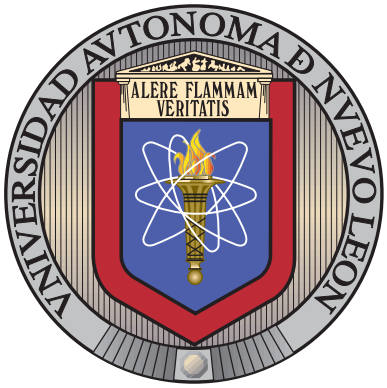
Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

Subdirección Académica



Detección de melanoma de piel mediante segmentación semántica

por

Mario Alberto Flores Hernández

como requisito parcial para obtener el grado de

licenciatura en ingeniería en mecatrónica

Febrero 2021

Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

Subdirección Académica

Detección de melanoma de piel mediante segmentación semántica

por

Mario Alberto Flores Hernández

como requisito parcial para obtener el grado de

licenciatura en ingeniería en mecatrónica

Febrero 2021

Universidad Autónoma de Nuevo León
Facultad de Ingeniería Mecánica y Eléctrica
Subdirección Académica

Los miembros del Comité de Tesis recomendamos que la Tesis «Detección de melanoma de piel mediante segmentación semántica», realizada por el alumno Mario Alberto Flores Hernández, con número de matrícula 1719126, sea aceptada para su defensa como requisito parcial para obtener el grado de Licenciatura en Ingeniería en Mecatrónica.

El Comité de Tesis

Dra. Satu Elisa Schaeffer
Asesora

Dr. Romeo Sánchez Nigenda
Revisor

Dra. Sara Elena Garza Villarreal
Revisora

Vo. Bo.

Dr. Fernando Banda Muñoz
Subdirector Académico

San Nicolás de los Garza, Nuevo León, febrero 2021

Índice general

Índice de figuras

Índice de Cuadros

Notaciones

A continuación se resumen las notaciones usadas a lo largo de este trabajo de tesis, incluyendo el símbolo, significado y equivalente en el idioma inglés.

Símbolo	Definición	Equivalente al inglés
x	dato de entrada	*input*
y	dato de salida	*output*
$\hat{y}$	salida estimada	*estimated output*
d	dimensión de entrada	*input dimension*
d_{o}	dimensión de salida	*output dimension*
n_s	número de muestras	*samples*
e_i	entrada del modelo	*node input*
p_i	peso sináptico	*synaptic weight*
ϕ	función de activación	*activation function*

Agradecimientos

Quiero agradecer primeramente a mi madre Patricia Hernández Romero por siempre representar un ejemplo de esfuerzo y superación en cada uno de los obstáculos que se han presentado.

A mi padre Mario Alberto Flores Rosales (†), quien me enseñó las virtudes del trabajo y la disciplina.

A la Dra. Satu Elisa Schaeffer por la asesoría, paciencia y consejos otorgados durante el desarrollo de este trabajo.

Al Dr. Romeo Sánchez Nigenda y a la Dra. Sara Elena Garza Villarreal por el tiempo y comentarios otorgados para mejorar este trabajo.

A mis compañeros de la FIME y del ITESM quienes me han otorgado su tiempo para enriquecer y ampliar mis conocimientos y que además de eso, me han servido como modelos a seguir y ejemplos de superación.

Resumen

Mario Alberto Flores Hernández.

Candidato para obtener el grado de Licenciatura en Ingeniería en Mecatrónica.

Universidad Autónoma de Nuevo León.

Facultad de Ingeniería Mecánica y Eléctrica.

Título del estudio: Detección de melanoma de piel mediante segmentación semántica.

Número de páginas: 43.

Objetivos y método de estudio: El presente trabajo de tesis habla sobre la aplicación del aprendizaje profundo o también denominado *deep learning*, para la detección de melanoma de piel mediante la aplicación de las redes neuronales profundas. El melanoma es un padecimiento que se origina en las células de la piel cuando los melanocitos (las células que dán el color marrón a la piel), comienzan a crecer sin control causando estragos a quien la padece y que sin una detección y tratamiento temprano, puede aumentar exponencialmente el riesgo de fallecimiento del afectado. Una red neuronal es un modelo matemático que pretende simular el proceso de aprendizaje de las neuronas biológicas mediante el recibimiento de señales binarias o normalizadas, la asignación de un peso sináptico o importancia para cada señal y una función de activación que discrimine o realce dichas señales. Es posible desarrollar un modelo cuyo entrenamiento dé como resultado un sistema que ejecute una secuencia específica de transformaciones con el dato entrante, mediante la optimización automática de pesos sinápticos en las distintas capas del filtrado, para transformar la señal o dato entrante al valor deseado a la salida. El dato entrante, en el caso de imágenes a color, trata de tres matríces correspondientes a los tres canales de colores primarios (azul, verde, rojo), y la salida deseada trata de un mapa probabilístico correspondiente al mapa de segmentación en el cual se ilustran las regiones de la imagen mediante colores distintos, refiriéndose mediante su color a las cate-

goría correspondiente de dicha región (tejido sano, tejido melanoma). Para obtener un modelo que realice dicha serie de transformaciones se requiere de dos procesos: *reducción* y *reconstrucción*. La fase de *reducción* trata de reducir la dimension de la imagen de entrada conservando su información característica, haciendo más fácil su computación en los procesos siguientes; y la fase de *reconstrucción*, como el nombre indíca, corresponde a la reconstrucción de la imagen basándose en una predicción aplicada al dato previamente codificado en la fase de *reducción*. Para finalizar se realiza un escalado del mapa probabilístico obtenido obtenido de la predicción para crear la máscara de segmentación con sus correspondientes regiones categorizadas.

CONTRIBUCIONES Y CONCLUSIONES: La principal contribución del presente trabajo de tesis es la de establecer los pasos requeridos para llevar a cabo el proceso de implementación de redes neuronales profundas para realizar tareas de segmentación semántica. El lenguaje utilizado en esta implementación fue el de *Python*, debido a que sus librerías permiten desplegar ágilmente modelos neuronales y utilizar métricas que permitan evaluar y experimentar su funcionamiento. Una de las métricas utilizadas para evaluar dicha hipótesis es la del índice de Jaccard, su función es comparar un mapa binario con un mapa probabilístico obtenido por el modelo generado y determinar la similitud entre ambos.

Para la implementación se propuso el uso de la arquitectura `FPN`[1], la cuál es compatible con el codificador `ResNet`, ambos consisten en secuencias de convolución y deconvolución que realizarán las tareas de reducir las dimensiones de los datos en un solo dato multi-dimensional y de recrear el mapa probabilístico correspondiente a las computaciones del modelo. Para validar la selección de la arquitectura y codificador se establecieron una serie de experimentos dependientes unos de otros en donde se determina la mejor configuración de parámetros para la creación y el entrenamiento del modelo así como la validación de las máscaras obtenidas mediante sobreposición de pixeles de ambas máscaras, mediante el coeficiente de dados y el criterio de Jaccard.

Firma de la asesora: _____
Dra. Satu Elisa Schaeffer

[1]Feature Pyramid Network

Capítulo 1

Introducción

La piel es considerada como el órgano más grande del cuerpo humano; está compuesta por tres capas: *epidermis*, *dermis* e *hipodermis*, tal como se ilustra en la figura 1.1. La función principal de la piel es la de proteger al cuerpo de las hostilidades del medio ambiente tales como la radiación solar y los factores externos como las bacterias, proteger a los órganos y otros tejidos internos, y también funciona como una interfaz que nos permite interactuar con el medio ambiente otorgándonos la capacidad de percibir sensaciones tales como el tacto y la temperatura.

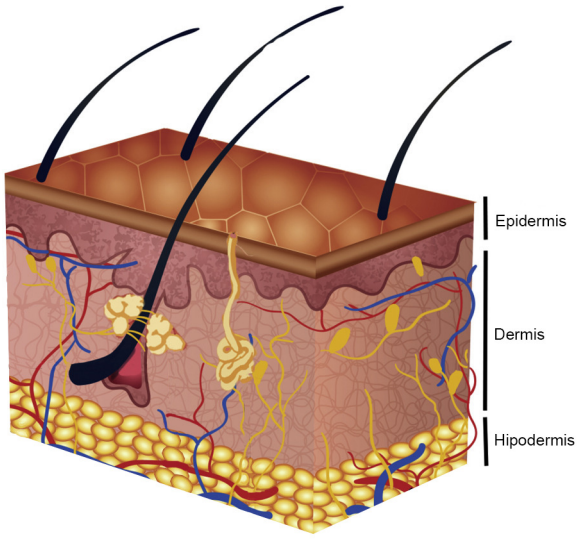


Figura 1.1: Ilustración de las capas de la piel y sus apéndices [22].

Sin embargo, debido a la exposición continua a la radiación solar o artificial, un porcentaje de la población llega a desarrollar anomalías relacionadas a la piel. Una de estas anomalías es la denominada *melanoma*, la cuál es una mutación originada en los melanocitos (células encargadas de otorgar la pigmentación marrón a la piel), en donde estas células crecen sin control y se esparcen a otras partes del cuerpo, ocasionando graves daños a la salud de quien la padece y que sin una detección temprana puede aumentar rápidamente el riesgo de fallecimiento [14].

Figura 1.2: Ejemplo de melanoma [13].

Su detección temprana es imprescindible para reducir el porcentaje de mortalidad que esta enfermedad representa, de ahí el porqué es importante no solamente trabajar en encontrar un tratamiento, sino también (y de forma simultánea), trabajar e investigar en el uso de tecnologías emergentes que permitan su detección temprana, fiable y al alcance de la mayoría.

En los últimos años se han logrado muchos avances en cuanto al desarrollo de *software* inteligente, una de las tecnologías que han adquirido mayor importancia es la *red neuronal*, la cual se trata de un modelo matemático que tiene la capacidad de *aprender* mediante el uso de bases de datos y mediante funciones de optimización que permite la configuración de dicho modelo y le otorga la capacidad de predecir, clasificar o construir datos ya sean futuros o desconocidos. Algunos de los sectores que se han beneficiado más de esta tecnología son: el sector automotriz (pilotos automáticos), el sector de manufactura (optimización de procesos), el sector de entretenimiento (recomendaciones personalizadas), el sector médico (diagnóstico de

imágenes).

1.1 Hipótesis

El presente trabajo de tesis tiene como hipótesis el uso de la inteligencia artificial para la detección y clasificación del melanoma de piel; mediante la implementación de un modelo basado en el aprendizaje profundo, es posible crear una aplicación de *software* que entrene y optimice un modelo para realizar la tarea de la *segmentación semántica*, la cual se trata de la transformación y predicción de pixeles para obtener la clasificación de las diferentes regiones que representa una imágen dermatológica. Y de esta manera obtener una aplicación de reconocimiento visual cuya robustez pueda ser verificada a partir de una serie de experimentos.

1.2 Objetivos

Primero en *objetivo general*, se habla de manera conceptual la problemática a resolver tales como cuales son las situaciones en las que podemos optimizar la resolución de un problema mediante el uso de la red neuronal, posteriormente en los *objetivos específicos* se describe de forma puntual los pasos a realizar en el presente trabajo de tésis para llegar al resultado deseado.

El *objetivo general* de este trabajo de tesis es la creación de una aplicación con la capacidad de reconocer y segmentar melanoma en la piel de forma automatizada, esto con el motivo de asistir a los médicos dedicados a esta labor a optimizar y extender la detección temprana, consiguiendo así reducir la tasa de mortalidad de este padicimiento.

El *objetivo específico*, es el de implementar un modelo de red neuronal cuya entrada sean imágenes que pueden o no contener la presencia de alguno de los ti-

pos de cáncer de piel comunes, y como salida del modelo se obtenga un mapa de características donde de haber presencia de el cáncer este se distinga mediante una región colorada en el espacio que abarca. El modelo debe cumplir con las siguientes características:

- El modelo debe ser capaz de trabajar con imágenes a color o en blanco y negro.
- El modelo debe contar con una función capaz de evaluar la precisión.
- El modelo debe contar con un algoritmo capaz de optimizar la precisión.
- El modelo debe ser capaz de segmentar correctamente en imágenes no utilizadas en los datos de entrenamiento.

(a) imágen real de entrada (b) máscara de salida

Figura 1.3: Ejemplo de segmentación: entrada y salida.

En la figura 1.3a se observa como la imagen entrante presenta un caso de melanoma, mientras que la figura 1.3b se puede observar la misma imagen después de pasar por una secuencia de transformaciones, esto se denomina *segmentación semántica* y se refiere a la acción de separar y clasificar en una o más categorías los objetos detectables en una imagen.

1.3 Estructura de la Tesis

A continuación se da una breve explicación sobre el orden en el que se presentan los capítulos de este trabajo de tesis, así como una breve descripción de su contenido.

En el capítulo 2 se habla sobre los antecedentes relacionados al presente trabajo de tesis, primero se empieza definiendo la naturaleza del problema con el que se pretende tratar, después algúnos conceptos clave que serán necesarios para la comprensión de la implementación propuesta tales como las dimensiones de los datos y los algoritmos de evaluación y optimización.

En el capítulo 3 se recopilan trabajos relacionados al método o problemática en cuestión, se estudia las características de dichos trabajos y se busca un punto de convergencia entre estos y el presente trabajo de tésis con el fin de comparar las áreas de oportunidad.

En el capítulo 4 se define el proceso de transformación de los datos del sistema propuesto en este trabajo de tesis, desde las características de los datos a la entrada.

El capítulo 5 describe a profundidad la implementación de la propuesta, desde la descripción de los datos utilizados para el entrenamiento del modelo y la verificación de este, la arquitectura específica utilizada

Finalmente, en el capítulo 6 se exponen los resultados obtenidos de la implementación del producto científico en el capítulo anterior, así como un análisis y conclusión final sobre los valores obtenidos en precision y tiempo de entrenamiento.

Capítulo 2

Antecedentes

En este capítulo se introduce de forma teórica conceptos relacionados con el trabajo propuesto, primero se define que es el *cáncer de piel*, cuales son los factores que influyen en el desarollo de este padecimiento, los tipos de cáncer y las diferencias entre estos. Después algunos conceptos relacionados con las redes neuronales necesarios para un entendimiento sólido del *aprendizaje profundo*, tales como los elementos clave que conforman a la red neuronal, la manera en la que esta evalúa su precisión, y el algoritmo de optimización.

2.1 Cáncer de Piel

El *cáncer de piel* es una enfermedad que suele relacionarse con la aparición de lunares o manchas que no se encontraban previamente, se pueden manifestar como manchas oscuras o rojizas, bultos y/o escamas en la superficie de la piel y afecta a todos los tonos de la piel por igual. Esta enfermedad se desarrolla principalmente en partes del cuerpo expuestas al sol, sin embargo también se puede desarrollar en partes que no suelen exponerse. Algunos factores como la exposición a los rayos ultravioletas (UV), el uso de sustancias como el tabaco o el envejecimiento son factores correlacionados con la aparición del *cáncer de piel*, existen dos factores de envejecimiento y se dividen en dos grupos: *intrínsecos* y *extrínsecos*.

Los factores *intrínsecos* son aquellos que suceden de forma interna en la piel, un ejemplo de esto es el envejecimiento cronológico, el cual es un proceso natural que consiste en degradación del colágeno, la elastina y el adelgazamiento de la epidermis debido al envejecimiento celular al paso de los años y de el efecto de algunas hormonas.

Los factores *extrínsecos* son aquellos que suceden de forma agena al organismo, como es el caso del *foto-envejecimiento* el cual sucede cuando nos encontramos expuestos a los rayos ultravioletas (UV). Este factor de envejecimiento genera lesiones en las cadenas de ácido desoxirribonucleico (ADN) debido a la oxidación y afecta la regeneración de celulas, al sistema inmune y a la forma en la que se regula la pigmentación [17].

2.1.1 Tipos de Cáncer de Piel

El cáncer de piel es un padecimiento que se puede manifestar en una gran variedad de formas distintas, algúnas de estas formas pueden representar un enorme riesgo para la salud de quien la padece mientras que otras pueden permanecer mucho tiempo sin afectar la calidad de vida del paciente; se clasifican principalmente en dos tipos: *benigno* y *maligno*.

Los tumores del tipo *benignos* no suelen ser considerados cáncer como tal, ya que tienen un crecimiento lento y no suelen extenderse a otras partes del cuerpo, por otro lado los tumores *malignos*, si representan un riesgo mayor ya que crecen rapidamente y suelen hacer *metástasis*, lo cual significa que migra a otras partes del cuerpo.

2.2 Redes Neuronales

Hasta hace algunos años el desarrollo de aplicaciones de software solía realizarse de forma robústa, por ejemplo, si deseáramos desarrollar una aplicación de reconocimiento de imágenes faciales de la forma tradicional, sería necesario un grupo de expertos en dicho rubro para que definan una secuencia de transformaciones específicas para lograr esa función, donde la variabilidad de las imágenes que dicho sistema permitiría depende específicamente de la complejidad del mismo, sin embargo mediante las redes neuronales es posible resolver el mismo problema con la ventaja de que la variabilidad de imágenes que pueden ser reconocidas por el sistema, depende de los datos utilizados para el entrenamiento, el model de una red neuronal *aprende* de los datos proporcionados y crea un modelo capaz de clasificar, predecir o reconstruir datos conocidos y desconocidos. Algunos de los puntos clave que conforman un sistema basado en redes neuronales son:

Datos Se debe determinar la naturaleza y la dimensión de los datos que entrarán al modelo.

Modelo Se debe crear un sistema que transforme los datos de entrada, en la salida deseada.

Evaluación El modelo debe tener la capacidad de evaluar su precisión.

Optimización El modelo debe contar con un algoritmo que optimice la precisión.

2.2.1 Imágenes como Datos

Las imágenes se pueden definir como una matriz de $m \times n \times 1$ pixeles en el caso de las imágenes de un solo canal (blanco y negro) y en el caso de imágenes a color es de $m \times n \times k$, donde k es igual al numero de canales de color que tenga

la imagen, siendo tres en el caso de imágenes RGB[1], es importante definir si las imágenes ingresarán al modelo a color o en blanco y negro debido a que esto define el numero de nodos de entrada de este, ya que para cada pixel debe corresponder un nodo de entrada y en el caso de imágenes a color también deberá tener un número de capas de nodos de entrada igual al número de canales que tenga la imagen.

[Pend: Draw grids]

2.2.2 Modelo

Un *modelo* se puede definir como el bloque intermedio entre los datos de entrada (*input*) y los datos de salida (*output*), es el sistema encargado de transformar los datos de entrada en la salida deseada y consta de los siguientes componentes:

La unidad principal que compone el modelo de la red neuronal es el perceptrón, denominado *nodo* en esta literatura. Se trata de una analogía matemática basada en el comportamiento de la neurona biológica, el nodo recibe n entradas, las cuales son multiplicadas por el peso p, se suma el producto de las entradas por sus pesos $e \times p$ y finalmente se multiplica por la función de activación ϕ.

Partiendo de lo anterior, creando arreglos con estos nodos podemos formar estructuras denominadas *redes neuronales*, las cuales cuentan con los siguientes ele-

[1]RGB: Se refiere a los tres canales de color rojo, verde y azul, por sus siglas en inglés (*red, green, blue*).

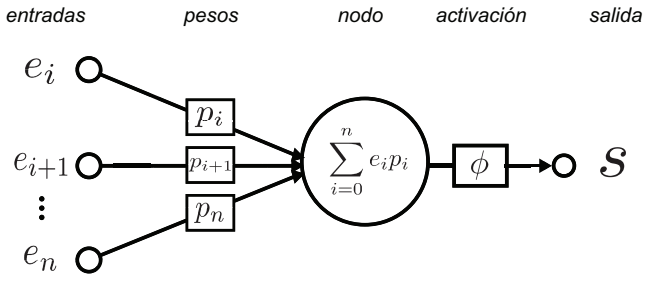


Figura 2.1: Ejemplo de un perceptrón.

mentos estructurales:

Capa de entrada (*input layers*) Se trata de la primera capa del modelo, en el caso de imágenes a cada pixel le corresponde un nodo de entrada. Estos nodos deben tener las mismas dimensiones que las imágenes de entrada.

Capa oculta (*hidden layer*) Las dimensiones de estos nodos pueden ser diferentes a los de entrada y tener varias capas ocultas, sin embargo esto puede afectar la complejidad y precisión del modelo.

Capa de salida (*output layer*) Esta es la capa de salida del modelo, las dimensiones de la capa de salida determinan las dimensiones del dato resultante

Función de activación (*activation*) Se trata de una función dentro de cada nodo que interactúa con el valor entrante y se multiplica por el peso.

Pesos (*weight*) Se trata de un valor entre el nodo de la capa actual y el nodo siguiente inicializado de forma aleatoria y después ajustado por un algoritmo para aproximar la salida al valor real.

Sesgo (*bias*) Se trata de un valor constante que se suma al resultado de la función de activación y el peso, de esta forma se puede ajustar que tan fácil o difícil es activar un nodo.

2.2.3 Evaluación y Optimización

Para realizar el proceso de *aprendizaje* del modelo, primero se debe evaluar cuál es el estado actual de las predicciones. Para esto es necesario evaluar que distancia existe entre el valor predicho y el valor real, esto se puede lograr mediante una *función de pérdida* que se encargue de determinar que tanta diferencia (*loss*) existe en las estimaciones del modelo con los pesos actuales.

Un ejemplo de la función de pérdida es el de la función logarítmica del costo (*log-likelihood*), como se muestra a continuación

$$l(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_{j=1}^{q} (y_j \log \hat{y}_j), \tag{2.1}$$

q representa el número de clases entre las cuales predecir, y_j representa el valor de salida real, $\hat{y_j}$ el valor estimado de salida por el modelo y j la posición indizada de clase.

Ya que se tiene calculada la pérdida del modelo en su configuración de pesos actual, es necesario actualizar el valor de todos los pesos dentro del modelo para poder acercarnos al valor real. Aquí es donde participa el algoritmo de *gradiente descendiente* [1], dicho algoritmo se puede representar con la siguiente ecuación

$$\text{error} = \frac{\partial}{\partial m} = \frac{2}{N} \sum_{i=1}^{N} -X_i(y_i - (mX_i + b)), \tag{2.2}$$

$$\frac{\partial}{\partial b} = \frac{2}{N} \sum_{i=1}^{N} (y_i - (mX_i + b)), \tag{2.3}$$

donde m y b son valores enteros que representan el peso de dos parámetros de un modelo, los cuáles serán optimizados para acercar la salida del modelo a la salida real.

Capítulo 3

Estado del Arte

En este capítulo se estudia las literaturas relacionadas con el presente trabajo de tesis con el objetivo de hacer una comparativa entre distintos métodos para resolver el mismo problema, o implementaciones similares para resolver problemas distintos.

En la primera sección, *trabajos similares*, se recopilan trabajos con características relacionadas al presente trabajo de tesis, ya sea relacionados con el método o con el problema que se pretende resolver, se describe el tema que abarcan y los puntos que lo relacionan con el trabajo aquí presente.

En la segunda sección, *análisis comparativo*, se comparan las distintas características de los trabajos revisados, de esta forma podemos determinar las principales diferencias así como las ventajas y desventajas de cada trabajo. Así mismo se recopilaran todas estas diferencias en una tabla para visualizar mejor estas características.

Finalmente en las *áreas de oportunidad*, se realiza una conclusión acerca de los resultados en la tabla comparativa.

3.1 Trabajos Similares

A continuación se mencionan los trabajos revisados en los que se puede obtener un punto de convergencia ya sea en los métodos similares utilizados para resolver problemas diferentes o en su defecto métodos distintos para resolver problemas similares.

Badrinarayanan *et al.* [2] en este se describe la arquitectura denominada `SegNet` utilizada principalmente para la conducción autónoma y la detección de peatones, se describe como una arquitectura convolucional, con la característica de que es una jerarquía de codificador-decodificador donde a cada codificación (*encoding*) corresponde una capa de agrupación de máximos (*max pooling*) y un decodificador. Inspirado principalmente en la arquitectura `VGG16`, con la diferencia de que el componente clave en esta arquitectura es el uso de convoluciones en lugar de la capa completamente conectada de nodos (*fully-connected-layer*) a la salida.

Ronneberger *et al.* [18] proponen uno de los primeros modelos de la redes neuronales convolucionales denominada `U-net` dirigida al análisis de imágenes del area médica. Este trabajo consiste en una arquitectura con dos trayectorias: la trayectoria de contracción de la imagén y la trayectoria de expansión. En la trayectoria de contracción de la imagen se aplica una secuencia de convoluciones de dimension 3×3 con solapado, seguido de una función de activación de rectificación linear unitaria (ReLu) y posteriormente una operación de agrupación de máximos (*max pooling*) para la compresión de la imagen (*downsampling*). Por otro lado, la trayectoria de expansión consiste de el escalado de la imagen (*upsampling*), en cada paso se aplica una capa de convolución de 2×2 a la cuál se concatena una imagen recortada del mapa de características de la trayectoria de contracción y dos convoluciones de 3×3 seguidos de una activación ReLu cada una. En total la arquitectura tiene 23 convoluciones (U-net).

Chen *et al.* [4] en este trabajo describen la arquitectura denominada `Deep Lab`, la cual propone un acercamiento distinto a la parte de la reconstrucción en las capas finales del modelo. Al igual que otras arquitecturas cuenta con capas de convolución. agrupación de máximos y activación de rectificación linear unitaria, sin embargo, el componente clave aquí es la denominada convolución de hoyos (*atrous convolution*), la cual es una modificación al filtro de convolución basado en la transformada de *Wavelet* para el escalado del mapa de características (*upsampling*). De esta forma se obtiene una alternativa al uso de capas de deconvolución y así se aumenta la resolución del mapa de características a la salida del modelo sin aumentar el tiempo de computación o la cantidad de parámetros.

Teichmann *et al.* [21] reducen la carga computacional en los modelos de redes neuronales convolucionales mediante la unificación de las tres tareas principales de estas: clasificación, detección y segmentación. Denominada como multi-red neuronal `MultiNet` se trata de una arquitectura tipo codificador-decodificador en donde el decodificador esta compuesto de tres ramas las cuales corresponden a las tareas de clasificación, detección y segmentación por lo que se realizan las tres tareas de forma simultanea partiendo de la misma entrada (*multi-tasking*) y usando la salida de cada una de las ramas para el ajuste fino de los parámetros del modelo (*fine-tuning*).

Kroner *et al.* [10] describen un modelo de codificación-decodificación contextual basado en el proceso en el que los humanos obtenemos la información espacial de los escenarios visuales complejos, mediante un mapa topográfico de las salientes. Se basa en más en el proceso biológico con el que se obtiene la información espacial que en los métodos matemáticos para calcularla, dicho lo anterior, los principales componentes para determinar una saliente son color, la intensidad y la orientación.

Kadampur y Al Riyaee [9] proponen el úso de los servicios de nube en conjunto con las redes neuronales profundas y el diseño de modelos utilizando modelos pre-entrenados. Para esto utilizan la herramienta de `Deep Learning Studio`, la cual es un software que permite el uso de tarjetas gráficas (*GPU*) localizadas en la nube

o localmente, y también permite el uso de múltiples tarjetas de forma simultanea (*multi-GPU*), la arquitectura usada para la detección de cáncer de piel en este trabajo es de una red neuronal profunda de clasificación, por lo que el modelo se codifica mediante convoluciones y se decodifica mediante capas de aplanamiento (*flatten layer*) para posteriormente entrar a una capa de nodos completamente conectada (*fully-connected-layer*) y obtener dos posibles resultados: es cáncer o no es cáncer.

Zhou *et al.* [25] hablan sobre el aprendizaje de máquina `Machine Learning` y las redes neuronales para la predicción teórica de nano-materiales. Primero determinaron el número de capas y la heterogeneidad del material para posteriormente entender los fenómenos visuales como el parpadeo y la emisión cuántica, concluyendo que las intensidades RGB están correlacionadas al número de capas en el material del grafeno.

Luc *et al.* [12] proponen una alternativa al proceso de decodificación. Normalmente en las tareas de segmentación es necesaria la parte de codificación para realizar las predicciones a grado pixel y la decodificación para reconstruir la imagen con las categorías segmentadas. Este trabajo propone el uso de una red generativa adversaria `GAN`, para utilizarse como decodificador, de esta manera se obtiene un mapa de etiquetas (*map layer*) con una mayor resolución.

Jain *et al.* [8] desarrollan una herramienta de procesamiento de imágenes médicas relacionadas al melanoma, basándose en sus características físicas como: asimetría, color, bordes y diámetro. Se basa en la técnica de segmentación de imágenes mediante el uso de filtros de contraste, detección de ejes, etc. A diferencia de los otros trabajos mencionados, este no consiste del uso de redes neuronales sino de filtrados de imágen mediante distintos kernels de convoluciones para obtener la máscara.

3.2 Análisis Comparativo

En esta sección se evalúan los trabajos revisados con el trabajo propuesto en este trabajo de tesis, para esto se define una serie de características que servirán como puntos de referencia entre el trabajo propuesto y los trabajos relacionados, dichas características son las siguientes:

Modelo Tipo de arquitectura del modelo.

Clasificación El sistema puede clasificar entre distintas categorías.

Segmentación El sistema puede segmentar las imágenes.

Supervisado El sistema requiere de datos de datos de entrenamiento.

Pre-entrenamiento El sistema puede inicializarse con pesos pre-entrenados como alternativa a la inicialización con pesos aleatorios.

Evaluación El sistema cuenta con una función de evaluación y un algoritmo de optimización.

Salida Características de los datos obtenidos en la salida del modelo.

Cuadro 3.1: Similitudes y diferencias entre los trabajos revisados; las características implementadas se representan con ✓, mientras que las no implementadas con ×.

Trabajo	**Modelo**	Clasificación	Segmentación	Supervisado	Pre-entrenamiento	Evaluación	Salida
Badrinarayanan *et al.* [2]	`SegNet`	✓	✓	✓	✓	✓	mapa de etiquetas
Ronneberger *et al.* [18]	`U-net`	✓	✓	✓	×	✓	mapa de etiquetas
Chen *et al.* [4]	`DeepLab`	✓	✓	✓	×	✓	mapa de etiquetas
Teichmann *et al.* [21]	`MultiNet`	✓	✓	✓	×	✓	mapa de etiquetas
Kroner *et al.* [10]	`VGG16`	✓	✓	✓	✓	✓	mapa de calor
Kadampur y Al Riyaee [9]	`CNN`	✓	×	✓	×	✓	mapa de etiquetas
Zhou *et al.* [25]	`ML / SVM`	✓	×	✓	×	✓	etiqueta
Luc *et al.* [12]	`CNN/GAN`	✓	✓	✓	✓	✓	mapa de etiquetas
Jain *et al.* [8]	`A.B.C.D`	×	✓	×	×	×	etiqueta
Propuesta de tesis	`FPN`	✓	✓	✓	✓	✓	mapa de etiquetas

3.3 Áreas de Oportunidad

En esta sección se señalan las características de los trabajos mencionados en la sección anterior y se compara con las características del método propuesto en este trabajo de tesis para obtener una comparativa sobre las áreas de oportunidad.

Para las tareas de segmentación es fundamental contar con un codificador y un decodificador eficientes, sin embargo aún más importante es tener la capacidad de evaluarse así mismo y optimizarse, a diferencia de algunas técnicas más robustas de diseño, las redes neuronales convolucionales permiten esas funcionalidades. Otro aspecto importante a considerar es la capacidad del úso de pesos pre-entrenados a la hora de entrenar utilizando nuestros datos, ya que incluso cuando los pesos pre-entrenados fueron obtenidos mediante bases de datos no necesariamente relacionadas con el o los objetos que nosotros queremos detectar, pero con similitudes físicas, en teoría ofrece un mejor desempeño y reduce el tiempo de entrenamiento ya que de lo contrario, se tendría que inicializar el modelo con pesos distribuidos de forma aleatoria, por lo que la precisión inicial del modelo sería cercana al cero por ciento. En este trabajo de tesis, el modelo propuesto cuenta con la capacidad de utilizar pesos pre-entrenados, el reto está en seleccionar un conjunto de pesos que mejoren la precisión en el caso de imágenes médicas, ya que si los pesos utilizados fueron obtenidos con imágenes que difieren mucho de las imágenes que se utilizadas en el presente trabajo, podría verse comprometida la precisión o el tiempo de entrenamiento.

Capítulo 4

Solución Propuesta

El objetivo de este trabajo de tesis es implementar una técnica para la clasificación y segmentación de regiones de tumores del tipo melanoma en imágenes dermatológicas mediante el úso del aprendizaje profundo (*deep learning*). Para comprobar la hipótesis se desarrolló una aplicación de software cuya función es la de ser entorno de trabajo mediante el cuál se puedan procesar datos para el entrenamiento de modelos, trabajar con distintas arquitecturas de redes, así como modificar los parámetros del entrenamiento para hacer el ajuste fino del modelo.

En éste capítulo se describe la estructura de la implementación propuesta, comenzando con la *metodología*, en donde se describe el conjunto de procedimientos por los cuales debe atravesar los datos desde su entrada como una imágen ya sea de un canal o triple canal, hasta su salida como máscara de segmentación con sus respectivas regiones clasíficadas a su categoría correspondiente. Posteriormente en *implentación de la solución* se describe el lenguaje utilizado para la implementación, las librerías utilizadas; así como el análisis llevado a cabo para determinar cuál configuración de parámetros ofrece el mejor resultado y una comparativa o *benchmarking*.

4.1 Metodología

La herramienta propuesta cuenta con tres fases principales:cargado y filtrado de datos, entrenamiento y predicción, la fase de *cargado de datos* consiste en adquirir y corregir las imágenes que serán usadas en las fases siguientes, el *entrenamiento* trata de la obtención de un modelo a partir de dos datos: la imagen real (entrada) y la máscara conocida (salida deseada) que corresponde a la región a clasificar, el proceso de entrenamiento consta de un número determinado de ciclos también denominados épocas (*epochs*) en dónde se prueban distintas configuraciones y se conserva la configuración con el mayor porcentaje de precisión. Al finalizar el entrenamiento se obtiene un modelo mediante el cuál se realiza la segunda fase, *predicción*, ahora únicamente se requiere un dato de entrada el cuál es la imágen de la cual se desconoce la clasficación de sus regiones, y como resultado se obtiene una estimación de la máscara de segmentación desconocida con sus regiones correctamente clasificadas.

4.1.1 Características de los Datos de Entrada

Para poder llevar a cabo la creación del modelo primero es necesario tener un conjunto de datos (*dataset*) que es el que será utilizado para el entrenamiento y la validación de la configuración de pesos sinápticos en la época presente. A partir de éste momento se referirá como *muestra* al conjunto de datos de entrada del entrenamiento del modelo; como se mencionó al comienzo del capítulo, para llevar a cabo el entrenamiento es necesario otorgar al sistema dos entradas: la imágen real y la máscara conocida de las regiones ya clasíficadas. Por lo tanto una muestra es el

Figura 4.1: Diagrama de flujo general de la herramienta propuesta.

equivalente a la concatenación de una imágen dermatolólogica y la máscara conocida correspondiente. Con una sola muestra se puede generar un modelo sin embargo la precisión sería muy baja, generalmente entre mayor sea la cantidad de muestras se puede obtener una mejor precisión. No obstante depende mucho de la variabilidad de las imágenes, ya que si se tratara de entrenar un modelo de muchas categorías, la variabilidad haría mas complejo el modelo y por ende requiriría de un mayor número de épocas para alcanzar una precisión viable (*underfit*); así como el úso de un *dataset* muy extenso de pocas categorías y con imágenes muy similares podría causar justamente lo contrario (*overfit*), lo cuál no permitiría realizar predicciones correctamente en imágenes no tan parecidas a las usadas en el entrenamiento.

4.1.2 Codificación de Características

Para poder realizar una predicción de la máscara segmentada, primero es necesario reducir las dimensiones de las imágenes sin afectar la información que estas representan. Para conseguir esto se utiliza un filtro de *convolución de matríces*. Una imágen es una función bidimensional $z = f(x, y)$, donde x y y son coordenadas espaciales y z es el valor de la intensidad de la imágen en el punto (x, y), en el caso de las imágenes a color se tienen tres funciones donde cada una representa la intensidad del pixel del canal correspondiente (rojo, verde o azul) [16, p. 100]. La convolución de matríces se define como la obtención de una matríz C a partir de dos matríces A y B. La matríz $A_{m\times n}$ sería el mapa de intensidad de pixeles de la imágen mientras que la matríz $B_{(2N+1)\times(2N+1)}$ es el kernel de la convolución, mediante la ecuación 4.1 se obtiene la matríz $C = A * B$

$$c_{ij} = \frac{1}{b}\sum_{r=1}^{2N+1}\sum_{s=1}^{2N+1}(a_i - N + r - 1, i - N + r - 1), \tag{4.1}$$

donde $b = \sum_{i,j=1}^{2N+1} b_{i,j}$, si $b = 0$ se toma $b = 1$.

El filtro de convolución es la base del proceso de codificado de la imagen, la co-

dificación se refiere a la transformación de una imagen a un vector de características de alta dimensión (*downsampling*), una vez que se tiene el mapa de características sigue con el proceso de *upsampling*.

4.1.3 Modelo FPN

La arquitectura de red neuronal tipo red piramidal de características, por sus siglas en inglés FPN[1], se trata de una estructura que funciona en conjunto con el codificador `ResNet`, mientras el codificador tiene la tarea de realizar el *downsampling* de las dimensiones espaciales de la imágen de entrada y convertirlas a un mapa de característica , el decodificador hace justamente lo opuesto (*up-sampling*), a partir del mapa de características producto de la codificación, realiza una secuencia de deconvoluciones que construyen la máscara de segmentación estimada.

[Pend: corregir ×]

En la figura 4.3 se puede apreciar la arquitectura y la transformación de las dimensiones de los datos, el número a lado izquiero de las capas de convolución representa la resolución en proporción a la entrada y el número de lado derecho representa el número de canales. El número de canales aumenta mientras que la resolución disminuye.

[1]FPN: Feature Pyramid Network

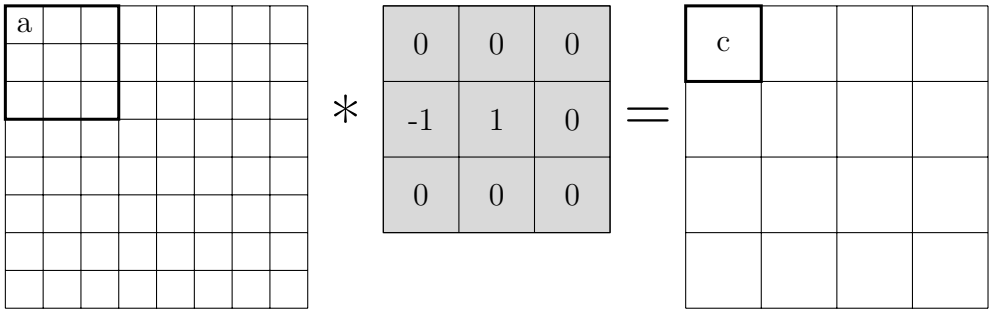


Figura 4.2: Representación del proceso de obtención de la matriz convolucionada.

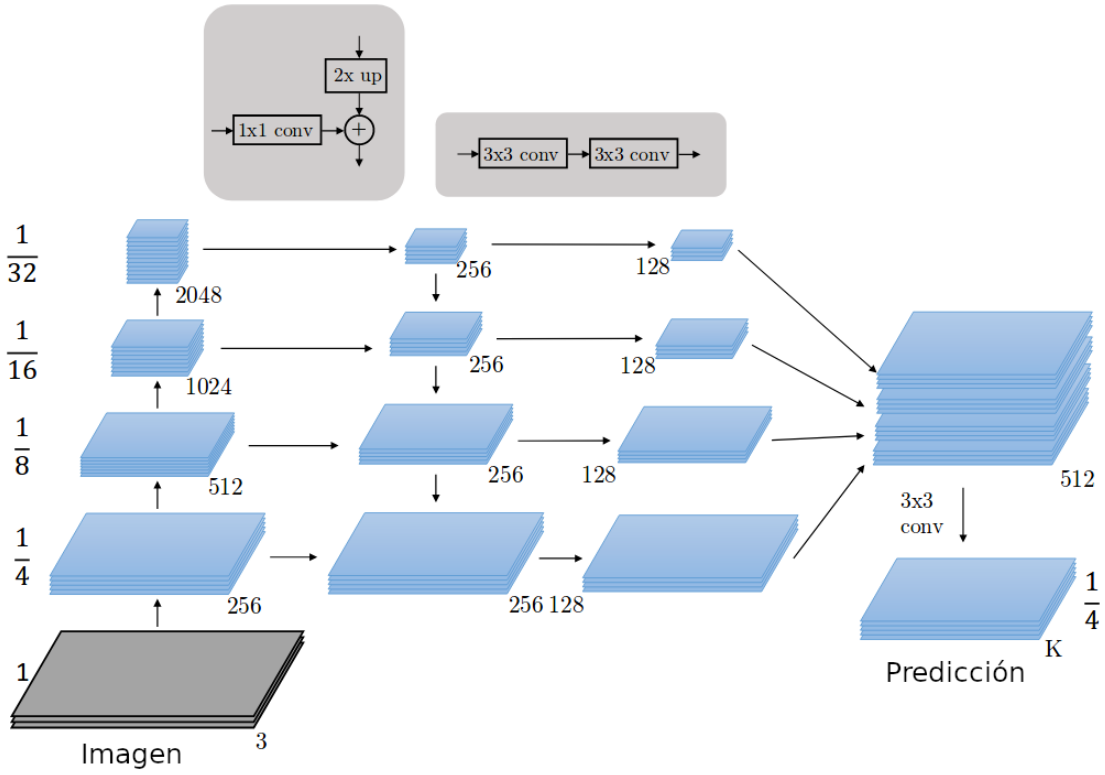


Figura 4.3: Representación de la arquitectura FPN [11].

4.2 Implementación de la Solución

Para la implementación de la herramienta propuesta, se utilizó `Python` como lenguaje principal. `Python` es un lenguaje de alto nivel con un extenso repositorio de paquetes desarrollados por la comunidad, también es un lenguaje ligero ideal para el procesamiento de datos y de operaciones con datos de altas dimensiones (*high-dimensional data*), debído a esto último, se consideró `Python` la mejor opción para la implementación de la red neuronal profunda.

Si bien otros lenguajes tienen una mayor velocidad en la ejecución de cálculos complejos, es la facilidad con la que se pueden desplegar implementaciones complejas en relativamente pocas líneas de código lo que llevo a la conclusión de que `Python` es la opción ideal para la implementación propuesta, como el hecho de que existen muchas documentaciones e implementaciones similares en este lenguaje.

A continuación se especifican las librerías claves utilizadas para llevar a cabo la implementación, así mismo se da una breve descripción general sobre la misma y su importancia durante el desarollo de la propuesta.

Cuadro 4.1: Librerías utilizadas para la implementación de la propuesta.

Librería	**Descripción**
`Torch.vision`	Este paquete es una colección de módulos relacionados con las operaciones de tensores y la aceleración de cálculos mediante hardware. Se usa en conjunto con `NumPY` y cuenta con las arquitecturas de redes neuronales típicas, así como herramientas para el cargado de datos y la configuración de parametros de los modelos.
`Quvel Segmentation Models`	Este paquete cuenta las arquitecturas de las redes neuronales de segmentación semántica típicas, se trata del marco de trabajo mediante el cuál podemos crear y configurar la arquitectura de la red para después realizar el entrenamiento.
`Numpy`	Esta librería fue desarrollada para habilitar el uso de vectores y matríces de grandes dimensiones y para la computación numérica en general, ya que `Python` originalmente no cuenta con el soporte para esto.
`OpenCV`	Es una librería dedicada al procesamiento de imágenes y a la visión computacional, cuenta con funciones que permiten cargar imágenes y obtener su matríz de pixeles así como aplicar efectos y transformaciones en estas.
`Albumentations`	Esta librería es principalmente útil para el pre-procesado de las imágenes al momento de realizar el entrenamiento, cuenta con funciones que permiten crear secuencias de transformaciones específicas para distíntos subconjuntos de datos así como la capacidad de transformar imágenes a tensores.

En el cuadro 4.2 se especifícan las características del equipo en el cuál fue realizado el experimento.

Cuadro 4.2: Especificaciones técnicas de los componentes.

Componente	**Descripción**
CPU	AMD Ryzen 5 3600x, 3.80 GHz, de seis núcleos.
GPU	Nvidia GeForce 1050ti, 4 GB.
MOBO	Gigabyte Gaming AB-350.
RAM	HyperX Fury, 16 GB, 2433 MHz.

4.2.1 Computación Asistída por Hardware

Los cálculos llevados en el proceso de entrenamiento involucra realizar operaciones con datos de altas dimensiones (*high-dimensional data*) los cuales requieren una gran cantidad de recursos de procesamiento, estas operaciones al estar relacionadas con vectores y matríces pueden ser ejecutadas rapidamente por la unidad GPU[2], entre mayor sea la velocidad de procesamiento de la GPU más rápido es el entrenamiento de nuevos modelos y con mayor resolución.

4.2.2 Módulos de la Implementación

La herramienta desarrollada en el presente trabajo de tesis se divide en cuatro módulos funcionales y un módulo central encargado de controlar la ejecución de los mismos. En la figura 4.4 se puede observar la estrúctura modular del software desarrollad, dentro de dichos módulos se encuentran las funciones específicas de esa categoría y en el módulo principal se encuentran los parámetros del modelo.

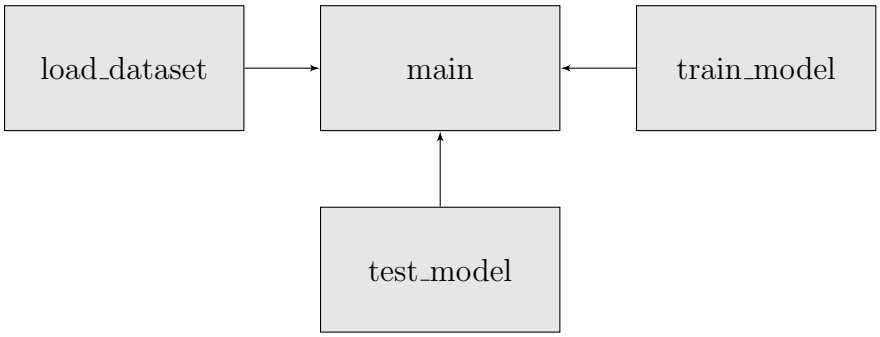


Figura 4.4: Distribución de los módulos en la herramienta desarrollada.

[2]GPU: Graphics Processor Unit

4.2.3 Lectura de Entrada

El proceso de cargado de datos y filtrado es un proceso muy importante para la ejecución correcta en las siguientes transformaciones, para el entrenamiento del modelo se utilizará una base de datos obtenida de la Asociación de Recolección de Imágenes dermatolólogicas, por sus siglas en inglés (*ISIC*[3]). La base de datos de ISIC cuenta con diez mil imágenes de entrenamiento con resoluciones distintas, con diferentes tonos de iluminación y rotación, así como su máscara de segmentación conocida y dos mil imágenes para realizar pruebas, también con resoluciones no estandarizadas y sin máscara de segmentación.

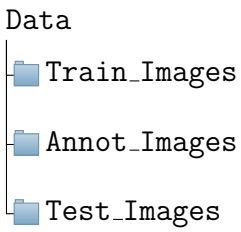


Figura 4.5: Representación del árbol de directorios de imágenes.

Lo primero que se desarrolló fue un asistente de cargado y filtrado de imágenes, en el código 4.1 se puede apreciar el código fuente y las funciones utilizadas para tal motivo. Al ser utilizado el objeto *dataset* y alimentar los argumentos con los directorios de las imágenes se obtiene como resultado un objeto de dimensión $N \times 2 \times (m \times n)$ donde N es el número de muestras cargadas y el primer elemento de la fila es la matríz de la imágen real, y el segúndo la matríz de la máscara conocída.

```
1    # load_dataset.py
2    class dataset ():
3        def __init__(self, images_dir, masks_dir, classes=None,):
4        # Extraccion de ids de imagenes y mascaras.
5        self.ids = listdir(images_dir)
6        self.images_fps = [os.path.join(images_dir(img_id)
7                           for img_id in self.ids]
```

[3]ISIC: International Skin Imaging Collaboration.

```
        self.masks_fps = [os.path.join(mask_dir,
                          (img_id[:-4] + '_segmentation.png'))
                          for img_id in self.ids]
    # Lectura de imagenes y correcion de canales.
    def __getitem__(self,i):
        image = cv2.imread(self.images_fps[i])
        image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
        mask = cv2.imread(self.masks_fps[i],0)
        return image, mask
```

Código 4.1: Código fuente del asistente de cargado de imágenes.

```

    # main.py.
    import load_dataset as ld
    train_dir = 'Data/Train_Images'
    mask_dir = 'Data/Train_Images'
    set_de_datos = ld.dataset(train_dir, mask_dir)
    # Al indicar la posicion dentro del dataset se obtiene una
   muestra dividida en dos objetos:
    imagen, mascara = set_de_datos[1]
```

Código 4.2: Invocación de la función contenida dentro de la clase.

En el código 4.2 se invoca la función `dataset`, la cuál tiene por argumento la dirección de las imágenes reales y las máscaras conocidas, con esto se crea el conjunto de datos que posteriormente es utilizado en la sección de entrenamiento.

4.2.4 Pre-procesamiento

El *pre-procesamiento* se define como una secuencia de transformaciones que se aplican al momento previo del entrenamiento y afecta a un subconjunto específico de los datos. Para el diseño de la secuencia de transformaciones tanto para el subconjunto de entrenamiento como el subconjunto fueron consideradas las siguientes características.

Resolución El codifícador admite resoluciones específicas, debído a las dimensiones admitidas para el filtrado de convolución de matríces.

Número de muestras La cantidad de datos disponibles para el entrenamiento afecta la precisión final, si el conjunto de datos es limitado se pueden realizar transformaciones de las imágenes presentes para obtener más datos de entrenamiento. Así como tener un conjunto muy extenso y con poca variabilidad afectaría también la capacidad de predecir imágenes generalizadas, se puede ajustar mediante el pre-procesamiento.

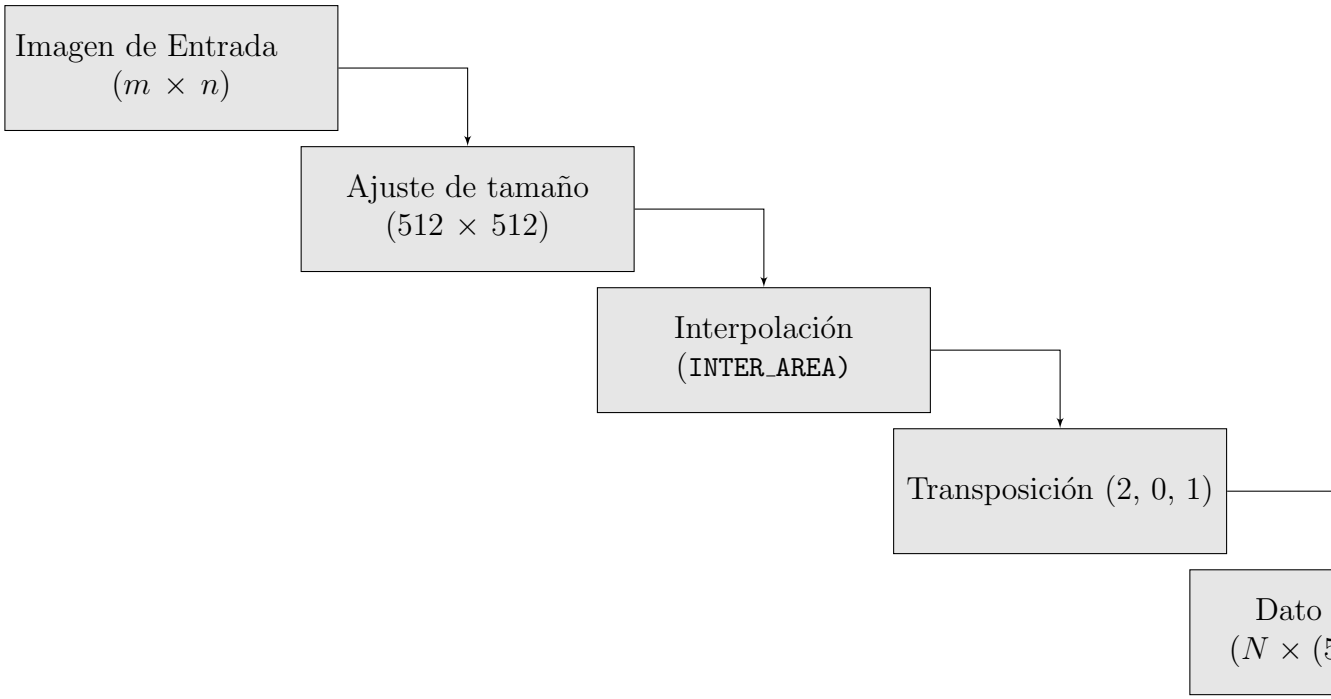


Figura 4.6: Línea de pre-procesamiento.

[Pend: arreglar textos]

Debido a que el conjunto de datos es extenso y solo clasifíca entre dos categorías, no es necesario realizar demasiadas trasnformaciones, no obstante es importante trasponer la matríz de entrada para convertir la matríz a tensor.

4.2.5 Entrenamiento y Validación

La secuencia de entrenamiento consiste de un búcle en el cuál se condensan las imágenes en un grupo (*batch*), para ser procesadas por el codificador de forma simultanea, esto para acelerar la velocidad de entrenamiento. Sin embargo, a mayor tamaño de batch también se requiere mayor memoria disponible por la unidad procesadora de gráficos y también la precisión del entrenamiento se ve influenciado. Posteriormente se evalúa mediante el *índice de Jaccard*, como se muestra en la ecuación 4.2.

```
1
2    # Variable inicializada en cero para almacenar la precision
   maxima obtenida.
3    max_score = 0
4    for i in range(0,40):
5        print('\n Epoch: {}'.format(i))
6        train_logs = train_epoch.run(train_loader)
7        valid_logs = valid_epoch.run(valid_loader)
8
9        if max_score < valid_logs['iou_score']:
10           max_score = valid_logs['iou_score']
11           torch.save(model,('Model/'+encoder+'.pth'))
12           print('Highest Score Model Saved: {}'.format(max_score)
   )
13       if i == 25:
14           optimizer.param_groups[0]['lr'] = 1e-5
15           print('decreased decoder learning rate to 1e-5')
16     lr= 0.0001),])
```

Código 4.3: Búcle de entrenamiento.

Para poder determinar la eficiencia durante el *entrenamiento* y optimizar los pesos sinápticos es necesario hacer uso de métricas, dichas métricas se aplicarán sobre el subconjunto de validación. Los datos utilizados fueron divididos en una proporción de 70 % para entrenamiento y 30 % para la validación, el subconjunto de

validación no es usado durante el entrenamiento, sino qué, al terminar una época se verifíca con ese subconjunto la precisión.

La métrica utilizada para evaluar el modelo es la del *índice de Jaccard* (*Intersection Over Union*) por sus siglas en inglés (IoU) [19] de la siguiente ecuación,

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \tag{4.2}$$

donde A y B se pueden interpretar como la máscara real y la máscara estimada, y debido a que las imágenes consisten de pixeles, la ecuación anterior se puede representar en el caso de objetos discretos como

$$J = \frac{1}{n} \sum_{c=1}^{k} W_c \sum_{i=1}^{n} \frac{y_i^c \hat{y}_i^c}{y_i^c + \hat{y}_i^c - y_i^c \hat{y}_i^c} \tag{4.3}$$

donde y_i^c y $\hat{y}_i^c$ son valores binarios y corresponden a la probabilidad de que el pixel i corresponda a la clase c de un número k de clases.

La librería de Yakubovskiy [24] cuenta con distintas métricas ya implementadas dentro de su `API`[4], dichas métricas se pueden configurar previamente al entrenamiento. En el código 4.4 se puede apreciar la selección de la función de métrica, la función de pérdida y el optimizador de pesos del modelo.

```
1  # main.py.
2  import segmentation_models_pytorch as smp
3  # Metrica de evaluacion (Jaccard Index)
4  metric = [smp.utils.metrics.IoU(threshold=0.5),]
5  # Funcion de perdida (Dice Loss)
6  loss = smp.utils.losses.DiceLoss()
7  # Funcion optimizadora (Adam)
8  optimizer = torch.optim.Adam([dict(params=model.parameters(),
```

[4]Interfaz de programación por sus siglas en inglés, *Application Programming Interface*.

```
    lr= 0.0001),])
```

Código 4.4: Ejemplo de configuración de parámetros.

4.2.6 Predicción de Máscaras

Una vez obtenido un modelo con precisión buena (*good fit*), el objetivo es generar máscaras desconocidas a partír de las imágenes del set de pruebas. Para esto es necesario popular de nuevo un *dataset* de dimensión $N \times (m \times n)$, básicamente es un vector de matríces donde N es el número de imágenes para realizar la prueba, y $m \times n$ es la resolución de imágen de la cuál se desea obtener la máscara de segmentación.

```
# test_model.py
def test_model(m_path, t_path, encoder, weights, classes, device):
# Cargar el modelo generado
model = torch.load(m_path)
prep_fn = smp.encoders.get_preprocessing_fn(encoder, weights)
# Crear el subconjunto de datos para pruebas
vis_dataset = ds.testing_data(t_path,
classes,augmentation=tfm.get_validation_augmentation())
test_dataset = ds.testing_data(t_path,
classes,
augmentation=tfm.get_validation_augmentation(), preprocessing=tfm.get_preprocessing(prep_fn))
# Seleccionar una muestra al azar.
n = np.random.choice(len(vis_dataset))
vis = vis_dataset[n].astype('uint8')
test_image = test_dataset[n]
# Realizar una prediccion de mascara con la imagen aleatoria.
x_tensor = torch.from_numpy(test_image).to(device).unsqueeze(0)
pred_mask = model.predict(x_tensor)
pred_mask = (pred_mask.squeeze().cpu().numpy().round())

#Visualizar la imagen real y la mascara predicha.
```

```
22    ds.visualize(image=vis, predicted=pred_mask)
```

Código 4.5: Código fuente del modulo de pruebas.

```
1    # main.py
2    if test_sample:
3    tsm.test_model(model_path, test_dir, ENCODER, ENCODER_WEIGHTS,
   CLASSES, DEVICE)
```

Código 4.6: Uso de la función definida en el modulo de pruebas.

En el código 4.5, se define la función que se encarga de cargar el modelo y cargar el subconjunto de datos para pruebas mientras que en el código 4.6 se da un ejemplo del uso de dicha función. Dando como resultado una gráfica con la imágen de entrada y la máscara estimada para tal.

Con esto finalizarían los procesos de cargado, entrenamiento y predicción de la herramienta propuesta, la línea de transformaciones se diseña en base a la cantidad de datos y las características que estas contienen. A sí mismo dichas transformaciones se llevan a cabo para adaptar las dimensiones de la imagen de entrada con el codificador, posteriormente y durante el entrenamiento se evalúa la configuración de los pesos sinapticos mediante el coeficiente de dados y se optimizan dichos pesos mediante el optimizador *adam*, el cual es una alternativa al método de gradiente descendiente y está diseñado para actualizar los pesos sinápticos en elementos de naturaleza iterativa como es el caso de las imágenes.

Capítulo 5

Experimentos

En este capítulo se presentan los experimentos llevados a cabo durante el desarrollo de la herramienta propuesta. Las siguientes secciones corresponden a diferentes experimentos llevados a cabo para validar la hipótesis del presente trabajo de tesis, cada experimento contiene tres subsecciones: *diseño experimental*, *resultados* y *discusión*.

En la subsección de *diseño experimental*, se detalla el experimento y el parámetro que se pretende definir en base a este, todos los procesos realizados por la herramienta desarrollada en el presente trabajo de tesis se ejecutan de forma secuencial, debido a esto es necesario determinar el mejor parámetro en cada paso de la transformación.

En *resultados*, se reporta e interpreta todos los resultados obtenidos mediante el experimento llevado a cabo en esa sección, cada experimento cuenta con su propia sección y se determina el mejor parámetro para realizar el experimento siguiente.

Para finalizar en *discusión*, se llega una conclusión basándose en los resultados obtenidos y la información que representan las gráficas.

5.1 Estadísticas de los Datos de Entrada

Este experimento tiene como objetivo determinar el codificador ideal para la arquitectura implementada basándose en la información estadística del conjunto de datos de las imágenes, también determinar las transformaciones necesarias para coincidir con las dimensiones requeridas por la arquitectura y las ofrecidas por la imagen.

5.1.1 Diseño Experimental

Las imágenes representan un mapa de características o de valores numéricos en forma de matríz, dichas características pueden ser analizadas para determinar la secuencia correcta de transformaciones en la línea de pre-procesado para acoplar correctamente las dimesiones de entrada de la imagen con las entradas disponibles del codificador.

Muestras El número de muestras del conjunto de datos puede determinar las transformaciones posteriores tanto para reducir *sobreajuste*, como para mejorar el *subajuste*.

Clases La cantidad de clases entre las que el modelo tiene que clasificar determina la complejidad computacional de éste, por lo tanto también determina transformaciones requeridas.

Histograma de Resoluciones Determinar cuál es la frecuencia de dimensiones en el conjunto de datos.

5.1.2 Resultados

5.1.3 Discusión

5.2 Evaluación del Aprendizaje

Una vez seleccionado el codificador ideal, es necesario evaluar la configuración del modelo en la época presente y configuración actual, al finalizar cada ciclo de entrenamiento se realiza una validación con el porcentaje del conjunto de datos que no fue utilizado durante éste.

5.2.1 Diseño Experimental

Para determinar un valor probabilistico entre dos mapas de valores binarios es necesario utilizar un coeficiente que evalúe la similitud entre ambos conjuntos de datos, siendo el conjunto A la máscara real o *ground truth*, y el conjunto B el mapa obtenido por la computación del modelo se utilizó el criterio de dados para determinar la probabilidad $(0\,\% - 100\,\%)$, la ecuación del criterio de dados dice que la probabilidad de que dos mapas binarios sean iguales es dos veces la cantidad de pixeles sobrepuestos o idénticos entre la suma de los pixeles de ambos mapas,

$$\mathrm{DC} = \frac{2|A \cap B|}{|A| + |B|} \tag{5.1}$$

donde A es un mapa binario de pixeles que representa la máscara real y B la máscara de segmentación obtenida por el modelo.

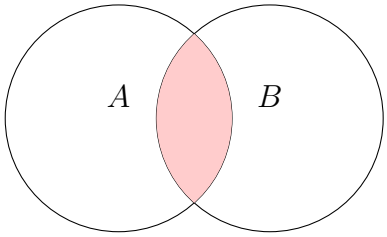


Figura 5.1: Representación de la región de sobreposición de pixeles en el coeficiente de dados.

5.2.2 Resultados

5.2.3 Discusión

5.3 Criterio de Jaccard

En la sección anterior, el criterio de dados es utilizado para evaluar la similitud entre un mapa binario y un mapa probabilistico correspondiente a la máscara conocida y la máscara estimada, durante el entrenamiento del modelo. Sin embargo para validar el resultado final es necesario utilizar una métrica similar, que penalice más el error o la diferencia entre ambos mapas.

5.3.1 Diseño Experimental

Para determinar un porcentaje de similitud entre la máscara estimada con la máscara conocida o también denominada verdad absoluta (*ground truth*) podemos utilizar una métrica como la de la ecuación 4.2, el índice de Jaccard. Este criterio es relativamente similar al coeficiente de dados de la ecuación 5.1, mediante el solapado de ambas máscaras se promedía la coincidencia entre pixeles, sin embargo en el criterio IoU el error tiene mayor penalización, por lo que se podría decir que es una versión mas robusta de evaluación ideal para evaluar el modelo final.

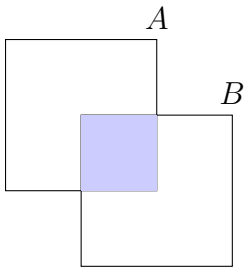


Figura 5.2: Representación de la región de sobreposición de pixeles en el índice de Jaccard.

5.3.2 Resultados

5.3.3 Discusión

Capítulo 6

Conclusiones

6.1 Discusión

6.2 Trabajo a Futuro

Glosario

dermis

Se trata de la capa intermedia de la piel, es más gruesa que la epidermis y está conformada de folículos pilosos, glándulas sudoríparas, vasos sanguíneos y nervios los cuales están sostenidos por colágeno. 1

epidermis

Se trata de la capa mas superficial de la piel, es también la mas delgada y está conformada en su mayoría por células llamadas queratinocitos o células escamosas, células basales y melanocitos. 1

hipodermis

Se trata de la capa mas profunda de la piel, la principal función de esta capa es regular la temperatura y amortiguar los golpes externos para proteger a los órganos. 1

red neuronal

Modelo matemático optimizado mediante funciones de cálculo para dar una salida deseada en base a una entrada. 2

Bibliografía

[1] Bachman, D. (2007), *Advanced calculus demystified*, McGrawHill.

[2] Badrinarayanan, V., A. Kendall y R. Cipolla (2015), «SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation», *CoRR*, **abs/1511.00561**, 1511.00561[cs.CV], URL http://arxiv.org/abs/1511.00561.

[3] Boni, R., C. Schuster, B. Nehrhoff y G. Burg (2002), «Epidemiology of skin cancer», *Neuroendocrinology Letters*, **23**, págs. 48–51.

[4] Chen, L., G. Papandreou, I. Kokkinos, K. Murphy y A. L. Yuille (2016), «DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs», *CoRR*, **abs/1606.00915**, 1606.00915[cs.CV], URL http://arxiv.org/abs/1606.00915.

[5] Codella, N. C. F., V. Rotemberg, P. Tschandl, M. E. Celebi, S. W. Dusza, D. Gutman, B. Helba, A. Kalloo, K. Liopyris, M. A. Marchetti, H. Kittler y A. Halpern (2019), «Skin Lesion Analysis Toward Melanoma Detection 2018: A Challenge Hosted by the International Skin Imaging Collaboration (ISIC)», *CoRR*, **abs/1902.03368**, 1902.03368[cs.CV], URL http://arxiv.org/abs/1902.03368.

[6] Codella, N. C. F., V. Rotemberg, P. Tschandl, M. E. Celebi, S. W. Dusza, D. Gutman, B. Helba, A. Kalloo, K. Liopyris, M. A. Mar-

chetti, H. Kittler y A. Halpern (2019), «Skin Lesion Analysis Toward Melanoma Detection 2018: A Challenge Hosted by the International Skin Imaging Collaboration (ISIC)», *CoRR*, **abs/1902.03368**, 1902.03368, URL http://arxiv.org/abs/1902.03368.

[7] GJ., T. y G. SR. (1993), «PRINCIPLES OF ANATOMY AND PHYSIOLOGY», URL https://www.clinimed.co.uk/wound-care/wound-essentials/structure-and-function-of-the-skin#:~:text=The%20skin%20consists%20of%20two,the%20functions%20of%20the%20skin.

[8] Jain, S., V. jagtap y N. Pise (2015), «Computer Aided Melanoma Skin Cancer Detection Using Image Processing», *Procedia Computer Science*, **48**, págs. 735 – 740, international Conference on Computer, Communication and Convergence (ICCC 2015), URL http://www.sciencedirect.com/science/article/pii/S1877050915007188.

[9] Kadampur, M. A. y S. Al Riyaee (2020), «Skin cancer detection: Applying a deep learning based model driven architecture in the cloud for classifying dermal cell images», *Informatics in Medicine Unlocked*, **18**, pág. 100 282, URL http://www.sciencedirect.com/science/article/pii/S2352914819302047.

[10] Kroner, A., M. Senden, K. Driessens y R. Goebel (2020), «Contextual encoder–decoder network for visual saliency prediction», *Neural Networks*, **129**, págs. 261 – 270, URL http://www.sciencedirect.com/science/article/pii/S0893608020301660.

[11] Lin, T., P. Dollár, R. B. Girshick, K. He, B. Hariharan y S. J. Belongie (2016), «Feature Pyramid Networks for Object Detection», *CoRR*, **abs/1612.03144**, 1612.03144, URL http://arxiv.org/abs/1612.03144.

[12] Luc, P., C. Couprie, S. Chintala y J. Verbeek (2016), «Semantic Segmentation using Adversarial Networks», *CoRR*, **abs/1611.08408**, 1611.08408, URL http://arxiv.org/abs/1611.08408.

[13] Marazzi, D. P. (2012), «Science Photo Library», URL https://ichef.bbci.co.uk/news/660/media/images/77303000/jpg/_77303278_skin_cancer-spl-1.jpg.

[14] medical, T. A. C. S. y editorial content team (2019), «What Are Basal and Squamous Cell Skin Cancers?», URL https://www.cancer.org/cancer/basal-and-squamous-cell-skin-cancer/about/what-is-basal-and-squamous-cell.html.

[15] of Artificial Intelligence, B. A. (2020), «Suggested Notation for Machine Learning», https://github.com/Mayuyu/suggested-notation-for-machine-learning.

[16] Palomares, F. G., J. A. M. Serrá y E. A. Martínez (2016), «Aplicación de la convolución de matrices al filtrado de imágenes», *Modelling in Science Education and Learning*, **9**(1), págs. 97–108.

[17] Rittie, L. y G. J. Fisher (2015), «Natural and Sun-Induced Aging of Human Skin», *Cold Spring Harb Perspect Med, doi: 10.1101/cshperspect.a015370*.

[18] Ronneberger, O., P. Fischer y T. Brox (2015), «U-Net: Convolutional Networks for Biomedical Image Segmentation», *CoRR*, **abs/1505.04597**, 1505.04597[cs.CV], URL http://arxiv.org/abs/1505.04597.

[19] Seferbekov, S. S., V. Iglovikov, A. Buslaev y A. Shvets (2018), «Feature Pyramid Network for Multi-Class Land Segmentation.», en *CVPR Workshops*, págs. 272–275.

[20] Shelhamer, E., J. Long y T. Darrell (2016), «Fully Convolutional Networks for Semantic Segmentation», *CoRR*, **abs/1605.06211[cs.CV]**, 1605.06211[cs.CV], URL http://arxiv.org/abs/1605.06211.

[21] Teichmann, M., M. Weber, J. M. Zöllner, R. Cipolla y R. Urtasun (2016), «MultiNet: Real-time Joint Semantic Reasoning for Autono-

mous Driving», *CoRR*, **abs/1612.07695**, 1612.07695[cs.CV], URL http://arxiv.org/abs/1612.07695.

[22] TODOROVA, K. y A. MANDINOVA (2020), «Novel approaches for managing aged skin and nonmelanoma skin cancer», *Adv. Drug Deliv. Rev., https://doi.org/10.1016/j.addr.2020.06.004*.

[23] WU, H., J. ZHANG, K. HUANG, K. LIANG y Y. YU (2019), «FastFCN: Rethinking dilated convolution in the backbone for semantic segmentation», *arXiv preprint arXiv: 1903.11816[cs.CV]*.

[24] YAKUBOVSKIY, P. (2020), «Segmentation Models Pytorch», https://github.com/qubvel/segmentation_models.pytorch.

[25] ZHOU, J., B. HUANG, Z. YAN y J.-C. G. BÜNZLI (2019), «Emerging role of machine learning in light-matter interaction», *Light: Science & Applications*, **8**(1), págs. 1–7.

Resumen autobiográfico

Mario Alberto Flores Hernández

Candidato para obtener el grado de
Licenciatura en Ingeniería en Mecatrónica

Universidad Autónoma de Nuevo León
Facultad de Ingeniería Mecánica y Eléctrica

Tesis:
Detección de melanoma de piel mediante segmentación semántica

Nací el 4 de junio de 1997 en la ciudad de Monterrey, Nuevo León. Hijo del Sr. Mario Alberto Flores Rosales y la Sra. Patricia Hernández Romero. Comencé mis estudios de Ingeniería En Mecatrónica en agosto de 2014 en la Universidad Autónoma de Nuevo León, en marzo de 2019 llevé a cabo el diplomado de Innovación Biomédica.