

Benemérita Universidad Autónoma de Puebla
Facultad de Ciencias Físico Matemáticas
Licenciatura en Física



Segmentación de imágenes radiológicas con U-Net
TESIS
Para obtener el título de:
Licenciado en Física
PRESENTA:

C. Miguel Agustín Teyssier García

Directores de Tesis: Dr. José Rubén Conde Sánchez (FCFM - BUAP)
Dr. Fernando Moreno Barbosa (Facultad de Medicina - BUAP)

Enero 2025, Puebla, Pue. Méx.

Título: Segmentación de imágenes radiológicas con U-Net

Estudiante: C. Miguel Agustín Teyssier García

JURADO

Dr. Jorge Velázquez Castro
Presidente

Dr. Lucio Fidel Rebolledo Herrera
Secretario

Dra. Gladys Denisse Salgado Suárez
Vocal

M. en C. Sergio Adán Juárez
Suplente

Dr. José Rubén Conde Sánchez
Director de tesis

Dr. Fernando Moreno Barbosa
Asesor externo

Resumen

La presente tesis se centra en el desarrollo y la aplicación de técnicas de segmentación basadas en inteligencia artificial y aprendizajes automáticos para imágenes radiológicas, con el objetivo principal de lograr una segmentación precisa de estructuras anatómicas y una adaptabilidad a diferentes tipos de imágenes. Para ello, se empleó la arquitectura de red neuronal convolucional U-Net, reconocida por su capacidad para preservar la información espacial y segmentar con precisión estructuras anatómicas en imágenes médicas.

La tesis se dividió en varias etapas. Primero, se realizó un análisis y preprocesamiento de las imágenes radiológicas, abarcando modalidades como radiografías, tomografías computarizadas (CT) y resonancias magnéticas (RM). Posteriormente, se implementó y entrenó la red U-Net utilizando un conjunto de datos representativo y variado, lo que permitió que el modelo se adaptara a la diversidad de características presentes en las imágenes radiológicas.

Los resultados obtenidos mostraron una segmentación precisa de estructuras anatómicas en todas las modalidades de imágenes evaluadas. Además, la adaptabilidad de U-Net se evidenció en su capacidad para generalizar a diferentes tipos

de imágenes radiológicas, garantizando resultados consistentes y confiables independientemente de la modalidad de imagen.

Palabras Clave: U-Net, segmentación, imagen radiológica.

Agradecimientos

Agradezco a mis papás por darme la oportunidad de estudiar, también agradezco a mis amigos que conocí a lo largo de la carrera; fueron un gran apoyo y motivación en momentos difíciles.

Dedicatoria

A mi familia y a todos mis amigos.

Índice general

Resumen	5
1. Introducción	1
1.1. Planteamiento del problema	3
1.2. Motivación	4
1.3. Justificación	5
1.4. Objetivos	6
1.4.1. Objetivo general	6
1.4.2. Objetivos particulares	7
1.5. Metodología propuesta	7
1.6. Organización de la tesis	8
2. Antecedentes	11
2.1. Redes convolucionales	11
2.1.1. ResNet	12
2.1.2. U-Net 2015	14
2.2. Avances en U-Net	16
2.2.1. DoubleU-Net	17
2.2.2. UNet ++	19
2.3. Procesamiento digital de imágenes	21
2.3.1. Definiciones	21
2.3.2. Procesamiento	23
2.3.3. Imágenes DICOM	24

2.3.4. Pydicom	25
2.4. Rayos X	26
2.4.1. Producción	26
2.5. Neumotórax	28
2.5.1. Síntomas	28
2.5.2. Diagnóstico	29
3. Marco teórico	31
3.1. Arquitectura de la red neuronal	31
3.1.1. Convolución.	31
3.1.2. Mapas de características	33
3.1.3. Función de activación	34
3.1.4. Max pooling	35
3.1.5. Stride	36
3.1.6. Padding	36
3.1.7. Sesgo	37
3.1.8. Camino de expansión	38
3.1.9. Mapas de segmentación	39
3.2. Entrenar el modelo	39
3.2.1. Función de costo	42
3.2.2. Mapas de pesos	44
3.2.3. Learning rate	45
3.2.4. Descenso de gradiente estocástico (SGD)	45
3.2.5. Épocas (Epochs)	46
4. Desarrollo y resultados	49
4.1. Procesamiento de datos	49
4.1.1. Cálculo	50
4.2. Código U-NET	52
4.3. Pruebas con imágenes conocidas	54

4.4. Resultados	55
Conclusiones	65
Bibliografía	67
A. Código	71
B. Código	75

Índice de figuras

1.1. Metodología propuesta.	7
2.1. Arquitectura de una Red Neuronal convolucio- nal clásica con 2 capas ocultas de Neuronas[10].	12
2.2. Estructura de U-Net, (ejemplo para 32x32 píxe- les en la resolución más baja). Cada cuadro azul corresponde a un mapa de características de múltiples canales. El número de canales se indica en la parte superior del cuadro. El tamaño x-y se proporciona en el borde inferior izquierdo del cuadro. Los cuadros blancos representan mapas de características copiados. Las flechas indican las diferentes operaciones [28].	14
2.3. La estructura de Double U-Net es una combi- nación de dos arquitecturas: VGG19, ASPP y U-Net [17].	18

2.4.	U-Net++ sigue el diseño de U-Net (backbone), mientras que los mapas de características muestreados ascendentes se suman con los mapas de características omitidos del codificador [43]. b) Las conexiones de salto rediseñadas que conectan las dos subredes, representadas en verde y azul, son remodeladas para mejorar la transferencia de información entre el encoder y el decoder.	20
2.5.	Una imagen con diferentes resoluciones aumenta el número de píxeles [40].	22
2.6.	Esta fotografía es una imagen (las imágenes son matrices formadas por elementos discretos), donde cada punto más pequeño se representa un píxel [7].	23
2.7.	Imágenes tipo DICOM, usadas típicamente para formatos médicos[8].	25
2.8.	Espectro Electromagnético	27
2.9.	Del lado izquierdo hay aire en el espacio pleural mientras que en el derecho no [30].	29
2.10.	Un radiografía de tórax [36].	30
2.11.	La respectiva mascara de la imagen 2.10, que identifica las zonas que contiene aire marcada de blanco, para un espacio pleural sano tendira que verse negro totalmente ya que no hay aire dentro[36].	30
3.1.	La convolución de 2 matrices, dando como resultado una tercera [10].	32

3.2.	Algunos de los tipos de kernels más usados son los que generan los mapas de características [10].	33
3.3.	Función de activación ReLu.	35
3.4.	La aplicación del Max Pool de 2x2 se puede ver como un método para extraer y resumir las características más importantes de la imagen. [5]. .	35
3.5.	Comparativa entre stride=1 y un stride=2 [18]. .	36
3.6.	Como funciona el padding tras una convolución.	36
3.7.	Una mapa de segmentación hecho con Style GAN [32].	39
3.8.	Descenso de gradiente como encuentra el mínimo de la función entre los pesos contra la función de costo.	41
3.9.	Comparativa entre las diferentes maneras de encontrar el mínimo con el algoritmo gradiente descendente [1].	41
3.10.	Se tiene la radiografía, la mascara y las mascara que debe predecir la U-Net, esta con pocas épocas de entrenamiento.	46
3.11.	Se tiene la radiografía, la mascara y las mascara que debe predecir la U-Net, esta con un aumento de épocas diferencia de la imagen anterior. . . .	47
4.1.	Arquitectura de U-Net.	50
4.2.	La reducción del volumen para después ser juntado con otro.	52
4.3.	El nuevo volumen después del Up-Conv concatenándose con su análogo	53
4.4.	Radiografía con neumotórax.	54
4.5.	Radiografía con neumotórax.	54

4.6. Radiografía con neumotórax.	54
4.7. Radiografía sin neumotórax.	54
4.8. Radiografía sin neumotórax.	54
4.9. Radiografía sin neumotórax.	54
4.10. Ejemplo de una matriz de confusión. Los valores en verde corresponden a predicciones correctas, mientras que los valores en rojo representan erro- res de clasificación.	55
4.11. Tabla con todas las métricas de evaluación más usadas.	57
4.12. Radiografías, para 20 épocas de entrenamiento. .	59
4.13. Las correspondientes figuras 4.3, 4.4 y 4.5 con sus mascarar donde se encuentra el neumotórax y la predicción con 20 épocas de entrenamiento.	60
4.14. Estas radiografías son análogas a las mostradas en las figuras 4.6, 4.7 y 4.8. La máscara original se presenta en negro, ya que no se evidencia la presencia de la enfermedad neumotórax.	61
4.15. Estas imágenes son tomadas de una base de da- tos diferentes, donde se podría decir que son ra- diografías mejor tomadas.	62
4.16. Comparativa de ResNet contra U-Net bajo las mismas condiciones.	63

Capítulo 1

Introducción

En el campo de la visión por computadora y el procesamiento de imágenes, la segmentación semántica desempeña un papel importante al permitir la identificación y delimitación precisa de objetos y/o regiones de interés en imágenes. Esta capacidad tiene aplicaciones importantes en diversos campos, desde la medicina, la robótica y hasta la agricultura. En este contexto, las redes neuronales convolucionales han demostrado ser herramientas importantes para abordar problemas de la segmentación semántica.

A través de los años han surgido numerables arquitecturas para avanzar en la solución de esta problemática. Una de las arquitecturas más destacadas para la segmentación semántica es U-Net, propuesta por Ronneberger en 2015 [28]. U-Net ha sobresalido por su eficacia en la segmentación de imágenes biomédicas, donde la precisión y la capacidad de capturar detalles sutiles son fundamentales para aplicaciones clínicas y de investigación. Sin embargo, su utilidad se extiende más allá del campo biomédico y ha sido aplicada con éxito en una variedad de dominios, incluyendo la visión por computadora general.

La arquitectura U-Net [28] se caracteriza por su diseño distintivo en forma de U, compuesta por un camino de contracción (encoder) y un camino de expansión (decoder). Estos componentes trabajan en conjunto para capturar información contextual a diferentes escalas y generar segmentaciones precisas conservando detalles importantes de la imagen original. La conexión “skip” entre las capas del encoder y del decoder es una de las características clave de U-Net, permitiendo la preservación de información detallada durante el proceso de up-sampling.

La presente tesis propone explorar y analizar las aplicaciones de la red neuronal U-Net en la segmentación semántica de imágenes biomédicas, también su potencial en otros campos de la visión por computadora. Se examinarán estudios relevantes que emplean esta arquitectura, evaluando su rendimiento, ventajas y limitaciones en diferentes contextos. Además, se llevará a cabo un análisis detallado de la arquitectura y funcionamiento de U-Net, proporcionando una comprensión de sus mecanismos.

La tesis se sitúa en la intersección entre la visión por computadora, el aprendizaje profundo y la aplicación práctica en campos como la medicina, la biología y la ingeniería, con el objetivo de aprovechar el potencial de U-Net para abordar desafíos de segmentación semántica en diversas aplicaciones.

1.1. Planteamiento del problema

En el campo de la segmentación de imágenes radiológicas, se enfrenta el desafío de obtener resultados precisos y eficientes para apoyar la toma de decisiones clínicas. A pesar de los avances en técnicas de aprendizaje profundo, para la arquitectura U-Net aún existen áreas de mejora y desafíos por abordar.

Uno de los problemas principales radica en la necesidad de optimizar la segmentación de imágenes médicas para obtener resultados más precisos y rápidos, especialmente en entornos clínicos donde la eficiencia y la exactitud son críticas para el diagnóstico y tratamiento de enfermedades. La complejidad de las redes neuronales convolucionales utilizadas en la segmentación de imágenes puede afectar el rendimiento y la escalabilidad de los modelos, lo que limita su aplicación en entornos con recursos limitados.

Además, la generalización de los modelos de segmentación a diferentes tipos de imágenes médicas y la adaptabilidad a nuevas aplicaciones clínicas siguen siendo desafíos importantes a abordar. La capacidad de los modelos de segmentación para trabajar con diversos conjuntos de datos y adaptarse a diferentes contextos clínicos es fundamental para su utilidad práctica y su impacto en la atención médica.

Por lo tanto, es necesario investigar y desarrollar estrategias innovadoras que mejoren la precisión, eficiencia y generalización de la segmentación de imágenes biomédicas, con el objetivo de

superar los desafíos actuales y avanzar hacia soluciones más efectivas y aplicables en entornos clínicos diversos. Esta tesis de licenciatura propone abordar estos desafíos y contribuir al avance en la segmentación de imágenes médicas para mejorar la atención y el diagnóstico en el campo biomédico.

1.2. Motivación

La segmentación de imágenes médicas es un área de investigación novedosa y a tomado gran importancia en el campo de la salud, actualmente esta jugando un papel fundamental en el diagnóstico, tratamiento y seguimiento de enfermedades. La aplicación de técnicas de aprendizaje profundo, como la arquitectura U-Net, ha revolucionado la forma en que se abordan los desafíos de la segmentación de imágenes biomédicas, permitiendo una identificación precisa de estructuras anatómicas y patológicas en imágenes complejas.

En esta tesis presentaremos una visión integral de las últimas investigaciones y avances en el uso de U-Net para la segmentación de imágenes médicas. Desde la presentación de U-Net como una red neuronal convolucional profunda que mejora significativamente el rendimiento en tareas de segmentación, hasta la propuesta de un modelos más avanzados o mejorados.

Además, la aplicación de U-Net en la investigación radiológica para mejorar el cuidado del paciente destaca la relevancia y el impacto positivo que esta tecnología puede tener en la

práctica clínica. Por lo tanto, una tesis de licenciatura que se enfoque en la aplicación de U-Net en la segmentación de imágenes médicas no solo sería una oportunidad para explorar un campo en constante evolución, sino que también podría contribuir al avance de la ciencia y la tecnología médica.

Esta investigación podría abrir nuevas perspectivas sobre cómo mejorar la precisión, la eficiencia y su uso en entornos con recursos limitados, brindando beneficios tangibles tanto para los profesionales de la salud como para los pacientes, además de reafirmar el conocimiento sobre redes neuronales e inteligencia artificial.

1.3. Justificación

La segmentación de imágenes médicas permite la identificación y delimitación precisa de estructuras anatómicas y patológicas en imágenes biomédicas. El uso de técnicas de aprendizaje profundo, como la arquitectura U-Net, ha demostrado ser altamente efectivo en este tipo de tareas, proporcionando resultados precisos y eficientes.

Se discuten variantes de redes convolucionales como ResNet e igualmente las variantes de U-Net, como DoubleU-Net, U-Net++, R2U-Net y modelos explicativos, que han mostrado mejoras y métricas favorables en la segmentación de imágenes médicas como lo son Rayos-X, Tomografías, etc.

Además, se destaca la importancia de fusionar información

de diferentes modalidades para mejorar la segmentación, así como la necesidad de mejorar la interpretabilidad y generalización de los modelos aplicados a imágenes médicas. Se mencionan avances en la arquitectura U-Net, como módulos de atención y estructuras residuales, que han permitido la segmentación precisa de diversas lesiones, lo cual es fundamental para el diagnóstico y tratamiento de enfermedades.

Por lo tanto, una tesis de licenciatura que investigue la aplicación de la arquitectura U-Net en la segmentación de imágenes médicas sería altamente relevante y oportuna. Esta investigación podría explorar cómo las variantes de U-Net pueden mejorar la precisión y generalización de los modelos, así como su aplicabilidad en la investigación radiológica para mejorar el cuidado del paciente.

1.4. Objetivos

1.4.1. Objetivo general

Evaluar el rendimiento de la red convolucional U-Net en la segmentación de imágenes radiológicas, específicamente radiografías de tórax.

Implementar las formas de construir la Red Neuronal Convolucional U-Net, sus posibles mejoras y compararlas para obtener una mejora en la arquitectura.

1.4.2. Objetivos particulares

- Investigar, analizar y seleccionar una arquitectura variante de U-Net para la segmentación de imágenes radiológicas del tórax.
- Evaluar los resultados de la red convolución U-Net para compararlos con otros métodos de segmentación de imágenes y su costo computacional.
- Validar el algoritmo utilizado para la segmentación.

1.5. Metodología propuesta

La metodología propuesta para esta tesis incluye la obtención de los datos, preprocesamiento, procesamiento y posprocesamiento de los datos; y para ello se propone el diagrama de la metodología propuesta, Figura 1.1

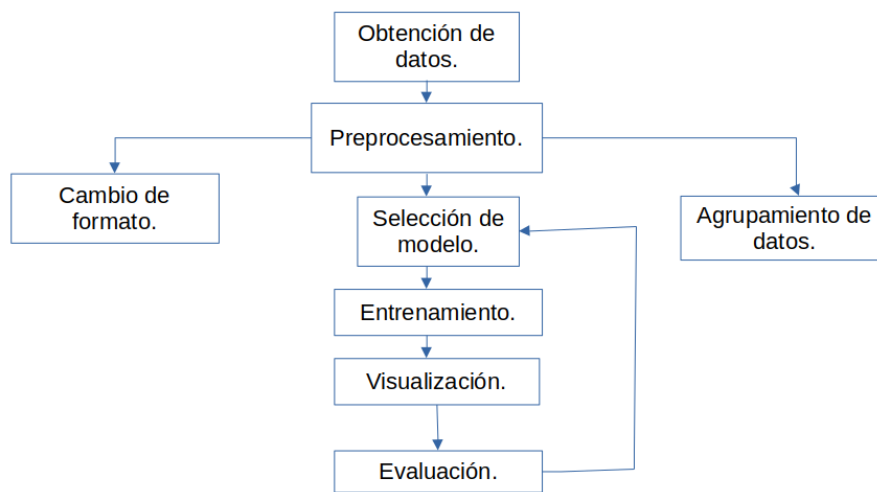


Figura 1.1: Metodología propuesta.

- Fase 1: Encontrar un dataset con imágenes que cumplan los requisitos de tamaño y su respectiva mascara.

- Fase 2: Preprocesamiento.
 1. Dividir dataset para entrenamiento.
 2. Organizar el dataset separando las radiografías con su respectiva mascara.
 3. Aumento de datos para hacer un mejor entrenamiento.
 4. Se decodifica la imagen JPEG en formato tensorial, asegurando que tenga 3 canales de color (RGB).
 5. Convertir la máscara en una imagen binaria, asegurando que todos los valores sean 0 o 1.
- Fase 3: Selección de la arquitectura para entrenar.
- Fase 4: Entrenamiento con diferentes cantidades de épocas.
- Fase 5: Resultados y conclusiones.

1.6. Organización de la tesis

La presente tesis se encuentra organizada en 5 capítulos, cuyo contenido se describe a continuación:

- Capítulo 1 presenta la introducción, justificación y los objetivos.
- Capítulo 2 presenta el estado del arte.
- Capítulo 3 explica los conceptos necesarios para la comprensión del trabajo y las herramientas requeridas para desarrollarlo.
- Capítulo 4 presenta los resultados obtenidos de las pruebas de segmentación de imágenes.

- Capitulo 5: Conclusiones en esta sección se hace una breve descripción de los resultados de la tesis.

Esta tesis contribuye en el entendimiento y la aplicación de U-Net como una herramienta efectiva para la segmentación semántica de imágenes en diversos escenarios. A través de la exploración de casos de estudio y experimentos prácticos, se pretende ofrecer una visión de las capacidades y posibles áreas de mejora de esta arquitectura, así como también identificar posibles direcciones futuras de investigación.

Capítulo 2

Antecedentes

2.1. Redes convolucionales

Las redes neuronales convolucionales (CNN, por sus siglas en inglés, Convolutional Neural Network) son un tipo de red neuronal artificial diseñada específicamente para procesar datos con una estructura de cuadrícula. El ejemplo más popular son las imágenes.

Una CNN es una arquitectura que aprende directamente a partir de datos [24]. Son particularmente útiles para identificar patrones en imágenes con el fin de reconocer objetos, clases y categorías [21]. Estas redes pueden tener decenas o cientos de capas, donde cada una aprende a detectar diferentes características de una imagen aplicando filtros a las imágenes de entrenamiento con distintas resoluciones [33]. Las CNN constan de capas de convolución, activación, y agrupación, su funcionamiento se basa en el procesamiento de datos para comprender sus características particulares y proporcionar una salida de clasificación [24]. Las neuronas de una red convolucional corresponden a campos receptivos de manera similar a las neuronas en la corteza visual primaria de un cerebro biológico, lo que las hace altamente efectivas en tareas de visión

artificial, como la clasificación y segmentación de imágenes [16].

Se han logrado avances significativos en tareas como la clasificación de imágenes, la detección de objetos y la segmentación semántica.

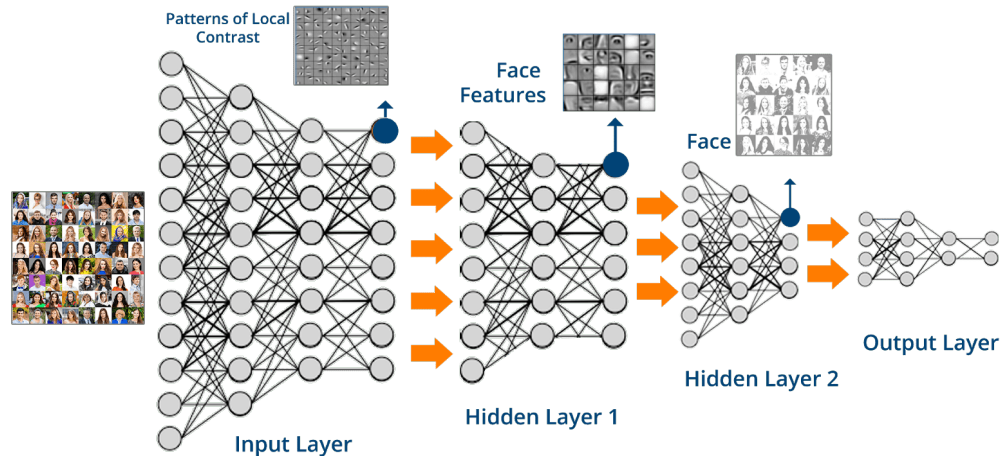


Figura 2.1: Arquitectura de una Red Neuronal convolucional clásica con 2 capas ocultas de Neuronas[10].

La figura 2.1 muestra un tipo de CNN que identifica rostros, donde en la capar de entrada cada neurona representa un valor de un pixel, la primera capa extrae características de bajo nivel para después ser agrupadas y halla una combinación de características, las conexiones entre capas representan los pesos que se asigna a las conexiones entre capas para una vez entrenada tome las mejores decisiones para identificar el rostro.

2.1.1. ResNet

Redes Residuales, también conocida como ResNet, es una de las muchas variedades de redes convolucionales. Puede tener

diferentes profundidades desde 18, 34, 50, 101, 152 capas y más. Cada aumento en la profundidad se logra añadiendo más bloques residuales [15]. A pesar de su profundidad, ResNet tiene una complejidad computacional menor en comparación con otras arquitecturas profundas, como VGG19[42] (de la cual se habla más adelante), lo que la hace más eficiente en términos de recursos computacionales.

La arquitectura de ResNet inspirada en “Plain Network” se compone de bloques de capas que incluyen conexiones de atajo (shortcut connections) para facilitar el aprendizaje de funciones residuales. Los componentes clave de su arquitectura incluyen los bloques residuales, que consisten en dos o más capas convolucionales, generalmente con filtros de 3×3 , y una conexión de atajo que suma la entrada del bloque a la salida de las capas convolucionales, permitiendo que la red aprenda la diferencia entre la entrada y la salida deseada. Las conexiones de atajo pueden ser de identidad cuando las dimensiones de entrada y salida son las mismas, o pueden incluir una transformación, como convoluciones, si las dimensiones cambian, asegurando así que la información fluya a través de la red sin ser alterada[15].

La red también utiliza capas de convolución para extraer características de las imágenes, empleando en la arquitectura básica capas de convolución de 3×3 . Además, se utiliza el pooling, o submuestreo, para reducir la dimensionalidad de las características extraídas, lo que ayuda a disminuir la carga computacional y a controlar el sobreajuste. Finalmente, al final de la red, se emplea una capa de pooling global seguida de una

capa completamente conectada que produce las probabilidades de clasificación para las diferentes clases[15].

2.1.2. U-Net 2015

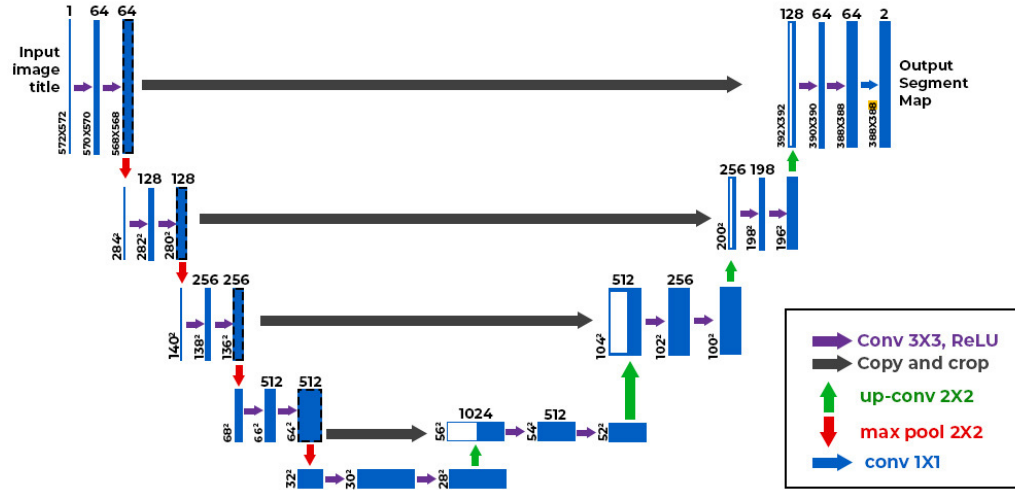


Figura 2.2: Estructura de U-Net, (ejemplo para 32x32 píxeles en la resolución más baja). Cada cuadro azul corresponde a un mapa de características de múltiples canales. El número de canales se indica en la parte superior del cuadro. El tamaño x-y se proporciona en el borde inferior izquierdo del cuadro. Los cuadros blancos representan mapas de características copiados. Las flechas indican las diferentes operaciones [28].

U-Net es una red neuronal diseñada para segmentar imágenes, especialmente en campos como la medicina. Su estructura en forma de U (Figura 2.2) se divide en una parte descendente (encoder) que extrae características importantes de la imagen, y una parte ascendente (decoder) que reconstruye la segmentación. Algo especial de U-Net son las conexiones residuales, que conectan capas del encoder con capas correspondientes del decoder, preservando detalles importantes de la imagen original. Esto ayuda a mejorar la precisión de la segmentación al evitar

la pérdida de información durante el proceso de reconstrucción.

U-Net presenta varias ventajas para la segmentación de imágenes médicas. En primer lugar, su capacidad para extraer características de alta dimensión le permite identificar con precisión detalles y patrones complejos en las imágenes médicas. Esto es fundamental para obtener resultados precisos en aplicaciones médicas donde la precisión es crucial para el diagnóstico y tratamiento adecuado [28] [9]. Otra ventaja es que hace un uso intensivo del aumento de datos, lo que le permite entrenar eficazmente la red con un número limitado de muestras anotadas. Esta estrategia mejora la generalización del modelo, combinada con su rapidez y eficiencia tanto en el entrenamiento como en la predicción, la convierte en una opción adecuada para aplicaciones en tiempo real o con grandes conjuntos de datos. Por último, emplea conexiones residuales que saltan capas, lo que ayuda a mitigar el problema de desvanecimiento del gradiente y facilita el flujo de información a través de la red.

U-Net ofrece características únicas que la hacen altamente efectiva en el campo de la segmentación. Aunque en esta tesis se utilizará para imágenes radiológicas, se tiene registro de una gran variedad de usos, como la segmentación de células individuales [4], lo cual es crucial para aplicaciones médicas como el seguimiento y análisis de células en imágenes. Su potencial para aplicaciones clínicas, como la segmentación de imágenes de TC (Tomografía Computada) de pulmón en pacientes con COVID-19, promete contribuir significativamente a la comprensión y tratamiento de diversas condiciones médicas. U-Net se destaca

como una herramienta poderosa y efectiva para la segmentación de imágenes médicas, con aplicaciones prometedoras en la medicina y la investigación biomédica [28].

2.2. Avances en U-Net

U-Net original ha sido objeto de numerosas mejoras y variaciones dentro de la comunidad de investigación en aprendizaje profundo y visión por computadora. Estas mejoras y variaciones se enfocan principalmente en aspectos como la arquitectura de red, las capas convolucionales, los módulos de atención y las técnicas de regularización, con el objetivo de potenciar tanto el rendimiento como la generalización del modelo resultante.

Para esta tesis se implementará U-Net utilizando el lenguaje de programación Python debido a su versatilidad. Además, Python cuenta con bibliotecas y frameworks especializados en aprendizaje profundo y procesamiento de imágenes, como TensorFlow, PyTorch y Keras, que facilitan el desarrollo de una CNN.

Algunas CNN basadas en U-Net

- Double U-Net.
- V-Net.
- Nested U-Net.
- Attention U-Net.

- Recurrent U-Net.
- Residual U-Net.
- Dense U-Net.
- Attention Gate U-Net.
- SE U-Net (Squeeze-and-Excitation U-Net).
- U-Net++.
- R2U-Net (Recurrent Residual U-Net).

2.2.1. DoubleU-Net

La arquitectura de DoubleU-Net incorpora dos modelos U-Net en secuencia, con dos codificadores y dos decodificadores. El primer codificador en DoubleU-Net se basa en el modelo pre-entrenado VGG-19 [42] que proporciona una base sólida para la extracción de características, como se muestra en la figura 2.3, esto puede ayudar a mejorar el rendimiento y la eficiencia del modelo DoubleU-Net en tareas de segmentación de imágenes al proporcionar una representación inicial de alta calidad de las imágenes antes de continuar con las capas de decodificación y upsampling, además, DoubleU-Net incluye Atrous Spatial Pyramid Pooling (ASPP) para capturar información contextual a múltiples escalas. Estas mejoras arquitectónicas contribuyen a una mejoría en el rendimiento de segmentación en comparación con U-Net tradicional [17].

Siendo VGG-19 un modelo de red neuronal profunda [42] que ha sido pre-entrenado en un gran conjunto de datos, como ImageNet, para tareas de clasificación de imágenes. Este modelo se caracteriza por su profundidad (contiene 19 capas) y su arquitectura relativamente simple pero efectiva, que consiste principalmente en capas convolucionales apiladas seguidas de capas de pooling y activación.

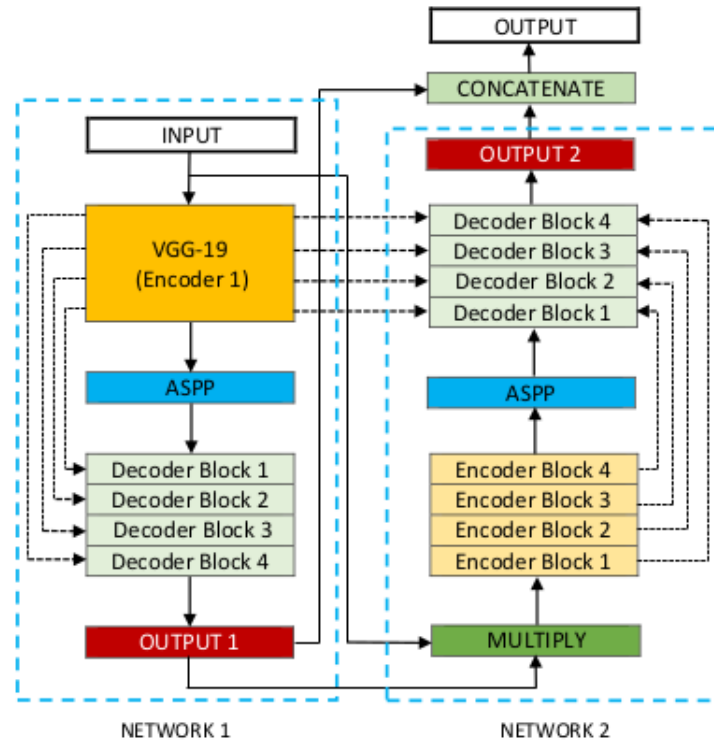


Figura 2.3: La estructura de Double U-Net es una combinación de dos arquitecturas: VGG19, ASPP y U-Net [17].

DoubleU-Net se destaca por varias ventajas sobre U-Net tradicional. En primer lugar,[17] se ha evidenciado un mejor rendimiento en diversos conjuntos de datos médicos, donde DoubleU-Net supera a U-Net y otros modelos base en términos de precisión de segmentación. Específicamente, produce máscaras de segmentación más precisas, incluso en imágenes

desafiantes con detalles diminutos como pólipos pequeños y planos, lo que resalta su capacidad para abordar escenarios clínicamente relevantes.

Además, se ha observado que DoubleU-Net exhibe una generalización mejorada y mayor robustez, lo que lo hace apropiado para pruebas de evaluación cruzada en diferentes conjuntos de datos. Este aspecto sugiere que puede servir como una línea base sólida para la segmentación de imágenes médicas y pruebas de generalización de modelos de aprendizaje profundo, proporcionando así resultados más consistentes y fiables en una variedad de contextos clínicos.

2.2.2. UNet ++

La arquitectura de UNet++ representa una mejora significativa sobre U-Net al incorporar conexiones de salto densas y anidadas para minimizar la brecha semántica entre las características del encoder y del decoder. El diseño de UNet++ sigue la estructura fundamental de U-Net [43], con un encoder encargado de capturar características a diferentes niveles de abstracción y un decoder que reconstruye la imagen segmentada utilizando estas características.

Las conexiones de salto densas y anidadas permiten una fusión de características entre el encoder y el decoder [43], [31] como en "Fully Convolutional Network" (FCN), los mapas de características muestreados ascendentes se suman con los mapas de características omitidos del codificador, mientras que

U-Net los concatena y agrega convoluciones y no linealidades entre cada paso de muestreo ascendente.

Estas conexiones de salto ayudan a capturar detalles finos y mejorar la precisión de la segmentación al reducir la pérdida de información durante la propagación hacia atrás. Al combinar información semántica y de apariencia [31], la arquitectura de salto permite que la red aprenda a capturar tanto la estructura general de la imagen como los detalles específicos necesarios para la segmentación precisa. Otra característica importante es la implementación de la supervisión profunda, que consiste en agregar capas de supervisión adicionales para guiar el proceso de optimización y mejorar la precisión de la segmentación (Figura 2.4).

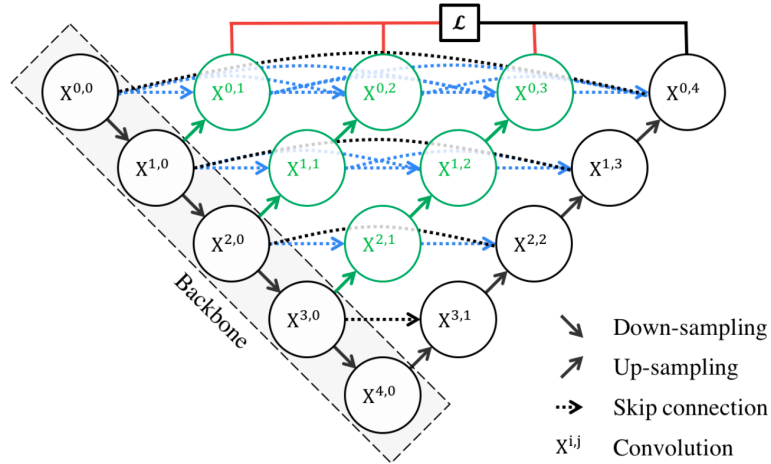


Figura 2.4: U-Net++ sigue el diseño de U-Net (backbone), mientras que los mapas de características muestreados ascendentes se suman con los mapas de características omitidos del codificador [43]. b) Las conexiones de salto rediseñadas que conectan las dos subredes, representadas en verde y azul, son remodeladas para mejorar la transferencia de información entre el encoder y el decoder.

Las diferencias clave entre UNet++ y los modelos tradicionales U-Net y FCN [44] son notables y pueden influir signifi-

cativamente en su desempeño en la segmentación de imágenes médicas. En primer lugar, U-Net y FCN están limitados por una profundidad fija, lo que puede afectar su capacidad para manejar la variabilidad en los datos y los objetos de interés [44]. Segundo una mejora en la capacidad del modelo para realizar una segmentación precisa y detallada en diversas aplicaciones [43].

Estas diferencias resaltan las ventajas y mejoras significativas que UNet++ ofrece en el campo de la segmentación de imágenes médicas en comparación con los enfoques tradicionales como U-Net y FCN.

2.3. Procesamiento digital de imágenes

2.3.1. Definiciones

Se presentan los conceptos básicos que son usados en el procesamiento digital de imágenes.

Píxel

El píxel es la unidad mínima que forma una imagen digital [38], la cual puede ser interpretada por monitores, pantallas de celulares, entre otros equipos. Se puede decir que, entre mayor sea la cantidad de píxeles en una imagen, mejor será la calidad de la imagen.

Imagen digital

Una imagen digital o gráfico digital es una representación bidimensional de una imagen a partir de una matriz numérica [37] $M(i, j)$ donde i y j representan la ubicación de los píxeles.

Resolución

La resolución de una imagen indica la cantidad de detalles que puede observarse en esta [40]. La resolución de una imagen esta relacionado con la cantidad de píxeles que la describen como se ve en la figura 2.5.

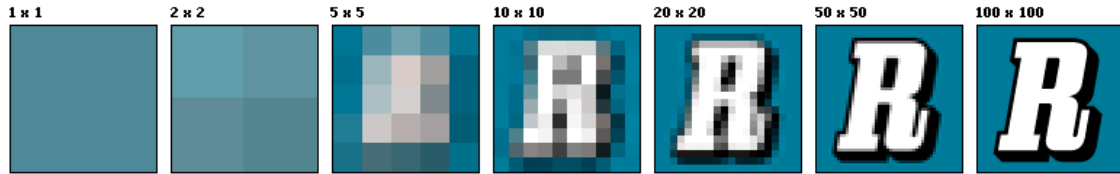


Figura 2.5: Una imagen con diferentes resoluciones aumenta el número de píxeles [40].

Segmentación

La segmentación es un proceso de clasificación por píxel que asigna una categoría a cada píxel de la imagen analizada[41] con el fin de separar las regiones de interés del resto de la imagen [22]. Los algoritmos de segmentación de imágenes monocromáticas se basan en dos propiedades principales de los valores a escala de grises como se muestra en la figura 2.6 : Similitud y discontinuidad [12]:

- Similitud: Consiste en dividir una imagen basándose en el cambio brusco de los niveles de grises de la imagen, como puede ser la detección de bordes de una imagen.
- Discontinuidad: Se basa en dividir la imagen en regiones que son similares de acuerdo a criterios predefinidos, algunos ejemplos son la umbralización, crecimiento de regiones y fusión de regiones.

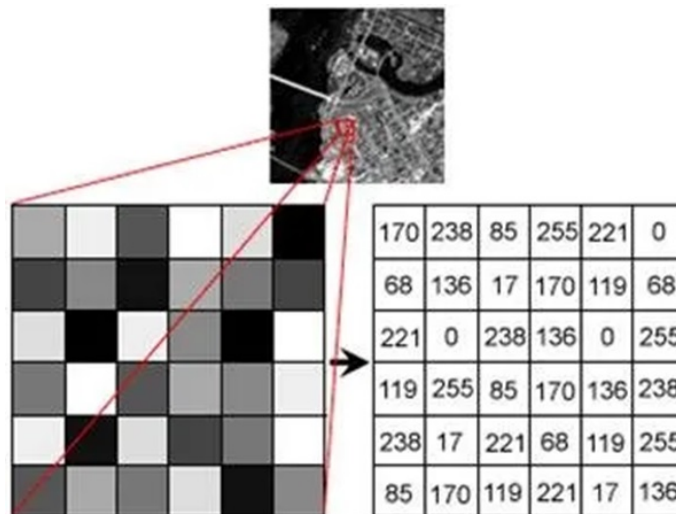


Figura 2.6: Esta fotografía es una imagen (las imágenes son matrices formadas por elementos discretos), donde cada punto más pequeño se representa un píxel [7].

Máscaras

Una máscara es una imagen binaria que indica qué píxeles de la imagen original deben ser considerados o ignorados en un proceso particular. Los píxeles de la máscara tienen valores típicamente de 0 (negro) y 1 (blanco), donde 1 indica las áreas de interés y 0 indica las áreas a ignorar.

2.3.2. Procesamiento

Una radiografía es una imagen registrada en una placa o película fotográfica, o de forma digital (radiología digital directa o indirecta) en una base de datos. Esta imagen se obtiene al exponer al individuo a una fuente de radiación ionizante, un objeto entre la fuente de radiación y el receptor, las partes más densas del objeto aparecen con diferentes tonos dentro de una escala de grises [39].

La escala de grises se refiere a que estos píxeles contienen una escala de tonos de grises con 256 niveles diferentes, re-

presentados en 8 bits; un píxel negro indica la ausencia total de color, representado por “0”, mientras que un píxel blanco indica la presencia total de color, representado por “255”.

2.3.3. Imágenes DICOM

DICOM (Digital Imaging and Communications in Medicine) [8] es un estándar internacional para almacenar e intercambiar imágenes médicas y sus metadatos entre dispositivos como escáneres, equipos informáticos y servidores. Es el estándar más frecuente para almacenar imágenes en el campo médico y permite capturar y almacenar volúmenes de imágenes y datos de pacientes en un solo almacén de datos. Las imágenes guardadas en formato DICOM tienen un alto rango dinámico (hasta 16 bits) y son compatibles con muchos fabricantes de software y dispositivos médicos [26] [6].

MATLAB y su herramienta Image Processing Toolbox permiten acceder y trabajar directamente con imágenes médicas en formato DICOM, y existen visores DICOM en línea como DICOM Library que permiten compartir, explorar y analizar archivos DICOM.

Dado que se utiliza el lenguaje de programación Python, es conveniente que las imágenes tengan un formato en mapa de bits como PNG y JPG. Estos formatos de imagen son ampliamente soportados por navegadores web, sistemas operativos y aplicaciones, lo que facilita la visualización y el intercambio de imágenes en múltiples plataformas y contextos.

Para esta tesis, se emplearán datasets en formato JPG, lo cual requiere cambiar de formato (véase el apéndice, código A).

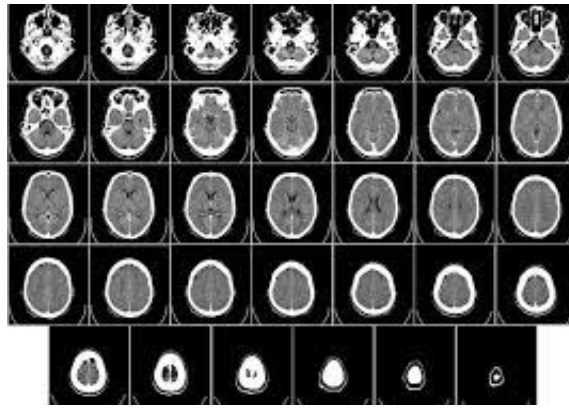


Figura 2.7: Imágenes tipo DICOM, usadas típicamente para formatos médicos[8].

2.3.4. Pydicom

Pydicom es una biblioteca de software desarrollada para la manipulación y análisis de archivos en el formato DICOM (Digital Imaging and Communications in Medicine), que es un estándar global para el almacenamiento y transmisión de información médica e imágenes. Esta biblioteca, implementada en el lenguaje de programación Python, permite a los usuarios leer, modificar y escribir datos en archivos DICOM de manera eficiente y conforme a las convenciones de Python[27].

Pydicom es particularmente valioso en entornos de investigación y desarrollo clínico, donde la precisión y la capacidad de personalización son esenciales [27] .

2.4. Rayos X

Los rayos X son una forma de radiación electromagnética de alta energía, cuya longitud de onda se encuentra aproximadamente en el rango de 0.01 a 10 nanómetros (nm), lo que equivale a energías de fotones entre 100 eV y 100 keV. Este rango los coloca entre la radiación ultravioleta y los rayos gamma dentro del espectro electromagnético.

2.4.1. Producción

- **Rayos X de frenado (Bremsstrahlung):** Cuando electrones acelerados a altas energías (típicamente entre 30,000 y 150,000 volts (30–150 kV) son desacelerados bruscamente al interactuar con núcleos atómicos, emiten radiación continua en forma de rayos X.
- **Rayos X característicos:** Al bombardear un material con electrones de alta energía, se ionizan electrones internos (ej., de la capa K). Cuando un electrón de una capa superior llena el vacante, se emite un fotón con energía igual a la diferencia entre niveles, generando un espectro discreto característico del elemento.

Desde un punto de vista médico, los rayos X se utilizan principalmente para obtener imágenes internas del cuerpo humano sin necesidad de cirugía.

En una máquina de rayos X, se acelera un haz de electrones mediante una diferencia de potencial y se hace colisionar contra un ánodo metálico (generalmente de tungsteno). Al interactuar con los átomos del ánodo, los electrones generan rayos X por los

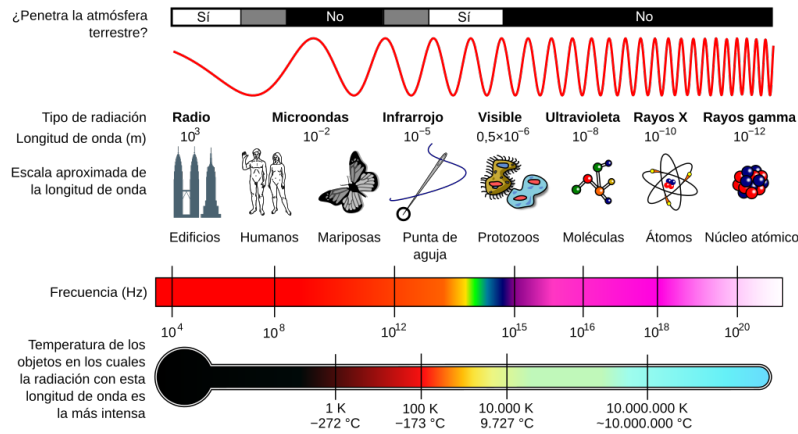


Figura 2.8: Espectro Electromagnético

procesos de frenado (Bremsstrahlung) y transiciones electrónicas internas (rayos X característicos).

Cuando el haz de rayos X atraviesa el cuerpo humano en Tejidos blandos (como músculo, grasa y órganos) tienen baja densidad y número atómico, por lo que absorben pocos rayos X. Estos fotones pasan casi sin ser atenuados en tejidos densos (como huesos o estructuras calcificadas) tienen mayor número atómico y densidad, por lo que absorben más fotones mediante absorción fotoeléctrica.

Detrás del paciente hay un detector (puede ser una placa fotográfica antigua, una película radiográfica, o un sensor digital). Este detector recibe los fotones que no fueron absorbidos por el cuerpo las zonas donde más rayos X pasan (por ejemplo, los pulmones) se ven más oscuras en la imagen, las zonas donde se absorben más rayos X (por ejemplo, los huesos) se ven más claras.

Este contraste genera una imagen en escala de grises que permite visualizar la anatomía interna.

2.5. Neumotórax

Un neumotórax es la entrada de aire en el espacio interpleural, que se encuentra entre la pleura visceral (que rodea los pulmones) y la pleura parietal (que rodea el tórax). Esto provoca un colapso pulmonar, ya que el aire comprime el pulmón y reduce su capacidad para expandirse y oxigenar el cuerpo adecuadamente.

La presión intrapleural es subatmosférica durante todo el ciclo respiratorio, y es el resultado de las fuerzas elásticas opuestas del pulmón y de la pared torácica[34]. La presión intraalveolar es relativamente positiva en relación con la intrapleural, ayudando a mantener el pulmón en expansión durante la respiración. La entrada incidental de aire en el espacio pleural provoca una pérdida de la presión negativa intrapleural y el pulmón tiende a colapsarse por su propia fuerza retráctil, hasta llegar a un nuevo punto de equilibrio entre la presión intrapleural y la presión alveolar[34]. Si la cantidad de aire es importante, la presión pleural se aproxima a la atmosférica, lo que puede causar un colapso de todo el pulmón.

2.5.1. Síntomas

- Dolor torácico repentino y agudo
- Dificultad para respirar
- Respiración superficial y rápida
- Tos
- Cianosis (coloración azulada de la piel)

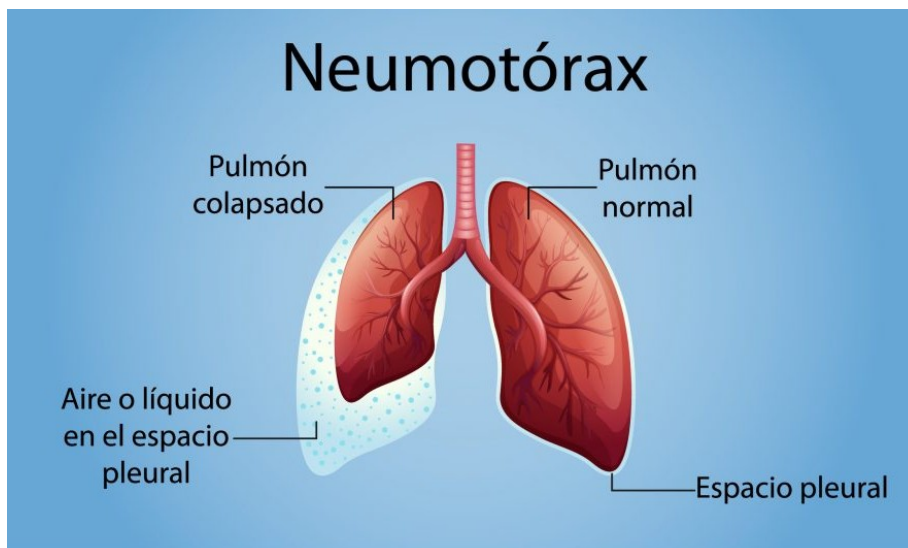


Figura 2.9: Del lado izquierdo hay aire en el espacio pleural mientras que en el derecho no [30].

La presión intrapleural es subatmosférica durante todo el ciclo respiratorio, y es el resultado de las fuerzas elásticas opuestas del pulmón y de la pared torácica[34].

2.5.2. Diagnóstico

Los síntomas y la exploración física permiten hacer un diagnóstico de sospecha [34]. El diagnóstico lo proporciona la radiografía posteroanterior de tórax al identificar la línea de la pleura visceral, con ausencia de trama vascular periférica.



Figura 2.10: Un radiografía de tórax [36].



Figura 2.11: La respectiva mascara de la imagen 2.10, que identifica las zonas que contiene aire marcada de blanco, para un espacio pleural sano tendira que verse negro totalmente ya que no hay aire dentro[36].

Capítulo 3

Marco teórico

3.1. Arquitectura de la red neuronal

Aunque existen diferentes tipos de CNN, lo que diferencia una de otras sería lo que llamamos arquitectura, con esto nos referimos a la forma en la que se acomodan las funciones.

La figura 2.2 muestra la forma y orden en la que las funciones están acomodadas y justo aquí es donde se puede hacer una mejora de la arquitectura.

3.1.1. Convolución.

El mismo nombre de la red neuronal convolucional indica que la red aplica la operación matemática de convolución en al menos una de sus capas.

Sea una convolución definida por la ecuación 3.1, donde f y g son funciones y t un desplazamiento.

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\nu)g(t - \nu)d\nu \quad (3.1)$$

Para el procesamiento de imágenes, se toma la convolución matricial, que se define como la ecuación 3.2 donde se multiplica una matriz A por otra llamada *kernel*.

Dadas la matriz de entrada $A \in \mathbb{R}^{m \times n}$, representada como $A = (a_{i,j})$, y la máscara o kernel de convolución $C \in \mathbb{R}^{(2N+1) \times (2N+1)}$, denotada como $C = (c_{l,s})$, con $2N+1 < m, n$, se define la convolución como

$$D := A * C,$$

donde $D = (d_{i,j})$ es la matriz resultante de aplicar la operación de convolución [11].

$$d_{i,j} = \frac{1}{k} \sum_{l=1}^{2N+1} \sum_{s=1}^{2N+1} a_{(i-N+l-1),(j-N+s-1)} c_{l,s} \quad (3.2)$$

donde $k = \sum_{i,j=1}^{2N+1} c_{i,j}$ es la suma de coeficientes de la matriz, (cuando $c=0$ se toma $c=1$), también para $d_{i,j}$ se define $i = N+1, \dots, (m-N-1)$ y $j = N+1, \dots, (n-N-1)$.

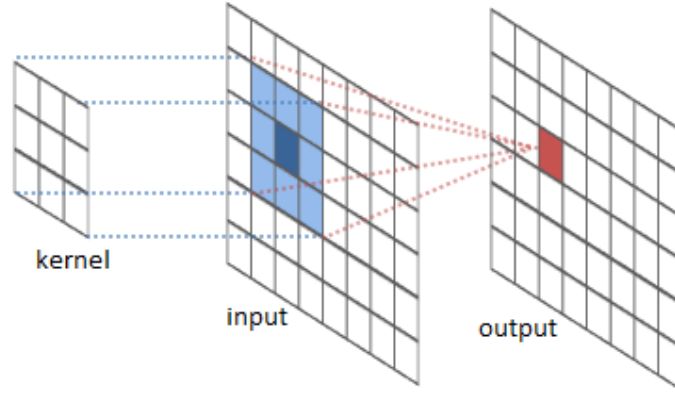


Figura 3.1: La convolución de 2 matrices, dando como resultado una tercera [10].

La Figura 3.1 muestra el proceso de convolución, donde un kernel extrae características de una imagen de entrada. El tamaño del kernel debe ser menor que el de la imagen, y esta última no requiere ser cuadrada. Un caso especial ocurre al

aplicar la convolución en los bordes de la matriz. En estas situaciones, es común asumir que la matriz está rodeada de ceros, aunque también pueden emplearse otras técnicas, como replicar los valores del borde o completarlos mediante simetría [11].

3.1.2. Mapas de características

Un kernel puede ser considerado como un tipo de filtro que permite llevar a cabo tareas como detectar bordes, líneas verticales y horizontales en imágenes, también de desenfocar o enfocar la imagen. El resultado de estas operaciones se conoce como mapas de características o mejor conocido en inglés como *feature maps*.

Los *feature channels* hace referencia a la profundidad del tensor cuyas capas están hechas de los *feature map* [28].

Enfoque	Desenfoque	Realce de bordes	Repujado
$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} -2 & -1 & 0 \\ -1 & 1 & 1 \\ 0 & 1 & 2 \end{bmatrix}$
Detección de bordes	Filtro de tipo Sobel	Filtro de tipo Sharpen	
$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & -2 & 1 \\ -2 & 5 & -2 \\ 1 & -2 & 1 \end{bmatrix}$	
Filtro Norte	Filtro Este	Filtro de tipo Gauss	
$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -2 & 1 \\ -1 & -1 & -1 \end{bmatrix}$	$\begin{bmatrix} -1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & 1 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 2 & 3 & 1 & 1 \\ 2 & 7 & 11 & 7 & 2 \\ 3 & 11 & 17 & 11 & 3 \\ 2 & 7 & 11 & 7 & 2 \\ 1 & 2 & 3 & 2 & 1 \end{bmatrix}$	

Figura 3.2: Algunos de los tipos de kernels más usados son los que generan los mapas de características [10].

Por ejemplo, si se introduce una imagen en formato RGB (por ejemplo, de tamaño $224 \times 224 \times 3$), los tres canales iniciales

corresponden a los colores Rojo, Verde y Azul.

3.1.3. Función de activación

Su propósito principal es introducir no linealidad en el proceso de aprendizaje, permitiendo a la red aprender y representar patrones complejos en los datos [14].

Cada neurona recibe una entrada, que es una combinación ponderada de las salidas de las neuronas en la capa anterior. La función de activación se aplica a esta entrada para determinar si la neurona debe activarse o no, y en qué grado. Esta toma la entrada de la neurona, que puede ser un valor positivo o negativo, y la transforma en un valor de salida, generalmente entre 0 y 1, o entre -1 y 1. Este valor de salida representa la intensidad de la activación de la neurona [3].

Existen diferentes tipos de funciones de activación, como Sigmoide, Tangencial, entre otras. La que se utiliza para esta tesis es la función ReLU, dada por la ecuación 3.3.

$$ReLU(x) = \begin{cases} 0 & \text{si } x \leq 0 \\ x & \text{si } x > 0 \end{cases} \quad (3.3)$$

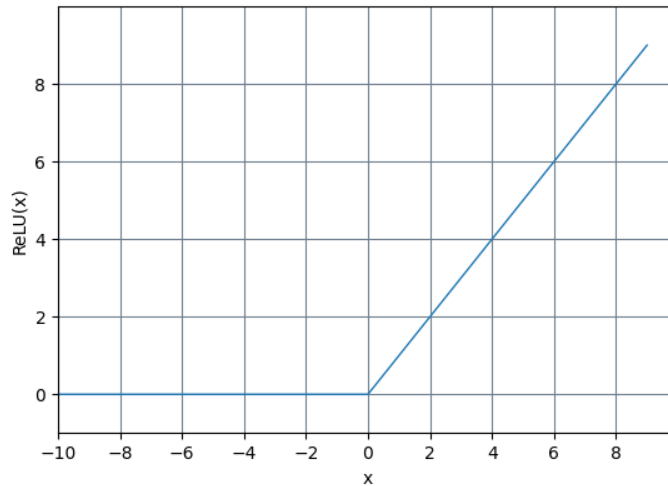


Figura 3.3: Función de activación ReLu.

3.1.4. Max pooling

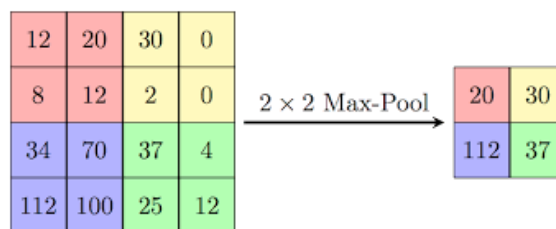


Figura 3.4: La aplicación del Max Pool de 2x2 se puede ver como un método para extraer y resumir las características más importantes de la imagen. [5].

Esta operación reduce las dimensiones espaciales (ancho y alto) de los mapas de características generados por capas convolucionales, como muestra la Figura 3.4. Para ello, extrae el valor máximo de cada región cuadrada (por ejemplo, de 2×2 píxeles) y genera una matriz más pequeña, en este caso, la mitad del original. Así, se disminuye el número de parámetros y el costo computacional de la red, además de mitigar el sobreajuste.

3.1.5. Stride

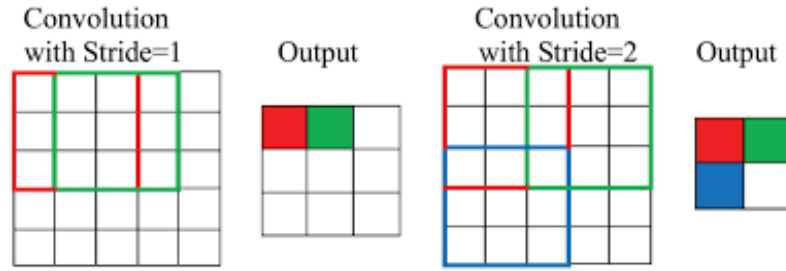


Figura 3.5: Comparativa entre stride=1 y un stride=2 [18].

El stride (o salto) se refiere al número de píxeles por los que se desplaza el kernel al realizar la operación de convolución. Como el ejemplo de la figura 3.5, la función determina cómo se mueve el filtro a través de la imagen de entrada y afecta directamente el tamaño del mapa de características resultante.

3.1.6. Padding

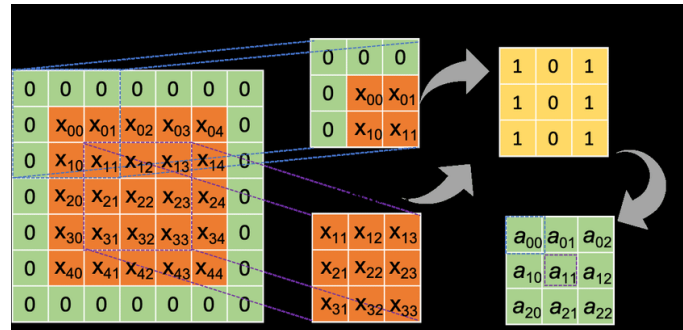


Figura 3.6: Como funciona el padding tras una convolución.

El padding es el proceso de agregar elementos adicionales, generalmente ceros, alrededor de la imagen de entrada antes de aplicar la convolución. Este proceso actúa como un borde protector alrededor de la imagen, lo que permite mantener las dimensiones espaciales durante la operación de convolución y

evitar la pérdida de información en los bordes[25].

Ciertamente, después de una convolución, las imágenes de salida son más pequeñas que las de entrada. Este proceso sigue una regla básica que se aplica en el procesamiento de imágenes.

- I_1 -Tamaño de la imagen de entrada.
- I_2 -Tamaño de la imagen de salida.
- k -Tamaño del Kernel.
- P - Padding.
- S -stride.

$$I_2 = 1 + \frac{I_1 - k + 2(P)}{S} \quad (3.4)$$

3.1.7. Sesgo

El término *Sesgo* (Bias en ingles) es un parámetro adicional que se añade a la salida de una neurona después de aplicar la función de activación. Los bias desempeñan un papel importante en el ajuste de la función de activación, lo que permite a la red neuronal modelar datos de manera más flexible y precisa.

El sesgo para redes neuronales ordinarias viene dado por:

$$y_i = f\left(\sum_i w_i * x_i + b\right) \quad (3.5)$$

- y_i es la neurona de salida.

- f es la función de activación.
- w_i los pesos.
- x_i son las entradas a la neurona.
- b sesgo.

La ecuación 3.5 se reescribe para CNN:

$$y_{i,j} = f\left(\sum_{n,m} (w_{n,m} * x_{i+n,j+m}) + b\right) \quad (3.6)$$

- (i, j) son las posiciones en los mapas de características.
- (n, m) las posiciones en los filtros.

3.1.8. Camino de expansión

Para el camino de expansión ocupa una función *Up-Conv* se refiere a convolucion para arriba y una concatenación, también llamada convolución transpuesta. Su función principal es realizar un proceso de up-sampling, es decir, aumentar la resolución de una imagen o de un mapa de características.

La convolución transpuesta se utiliza para invertir el efecto de la convolución estándar. Mientras que la convolución normal reduce la dimensión de los datos, la convolución transpuesta aumenta la dimensión. Esto se logra aplicando un filtro en la dirección opuesta a la de la convolución convencional, permitiendo que la red neuronal aprenda a rellenar detalles en la imagen generada o en el mapa de características [19].

3.1.9. Mapas de segmentación

En problemas de clasificación a nivel de píxel (segmentación semántica), la salida de la red es un mapa de segmentación donde cada píxel tiene un valor asociado a una clase específica. Si bien esta tesis se enfoca en otro tipo de resultados, esta aplicación representa una de las tareas más comunes para las CNN. Aunque esta tesis esta enfocada en otro tipo de resultados, suele ser unas de las aplicaciones mas comunes para CNN.

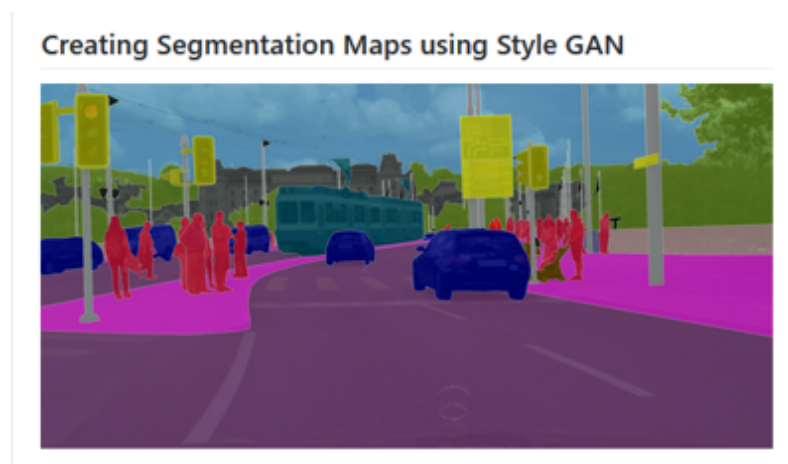


Figura 3.7: Una mapa de segmentación hecho con Style GAN [32].

Como se observa en la figura 3.7, el mapa de segmentación asigna colores distintos a cada categoría (por ejemplo, azul para coches, verde para árboles y gris para edificios), permitiendo una identificación precisa de los objetos en la imagen.

3.2. Entrenar el modelo

En el contexto de CNN existen 2 tipos de aprendizajes **supervisado** y **no supervisado** esto se refiere a si los datos de entrada están etiquetados y categorizados o caso contrario.

Para esta tesis, el tipo de aprendizaje es supervisado dado que se utilizan las imágenes de tórax y sus respectivas máscaras, posteriormente se etiquetan todos los datos para finalmente ser entrenada mediante el método de descenso de gradiente, el cual posee tres variantes muy comunes para el entrenamiento de algoritmos de aprendizaje automático: mini-batch, batch y estocástico, las cuales tiene como objetivo principal encontrar el mínimo de la función de costo con el uso de las derivadas.

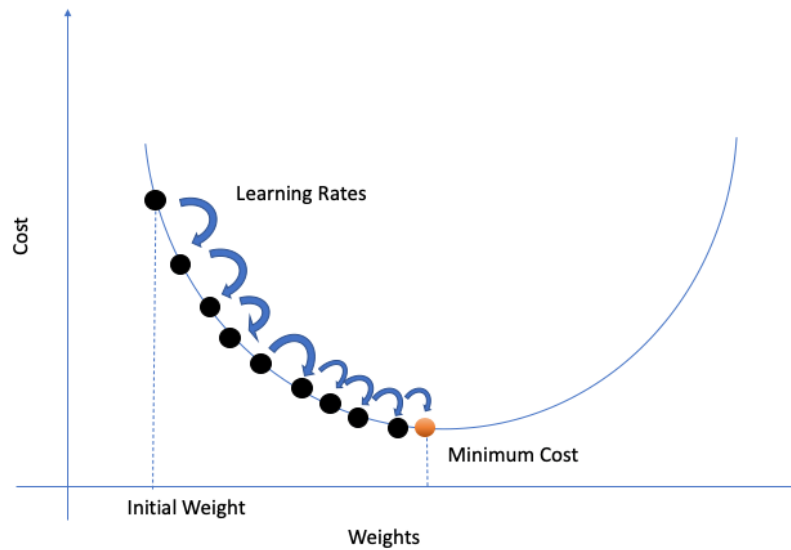


Figura 3.8: Descenso de gradiente como encuentra el mínimo de la función entre los pesos contra la función de costo.

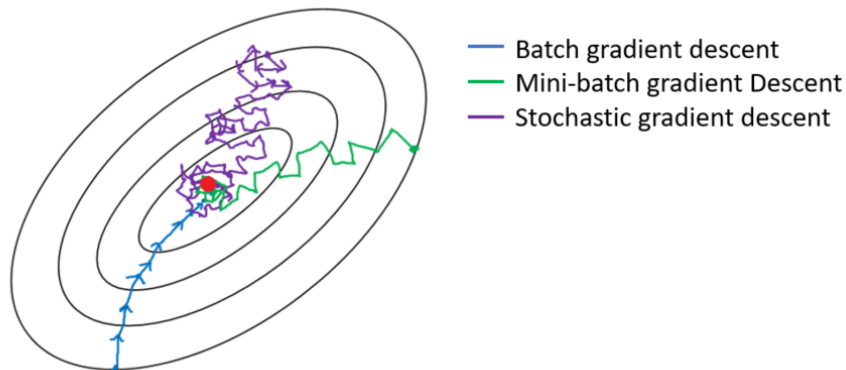


Figura 3.9: Comparativa entre las diferentes maneras de encontrar el mínimo con el algoritmo gradiente descendente [1].

Para U-Net [28], se emplea la variante estocástica, conocida como Gradiente Descendente Estocástico (SGD, por sus siglas en inglés). Esta elección resulta ventajosa dado el volumen de datos utilizado, ya que SGD procesa un ejemplo a la vez, reduciendo el costo computacional en comparación con métodos por lotes (batch). Además, la selección del método de entrena-

miento (SGD, mini-batch o batch) depende de factores como: (1) el tamaño del conjunto de datos, (2) los recursos computacionales disponibles, y (3) la arquitectura del modelo.

3.2.1. Función de costo

La elección de la función de costo adecuada depende del tipo de problema y del modelo específico que se esté utilizando. Para este caso se utiliza dos funciones: entropía cruzada y softmax sobre los mapas de activaciones finales.

Softmax

Es una función matemática utilizada en CNN para convertir un vector de números en probabilidades. La función Softmax se aplica típicamente en la capa de salida para producir una distribución de probabilidad sobre varias clases. Esta función calcula la probabilidad de que cada elemento pertenezca a una clase específica, normalizando las salidas para que sumen 1.

Esto permite interpretar las salidas de la red como probabilidades y facilita la clasificación de múltiples clases en un problema de aprendizaje supervisado.

$$p_i(x) = \frac{\exp a_i(x)}{\sum_{j=1}^K \exp a_j(x)} \quad (3.7)$$

donde $a_i(x)$ es la activación del i -ésimo canal de características, x es la posición del píxel con $x \in \Omega$ y $\Omega \subseteq \mathbb{Z}^2$ [28], K es

el número de clases, y $p_i(x)$ es la función máxima aproximada.

Para cuando $p_i(x) \approx 1$ para la j que tiene su máximo, entonces $a_i(x)$ y $p_k(x) \approx 0$ para todo otro k .

Cross-entropy (Entropía cruzada)

La entropía cruzada es una función de pérdida ampliamente utilizada en tareas de clasificación, especialmente en redes neuronales. Evalúa qué tan cerca está la distribución de probabilidad predicha por el modelo respecto a la distribución real de las clases.

Durante el entrenamiento, minimizar esta función implica ajustar los pesos de la red para que las salidas predichas se asemejen lo más posible a las etiquetas verdaderas. Esto se traduce en una mejora en la capacidad del modelo para clasificar correctamente las muestras.

En [28], se utiliza una versión ponderada de la entropía cruzada, representada por la siguiente ecuación:

$$E = - \sum_{x \in \Omega} w(x) \log(p_{l(x)}(x)) \quad (3.8)$$

donde Ω es el conjunto de píxeles de la imagen, $w(x)$ es un mapa de pesos que permite dar mayor importancia a ciertas regiones, y $p_{l(x)}(x)$ es la probabilidad predicha para la clase correcta $l(x)$ en el píxel x . Las clases posibles están en el conjunto $\{1, 2, \dots, K\}$.

3.2.2. Mapas de pesos

Los mapas de pesos (*weight maps*, en inglés) son una técnica utilizada para asignar distintos niveles de importancia a diferentes regiones de los datos durante el entrenamiento de un modelo. En tareas de segmentación de imágenes, permiten ajustar la contribución de ciertas áreas al cálculo de la función de pérdida, ayudando a enfocar el aprendizaje del modelo en regiones más relevantes.

Por ejemplo, un mapa de pesos puede asignar valores mayores a los píxeles cercanos a los bordes entre clases, lo cual es útil para mejorar la precisión en límites difíciles de segmentar. En el caso del modelo U-Net [28], los mapas de pesos se definen mediante la siguiente ecuación:

$$w(x) = w_c(x) + w_0 \cdot \exp \left(-\frac{(d_1(x) + d_2(x))^2}{2\sigma^2} \right) \quad (3.9)$$

donde $w_c(x) : \Omega \rightarrow \mathbb{R}$ equilibra las frecuencias de clase, y $d_1(x)$ y $d_2(x)$ representan la distancia desde el píxel x a los bordes de las dos regiones de objeto más cercanas. Esta formulación permite dar mayor peso a los píxeles situados entre objetos adyacentes, reforzando el aprendizaje en zonas de separación crítica. [28] w_0 es un valor que determina el peso adicional aplicado a las regiones cercanas a las fronteras entre objetos (como células tocándose). Un valor alto significa que se enfatiza la precisión en esas fronteras, incentivando a la red a aprender a distinguir objetos muy cercanos o en contacto. [28] σ es la desviación estándar que define qué tan rápidamente disminuye el peso adicional a medida que nos alejamos de las fronteras. Es decir, regula el radio de influencia de esas fronteras; valores

más pequeños hacen que solo las regiones muy cercanas a las fronteras tengan un peso elevado, mientras que valores mayores extienden esa influencia a regiones más amplias.

3.2.3. Learning rate

La tasa de aprendizaje, o *learning rate*, este parámetro decide el tamaño de los pasos que se dan al ajustar los pesos de un modelo durante la optimización. Dicho de manera sencilla, la tasa de aprendizaje regula la velocidad con la que un modelo *aprende* a partir de los datos como se muestra en la figura 3.8, representadas como flechas azules que dan el salto para llegar el mínimo de la función.

3.2.4. Descenso de gradiente estocástico (SGD)

Retomando las figuras 3.8 y 3.9 donde trata de encontrar el mínimo en una función mediante la variante estocástica. La función SGD esta dada por la ecuación 3.10, esta incluye la función de costo, la tasa de aprendizaje y una derivada.

$$W_2 = W_1 - \alpha * \nabla E(w) \quad (3.10)$$

- α - tasa de aprendizaje.
- W_1 - Parámetros del modelo (pesos).
- W_2 - El nuevo parámetro del modelo.
- ∇E - Derivada de la función de costo que depende de los pesos.

3.2.5. Épocas (Epochs)

Una época se refiere a una iteración completa de entrenamiento de una red neuronal en un conjunto de datos [13]. Durante cada época, la red neuronal procesa todas las muestras del conjunto de entrenamiento una vez, ajustando los pesos de las conexiones entre las neuronas para mejorar su capacidad de realizar tareas como clasificación, regresión, reconocimiento de imágenes, procesamiento del lenguaje natural, entre otros [23]. El entrenamiento de una red neuronal implica la presentación repetida de los datos de entrenamiento a la red, con el objetivo de que el modelo aprenda a reconocer patrones y relaciones en los datos. Cada época es una de estas presentaciones, y el proceso se repite varias veces hasta que el modelo alcanza un nivel aceptable de precisión o converge [29], las figuras, 3.12 y 3.13 muestra la importancia del numero de épocas en un entrenamiento.

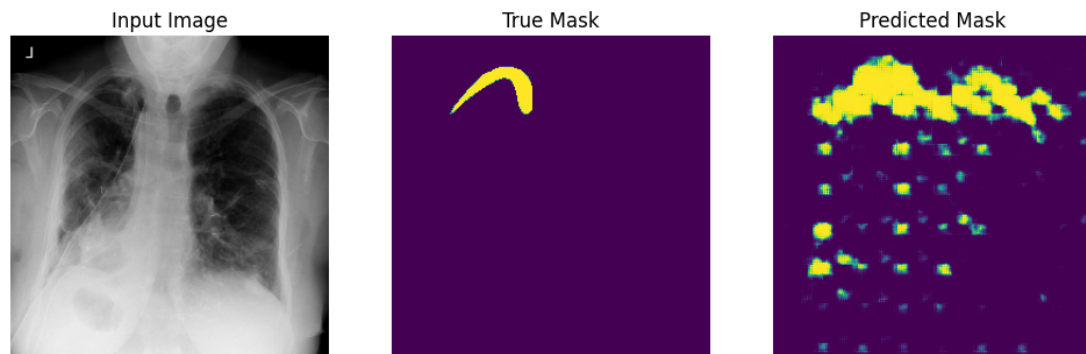


Figura 3.10: Se tiene la radiografía, la mascara y las mascara que debe predecir la U-Net, esta con pocas épocas de entrenamiento.

Las épocas pueden guardar puntos de control durante el entrenamiento, lo que permite recuperar el estado de la red neuro-

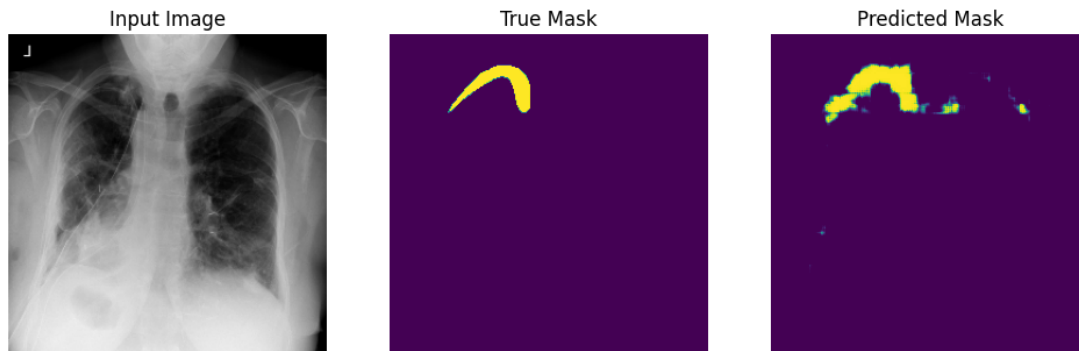


Figura 3.11: Se tiene la radiografía, la mascara y las mascara que debe predecir la U-Net, esta con un aumento de épocas diferencia de la imagen anterior.

nal en un momento específico si es necesario [20]. Por ejemplo, el tamaño del lote (batch size), el número de neuronas en cada capa y el número de épocas son algunos de los hiperparámetros que se pueden ajustar para mejorar el rendimiento de la red neuronal [2],[35].

Capítulo 4

Desarrollo y resultados

4.1. Procesamiento de datos

La primera parte del proceso tiene como objetivo descargar y descomprimir archivos desde una URL específica en el entorno de Kaggle. El conjunto de datos fue extraído de Kaggle¹.

En la página del dataset se encuentran varias carpetas: una contiene las máscaras de las imágenes y otra, las radiografías de entrenamiento. Una vez descargadas, se listan para verificar que cada radiografía tenga su correspondiente máscara. El conjunto de datos se divide en dos subconjuntos: uno para entrenamiento y otro para pruebas. Cada imagen, tanto la radiografía como su máscara, tiene una resolución de 1024×1024 píxeles. El dataset proporciona un total de 12,041 imágenes, de las cuales 10,668 se utilizan para entrenamiento y 1,373 para pruebas.

Las imágenes, en formato JPEG, se decodifican como tensores RGB de tres canales. Por su parte, las máscaras se transforman a escala de grises y posteriormente se binarizan,

¹<https://www.kaggle.com/datasets/vbookshelf/pneumothorax-chest-xray-images-and-masks>

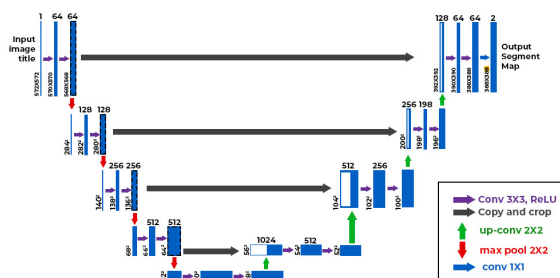


Figura 4.1: Arquitectura de U-Net.

tomando valores de 0 y 1.

El desarrollo del algoritmo se hizo en Python con la versión 3.10.12 en la cual existen una variedad de librerías que contienen las funciones necesarias para construir cualquier CNN. Para esta tesis las principales librerías son tensorflow y keras en sus versiones 2.17.0 y 3.4.1 respectivamente son librerías de código abierto especializadas en los entorno de aprendizajes automáticos.

4.1.1. Cálculo

Para poder describir el proceso de la Figura 4.1, es necesario que esta sea dividida por secciones, antes de cada convolución y antes de aplicar un MaxPooling. Para la imagen de entrada (input) con un tamaño de (572 x 572) píxeles, se le aplica la primera convolución (3 x 3), ajustando el tamaño de la imagen a un volumen siguiendo la ecuación 3.4 , obteniendo así (570 x 570 x 64). Se aplica la segunda convolución que genera un nuevo volumen de (568 x 568 x 64). Aquí acaba la primera sección, por lo tanto, se le aplica el primer MaxPooling (MP) (2 x 2).

En la segunda sección, después del MaxPooling, se obtiene un volumen de $(284 \times 284 \times 128)$. A este volumen se le aplica una primera convolución, resultando en un nuevo volumen de $(282 \times 282 \times 128)$. Posteriormente, se aplica una segunda convolución, obteniendo un volumen de $(280 \times 280 \times 128)$. En este punto, se aplica otro MaxPooling.

En la tercera sección, después del MaxPooling, se obtiene un volumen de $(140 \times 140 \times 128)$. A este volumen se le aplican dos convoluciones: la primera genera un volumen de $(138 \times 138 \times 256)$ y la segunda, un volumen de $(136 \times 136 \times 256)$. Para pasar a la siguiente sección, se aplica nuevamente el MaxPooling.

Para la cuarta y penúltima sección antes de llegar al fondo de la “U” referente a la figura 2.2, con un volumen inicial después del Max Pooling $(68 \times 68 \times 256)$, mientras que para la primera convolución $(66 \times 66 \times 512)$, para la segunda convolución $(64 \times 64 \times 512)$ y aplicamos el último Max Pooling.

Para la última parte, se tiene un volumen de $(32 \times 32 \times 512)$. Se aplica la primera convolución, obteniendo un volumen de $(30 \times 30 \times 1024)$, y una segunda convolución, obteniendo un volumen de $(28 \times 28 \times 1024)$.

A partir de este punto, se comienza una *Up-Convolution* o la reconstrucción de la imagen, en esta segunda parte se ocupan dos funciones principales: por su puesto la Up-Conv (2×2) y *copy and crop* que no es más que una concatenación y un corte

a el volumen.

Como se menciona en el capítulo anterior la *Up-Conv* es lo contrario a la convolución, lo que en el caso de la figura 2.2 se ve como multiplica las dimensiones de la imagen mientras que para los *Feature channels* los divide a la mitad, por ultimo se juntan los volúmenes análogos como se muestra en la figura 4.2, aunque es de notar que los volúmenes análogos son un poco mas grandes, entonces son cortados en la parte del centro como se ve en la figura 4.1.

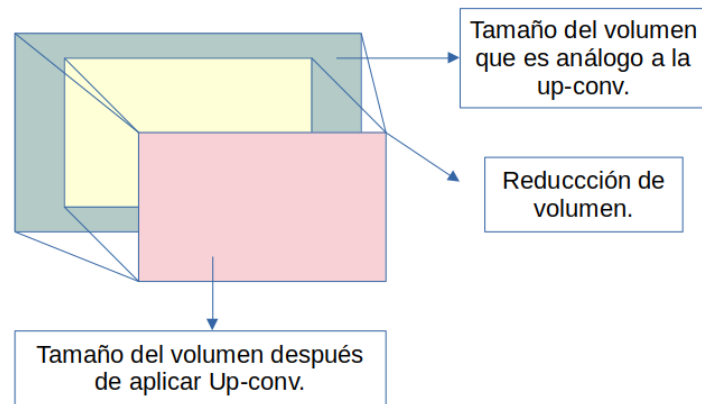


Figura 4.2: La reducción del volumen para después ser juntado con otro.

Resumiendo el proceso de expansión, hay tantas *Up-Conv* como Maxpool hubo en el proceso de contracción, después de la concatenación nuevamente una convolución, hasta llegar a la capa final donde en lugar de aplicar una convolución de (3x3), se aplica una de (1x1) 2 veces.

4.2. Código U-NET

En este apartado se presenta el código del algoritmo (Apendice A), el funcionamiento del algoritmo es el siguiente:

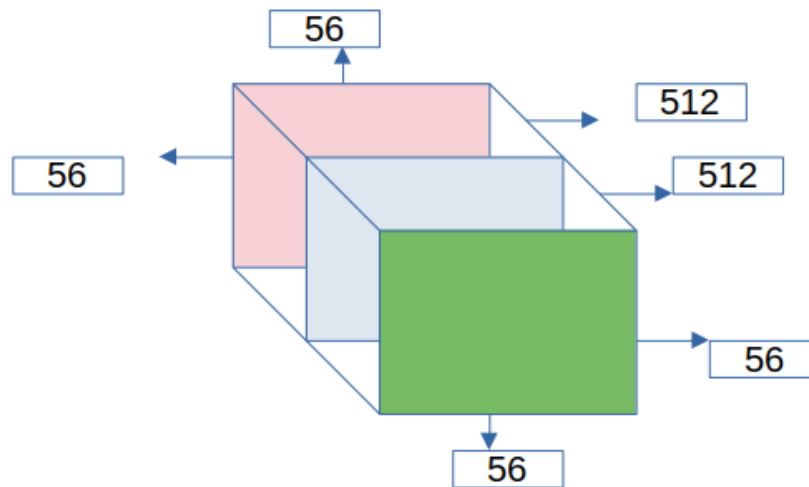


Figura 4.3: El nuevo volumen después del Up-Conv concatenándose con su análogo

1. Se descarga el *dataset* como un zip de un entorno kaggle.
2. Se verifica la descarga y se descomprimen en 12041 radiografías y 12041 mascararas.
3. Se verifica que cada radiografía tenga su mascara correspondiente.
4. Se definen las funciones que se usan y se divide el *dataset* para entrenamiento y para pruebas(10668 y 1373).
5. Desarrollo de la arquitectura.
6. Se entrena el modelo.
7. Se guarda los pesos como un archivo de extensión **.h5**.

En el apéndice B se muestra el algoritmo que evalúa individualmente las imágenes.

1. Se cargan los pesos que vienen con extensión **.h5**
2. Se coloca la arquitectura.

3. Se elige la imagen que se quiere evaluar.
4. Se desarrolla la función que evalúa con todas las métricas.
5. Se guarda las radiografías, las mascararas y la predicción.

4.3. Pruebas con imágenes conocidas



Figura 4.4: Radiografía con neumotórax.



Figura 4.5: Radiografía con neumotórax.



Figura 4.6: Radiografía con neumotórax.

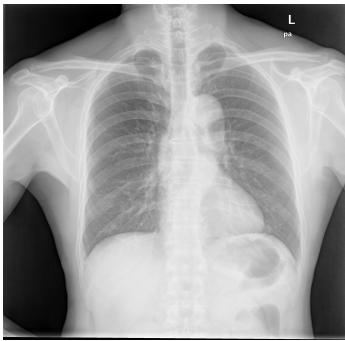


Figura 4.7: Radiografía sin neumotórax.

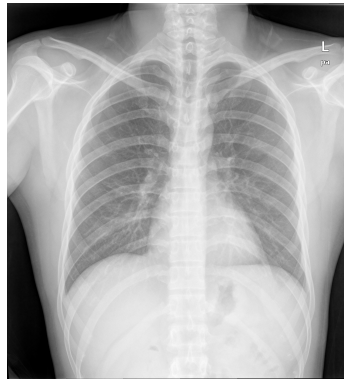


Figura 4.8: Radiografía sin neumotórax.



Figura 4.9: Radiografía sin neumotórax.

Se dispone de seis radiografías de tórax para evaluar el desempeño de U-Net. Las primeras tres imágenes (4.3, 4.4 y 4.5) corresponden a pacientes con diagnóstico de neumotórax, mientras que las restantes (4.6, 4.7 y 4.8) son radiografías de tórax de individuos sanos.

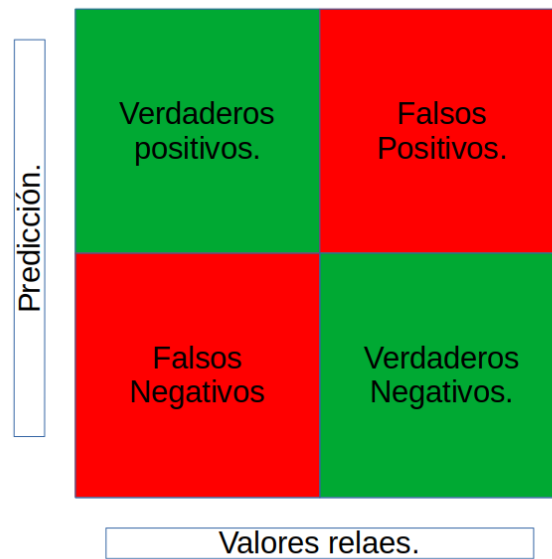


Figura 4.10: Ejemplo de una matriz de confusión. Los valores en verde corresponden a predicciones correctas, mientras que los valores en rojo representan errores de clasificación.

4.4. Resultados

En la matriz de confusión se identifican cuatro categorías principales:

- **Verdaderos Positivos (VP):** Casos en los que el modelo predice correctamente la clase positiva.
- **Falsos Positivos (FP):** Casos en los que el modelo predice incorrectamente la clase positiva.
- **Falsos Negativos (FN):** Casos en los que el modelo predice incorrectamente la clase negativa.
- **Verdaderos Negativos (VN):** Casos en los que el modelo predice correctamente la clase negativa.

La evaluación se realiza comparando los píxeles de la segmentación generada por la CNN (X) con la máscara de verdad de terreno (Y) mediante:

■ **Coeficiente de Dice (DSC/F1-Score):**

$$\text{DSC} = \frac{2|X \cap Y|}{|X| + |Y|} \quad (4.1)$$

Mide la superposición entre la segmentación predicha (X) y la verdad de terreno (Y).

Funcionamiento: Calcula el solapamiento normalizado entre las áreas predichas y reales. Un valor de 1 indica coincidencia perfecta. Es especialmente útil en segmentación de objetos pequeños, ya que no penaliza los desequilibrios de área como el IoU.

■ **IoU (Intersection over Union / Jaccard Index):**

$$\text{IoU} = \frac{|X \cap Y|}{|X \cup Y|} \quad (4.2)$$

Diferencia clave: El denominador incluye solo la unión de áreas (no la suma), lo que lo hace más sensible a falsos positivos. Se considera apropiado para tareas donde la precisión de contorno es crítica. Versión más estricta que penaliza fuertemente los errores.

■ **Sensibilidad (*Recall*):**

$$\text{Sensibilidad} = \frac{\text{VP}}{\text{VP} + \text{FN}} \quad (4.3)$$

Evalúa la capacidad para detectar áreas relevantes (VP: verdaderos positivos, FN: falsos negativos).

■ **Precisión:**

$$\text{Precisión} = \frac{VP}{VP + FP} \quad (4.4)$$

Indica la exactitud de las predicciones positivas.

■ **Especificidad:**

$$\text{Especificidad} = \frac{VN}{VN + FP} \quad (4.5)$$

Mide la habilidad para descartar regiones irrelevantes (VN: verdaderos negativos, FP: falsos positivos).

Exactitud	Precisión	Sensibilidad	Especificidad	F1 score
$\frac{VP + VN}{FN + VN + FP + VP}$	$\frac{VP}{VP + FP}$	$\frac{VP}{VP + FN}$	$\frac{VN}{VN + FP}$	$\frac{2 * \text{precision} * \text{sensibilidad}}{\text{precision} + \text{sensibilidad}}$

Figura 4.11: Tabla con todas las métricas de evaluación más usadas.

Evaluación

La duración del entrenamiento varía en función de los parámetros utilizados, particularmente del número de épocas empleadas para entrenar la arquitectura U-Net. En general, un mayor número de épocas tiende a mejorar la precisión de los resultados; sin embargo, esto también implica un aumento en el costo computacional y en el tiempo requerido para completar el entrenamiento. Es importante señalar que un número excesivo de épocas puede conducir al sobreentrenamiento del modelo, fenómeno conocido como *overfitting*.

A continuación, se presentan los resultados del entrenamiento para 20 y 40 épocas. El entrenamiento con 20 épocas requiere un tiempo estimado de 60 minutos. La evaluación del rendimiento del modelo se realizará utilizando el coeficiente de Dice y la métrica de Intersección sobre Unión (IoU), consideradas como las métricas más relevantes para este trabajo. También bajo las mismas condiciones se entreno ResNet, otro modelo de CNN con el cual se hace una comparación.

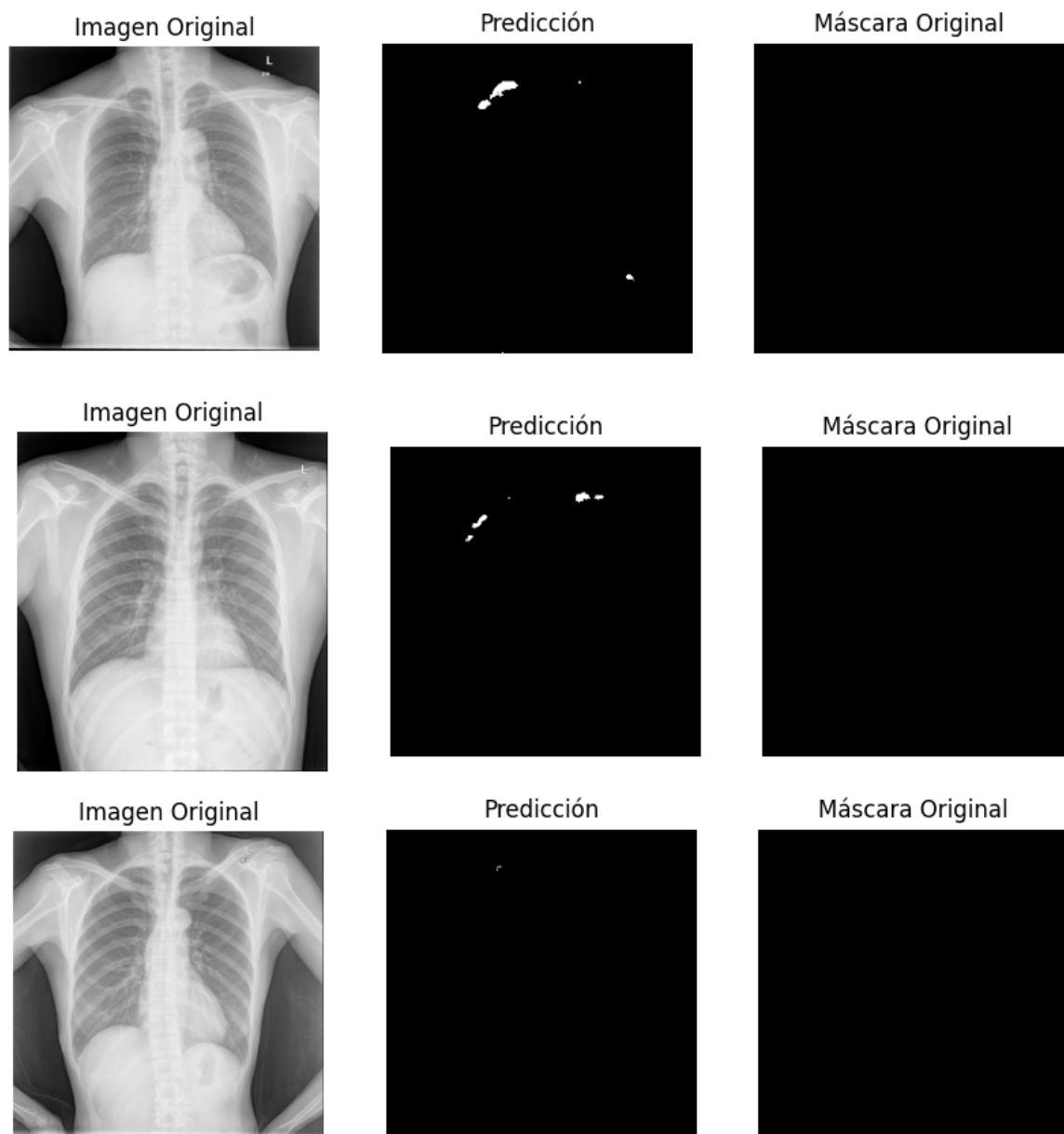


Figura 4.12: Radiografías, para 20 épocas de entrenamiento.

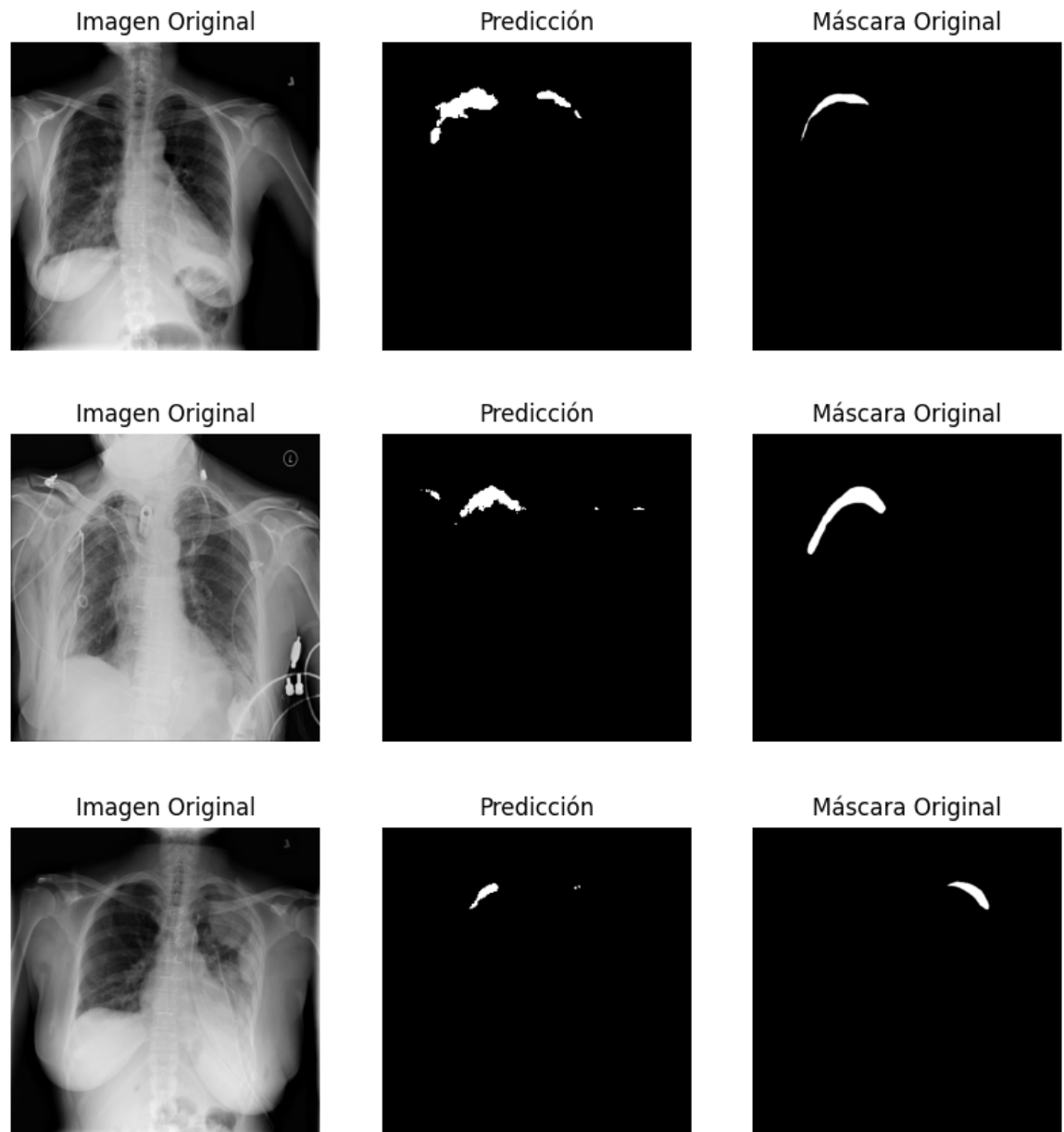


Figura 4.13: Las correspondientes figuras 4.3, 4.4 y 4.5 con sus mascararas donde se encuentra el neumotórax y la predicción con 20 épocas de entrenamiento.

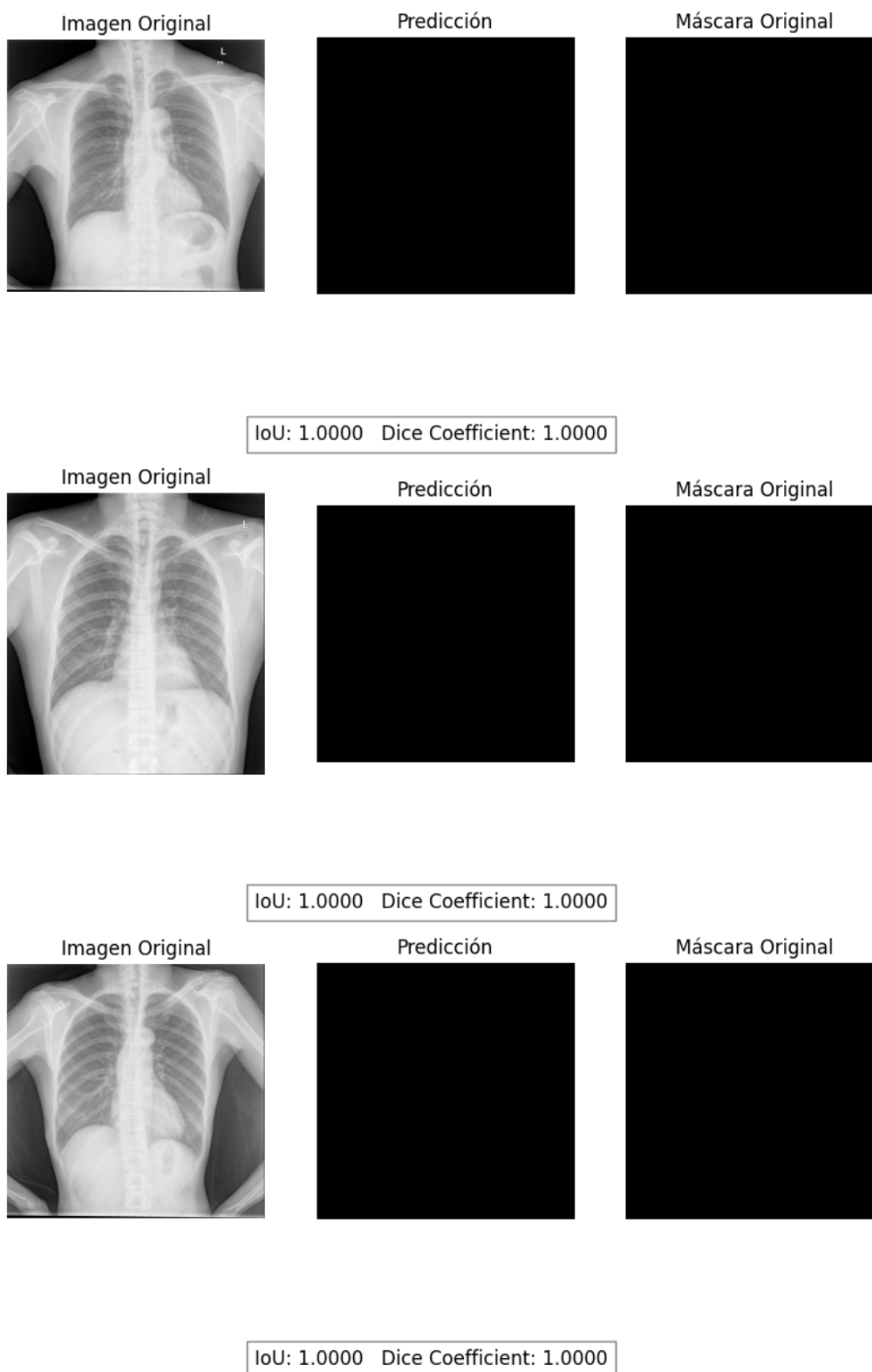


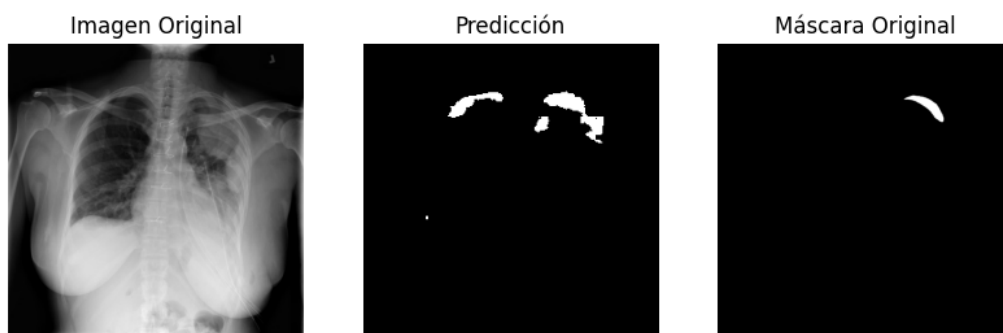
Figura 4.14: Estas radiografías son análogas a las mostradas en las figuras 4.6, 4.7 y 4.8. La máscara original se presenta en negro, ya que no se evidencia la presencia de la enfermedad neumotórax.



IoU: 0.2398 Dice Coefficient: 0.3868



IoU: 0.3089 Dice Coefficient: 0.4720



IoU: 0.1812 Dice Coefficient: 0.3068

Figura 4.15: Estas imágenes son tomadas de una base de datos diferentes, donde se podría decir que son radiografías mejor tomadas.

Coeficiente Dice de ResNet	Coeficiente Dice de U-Net.
0.00	1.00
0.00	1.00
0.00	1.00
0.01	0.3868
0.04	0.4720
0.01	0.3068

Figura 4.16: Comparativa de ResNet contra U-Net bajo las mismas condiciones.

Conclusiones

Este estudio confirmó que la arquitectura U-Net es una opción viable para tareas de segmentación, gracias a su eficiencia y relativamente bajo costo computacional. Como se muestra en la Figura 4.16, su desempeño es competitivo en comparación con otras arquitecturas profundas. No obstante, en escenarios de gran escala como la segmentación de imágenes médicas en entornos con recursos limitados el entrenamiento del modelo presentó ciertos desafíos.

- **Limitaciones computacionales:** la alta demanda de poder de procesamiento dificultó su implementación en contextos reales.
- **Desbalance de datos:** el conjunto de datos no balanceado obligó a emplear técnicas de aumento, las cuales, si bien funcionales, distan de ser ideales. Un dataset equilibrado mejoraría la capacidad de generalización del modelo.

Aunque los pesos preentrenados pueden desplegarse en equipos con recursos limitados, la necesidad de actualizaciones frecuentes y *fine-tuning* complica su adopción sostenible en comunidades remotas con conectividad restringida.

Hallazgos clave:

1. Las redes neuronales convolucionales (incluyendo U-Net) deben considerarse como **herramientas de apoyo**, nunca como reemplazo del diagnóstico médico profesional.
2. Estas tecnologías muestran potencial para superar barreras geográficas en la atención médica.
3. Proporcionan capacidades de evaluación preliminar en tiempo real.

Líneas futuras de investigación:

- Técnicas avanzadas de balanceo de datos (funciones de pérdida ponderadas, generación sintética controlada).
- Optimización del equilibrio entre precisión y consumo de recursos computacionales.
- Protocolos de actualización *offline* para entornos con conectividad intermitente.

La implementación efectiva de IA en medicina requiere tanto innovación técnica como soluciones inclusivas que prioricen las necesidades de poblaciones vulnerables.

Bibliografía

- [1] David Andrés. *Gradiente Descendente*. Consultado el: 1 de agosto de 2024. 2022. URL: <https://mlpillars.dev/machine-learning-es/gradiente-descendente/>.
- [2] Y. Bengio. “Deep Learning”. En: *IEEE Signal Processing Magazine* 33.4 (2016), págs. 132-133.
- [3] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.
- [4] Zhihao Chen. “Medical Image Segmentation Based on U-Net”. En: vol. 2547. Institute of Physics, 2023. DOI: 10.1088/1742-6596/2547/1/012010.
- [5] Computer Science Wiki. *Maxpool Sample 2*. Accessed: 2024-07-22. 2023. URL: <https://computersciencewiki.org/index.php/File:MaxpoolSample2.png>.
- [6] Wikipedia contributors. *DICOM*. Último acceso: 4 de julio de 2024. 2024. URL: <https://es.wikipedia.org/wiki/DICOM>.
- [7] DataSmarts. *Cómo manipular los píxeles de una imagen con OpenCV*. [En línea; acceso 19-ago-2024]. 2024. URL: <https://www.datasmarts.net/como-manipular-los-pixeles-de-una-imagen-con-opencv/>.
- [8] *Digital Imaging and Communications in Medicine (DICOM)*. <https://www.dicomstandard.org/>. Estándar de transmisión de imágenes médicas y datos entre hardware de propósito médico. 2021.
- [9] Journal Of Healthcare Engineering. “Retracted: U-Net-Based Medical Image Segmentation”. En: *Journal of Healthcare Engineering* 2023 (2023), pág. 9890389. ISSN: 2040-2309. DOI: 10.1155/2023/9890389.
- [10] El Laberinto de Falken. *Visión artificial con redes convolucionales (CNN)*. Accedido: 21 de mayo de 2024. 2019. URL: <https://www.ellaberintodefalken.com/2019/10/vision-artificial-redes-convolucionales-CNN.html>.
- [11] F. Giménez Palomares, J. A. Monsoriu Serrá y E. Alemany Martínez. “Aplicación de la convolución de matrices al filtrado de imágenes”. En: *Modelling in Science Education and Learning* 9.1 (2016), pág. 97. DOI: 10.4995/msel.2016.4524. URL: <https://doi.org/10.4995/msel.2016.4524>.

- [12] Rafael C. Gonzalez y Richard E. Woods. *Digital image processing*. Upper Saddle River, N.J.: Prentice Hall, 2008, págs. —. ISBN: 9780131687288 013168728X 9780135052679 013505267X. URL: http://www.worldcat.org/search?qt=worldcat_org_all&q=013505267X.
- [13] I. Goodfellow, Y. Bengio y A. Courville. *Deep Learning*. MIT Press, 2016.
- [14] Ian Goodfellow, Yoshua Bengio y Aaron Courville. *Deep Learning*. MIT Press, 2016.
- [15] Kaiming He et al. “Deep Residual Learning for Image Recognition”. En: *CoRR* abs/1512.03385 (2015). arXiv: 1512.03385. URL: <http://arxiv.org/abs/1512.03385>.
- [16] D. H. Hubel y T. N. Wiesel. “Receptive fields, binocular interaction and functional architecture in the cat’s visual cortex”. En: *The Journal of Physiology* 160.1 (1962), págs. 106-154. DOI: <https://doi.org/10.1113/jphysiol.1962.sp006837>. eprint: <https://physoc.onlinelibrary.wiley.com/doi/pdf/10.1113/jphysiol.1962.sp006837>. URL: <https://physoc.onlinelibrary.wiley.com/doi/abs/10.1113/jphysiol.1962.sp006837>.
- [17] Debesh Jha et al. “DoubleU-Net: A Deep Convolutional Neural Network for Medical Image Segmentation”. En: *arXiv* (jun. de 2020). URL: <http://arxiv.org/abs/2006.04868>.
- [18] Pranjal Khadka. *Understanding Convolutional Neural Network (CNNs)*. [En línea; acceso 19-ago-2024]. 2024. URL: <https://medium.com/@pranjalkhadka/understanding-convolutional-neural-network-cnns-e6f6e58fe249>.
- [19] KiKaBeN. “Up-sampling with Transposed Convolution”. En: (2017). URL: <https://kikaben.com/up-sampling-with-transposed-convolution/>.
- [20] D. P. Kingma y J. Ba. “Adam: A Method for Stochastic Optimization”. En: *arXiv preprint arXiv:1412.6980* (2014).
- [21] Alex Krizhevsky, Ilya Sutskever y Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. En: *Advances in neural information processing systems*. 2012, págs. 1097-1105.
- [22] Nora La Serna Palomino y Ulises Norberto Román Concha. “Técnicas de Segmentación en Procesamiento Digital de Imágenes”. En: *Revista de investigación de Sistemas e Informática* 6.2 (dic. de 2009), págs. 9-16. URL: <https://revistasinvestigacion.unmsm.edu.pe/index.php/sistem/article/view/3299>.
- [23] Y. LeCun, Y. Bengio y G. Hinton. “Deep learning”. En: *Nature* 521.7553 (2015), págs. 436-444.
- [24] Yann LeCun et al. “Gradient-based learning applied to document recognition”. En: *Proceedings of the IEEE* 86.11 (1998), págs. 2278-2324.

- [25] Devashree Madhugiri. “Padding in CNN: Its Importance in Convolutional Neural Networks”. En: *KnowledgeHut* (2023). Accessed: 2024-06-06. URL: <https://www.knowledgehut.com/blog/data-science/padding-in-cnn>.
- [26] MathWorks. *What is DICOM?* Último acceso: 4 de julio de 2024. 2024. URL: <https://la.mathworks.com/discovery/dicom.html>.
- [27] Pydicom Developers. *Pydicom: An open source DICOM library*. Una biblioteca de Python para trabajar con archivos DICOM. 2021. URL: <https://github.com/pydicom/pydicom>.
- [28] Olaf Ronneberger, Philipp Fischer y Thomas Brox. “U-Net: Convolutional Networks for Biomedical Image Segmentation”. En: (mayo de 2015). URL: <http://arxiv.org/abs/1505.04597>.
- [29] S. Russell y P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2010.
- [30] Ministerio de Salud Pública de Tucumán. *Neumotórax: qué es y cuáles son sus síntomas*. Accedido: 2024-07-17. 2024. URL: <https://msptucuman.gov.ar/neumotorax-que-es-y-cuales-son-sus-sintomas/>.
- [31] Evan Shelhamer, Jonathan Long y Trevor Darrell. “Fully Convolutional Networks for Semantic Segmentation”. En: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39 (4 2017). ISSN: 01628828. DOI: 10.1109/TPAMI.2016.2572683.
- [32] Signal y Technion Image Processing Lab. *Creating Image Segmentation Maps Using GANs*. Accessed: 2024-07-22. 2023. URL: <https://sipl.eelabs.technion.ac.il/projects/creating-image-segmentation-maps-using-gans/>.
- [33] Karen Simonyan y Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. En: *arXiv preprint arXiv:1409.1556* (2014).
- [34] Campos Soto. *Neumología: Manual de Diagnóstico y Terapéutica (Spanish Edition)*. Ergon Ediciones, 2006. ISBN: 9788484734093, 8484734099. URL: <https://libgen.li/file.php?md5=d33c1e5ef43e9a2a62b29c0639fa41f3>.
- [35] TensorFlow Developers. *TensorFlow: An end-to-end open-source machine learning platform*. <https://www.tensorflow.org>. 2024. URL: <https://www.tensorflow.org>.
- [36] vBookshelf. *Pneumothorax Chest X-ray Images and Masks*. [En línea; acceso 19-ago-2024]. 2024. URL: <https://www.kaggle.com/datasets/vbookshelf/pneumothorax-chest-xray-images-and-masks>.
- [37] Wikipedia contributors. *Imagen digital — Wikipedia, La enciclopedia libre*. [En línea; acceso 19-ago-2024]. 2024. URL: https://es.wikipedia.org/wiki/Imagen_digital.
- [38] Wikipedia contributors. *Píxel — Wikipedia, La enciclopedia libre*. [En línea; acceso 19-ago-2024]. 2024. URL: <https://es.wikipedia.org/wiki/P%C3%ADxel>.

- [39] Wikipedia contributors. *Radiografía* — *Wikipedia, La enciclopedia libre*. [En línea; acceso 19-ago-2024]. 2024. URL: <https://es.wikipedia.org/wiki/Radiograf%C3%ADa>.
- [40] Wikipedia contributors. *Resolución de imagen* — *Wikipedia, La enciclopedia libre*. [En línea; acceso 19-ago-2024]. 2024. URL: https://es.wikipedia.org/wiki/Resoluci%C3%B3n_de_imagen.
- [41] Wikipedia contributors. *Segmentación (procesamiento de imágenes)* — *Wikipedia, La enciclopedia libre*. [En línea; acceso 19-ago-2024]. 2024. URL: [https://es.wikipedia.org/wiki/Segmentaci%C3%B3n_\(procesamiento_de_im%C3%A1genes\)](https://es.wikipedia.org/wiki/Segmentaci%C3%B3n_(procesamiento_de_im%C3%A1genes)).
- [42] Zihan Yang. “Classification of picture art style based on VGGNET”. En: vol. 1774. IOP Publishing Ltd, feb. de 2021. DOI: 10.1088/1742-6596/1774/1/012043.
- [43] Zongwei Zhou et al. “UNet++: A Nested U-Net Architecture for Medical Image Segmentation”. En: (jul. de 2018). URL: <http://arxiv.org/abs/1807.10165>.
- [44] Zongwei Zhou et al. “UNet++: Redesigning Skip Connections to Exploit Multiscale Features in Image Segmentation”. En: *IEEE Transactions on Medical Imaging* 39 (6 jun. de 2020), págs. 1856-1867. ISSN: 1558254X. DOI: 10.1109/TMI.2019.2959609.

Apéndice A

Código

El código U-Net.

Listing A.1: Código U-Net.

```
base_model = tf.keras.applications.\
    MobileNetV2(input_shape=[256,256,3], \
        include_top=False, alpha =0.5)

# Use the activations of these layers
layer_names = [
    'block_1_expand_relu',    # 64x64
    'block_3_expand_relu',    # 32x32
    'block_6_expand_relu',    # 16x16
    'block_13_expand_relu',   # 8x8
    'block_16_project',       # 4x4
]
base_model_outputs = [base_model.get_layer(\
    name).output for name in layer_names]

# Create the feature extraction model
down_stack = tf.keras.Model(inputs=\
    base_model.input, outputs=)
```

```

        base_model_outputs)
down_stack.trainable = False

def upsample(filters , size , norm_type='\
batchnorm' , apply_dropout=False):
    initializer = tf.\
        random_normal_initializer(0. , 0.02)

    result = tf.keras.Sequential()
    result.add(
        tf.keras.layers.Conv2DTranspose(\
            filters , size , strides=2,
            padding='same' ,
            kernel_initializer=initializer , \
            use_bias=False))

    if norm_type.lower() == 'batchnorm':
        result.add(tf.keras.layers.\
            BatchNormalization())
    elif norm_type.lower() == 'instancenorm'\
        :
        result.add(InstanceNormalization())

    if apply_dropout:
        result.add(tf.keras.layers.Dropout\
            (0.5))
        result.add(tf.keras.layers.ReLU())

return result

```

```

up_stack = [
    upsample(512, 3), # 4x4 -> 8x8
    upsample(256, 3), # 8x8 -> 16x16
    upsample(128, 3), # 16x16 -> 32x32
    upsample(64, 3), # 32x32 -> 64x64
]

def unet_model(output_channels):
    inputs = tf.keras.layers.Input(shape\
        =[256,256,3])

    # Downsampling through the model
    skips = down_stack(inputs)
    x = skips[-1]
    skips = reversed(skips[: -1])

    # Upsampling and establishing the skip \
        connections
    for up, skip in zip(up_stack, skips):
        x = up(x)
        concat = tf.keras.layers.Concatenate\
            ()
        x = concat([x, skip])

    # This is the last layer of the model
    last = tf.keras.layers.Conv2DTranspose(
        output_channels, 3, strides=2, \
        activation='sigmoid',

```



```
padding='same')  #64x64 -> 128x128

x = last(x)

return tf.keras.Model(inputs=inputs, \
                        outputs=x)
```

Apéndice B

Código

El código para guardar y evaluar los resultados.

Listing B.1: Evaluación de resultados.

```
from google.colab import files
from tensorflow.keras.models import \
    load_model
from tensorflow.keras.preprocessing import \
    image
import numpy as np
import matplotlib.pyplot as plt

# Cargar archivos
files.upload() # Subir modelo
imagen = files.upload() # Subir imagen

# Funci n para preprocesar la imagen
def preprocess_image(img_path, target_size\
    =(224, 224)):
    img = image.load_img(img_path, \
        target_size=target_size)
    img_array = image.img_to_array(img)
```

```

img_array = np.expand_dims(img_array , \
                             axis=0)
img_array /= 255.0
return img_array

# Cargar el modelo (suponiendo que no hay \
  capas personalizadas)
model = load_model("unet_model.h5")

# Usar la imagen subida
img_path = list(imagen.keys())[0]

# Preprocesar la imagen
img_array = preprocess_image(img_path)

# Hacer la predicción
predicted_mask = model.predict(img_array)

# Asumiendo que es una máscara binaria (un \
  solo canal), se usa un umbral de 0.5
# La máscara predicha tendrá una forma \
  como (1, 224, 224, 1) por lo que \
  eliminamos las dimensiones adicionales
predicted_mask = (predicted_mask[0, :, :, 0] \
                  > 0.5).astype(np.uint8)

# Mostrar la imagen original y la máscara \
  predicha
plt.figure(figsize=(10, 5))

```

```
# Imagen original
plt.subplot(1, 2, 1)
original_image = image.load_img(img_path)
plt.imshow(original_image)
plt.title("Imagen - Original")

# M scara predicha
plt.subplot(1, 2, 2)
plt.imshow(predicted_mask, cmap="gray")
plt.title("M scara - Predicha")

plt.show()
```