos. Las transformacioens que se aplica a las imágenes son las mismas, giros horizontales, rotaciones un valor aleatorio de entre -90 y 90 grados así como sutiles cambios en los colores de las imágenes originales a través de los valores de saturación, exposición y tono.

YOLOv3 se entrenó durante 606 épocas y durante más de 200,000 iteraciones usando descenso de gradiente con momentum con los siguientes hiperparámetros: momento = 0.9, disminución de la tasa de aprendizaje = 0.0005 y tasa de aprendizaje = 0.001. Todas las imágenes durante esta etapa se redimensionaron a 608 × 608.

Al final de la etapa de entrenamiento, se evaluó el desempeño de los modelos generados sobre el conjunto de datos de validación. Se observó que los mejores resultados se obtuvieron de los pesos correspondientes a la iteración 150,000, alcanzando una precisión del 70 % y un *recall* del 77 %. Las siguientes figuras muestran la evolución de ambas métricas en diferentes iteraciones, es decir, para los diferentes valores de pesos obtenidos a través proceso de optimización.

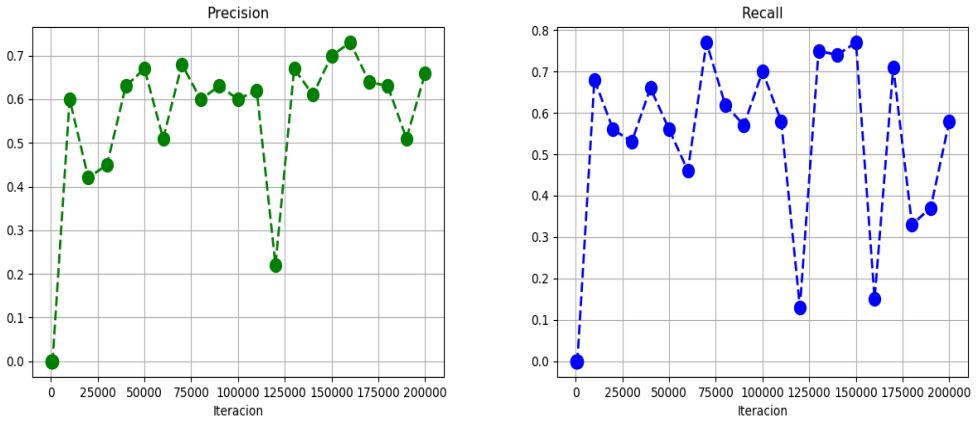


Figura 4.5: Desempeño sobre el conjunto de validación.

Al inspeccionar las métricas a través del proceso de validación y en base a las predicciones visuales obtenidas, podemos decir que la red neuronal profunda aprendió a distinguir entre las tres diferentes clases de lesiones a las que fue expuesta. Intentar

clasificar correctamente este tipo de imágenes dermatológicas es una tarea difícil tanto para los dermatólogos como para los modelos de aprendizaje profundo. En [30], se reporta que la inspección visual por dermatólogos expertos sin algún tipo de herramienta de ayuda se asocia con una exactitud diagnóstica de alrededor del 60 %.

Estos resultados iniciales apuntan a considerar la hipótesis de que el rendimiento de YOLOv3, al tratarse de un modelo de aprendizaje profundo, se puede mejorar con más imágenes médicas, en combinación con técnicas de procesamiento de imágenes como la segmentación para el aumento de datos.

Se creó un video de dos minutos compuesto por imágenes del conjunto de datos de prueba para ser procesado en tiempo real por YOLOv3. Cada imagen del video fue verificada por un dermatólogo para asegurar que la etiqueta asignada correspondiese con la lesión de piel en la imagen. YOLOv3 puede mostrar los frames por segundo (FPS), la clase predicha junto con los *bounding boxes* en la imagen. El vídeo se procesó a aproximadamente 28 FPS en una GPU NVIDIA GeForce GTX 1080.

Las siguientes imágenes son algunos de los cuadros del video de prueba. Es importante señalar que la red neuronal profunda nunca vio antes el conjunto de imágenes de prueba. Los ejemplos muestran el resultado una vez que YOLOv3 ha procesado una imagen con los pesos finales seleccionados de la etapa de validación. Todos los ejemplos son imágenes disponibles públicamente de diferentes sitios web de atención médica como [3], cuyo diagnóstico fue confirmado con la ayuda de un dermatólogo.

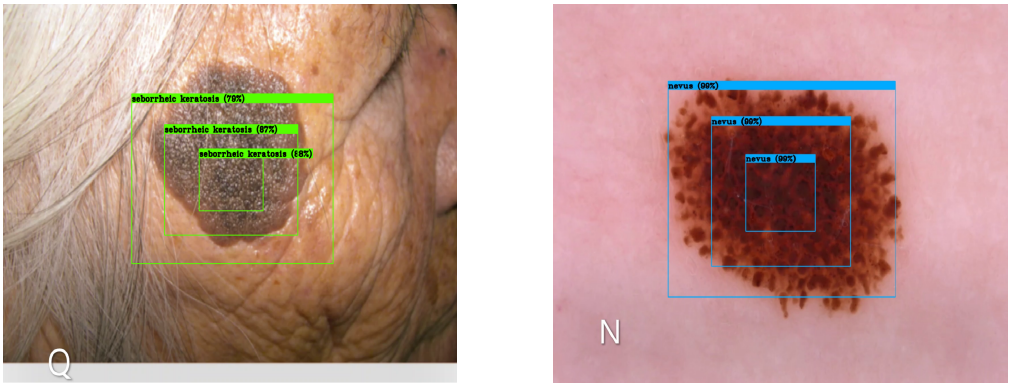


Figura 4.6: Imágenes del conjunto de prueba con la probabilidad de la clase predicha. Etiqueta real (letra blanca) queratosis (izquierda), detectada como queratosis (88 %). Etiqueta real (letra blanca) nevus (derecha), detectada como nevus (99 %).

4.4.3. Entrenamiento con imágenes segmentadas y aumento de datos

Al igual que en el caso anterior el problema se reduce a detectar los tres tipos de lesiones presentes en el conjunto de datos ISIC 2017. En este experimento se decide incluir como parte del aumento de datos las imágenes resultantes del proceso de segmentación. Como se comentó en el capitulo anterior, las imágenes fueron segmentadas mediante dos métodos diferentes, segmentación en el espacio HSV con grabcut y detección de bordes. La hipótesis es que aumentar el número de datos puede ayudar a mejorar el desempeño en el proceso de detección.

Todas las imágenes del conjunto de datos balanceado fueron segmentadas y añadidas como aumento de datos. Este nuevo conjunto está distribuido de la siguiente manera:

	Número de imágenes en cada partición		
Clase	Entrenamiento	Validación	Prueba
Melanoma	748	83	233
Nevus	748	83	233
Queratosis	748	83	233

Posteriormente, al igual que en los casos anteriores, se juntaron todas las imágenes en un mismo conjunto y este fue particionado en 90 % para entrenamiento y 10 % para validación.

El nuevo conjunto de datos formado por las imágenes originales y las imágenes segmentadas es también sometido a un segundo preprocesamiento como parte del aumento de datos. Este preprocesamiento aplica las mismas transformaciones antes descritas dadas por giros horizontales, rotaciones aleatorias entre -90 a 90 grados y cambios en los colores.

Como se comentó en el capítulo anterior, el método principal para segmentar las imágenes fue Grabcut en el espacio de color HSV, sin embargo, no todos los resultados entregados por este método son satisfactorios por lo que se puede experimentar con algún otro método que pudiese entregar buenos resultados. La figura 4.7 muestra un ejemplo donde se ilustra la situación anterior.

Para aquellas imágenes donde la segmentación no es aceptable ya sea porque el método eliminó por completo la lesión o bien la cantidad de piel es excesiva, se utilizó el método de segmentación por detección de bordes a través del filtro de Canny. Las figuras 4.8 y 4.9 muestran algunas de las etapas del proceso de segmentación por ambos métodos.

Puede verse que la segmentación de las lesiones no es perfecta, sin embargo esto no debería mermar el desempeño del modelo sino al contrario, esas imperfecciones pudiesen ayudar a que el modelo logre una mejor distinción entre las clases pues se conserva un

Figura 4.7: Imagen original de un melanoma (izquierda). Imagen segmentada incorrectamente por Grabcut (centro). Imagen segmentada por umbralización (derecha).

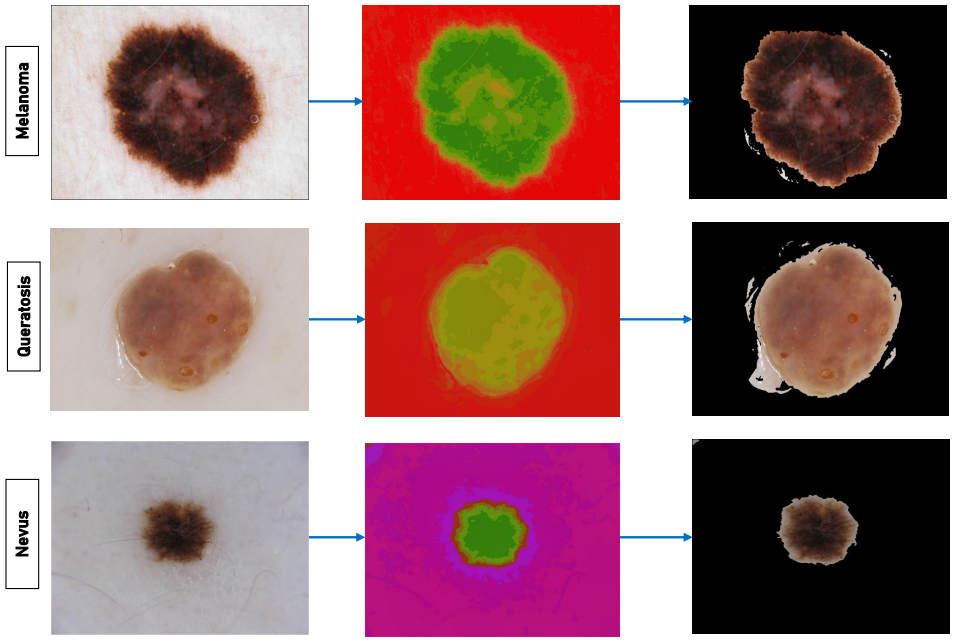


Figura 4.8: Lesiones de piel segmentadas por Grabcut en el espacio de color HSV.

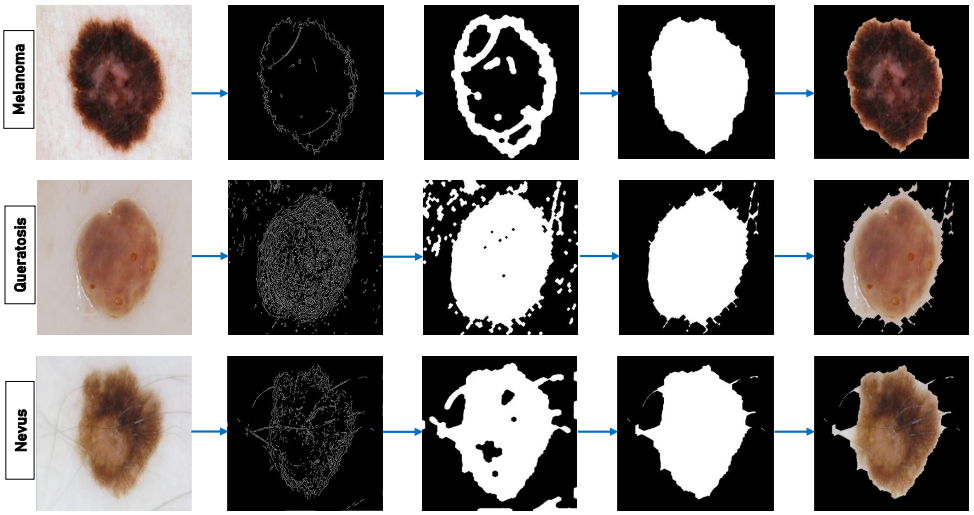


Figura 4.9: Lesiones de piel segmentadas a través de detección de bordes.

poco de la piel circundante. Se sabe que los dermatólogos durante su inspección visual, consideran los colores que exhibe la lesión así como aquellos que presenta la piel que le rodea al momento de evaluar un caso y ofrecer finalmente un diagnóstico.

El modelo se entrenó utilizando descenso del gradiente con momentum y con la misma selección de hiperparámetros que en el experimento anterior. El entrenamiento se llevó a cabo por 522 épocas y 130,000 iteraciones. Las gráficas que se muestran a continuación muestran el desempeño de los modelos con el conjunto de validación.

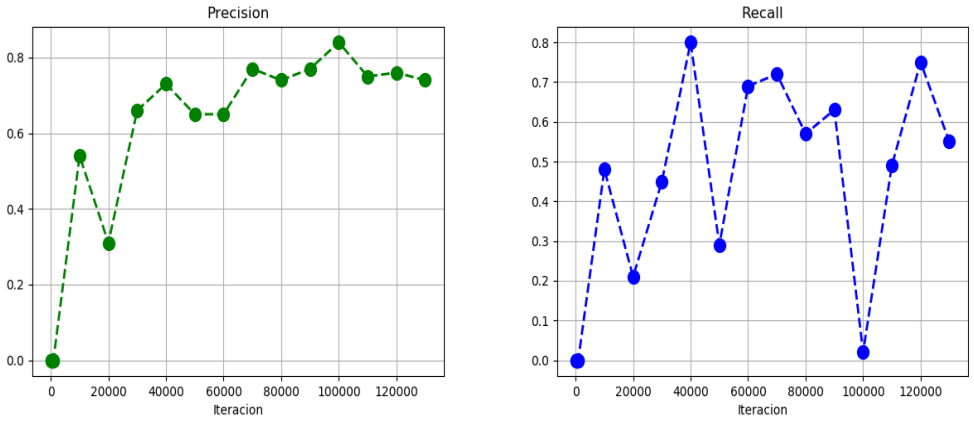


Figura 4.10: Desempeño sobre el conjunto de validación.

Puede verse que el mejor resultado se obtuvo en la iteración 40,000 donde se tuvo una precisión y un *recall* del 73 % y 80 %, respectivamente.

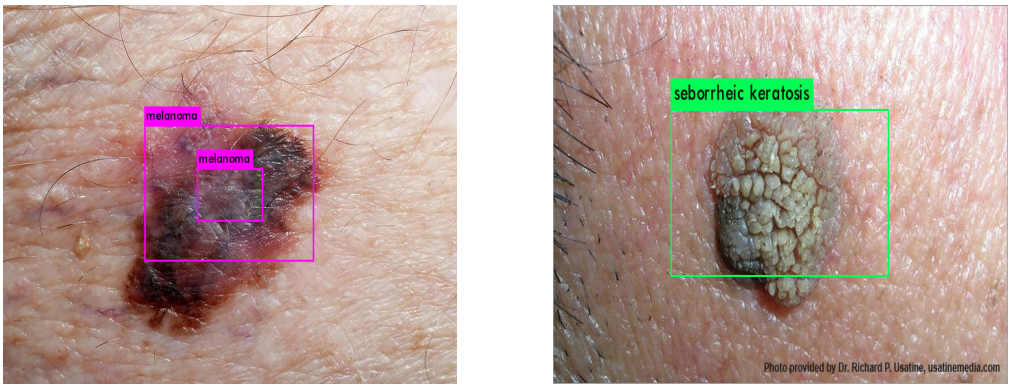


Figura 4.11: Predicciones para imágenes del conjunto de prueba con la probabilidad de la clase predicha. La imagen de la izquierda corresponde a un melanoma, la imagen de la derecha corresponde a queratosis seborreica. Las predicciones de clase son melanoma con 99 % (izquierda) y queratosis con 98 % (derecha).

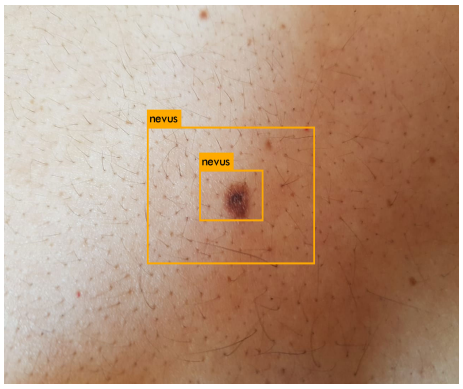


Figura 4.12: Imagen del conjunto de prueba correspondiente a un nevus. La predicción de clase es nevus con 100 %.

Discusión. Puede verse que para el problema de detectar tres diferentes tipos de lesiones en piel, los mejores resultados se obtuvieron en el experimento 4, esto se refleja en las métricas de precisión, *recall*, valor F1 e IoU. En este experimento se aumento el número de datos a través de transformaciones y de añadir nuevos ejemplos a través de las imágenes segmentadas. El otro aspecto que se pudo observar es que el modelo converge mucho más rápido a valores aceptables que en los experimentos 2 y 3. Con esto, se comprueba la hipótesis de que las segmentaciones tienen un impacto favorable en el desempeño del modelo. Sin embargo, a pesar de los métodos de preprocesamiento,

el conjunto ISIC 2017 sigue siendo un conjunto de datos pequeño. En la literatura por lo general se combina este con datos adquiridos por los autores así como de otras bases de datos (no abiertas) de menor tamaño.

En el experimento 1 donde se aborda el caso más general, detectar las lesiones como benignas o malignas, puede verse que contar con un número de datos de al menos 2,000 imágenes hace que el desempeño del modelo se incremente significativamente, aunque debemos tener en cuenta que en dicho experimento sólo se le pide detectar entre dos tipos de lesiones por lo que la tarea es más sencilla que distinguir entre los tres tipos particulares de lesiones de interés. Si se explora el conjunto de datos puede verse que en muchas ocasiones estos tres tipos de lesiones parecen ser idénticas de ahí la dificultad de identificar apropiada tanto para un especialista como para los modelos computacionales. Queda pendiente segmentar las imágenes, añadirlas como parte del aumento, entrenar el modelo y analizar los resultados que se obtienen.

Finalmente, en muchas tareas de detección es de especial interés contar con un buen puntaje de IoU (que tan buena es la predicción del *bounding box* del objeto de interés con respecto a la región marcada como *ground truth*), este se puede mejorar fácilmente etiquetando manualmente todas las imágenes, siendo este enfoque una tarea lenta pero necesaria en muchos casos. O bien, se podría aprovechar la segmentación de las lesiones para pegar todas sobre una misma máscara y dibujarles automáticamente su *bounding box* pues se conoce su tamaño.

Se presenta una pequeña tabla con el resumen de los resultados obtenidos.

- Detección entre lesiones malignas y beningnas.

Experimento	Iteración	Aumento de Datos	Segmentación	Precisión	Recall
1	80,000	Sí	No	**74 %**	**86 %**

- Detección de tres diferentes tipos de afecciones de la piel.

Experimento	Iteración	Aumento de Datos	Segmentación	Precisión	Recall
2	70,000	No	No	**67 %**	**77 %**
3	150,000	Sí	No	**70 %**	**77 %**
4	40,000	Sí	Sí	**73 %**	**80 %**

Capítulo 5

Conclusiones

Al final de este trabajo de tesis se logró poner en operación un sistema de detección de lesiones en piel cuyo funcionamiento se basa en un modelo de aprendizaje profundo y en técnicas de procesamiento de imágenes. Con base en los resultados mostrados podemos decir que el sistema logró aprender a diferenciar entre tres diferentes tipos de lesiones en piel a partir de las características extraídas por la red neuronal convolucional. Estas lesiones pueden llegar a ser visualmente muy similares por lo que el problema de clasificación es complejo tanto para los modelos computacionales como para los mismo especialistas en dermatología.

Estos resultados dejan en evidencia la poderosa capacidad de extracción de características de las redes neuronales profundas así como su capacidad de adaptación como solución a nuevos problemas completamente diferentes de aquellos para las que fueron originalmente concebidas.

Sin embargo, los resultados mostrados son todavía mejorables abordando el problema desde los siguientes flancos.

- El archivo de la ISIC es una base de datos de imágenes dermatológicas pública y abierta que con el paso de los años su archivo ha ido creciendo significativamente. El combustible de los modelos de aprendizaje profundo son los datos por lo que en tanto el archivo siga creciendo en número de imágenes y en diversidad de lesiones, se podrán generar mejores modelos y últimadamente alcanzar mejores resultados que los reportados en parte de la literatura actual. Existen también otras bases de datos como la biblioteca de imágenes Dermofit de la Universidad de Edimburgo que consta de un total de 1,300 imágenes, sin embargo, el *dataset* no es abierto, se requiere cubrir una tarifa así como con los requisitos para que al interesado le sea otorgada una licencia de uso.
- Se tiene actualmente el *dataset* de imágenes completamente segmentado por lo que se pueden aprovechar estas imágenes para pegar lesiones de diferente naturaleza en una misma imagen. Además como las imágenes ya se encuentran segmentadas, es posible generar directamente el *bounding box* por lo que el etiquetado sería automático. Esto permitiría explotar completamente las capacidades del detector pues sería capaz de aprender que pueden existir lesiones de diferente naturaleza, localizadas en diferentes regiones, en una misma imagen.

- YOLOv3, entrenada sobre *datasets* como COCO, es el estado del arte en sistemas de detección, sin embargo, no existen *benchmarks* de su desempeño en el contexto de imágenes médicas contra otros sistemas de detección. Estas pruebas se pueden comenzar de forma inmediata pues existen otros detectores comparables basados en redes neuronales profundas tales como SSD, Faster-RCNN, RetinaNet o incluso Mask-RCNN donde este último es capaz de realizar segmentación y clasificación en tiempo real.

Finalmente, algunas de las futuras líneas de desarrollo que puede tomar este trabajo de investigación son las siguientes:

- En vista del auge que vive el aprendizaje profundo hoy día, se propone explorar la opción de embeber una arquitectura de red neuronal profunda en un teléfono móvil para realizar tareas de detección haciendo uso de los *frameworks* actuales como TensorFlow Lite.
- En imágenes médicas se tiene el problema de que los datos disponibles muchas veces son escasos. Se propone explorar la posibilidad de utilizar el estado del arte en modelos generativos como las redes adversarias generativas (GANs) para producir nuevas imágenes dermatológicas con la finalidad de que al ser analizadas por especialistas médicos les resulten indistinguibles de las imágenes reales. De esta manera, se podría generar un *dataset* original de gran utilidad para futuros experimentos y para la comunidad interesada en abordar este problema.
- Se propone la puesta en funcionamiento de un sistema de computo distribuido del tipo cliente-servidor. Un sistema de este tipo le permitiría a un médico subir imágenes de su interés a un servidor remoto para ser evaluadas. El procesamiento de las imágenes se realizaría sobre el servidor a través de uno o dos modelos, previamente entrenados, de redes neuronales profundas como Inception v4. El resultado del proceso de inferencia para las imágenes solicitadas se le regresaría al médico como una etiqueta junto a la probabilidad asociada para tal diagnóstico.

Apéndice A

Apéndice

A.1. Desempeño en el conjunto de validación para el experimento 1

Iteración	Precisión	Recall	F1	IoU promedio
100	0	0	0	0
200	0	0	0	0
300	0	0	0	0
400	0	0	0	0
500	0	0	0	0
600	0	0	0	0
700	0	0	0	0
800	0	0	0	0
900	0	0	0	0
10000	0.55	0.69	0.61	36.39
20000	0.68	0.74	0.71	43.49
30000	0.71	0.87	0.78	60.26
40000	0.73	0.82	0.77	58.86
50000	0.7	0.64	0.67	47.58
60000	0.73	0.58	0.65	51.12
70000	0.72	0.82	0.77	57.15
80000	0.74	0.86	0.8	53.94
90000	0.77	0.65	0.71	52.91
100000	0.75	0.84	0.79	52.87
110000	0.74	0.83	0.78	50.16
120000	0.76	0.76	0.76	65.91
130000	0.73	0.8	0.76	51.36
140000	0.7	0.79	0.74	47.36
150000	0.71	0.59	0.64	51.28
160000	0.74	0.73	0.74	54.82
170000	0.73	0.78	0.75	61.45
180000	0.73	0.57	0.64	50.67
190000	0.73	0.8	0.76	55.37
200000	0.73	0.6	0.66	49.69

A.2. Desempeño en el conjunto de validación para el experimento 2

Iteración	Precisión	Recall	F1	IoU promedio
100	0	0	0	0
200	0	0	0	0
300	0	0	0	0
400	0	0	0	0
500	0	0	0	0
600	0	0	0	0
700	0	0	0	0
800	0	0	0	0
900	0	0	0	0
10000	0.52	0.57	0.55	33.2
20000	0.38	0.58	0.46	25.11
30000	0.61	0.72	0.66	49.85
40000	0.55	0.61	0.58	42.95
50000	0.56	0.79	0.65	44.42
60000	0.55	0.72	0.62	41.24
70000	0.67	0.77	0.72	59.27
80000	0.61	0.51	0.56	46.72
90000	0.45	0.55	0.5	31.32
100000	0.52	0.76	0.62	40.86
110000	0.6	0.77	0.68	44.58
120000	0.5	0.72	0.59	42.26

A.3. Desempeño en el conjunto de validación para el experimento 3

Iteración	Precisión	Recall	F1	IoU promedio
100	0	0	0	0
200	0	0	0	0
300	0	0	0	0
400	0	0	0	0
500	0	0	0	0
600	0	0	0	0
700	0	0	0	0
800	0	0	0	0
900	0	0	0	0
10000	0.6	0.68	0.63	48.75
20000	0.42	0.56	0.48	30.8
30000	0.45	0.53	0.49	32.87
40000	0.63	0.66	0.64	51.07
50000	0.67	0.56	0.61	57.48
60000	0.51	0.46	0.48	38.84
70000	0.68	0.77	0.72	59.98
80000	0.6	0.62	0.61	50.04
90000	0.63	0.57	0.6	57.14
100000	0.6	0.7	0.65	55.35
110000	0.62	0.58	0.6	52.9
120000	0.22	0.13	0.16	17.88
130000	0.67	0.75	0.71	60.45
140000	0.61	0.74	0.67	54.5
150000	0.7	0.77	0.73	64.51
160000	0.73	0.15	0.25	60.4
170000	0.64	0.71	0.67	56.57
180000	0.63	0.33	0.43	52.07
190000	0.51	0.37	0.43	40.2
200000	0.66	0.58	0.61	60.12

A.4. Desempeño en el conjunto de validación para el experimento 4

Iteración	Precisión	Recall	F1	IoU promedio
100	0	0	0	0
200	0	0	0	0
300	0	0	0	0
400	0	0	0	0
500	0	0	0	0
600	0	0	0	0
700	0	0	0	0
800	0	0	0	0
900	0	0	0	0
10000	0.54	0.48	0.51	42.64
20000	0.31	0.21	0.25	21.15
30000	0.66	0.45	0.54	46.1
40000	0.73	0.8	0.76	64.18
50000	0.65	0.29	0.4	41.81
60000	0.65	0.69	0.67	57.56
70000	0.77	0.72	0.74	64.01
80000	0.74	0.57	0.64	59.43
90000	0.77	0.63	0.69	56.6
100000	0.84	0.02	0.03	61.93
110000	0.75	0.49	0.59	56.13
120000	0.76	0.75	0.75	62.25
130000	0.74	0.55	0.63	67.5

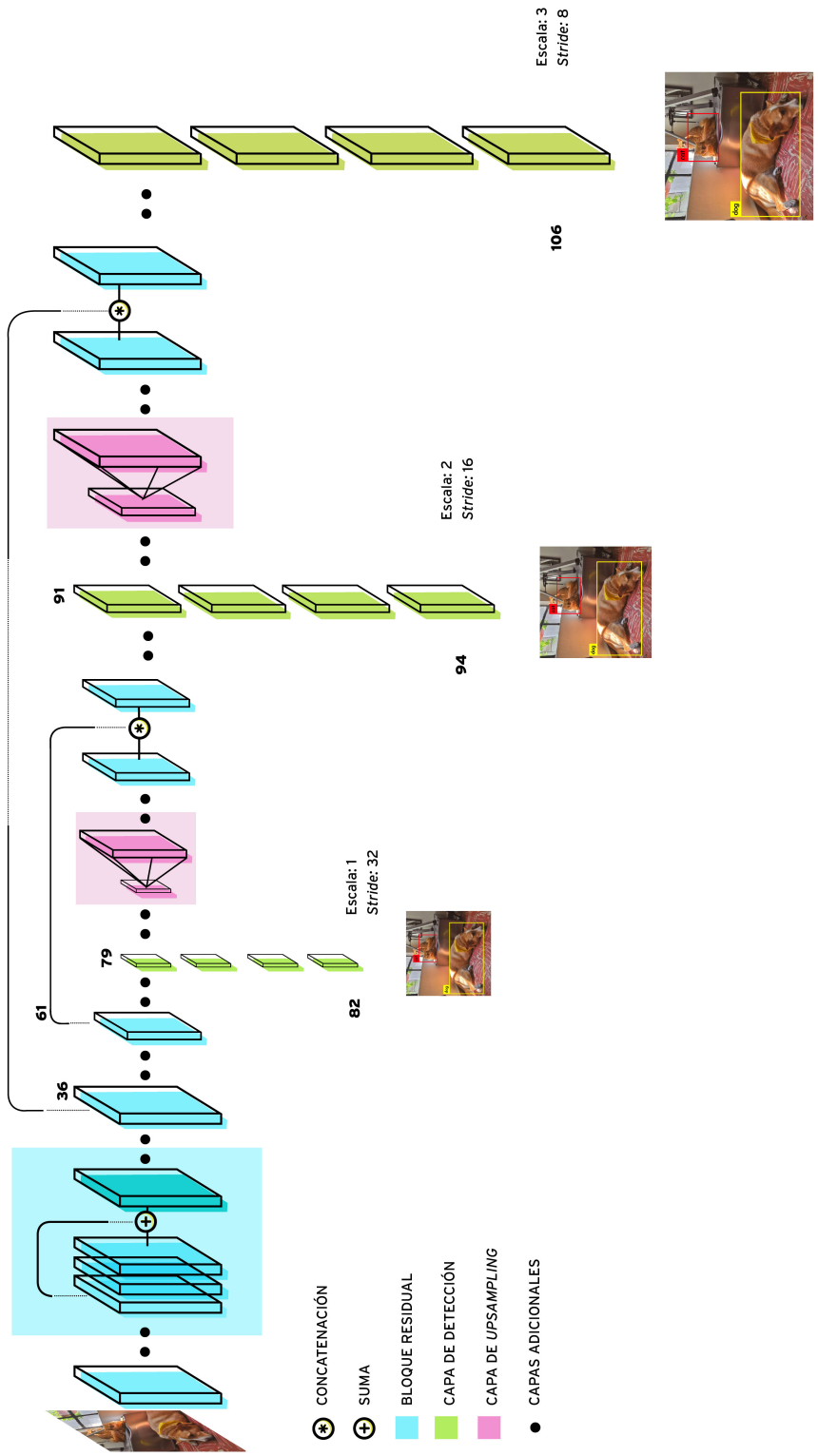


Figura A.1: Arquitectura de YOLOv3. Basado en el diagrama original de Ayoosh Kathuria.

Bibliografía

[1] (ACS), A.C.S.: Cancer facts & figures 2019 [internet], https://www.cancer.org/research/cancer-facts-statistics/all-cancer-facts-figures/cancer-facts-figures-2019.html, [En línea; accedido 22-mar-2019]

[2] (ACS), A.C.S.: If you have basal or squamous skin cancer, https://www.cancer.org/cancer/basal-and-squamous-cell-skin-cancer/if-you-have-basal-or-squamous-skin-cancer.html, [En línea; accedido 23-sep-2019]

[3] (ACS), A.C.S.: Skin cancer image gallery, https://www.cancer.org/cancer/skin-cancer/galleries/skin-cancer-image-gallery.html, [En línea; accedido 22-mar-2019]

[4] Amidi, A., Amidi, S.: Cs 230 - deep learning cheatsheets, https://stanford.edu/~shervine/teaching/cs-230/, [En línea; accedido 1-oct-2019]

[5] Canny, J.: A computational approach to edge detection. IEEE Transactions on pattern analysis and machine intelligence (6), 679–698 (1986)

[6] Celebi, M.E., Schaefer, G., Iyatomi, H.: Objective evaluation of methods for border detection in dermoscopy images. In: 2008 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society. pp. 3056–3059. IEEE (2008)

[7] Chollet, F.: Deep Learning with Python. Manning (Nov 2017)

[8] Chuong, C.M., Nickoloff, B., Elias, P., Goldsmith, L., Macher, E., Maderson, P., Sundberg, J., Tagami, H., Plonka, P., Thestrup-Pederson, K., et al.: What is the'true'function of skin? Experimental dermatology **11**(2), 159–187 (2002)

[9] Codella, N.C., Gutman, D., Celebi, M.E., Helba, B., Marchetti, M.A., Dusza, S.W., Kalloo, A., Liopyris, K., Mishra, N., Kittler, H., et al.: Skin lesion analysis toward melanoma detection: A challenge at the 2017 international symposium on biomedical imaging (isbi), hosted by the international skin imaging collaboration (isic). In: 2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018). pp. 168–172. IEEE (2018)

[10] DeVries, T., Ramachandram, D.: Skin lesion classification using deep multi-scale convolutional neural networks. arXiv preprint arXiv:1703.01402 (2017)

[11] Di Leo, G., Paolillo, A., Sommella, P., Fabbrocini, G.: Automatic diagnosis of melanoma: a software system based on the 7-point check-list. In: 2010 43rd Hawaii international conference on system sciences. pp. 1–10. IEEE (2010)

[12] Doukkali, F.: Clustering using k-means algorithm, https://www.kdnuggets.com/2018/07/clustering-using-k-means-algorithm.html, [En línea; accedido 25-sep-2019]

[13] Esteva, A., Kuprel, B., Novoa, R.A., Ko, J., Swetter, S.M., Blau, H.M., Thrun, S.: Dermatologist-level classification of skin cancer with deep neural networks. Nature **542**(7639), 115 (2017)

[14] Esteva, A., Robicquet, A., Ramsundar, B., Kuleshov, V., DePristo, M., Chou, K., Cui, C., Corrado, G., Thrun, S., Dean, J.: A guide to deep learning in healthcare. Nature Medicine **25** (01 2019). https://doi.org/10.1038/s41591-018-0316-z

[15] Fujisawa, Y., Otomo, Y., Ogata, Y., Nakamura, Y., Fujita, R., Ishitsuka, Y., Watanabe, R., Okiyama, N., Ohara, K., Fujimoto, M.: Deep-learning-based, computer-aided classifier developed with a small dataset of clinical images surpasses board-certified dermatologists in skin tumour diagnosis. British Journal of Dermatology **180**(2), 373–381 (2019)

[16] Gallegos Hernández, J.F., Nieweg, O.E.: Melanoma cutáneo (mc): diagnóstico y tratamiento actuales. Gaceta Médica de México **150**(S2), 175–182 (2014)

[17] Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the thirteenth international conference on artificial intelligence and statistics. pp. 249–256 (2010)

[18] Goodfellow, I., Bengio, Y., Courville, A.: Deep learning. MIT press (2016)

[19] Google: Classification, https://developers.google.com/machine-learning/crash-course/classification/accuracy, [En línea; accedido 2-nov-2019]

[20] Google: Seeing potential, https://about.google/stories/seeingpotential/, [En línea; accedido 28-sep-2019]

[21] Gron, A.: Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. O'Reilly Media, Inc., 1st edn. (2017)

[22] Haralick, R.M., Shapiro, L.G.: Computer and robot vision, vol. 1. Addison-wesley Reading (1992)

[23] Harangi, B.: Skin lesion classification with ensembles of deep convolutional neural networks. Journal of biomedical informatics **86**, 25–32 (2018)

[24] He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE international conference on computer vision. pp. 1026–1034 (2015)

[25] (ISIC), I.S.I.C.: Isic 2016: Skin lesion analysis towards melanoma detection, https://challenge.kitware.com/#challenge/560d7856cad3a57cfde481ba, [En línea; accedido 10-may-2019]

[26] Johr, R.H.: Dermoscopy: alternative melanocytic algorithms-the abcd rule of dermatoscopy, menzies scoring method, and 7-point checklist. Clinics in dermatology **20**(3), 240–247 (2002)

[27] Kanitakis, J.: Anatomy, histology and immunohistochemistry of normal human skin. European journal of dermatology **12**(4), 390–401 (2002)

[28] Katanforoosh, K., Kunin, D.: Initializing neural networks, https://www.deeplearning.ai/ai-notes/initialization/, [En línea; accedido 28-oct-2019]

[29] Katanforoosh, K., Kunin, D., Ma, J.: Parameter optimization in neural networks, https://www.deeplearning.ai/ai-notes/optimization/, [En línea; accedido 31-oct-2019]

[30] Kittler, H., Pehamberger, H., Wolff, K., Binder, M.: Diagnostic accuracy of dermoscopy. The lancet oncology **3**(3), 159–165 (2002)

[31] Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. pp. 1097–1105 (2012)

[32] Kumar, A.: Exponentially weighted average for deep neural networks, http://www.ashukumar27.io/exponentially-weighted-average/, [En línea; accedido 31-oct-2019]

[33] Mahbod, A., Schaefer, G., Wang, C., Ecker, R., Ellinge, I.: Skin lesion classification using hybrid deep neural networks. In: ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 1229–1233. IEEE (2019)

[34] Menegola, A., Tavares, J., Fornaciali, M., Li, L.T., Avila, S., Valle, E.: Recod titans at isic challenge 2017. arXiv preprint arXiv:1703.04819 (2017)

[35] Messadi, M., Cherifi, H., Bessaid, A.: Segmentation and abcd rule extraction for skin tumors classification. Journal of Convergence Information Technology **9**(2), 21 (2014)

[36] Nasiriany, S., Thomas, G., Wang, W., Yang, A.: A comprehensive guide to machine learning, https://www.eecs189.org/static/resources/comprehensive-guide.pdf, [En línea; accedido 2-nov-2019]

[37] Ocampo Blandón, C.F., et al.: Herramienta soporte al diagnóstico del melanoma usando imágenes dermatoscópicas= A Support Tool for Melanoma Diagnosis by using Dermoscopy Images. Ph.D. thesis, Universidad Nacional de Colombia-Sede Manizales

[38] OpenCV: Histogram equalization, https://docs.opencv.org/master/d5/daf/tutorial_py_histogram_equalization.html, [En línea; accedido 25-sep-2019]

[39] Peruch, F., Bogo, F., Bonazza, M., Cappelleri, V.M., Peserico, E.: Simpler, faster, more accurate melanocytic lesion segmentation through meds. IEEE Transactions on Biomedical Engineering **61**(2), 557–565 (2013)

[40] Proksch, E., Brandner, J.M., Jensen, J.M.: The skin: an indispensable barrier. Experimental dermatology **17**(12), 1063–1072 (2008)

[41] Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 779–788 (2016)

[42] Redmon, J., Farhadi, A.: Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767 (2018)

[43] Rumelhart, D.E., Hinton, G.E., Williams, R.J., et al.: Learning representations by back-propagating errors. Cognitive modeling **5**(3), 1 (1988)

[44] Schadendorf, D., Hauschild, A.: Melanoma in 2013: Melanoma—the run of success continues. Nature Reviews Clinical Oncology **11**(2), 75 (2014)

[45] Sethumadhavan, G., Sankaran, S.: Border detection and cancer propagation on spectral bands of malignant melanoma using six sigma threshold. In: 2009 Eighth IEEE/ACIS International Conference on Computer and Information Science. pp. 586–592. IEEE (2009)

[46] Tushar, F.I.: Automatic skin lesion segmentation using grabcut in hsv colour space. arXiv preprint arXiv:1810.00871 (2018)

[47] Vélez, A.: Cáncer de la piel y su detección temprana: la importancia del autoexamen, https://www.elmostrador.cl/agenda-pais/vida-en-linea/2016/11/07/cancer-a-la-piel-y-su-deteccion-temprana-la-importancia-del-autoexamen, [En línea; accedido 23-sep-2019]

[48] Werbos, P.J.: The roots of backpropagation: from ordered derivatives to neural networks and political forecasting, vol. 1. John Wiley & Sons (1994)

[49] (WHO), W.H.O.: How common is skin cancer?, https://www.who.int/uv/faq/skincancer/en/index1.html, [En línea; accedido 23-sep-2019]

[50] Yuheng, S., Hao, Y.: Image segmentation algorithms overview. arXiv preprint arXiv:1707.02051 (2017)