Universidad Autónoma de Baja California

FACULTAD DE INGENIERÍA

SISTEMA DE PROCESAMIENTO DE IMÁGENES PARA LA IDENTIFICACIÓN DE BIOMARCADORES EN LAS ETAPAS DE LA RETINOPATÍA DIABÉTICA

TESIS PARA OBTENER EL GRADO DE:
MAESTRÍA EN CIENCIAS

PRESENTA
Andre Ivann Herrera Chavez

DIRECTOR
Dra. Wendy Flores Fuentes

CODIRECTOR
Dr. Oscar Humberto Montiel Ross

Mexicali, B.C. a 26 de marzo de 2025

AGRADECIMIENTOS

La culminación de este trabajo representa no solo un logro académico, sino también el reflejo del esfuerzo conjunto de personas e instituciones que han sido fundamentales en este proceso.

En primer lugar, expreso mi más sincero agradecimiento a la Dra. Wendy Flores Fuentes, directora de esta tesis, por su orientación, paciencia y confianza. Me abrió las puertas de la Universidad Autónoma de Baja California (UABC) y me brindó la oportunidad de realizar una maestría. Su guía no solo ha sido clave en mi formación como investigador, sino también en mi desarrollo personal, enseñándome a abordar problemas reales y relevantes con un enfoque científico de alto nivel. Su exigencia, objetividad y compromiso fueron fundamentales para motivarme a mejorar en mis experimentos e innovaciones, alineándose perfectamente con mis objetivos profesionales. Sus retroalimentaciones precisas y directas han contribuido a mi crecimiento durante el posgrado.

De igual manera, extiendo mi profundo agradecimiento al Dr. Oscar Humberto Montiel Ross, codirector de este trabajo, por su apoyo constante y su invaluable orientación a lo largo de esta investigación. Su experiencia y conocimientos fueron esenciales para el desarrollo de este proyecto.

Asimismo, quiero expresar mi sincero reconocimiento al Dr. Eder Alejandro Rodríguez Martínez, por compartir su experiencia en visión por computadora y por su disposición para ayudarme a comprender muchisimo mejor el tema de aprendizaje profundo. Su apoyo constante desde el inicio de este proyecto ha sido fundamental, y valoro profundamente sus enseñanzas en cada etapa de esta investigación.

Agradezco a la Secretaría de Ciencia, Innovación y Tecnología para la Humanidad e Industria (SECIHTI) por el respaldo brindado durante la maestría, así como a la UABC, por proporcionar los recursos académicos y formativos necesarios para la realización de esta tesis.

También deseo expresar mi gratitud al Subcomité Académico por Campo del Conocimiento de Instrumentación y Mediciones Automáticas (SACC) por su evaluación y retroalimentación durante cada semestre.

Finalmente, quiero agradecer a mis compañeros, quienes con su apoyo y colaboración hicieron que este camino fuera más llevadero. Del mismo modo, expreso mi reconocimiento al personal docente y administrativo de la UABC, cuya disposición y ayuda facilitaron cada etapa de este proceso académico.

A todos ustedes, gracias por ser parte de este logro.

DEDICATORIA

A las futuras generaciones de investigadores, con la esperanza de que este trabajo sirva como un punto de partida y no como un comienzo desde cero.

A mi padre, quien en vida me enseñó el valor de la lógica, el razonamiento y la resolución de problemas, habilidades que han sido fundamentales en mi desarrollo personal y profesional. Incluso en programación, tus enseñanzas desde mi juventud han sido una guía constante en mi camino. Seguiré adelante honrándote con cada paso que dé en mi vida y en mi carrera.

A mi madre, por su amor incondicional, su fortaleza y su apoyo constante. Gracias por enseñarme a enfrentar los desafíos con determinación y de siempre hacer las cosas de la mejor manera posible.

A la Dra. Wendy Flores Fuentes y al Dr. Eder Alejandro Rodríguez Martínez, por su invaluable apoyo en mi formación académica y personal. Gracias por compartir su conocimiento, por guiarme con paciencia y por ayudarme a desarrollar habilidades que me han fortalecido como investigador y como persona a lo largo de mi posgrado.

RESUMEN

Esta tesis tiene como propósito el desarrollo de un sistema especializado de procesamiento de imágenes para la detección y segmentación de biomarcadores asociados a las etapas de la retinopatía diabética. El sistema propuesto integra técnicas avanzadas de visión por computadora y aprendizaje profundo, empleando arquitecturas como CNN (de sus siglas en inglés, Convolutional Neural Networks) y ViT (de sus siglas en inglés, Vision Transformers) para la clasificación de imágenes. Así como, SAM (de sus siglas en inglés, Segment Anything Model) para la segmentación y YOLO (de sus siglas en inglés, You Only Look Once) para la localización de biomarcadores, como microaneurismas, exudados duros, exudados suaves y hemorragias.

El enfoque incluye la implementación de preprocesamientos que mejoran la calidad de las imágenes mediante técnicas como CLAHE (de sus siglas en inglés, Contrast Limited Adaptive Histogram Equalization) y Filtro de Ben. Además, se utiliza la generación de imágenes sintéticas a través de CycleGan (de sus siglas en inglés, Cycle Generative Adversarial Network) y NST (de sus siglas en inglés, Neural Style Transfer) para ampliar y diversificar los conjuntos de datos, superando las limitaciones de disponibilidad de imágenes sintéticas.

El sistema fue validado mediante métricas de rendimiento como precisión, Accuracy, Recall, F1-Score y Confusion Matrix. Estas métricas permitieron medir la efectividad del modelo en la detección de biomarcadores y clasificación de las etapas de la retinopatía diabética.

ÍNDICE

Índice de Figuras

Índice de Tablas

CAPÍTULO 1

INTRODUCCIÓN

1.1. Antecedentes

La diabetes mellitus, reconocida como una de las enfermedades de mayor impacto a nivel global, afecta a millones de personas y constituye una de las principales causas de morbilidad y mortalidad en el mundo. Según datos de la Federación Internacional de Diabetes (FID), más de 537 millones de personas padecen esta condición, y se estima que este número podría alcanzar los 700 millones para el año 2045 [1].

Este trastorno microvascular es el resultado de un daño progresivo en los vasos sanguíneos de la retina, causado por la hiperglucemia persistente [2]. La retinopatía diabética (RD) se encuentra estrechamente ligada al tiempo de evolución de la diabetes y al control deficiente de la glucosa en sangre, factores que determinan tanto su aparición como su progresión [3]. En sus primeras etapas, la enfermedad puede ser asintomática, lo que dificulta su detección temprana y aumenta el riesgo de complicaciones graves, como el desprendimiento de retina, el edema macular y, en casos avanzados, la pérdida irreversible de la visión.

Según Tan et al [4] se detalla que la RD se caracteriza por la presencia de biomarcadores específicos que permiten clasificar sus etapas. En las fases iniciales, se observan microaneurismas y pequeñas hemorragias, mientras que en etapas avanzadas surgen exudados duros, exudados suaves y neovascularización. Estos biomarcadores son fundamentales para el monitoreo y manejo clínico de la enfermedad, dado que su detección oportuna puede prevenir la progresión hacia complicaciones irreversibles.

El Procesamiento Digital de Imágenes (PDI) ha emergido como una herramienta clave en oftalmología permitiendo resolver tareas como la detección de biomarcadores y la clasificación de las etapas de la RD en CFI (de sus siglas en inglés, Color Fundus Image). Según Shamrat et al. [5], se presentó un modelo llamado DRNet13, basado en CNN, diseñado para la clasificación de las etapas de la retinopatía diabética en cinco categorías: fondo de ojo normal, retinopatía diabética no proliferativa leve, retinopatía diabética no proliferativa moderada, retinopatía diabética no proliferativa severa y retinopatía diabética proliferativa. Este modelo integró un análisis detallado de los mapas de características para identificar las regiones más relevantes en el proceso de predicción.

Para la identificación de estos biomarcadores, [6] propone un método de clasificación multietiqueta basado en redes de grafos para detectar ocho tipos de lesiones (Cicatrices de láser, drusas, relación copa-disco, hemorragias, arteriosclerosis retiniana, microaneurismas, exuda-

dos duros y exudados suaves) en imágenes de fondo de ojo en la detección de biomarcadores de la Retinopatía Diabética. Por otro lado, la OCT (de sus siglas en inglés, Optical Coherence Tomography) permite obtener imágenes transversales de alta resolución, como lo mencionan los autores de [7], quienes además proponen un método basado en aprendizaje multiinstancia y multietiqueta (MIML-LR) para la detección y reconocimiento automático de múltiples lesiones en imágenes de OCT.

Sin embargo, además de las CNN, también existe otra forma para clasificar las etapas y biomarcadores de la RD, según Dosovitskiy et al [8] introducen el enfoque de los ViT, que utiliza mecanismos de atención para procesar imágenes como una serie de parches, capturando relaciones globales con alta precisión. Gu et al. [9] desarrollaron un modelo de clasificación de las etapas de la RD. Este modelo incluye un Bloque de Extracción de Características, que emplea transformadores para identificar con precisión áreas afectadas como hemorragias y exudados, y un Bloque de Predicción de Grado, que utiliza atención residual para diferenciar regiones relevantes en las imágenes.

Por otra parte, Toto et al. [10] utilizaron YOLOv7 para detectar exudados duros y clasificar la desorganización de las capas internas de la retina (DRIL) en imágenes de OCT. Asimismo, Alexander Kirillov et al. [11] presentaron el modelo SAM, que ha sido implementado para segmentar automáticamente regiones afectadas en imágenes oculares. Li et al. [12] propusieron un enfoque innovador denominado TP-DRSeg, que adapta el modelo de SAM para la segmentación de lesiones de RD asistida por indicaciones textuales. Este marco combina modelos de visión y lenguaje para incorporar conocimiento médico previo en el proceso de segmentación. La metodología introduce un codificador de conocimientos previos, que traduce conceptos médicos en información procesable, y un inyector alineado que integra estas indicaciones en la red de segmentación.

En cuanto a la generación de imágenes sintéticas, Gatys et al. [13] propusieron NST como un método para combinar el contenido de una imagen con el estilo de otra. Según en el trabajo de Singh et al. [14] se mezclaron estilos de imágenes de fondo de ojo con características específicas de la RD para generar nuevas imágenes sintéticas que mantuvieran las características relevantes para el diagnóstico. Este enfoque permitió incrementar la cantidad de datos de clases menos representadas en el conjunto de entrenamiento.

Por otra parte, en el trabajo de Zhu et al. [15] se introdujo CycleGAN, una metodología de aprendizaje no supervisado que permite transformaciones entre dominios de imágenes sin necesidad de pares alineados. En el contexto de la RD, Sadok et al. [16] utilizaron una variante denominada Least Squares Cycle-GAN para abordar la falta de imágenes etiquetadas.

Sin embargo, otros investigadores han destacado que la incorporación de mecanismos de atención en los modelos generativos que puede mejorar significativamente la calidad y fidelidad de las imágenes sintéticas. Por ejemplo, Zhu et al. [17] desarrollaron el modelo PA-cycleGAN, una variante de CycleGAN que integra un módulo de atención basado en bloques convolucionales y codificación de características.

1.2. Planteamiento del problema

Actualmente, la RD representa una de las principales causas de ceguera en adultos jóvenes a nivel mundial, especialmente en personas con diabetes mellitus mal controlada. Esta enfermedad afecta progresivamente la retina, provocando daños en los vasos sanguíneos que, en etapas avanzadas, pueden generar pérdida irreversible de la visión. La detección temprana de biomarcadores, como microaneurismas, hemorragias, exudados duros y suaves, es importante para clasificar las etapas de la enfermedad y prevenir complicaciones graves.

Los avances en inteligencia artificial, particularmente en el PDI, han permitido desarrollar modelos automatizados basados en CNN y ViT que logran identificar y clasificar biomarcadores de la retinopatía diabética. En modelos como DRNet13 [5] y TP-DRSeg [12] han demostrado ser efectivos al integrar técnicas avanzadas de segmentación y atención, mejorando la detección de regiones afectadas en imágenes médicas. No obstante, estas metodologías suelen ser complejas, lo que puede representar un desafío para su implementación. En contraste, el enfoque propuesto se caracteriza por su simplicidad, ofreciendo una alternativa más directa y eficiente que logra resultados similares o incluso superiores en la identificación y clasificación de biomarcadores de la retinopatía diabética.

El principal problema de esta investigación radica en la necesidad de desarrollar un sistema integral que permita la detección, segmentación y así como la clasificación de biomarcadores en etapas de la retinopatía diabética en imágenes del fondo de ojo. El primer enfoque consiste en aplicar métodos de clasificación y segmentación basados en CNN y ViT para identificar biomarcadores y etapas de la RD con alta precisión. El segundo enfoque propone el uso de técnicas generativas, como NST y CycleGAN, para ampliar y diversificar los conjuntos de datos, mejorando así la robustez de los modelos entrenados.

Estas limitaciones evidencian la necesidad de sistemas automatizados que mejoren la eficiencia y precisión en el diagnóstico, optimizando tanto el tiempo como el esfuerzo empleados por el personal médico. La propuesta de este trabajo busca abordar estas dificultades mediante el desarrollo de un enfoque más accesible y eficaz para apoyar al especialista en la toma de decisiones clínicas.

1.3. Justificación

La RD es una enfermedad progresiva que afecta a millones de personas en todo el mundo, y su diagnóstico temprano requiere un análisis meticuloso de CFI. Sin embargo, este proceso manual es costoso, consume tiempo considerable y depende de la experiencia del especialista, lo que puede resultar en diagnósticos tardíos o inconsistentes en contextos clínicos sobrecargados. Es por ello que resulta indispensable recurrir a sistemas automatizados que incorporen técnicas de inteligencia artificial, permitiendo un análisis preciso y eficiente. Además, estos sistemas contribuyen a reducir el desgaste físico y mental del médico al optimizar su tiempo y esfuerzo en contextos de alta demanda clínica.

Una de las ventajas principales del sistema propuesto radica en la integración de modelos basados en CNN, los cuales han demostrado un rendimiento superior en la detección y clasificación de biomarcadores. Además, se implementarán técnicas de generación de imágenes sintéticas, como NST y CycleGAN, para resolver la problemática de escasez de imágenes sintéticas, una limitación frecuente en el entrenamiento de modelos de aprendizaje profundo.

Este avance permitirá a los especialistas en oftalmología contar con una herramienta eficiente que reduzca los tiempos de diagnóstico y el margen de error, al tiempo que optimiza el uso de recursos en entornos clínicos con alta demanda. Asimismo, el sistema contribuirá a superar las barreras relacionadas con la disponibilidad de imágenes, gracias a la generación controlada de imágenes sintéticas.

1.4. Hipótesis

La implementación de un sistema automatizado basado en modelos de inteligencia artificial como CNN, YOLO y técnicas de generación de imágenes sintéticas como NST y CycleGAN, junto con la segmentación empleando SAM, permitirá detectar, segmentar y clasificar biomarcadores de la retinopatía diabética en CFI con una precisión y eficiencia comparables o superiores a los métodos tradicionales, reduciendo significativamente el tiempo y esfuerzo necesario para el diagnóstico clínico y apoyando al médico en la toma de decisiones.

1.5. Objetivos de la investigación

1.5.1. Objetivo general

Desarrollar un sistema automatizado basado en CNN, YOLO y técnicas de generación de imágenes sintéticas como NST y CycleGAN, junto con segmentación empleando el modelo SAM, para la detección, segmentación y clasificación de biomarcadores en etapas de la RD en CFI.

1.5.2. Objetivos específicos

- Implementar un modelo de clasificación basado en CNN para identificar las diferentes etapas y biomarcadores de la retinopatía diabética en CFI.
- Integrar YOLO para realizar detección de biomarcadores en CFI.
- Incorporar el modelo SAM para realizar una segmentación detallada de los biomarcadores en las imágenes analizadas.
- Generar imágenes sintéticas mediante técnicas como NST y CycleGAN para aumentar la diversidad y balancear el conjunto de datos de entrenamiento.
- Evaluar el desempeño del sistema utilizando métricas como precisión, sensibilidad, especificidad, f1-score y matrices de confusión.

CAPÍTULO 2

INVESTIGACIÓN DEL ESTADO DEL ARTE

El estudio de las lesiones oculares asociadas a la retinopatía diabética representa un desafío técnico y clínico de gran relevancia en la actualidad [18]. Tal como ocurre con otras áreas de la visión artificial aplicada a la medicina, el abordaje de esta enfermedad se ha consolidado en varias vertientes: la detección de biomarcadores [19], la segmentación de estructuras relevantes [20] y la generación de imágenes sintéticas para mejorar la robustez de los modelos [21]. Estos métodos se sitúan en el estado del arte, pues responden a la necesidad de un análisis más confiable y eficiente de la retinopatía diabética, que facilite tanto la investigación como la asistencia médica en entornos clínicos [22] y de telemedicina [23].

La motivación principal que impulsa estas investigaciones es la búsqueda de técnicas más precisas para localizar lesiones (por ejemplo, microaneurismas [24] o hemorragias) y estructuras críticas (vasos sanguíneos [25], disco óptico, mácula), con la intención de reducir el error humano [26]. Además, surge la necesidad de procesar cantidades crecientes de datos procedentes de diversas fuentes (hospitales, centros de investigación y dispositivos portátiles), lo cual demanda sistemas con capacidad de generalizar a escenarios no vistos. Por otro lado, el interés en la generación de imágenes sintéticas se enfoca en aumentar la diversidad de conjuntos de datos, equilibrar la representación de diferentes etapas de la enfermedad y mitigar la escasez de imágenes debidamente anotadas [27].

De manera particular, en la literatura se identifican enfoques que apuestan por métodos de atención o transformadores de ViT [28]. Dichos métodos se valen del mecanismo de autoatención para procesar la imagen en parches y así capturar correlaciones globales, superando algunas limitaciones de los modelos convencionales [29]. Con ello, se logra un análisis integral de la retina, maximizando la sensibilidad ante cambios sutiles de contraste o forma. Asimismo, han surgido técnicas que introducen arquitecturas de segmentación [30], como en el caso de SAM [31].

En lo que respecta a la generación de imágenes sintéticas, se han documentado avances significativos que recurren a algoritmos de aprendizaje profundo con capacidad generativa [32]. Por ejemplo, métodos como NST permiten adaptar la apariencia de fondos de ojo normales hacia apariencias de retinopatía diabética, al combinar el contenido de una imagen base con el “estilo” propio de otra que presenta la patología. Este enfoque fortalece los sistemas que, al entrenarse con estas imágenes híbridas, mejoran su habilidad de detectar variaciones sutiles y aprenden a distinguir patrones complejos [33]. En una línea cercana, las arquitecturas de con-

versión de dominios como CycleGAN [34] propician la traducción bidireccional de imágenes entre retinas sanas y afectadas, garantizando la consistencia de ciclo al reconstruir imágenes de forma casi indistinguible de las originales. De este modo, se amplía el repertorio de datos disponibles y se facilita el entrenamiento de algoritmos, incluso en etapas tempranas de desarrollo cuando no se dispone de grandes volúmenes de datos clínicos [35].

A continuación, se listan algunas estrategias relevantes empleadas en el estado del arte para maximizar la precisión y la efectividad de los sistemas de detección, segmentación y generación de imágenes sintéticas en retinopatía diabética:

- **Modelos de Segmentación Basados en la Atención [36]:**
  - Aprovechan mecanismos de atención para resaltar lesiones pequeñas o difusas.
  - Facilitan la segmentación adaptativa de biomarcadores en función de la complejidad de la imagen.
- **Transformadores de Visión (ViT) [8]:**
  - Dividen la imagen en parches, permitiendo correlaciones de largo alcance.
  - Potencian la robustez ante variaciones en la iluminación y el color de la retina.
- **Técnicas de Generación de Imágenes Sintéticas (NST [13], CycleGAN [34]):**
  - Aumentan la diversidad de conjuntos de datos para entrenar y validar modelos.
  - Permiten la adaptación estilística y la traducción de dominio sin pérdida de información anatómica.
- **Sistemas de Segmentación Multi-Label (SAM [31] u otros modelos):**
  - Permiten la segmentación simultánea de múltiples estructuras (disco óptico, vasos sanguíneos, biomarcadores, etc.) dentro de una misma imagen, asignando etiquetas independientes a cada región.
  - Están diseñados para minimizar errores en la delimitación de regiones críticas, asegurando una segmentación precisa y consistente para el diagnóstico médico.

Estos enfoques pueden combinarse creativamente para aprovechar las fortalezas de cada técnica y ofrecer soluciones más completas. Por ejemplo, se pueden generar imágenes con métodos basados en CycleGAN y, a su vez, entrenar modelos avanzados de segmentación para delinear biomarcadores en dichas imágenes sintéticas. De manera adicional, algunas propuestas integran CNN como parte del cálculo de la pérdida perceptual en modelos generativos [37], permitiendo capturar características de alto nivel y refinar la calidad visual de las imágenes sintéticas. Asimismo, se ha planteado la posibilidad de fusionar CNN y ViT [38] en un esquema híbrido, con el fin de unir la capacidad de extracción de características locales de las convoluciones con la atención global de los transformadores. Dicha integración mejora la detección de variaciones mínimas de tonalidad, forma y textura, un aspecto particularmente relevante en las lesiones tempranas de la retinopatía diabética.

En conclusión, el panorama actual de investigación y desarrollo en la detección, segmentación y generación de imágenes para retinopatía diabética se presenta con un alto nivel de

sofisticación. El empleo de transformadores, la adopción de modelos de segmentación basados en atención y la integración de métodos generativos ofrece un abanico de posibilidades para robustecer los procesos de diagnóstico y análisis. Dadas las tendencias recientes, se prevé que estas tecnologías continúen madurando y adapten nuevas mejoras que permitan un abordaje más preciso, automatizado y, a la larga, accesible para diversos entornos clínicos. La combinación ingeniosa de las estrategias mencionadas seguirá reduciendo las brechas entre la investigación en laboratorio y la práctica médica en el campo de la oftalmología.

CAPÍTULO 3

FUNDAMENTOS

3.1. Fisiología Ocular

El ojo está compuesto por varias estructuras clave. En la parte frontal, se encuentran la córnea, el iris, el cristalino y el humor acuoso, que ayudan a enfocar la luz que entra al ojo.

En la parte posterior, se encuentran las coroides, el epitelio pigmentario de la retina, la retina y el humor vítreo, que son fundamentales para la formación y procesamiento de las imágenes visuales (como se puede ver en la Figura 3.1) [39].

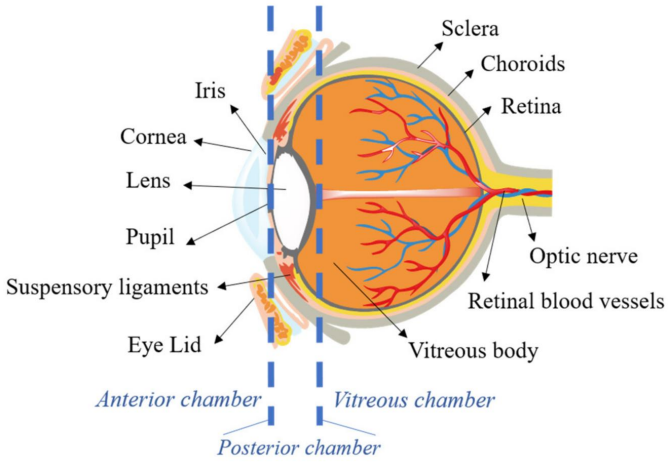


Figura 3.1: Diagrama de las estructuras principales del ojo humano.

3.1.1. Córnea

La córnea es el tejido más sensible del cuerpo humano, caracterizada por su transparencia, falta de vasos sanguíneos, delgadez y una alta densidad de nervios. Su grosor aumenta desde el centro hacia los bordes [40]. La córnea está compuesta por una gran cantidad de terminaciones nerviosas, incluyendo nervios sensoriales, fibras nerviosas simpáticas autónomas y nervios ciliares largos, lo que le otorga una densidad nerviosa superior a la de la piel [41]. El humor acuoso proporciona la glucosa y el oxígeno necesarios para el funcionamiento normal de la córnea [42].

3.1.2. Humor Acuoso

El humor acuoso es un líquido ocular transparente y ligeramente alcalino, que se produce continuamente por las células epiteliales del cuerpo ciliar, alcanzando una producción de aproximadamente 2,5 L/min en humanos. Este líquido se forma mediante difusión, ultrafiltración y secreción activa del plasma, y es fundamental para el suministro de nutrientes y oxígeno a la córnea y el cristalino, que son tejidos avasculares en la cámara anterior del ojo. La regulación del humor acuoso es crucial para mantener la presión ocular adecuada [43].

3.1.3. Iris

El iris se localiza detrás de la córnea y es una extensión del cuerpo ciliar, compuesto por endotelio, epitelio y estroma. La pupila, situada en el centro del iris, controla la cantidad de luz que ingresa a la retina. Por su parte, el cuerpo ciliar, situado en la parte anterior del iris, cumple funciones clave como la secreción y el drenaje del humor acuoso, además de contener un músculo que ajusta el cristalino para enfocar objetos o imágenes con precisión [44].

3.1.4. Cristalino

El cristalino es una estructura avascular, no inervada, transparente y con forma biconvexa, situada detrás del iris. La parte posterior del cristalino está en contacto con el humor vítreo, mientras que la parte anterior está en contacto con el humor acuoso. La cápsula del cristalino actúa como una barrera que regula los intercambios pasivos de metabolitos y desechos por difusión [45].

3.1.5. Nervio óptico

El nervio óptico está formado por millones de fibras nerviosas que transmiten mensajes visuales desde los ojos hacia el cerebro, permitiendo la visión. Cada ojo posee un nervio óptico que se conecta directamente con el cerebro, funcionando como una conexión unidireccional que solo envía señales visuales en esa dirección. Además de su función principal en la visión, algunas de las señales que transporta el nervio óptico también influyen en otros procesos y capacidades cerebrales [46].

3.1.6. Esclerótica

La esclerótica, conocida comúnmente como la parte blanca del ojo, es una estructura avascular, elástica y resistente que se encuentra debajo de la conjuntiva [47]. El nervio óptico atraviesa esta densa estructura, que constituye la capa externa del globo ocular y que suele verse afectada por factores externos como la presión intraocular [48].

3.1.7. Párpados

Los párpados son estructuras finas y móviles que cubren y resguardan la parte frontal del ojo. Su función principal es proteger el ojo de la luz excesiva y de posibles daños, además de

mantener la superficie ocular humectada al esparcir las lágrimas de manera uniforme. Los párpados constan de varias capas, como la piel, músculos, glándulas y nervios. Una capa clave es el músculo orbicular del ojo, que permite el cierre de los párpados, ya sea de manera voluntaria o como reflejo [49].

3.1.8. Coroides

La coroides es una capa altamente vascularizada e inervada situada entre la esclerótica y el epitelio pigmentario de la retina. Este tejido es importante para el suministro de nutrientes a la retina y para mantener el volumen y la temperatura del ojo. Histológicamente, la coroides está compuesta por cuatro capas: la supracoroidea, la capa de grandes vasos, la capa de vasos intermedios y la coriocapilar [50].

3.1.9. Epitelio Pigmentario Retiniano

El epitelio pigmentario de la retina (EPR) está compuesto por células indivisibles que forman una monocapa sobre la retina neural. Aunque estas células son en su mayoría indivisibles, pueden proliferar en ciertas condiciones patológicas. El EPR desempeña un papel protector para los tejidos internos del ojo y es responsable de la secreción de factores de crecimiento que mantienen la inmunidad ocular y protegen el ojo del daño oxidativo [51].

3.2. Retina

La retina es una capa de células fotorreceptoras y células gliales dentro del ojo que capta los fotones entrantes y los transmite a lo largo de vías neuronales como señales eléctricas y químicas para que el cerebro perciba una imagen visual. La retina está ubicada en el segmento posterior y forma el límite más interno entre las otras capas principales del ojo que incluyen la coroides vascular y la esclerótica fibrosa [52]. También es una capa de tejido nervioso sensible a la luz que recubre el interior del ojo y envía mensajes visuales a través del nervio óptico al cerebro [53].

3.2.1. Disco Óptico

El disco óptico es el punto en el ojo donde los axones de las células ganglionares se agrupan para formar el nervio óptico. En esta región, no hay receptores de luz como bastones o conos, lo que resulta en una zona sin percepción visual conocida como el punto ciego. Además, es por donde ingresan los vasos sanguíneos al ojo. La forma del disco óptico suele ser ligeramente ovalada en sentido vertical y no muestra variación significativa relacionada con el sexo, edad, peso o altura de la persona así como se ilustra en la Figura 3.2 [54, 55].

Figura 3.2: Imagen del disco óptico

3.2.2. Copa Óptica

La copa óptica es una depresión central dentro del disco óptico y representa la zona de menor densidad de fibras nerviosas en esta estructura. Su tamaño y forma varían entre individuos, pero en condiciones normales, tiene una orientación ovalada en sentido horizontal. A través de la copa óptica emergen los vasos sanguíneos principales de la retina, formando un punto clave en la circulación ocular, tal como se ilustra en la Figura 3.3 la ubicación de la copa óptica dentro del disco óptico [56].

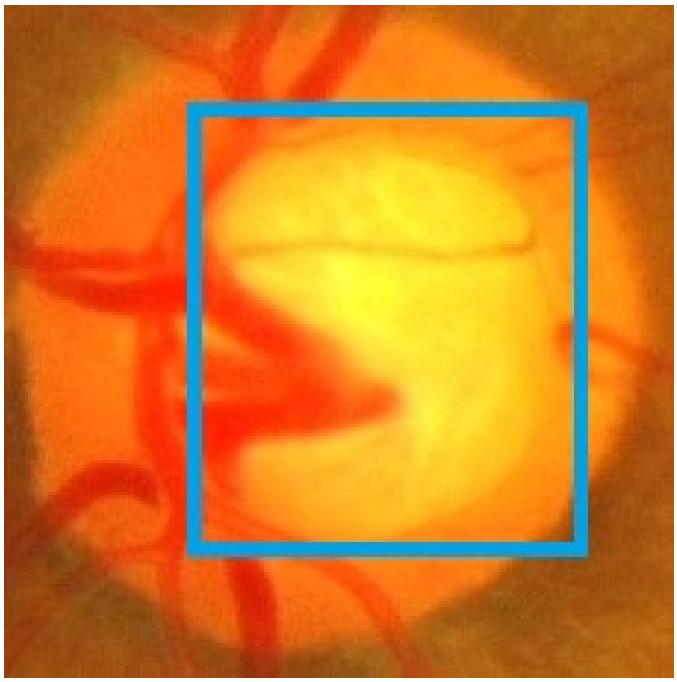


Figura 3.3: Imagen del disco óptico con un cuadro delimitador de color azul señalando la copa óptica

3.2.3. Vasos Sanguíneos

Los vasos sanguíneos de la retina forman una red ramificada que suministra oxígeno y nutrientes a las células de la retina. Se dividen en arterias y venas, las cuales pueden diferenciarse por su color y calibre. Las arterias tienen un tono más brillante y un diámetro menor, mientras que las venas son más oscuras y anchas. La disposición y el grosor de los vasos varían entre individuos, pero su organización general sigue un patrón característico. Como se ilustra en la Figura 3.4, los vasos sanguíneos emergen del disco óptico y se distribuyen a lo largo de la retina, formando una estructura clave para la circulación ocular [57].

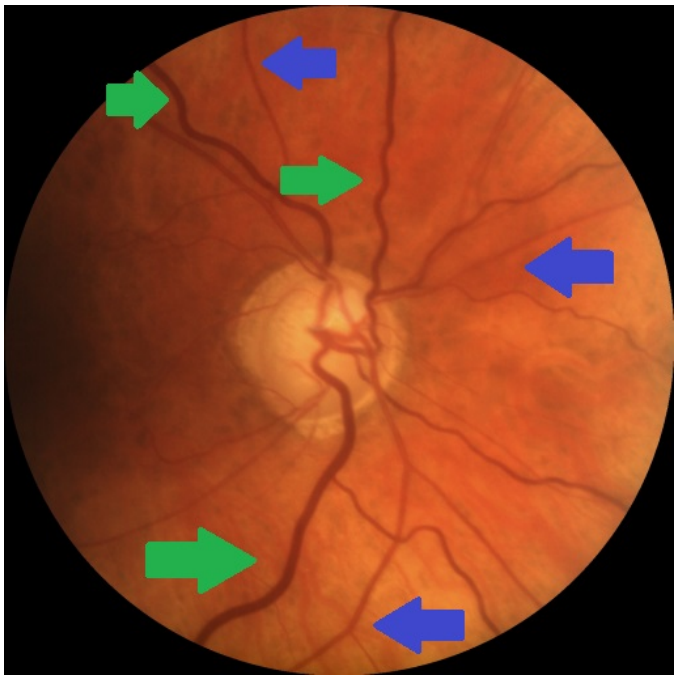


Figura 3.4: Imagen del fondo de ojo donde se observa la red de vasos sanguíneos de la retina. Las flechas azules señalan las arterias, mientras que las flechas verdes indican las venas.

3.2.4. Mácula

La mácula es una zona de la retina con una función altamente especializada, ubicada a aproximadamente dos diámetros del centro de la fóvea tal como se ilustra en la Figura 3.5. Esta región posee una gran cantidad de células de conos, que son responsables de la agudeza visual, la percepción del color y la visión central. Si la mácula o la fóvea están dañadas, es probable que se produzca una pérdida significativa de la visión central en el paciente debido a la reorganización y alteración de los fotorreceptores [58].

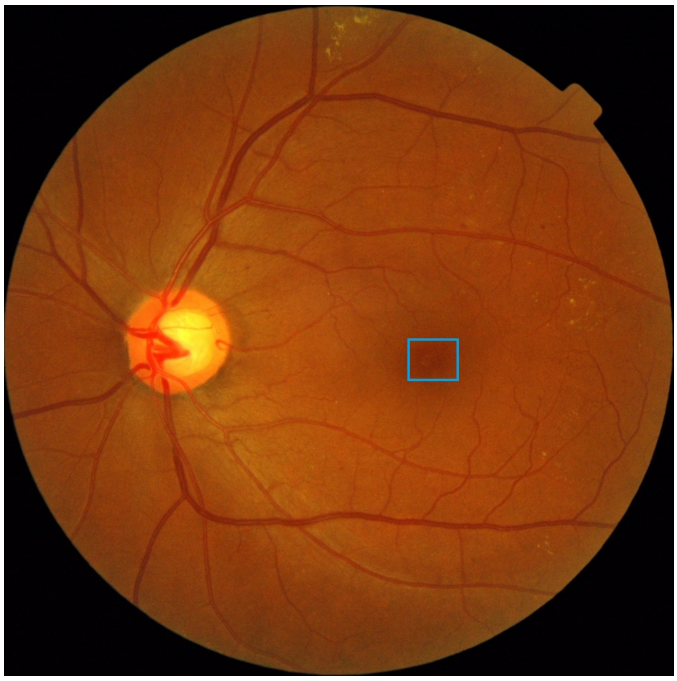


Figura 3.5: Imagen del fondo de ojo con un cuadro delimitador de color azul señalando la mácula.

3.2.5. Fóvea

La fóvea es una pequeña depresión ubicada en el centro de la mácula y representa la parte más delgada de la retina. Es responsable de proporcionar la visión más nítida y la mayor capacidad para distinguir colores, debido a la alta concentración de células de conos. Esta característica convierte a la fóvea en la región del ojo más involucrada en la visión central, la percepción del color y los detalles finos. Cuanto más densa es la disposición de estas células de conos, más precisa es la visión [58].

3.3. Modalidades de Imágenes en Oftalmología

El estudio y diagnóstico de enfermedades oculares se ha beneficiado del desarrollo de diversas modalidades de imágenes, cada una con características específicas que permiten la visualización detallada de diferentes estructuras del ojo. Entre las técnicas más utilizadas se encuentran CFI [59] y OCT [60].

3.3.1. Imágenes del Fondo de Ojo

La fotografía del fondo de ojo, también conocida como retinografía, es una técnica no invasiva utilizada para capturar imágenes detalladas de la retina. Se basa en cámaras oftalmológicas especializadas que emplean una fuente de luz, generalmente un flash, para iluminar el fondo ocular y registrar la luz reflejada en las estructuras retinianas [59].

En la Figura 3.6 se observa como este procedimiento permite obtener imágenes de alta resolución que son fundamentales para el diagnóstico y seguimiento de enfermedades como la retinopatía diabética [61].

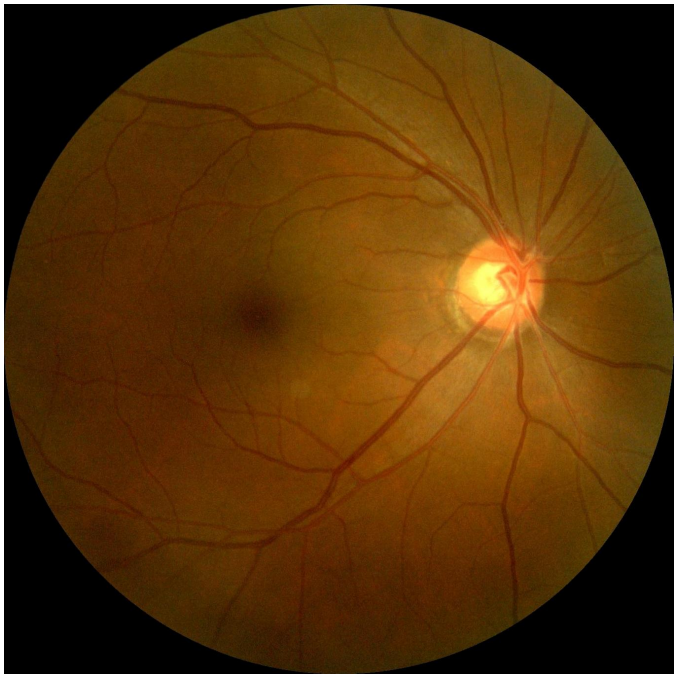


Figura 3.6: CFI sana.

3.3.2. Tomografía de Coherencia Óptica

La OCT es una técnica de imagen no invasiva basada en interferometría de baja coherencia que permite obtener cortes transversales de la retina con una resolución micrométrica así como se muestra en la Figura 3.7. A través de un haz de luz infrarroja, la OCT mide la reflexión de las capas retinianas, generando imágenes tridimensionales de alta resolución. Esta técnica es fundamental en la evaluación de patologías que afectan las capas internas de la retina y el nervio óptico [60].

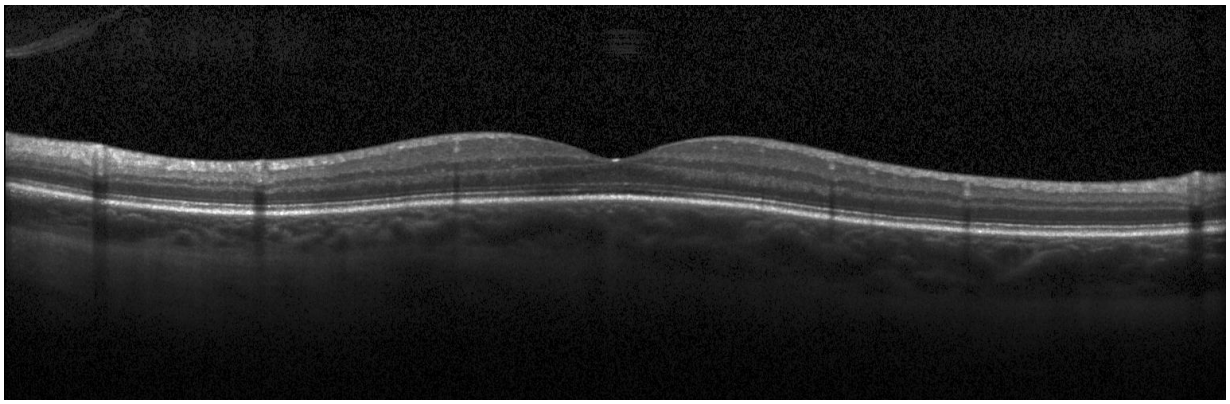


Figura 3.7: Imagen original de una tomografía de coherencia óptica (OCT) de una retina sana.

3.4. Retinopatía Diabética

La retinopatía diabética (RD) es una complicación microvascular de la diabetes mellitus, caracterizada por alteraciones en la circulación retiniana que pueden comprometer la función visual. Es una de las principales causas de ceguera irreversible en la población adulta a nivel mundial y representa un desafío significativo para los sistemas de salud debido a su prevalencia en aumento [62].

El impacto global de la enfermedad es preocupante, ya que se estima que millones de personas en todo el mundo padecen RD, especialmente en países con alta incidencia de diabetes [63]. De hecho, la detección temprana y el manejo oportuno de la enfermedad tanto en NPDR (de sus siglas en inglés, No Proliferative Diabetic Retinopathy) y PDR (de sus siglas en inglés, Proliferative Diabetic Retinopathy) han demostrado ser fundamentales para reducir la pérdida visual y mejorar la calidad de vida de los pacientes así como se muestra en la Figura 3.8 [64].

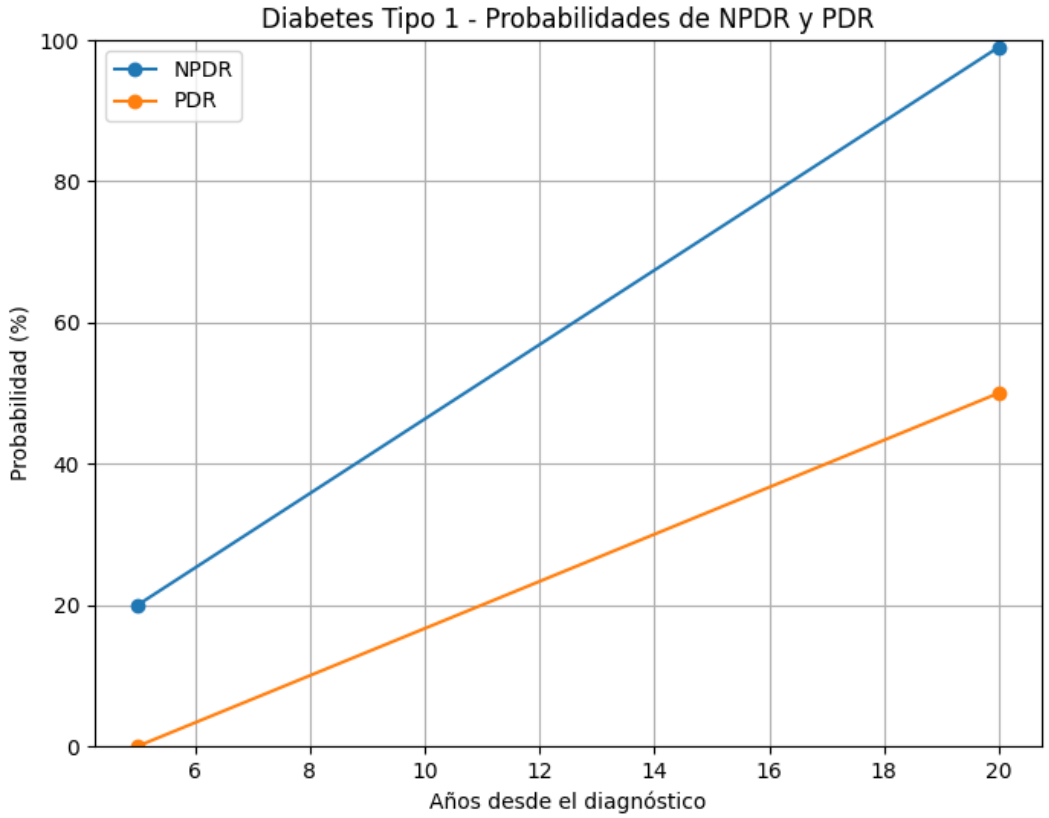
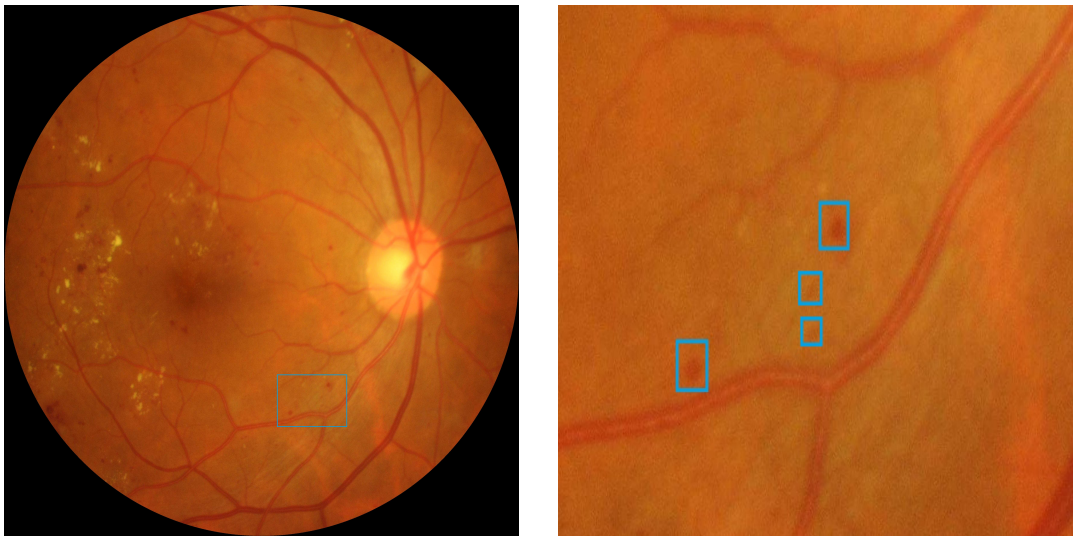


Figura 3.8: Probabilidad de desarrollar retinopatía diabética en pacientes con diabetes tipo 1 según los años desde el diagnóstico.

3.4.1. Biomarcadores en Retinopatía Diabética

Microaneurismas

Los microaneurismas son dilataciones focales de los capilares retinianos que se manifiestan como pequeñas estructuras redondeadas de color rojo en CFI así como se muestra en la Figura 3.9. Su presencia indica un daño temprano en la barrera hematorretiniana [65].



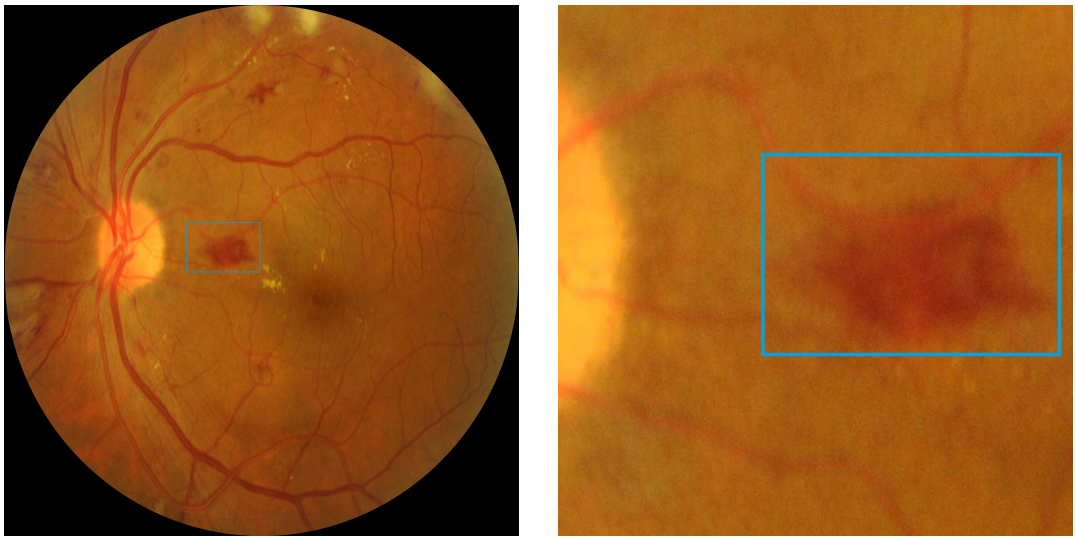
(a) (b)

Figura 3.9: Comparación de la visualización de microaneurismas en una CFI con retinopatía diabética: (a) CFI donde los cuadros delimitadores de color azul encierran las microaneurismas identificadas, (b) Ampliación de la misma imagen, proporcionando una vista más detallada de la microaneurisma de la retina.

Hemorragias

Las hemorragias son extravasaciones de sangre que pueden ocurrir en diferentes niveles de la retina, dependiendo de la capa afectada. Se manifiestan como lesiones de color rojo con diversas formas y tamaños en la CFI, tal como se observa en la Figura 3.10.

Estas hemorragias pueden estar asociadas a patologías como la retinopatía diabética, oclusiones vasculares y su evolución puede implicar la formación de cicatrices que puede afectar la visión [66].

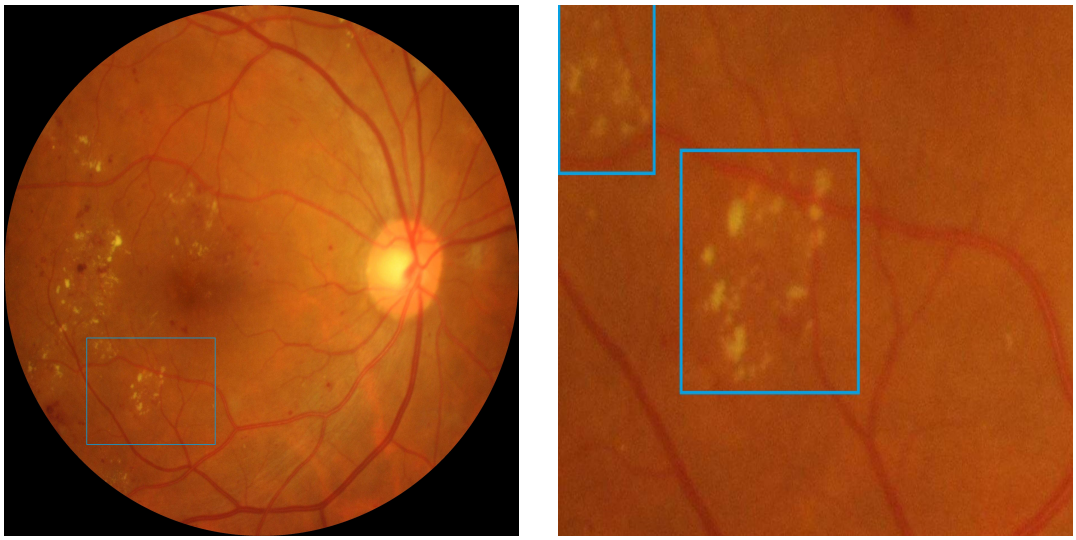


(a) (b)

Figura 3.10: Comparación de la visualización de hemorragias en una CFI con retinopatía diabética: (a) CFI donde los cuadros delimitadores de color azul encierran las hemorragias identificadas, (b) Ampliación de la misma imagen, proporcionando una vista más detallada de la hemorragia de la retina.

Exudados Duros

Los exudados duros son depósitos lipídicos que se presentan como lesiones amarillentas con un aspecto ceroso y bordes bien definidos en la CFI, como se muestra en la Figura 3.11. Su formación se debe a la extravasación de lípidos y proteínas plasmáticas desde los capilares dañados, lo que indica una alteración en la barrera hematorretiniana. Estos exudados pueden variar en tamaño y distribución, apareciendo de forma aislada o agrupada [67].



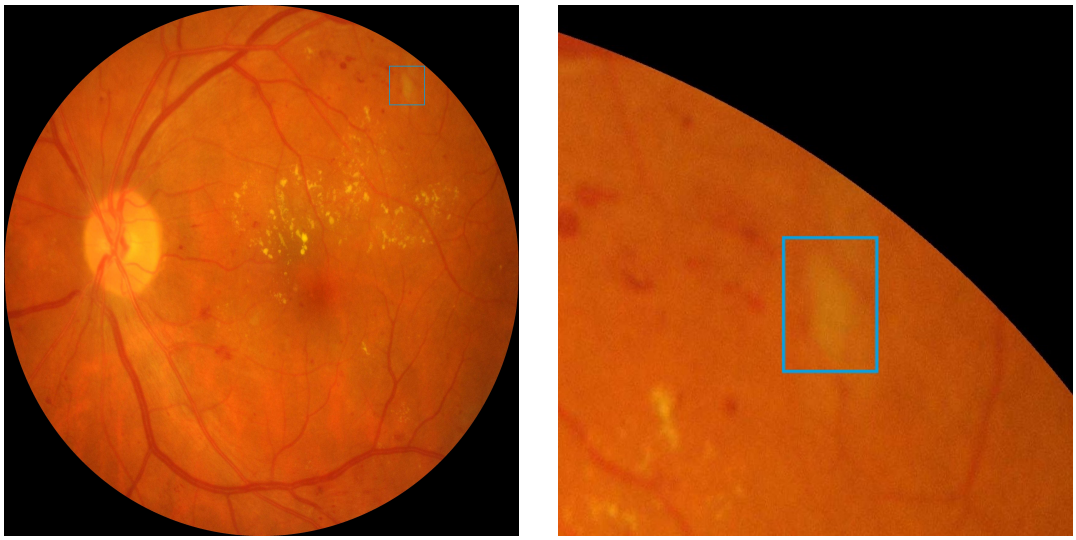
(a) (b)

Figura 3.11: Comparación de la visualización de exudados duros en CFI con retinopatía diabética: (a) CFI donde los cuadros delimitadores de color azul encierran los exudados duros identificados, (b) Ampliación de la misma imagen, proporcionando una vista más detallada de los exudados duros de la retina.

Exudados Suaves

Los exudados suaves, también conocidos como manchas algodonosas o microinfartos, son lesiones blanquecinas con bordes difusos que aparecen en la CFI, como se muestra en la Figura 3.12. Su origen se debe a la oclusión de las arteriolas terminales de la retina, lo que provoca isquemia localizada y acumulación de material axoplásmico en la capa de fibras nerviosas.

Estos exudados suelen asociarse con enfermedades que afectan la circulación retiniana, como la retinopatía diabética, la hipertensión arterial y ciertas patologías vasculares. Su presencia indica daño microvascular significativo y puede estar acompañada de otras alteraciones retinianas [68].



(a) (b)

Figura 3.12: Comparación de la visualización de exudados suaves en CFI con retinopatía diabética: (a) CFI donde los cuadros delimitadores de color azul encierran los exudados suaves identificados, (b) Ampliación de la misma imagen, proporcionando una vista más detallada de los exudados suaves de la retina.

3.4.2. Etapas de la Retinopatía Diabética

Su evolución se clasifica en diferentes etapas según la presencia y severidad de las lesiones retinales según U. Rajendra Acharya [69].

- **Sana:** En esta etapa, la retina no presenta signos de daño vascular. No se observan microaneurismas, hemorragias ni exudados, lo que indica una barrera hematorretiniana intacta y un flujo sanguíneo adecuado.
- **Leve:** Se caracteriza por la presencia de microaneurismas aislados, que son pequeñas dilataciones de los capilares retinianos. Pueden o no estar acompañados de exudados duros o manchas algodonosas. Esta etapa es indicativa de un daño temprano en la microcirculación, pero la función visual aún no está comprometida.
- **Moderada:** Se observa un aumento en la cantidad de microaneurismas y hemorragias intrarretinianas dispersas. Pueden aparecer manchas algodonosas y leve dilatación venosa, lo que refleja un mayor grado de isquemia y daño endotelial. En esta fase, el riesgo de progresión a formas más avanzadas es significativo.
- **Severa:** Se caracteriza por la presencia extensa de hemorragias y microaneurismas en los cuatro cuadrantes de la retina, con alteraciones venosas como dilataciones irregulares y bucles venosos. La hipoxia retiniana se intensifica, promoviendo la liberación de factores angiogénicos que pueden desencadenar la proliferación de nuevos vasos sanguíneos.
- **Proliferativa:** Es la fase más avanzada de la enfermedad. La hipoxia crónica induce la neovascularización, con el crecimiento de vasos sanguíneos frágiles y desorganizados en

la retina y el disco óptico. Estos vasos son altamente permeables y propensos a sangrados, aumentando el riesgo de hemorragias vítreas y desprendimiento traccional de la retina, lo que puede llevar a la pérdida severa de la visión.

El seguimiento periódico de la retina mediante imágenes de fondo de ojo es fundamental para detectar la progresión de la enfermedad y aplicar tratamientos oportunos como la fotocoagulación con láser o la terapia anti-VEGF [69].

3.5. Visión por Computadora

La visión por computadora es un campo de la inteligencia artificial que permite a las máquinas interpretar y analizar imágenes o videos de manera automática, imitando la capacidad visual humana. Su objetivo principal es extraer información útil de datos visuales para tareas como reconocimiento de objetos, segmentación de imágenes y detección de patrones [70]. El análisis de imágenes en visión por computadora requiere comprender su estructura y representación digital. En este contexto, los formatos de imagen y el rango de colores juegan un papel fundamental en la manipulación y procesamiento de datos visuales.

3.5.1. Lenguaje de Programación y Entorno de Desarrollo

El desarrollo de modelos de aprendizaje automático y aprendizaje profundo requiere un lenguaje de programación y entornos que faciliten la experimentación y visualización de datos. En este contexto, Python y Jupyter Notebook son dos de las herramientas más utilizadas.

Python

Python [1] es un lenguaje de programación de alto nivel, interpretado y de tipado dinámico. Se ha convertido en el estándar para la investigación y desarrollo en inteligencia artificial debido a su facilidad de uso, amplia comunidad. Su sintaxis clara y su enfoque en la productividad permiten una rápida implementación y prueba de modelos, convirtiéndolo en la opción preferida tanto en entornos académicos como en la industria [71].

Jupyter Notebook

Jupyter Notebook[2] es un entorno de desarrollo interactivo basado en celdas que permite ejecutar código Python de manera incremental. Se destaca por su capacidad de integrar texto formateado en gráficos y código en un solo documento, lo que facilita la exploración de datos, la documentación y la visualización de resultados. Además, permite probar fragmentos de código de forma modular sin necesidad de ejecutar el programa completo, lo que lo hace ideal para la investigación y el desarrollo de modelos de aprendizaje automático.

[1] Python. https://www.python.org/
[2] Jupyter Notebook. https://jupyter.org/

3.5.2. Librerías y Frameworks

El desarrollo y entrenamiento de modelos de aprendizaje automático y aprendizaje profundo requieren diversas librerías especializadas que facilitan el procesamiento de datos, la implementación de modelos y la evaluación de su desempeño. A continuación, se describen algunas de las más utilizadas en este campo.

NumPy

NumPy[3] es una librería fundamental para la estructura de datos en Python. Permite realizar operaciones eficientes sobre arreglos multidimensionales (ndarray), junto con funciones matemáticas. Es utilizada como base para muchas otras librerías, incluyendo TensorFlow y PyTorch.

Pandas

Pandas[4] proporciona estructuras de datos eficientes, como DataFrame y Series, que permiten la manipulación y análisis de datos estructurados. Facilita tareas como la carga, limpieza, transformación y exploración de datos, lo que la convierte en una herramienta esencial en el preprocesamiento de conjuntos de datos.

Scikit-Learn

Scikit-Learn [5] es una librería enfocada en el aprendizaje automático tradicional. Contiene implementaciones optimizadas de algoritmos de clasificación, regresión, clustering y reducción de dimensionalidad, así como herramientas para la validación cruzada, selección de características y ajuste de hiperparámetros. Es ampliamente utilizada para entrenar y evaluar modelos antes de recurrir a redes neuronales más complejas.

TensorFlow

TensorFlow [6] es un framework desarrollado por Google que permite la construcción y entrenamiento de modelos de aprendizaje profundo. Utiliza grafos computacionales para optimizar cálculos en hardware acelerado (GPUs y TPUs), permitiendo el desarrollo de modelos avanzados.

PyTorch

PyTorch [7], desarrollado por Facebook AI, es una alternativa a TensorFlow que ofrece una implementación más flexible y dinámica de redes neuronales.

[3] Numpy pip install. https://pypi.org/project/numpy/
[4] Pandas pip install. https://pypi.org/project/pandas/
[5] Scikit-Learn pip install. https://pypi.org/project/scikit-learn/
[6] Tensorflow pip install. https://pypi.org/project/tensorflow/
[7] Pytorch pip install. https://pypi.org/project/pytorch/

3.5.3. Estructura de la Imagen

Las imágenes digitales están compuestas por una matriz de píxeles, donde cada píxel almacena información de color e intensidad. Dependiendo de la naturaleza de la imagen y su uso en visión por computadora, existen diferentes formatos y modelos de color que definen la manera en que la información visual es codificada y procesada [72].

- **Valores de los Píxeles:**

  En una imagen digital, cada píxel contiene información que define su color o intensidad. En imágenes en escala de grises, cada píxel tiene un único valor en el rango de 0 a 255, donde 0 representa el negro absoluto, 255 el blanco absoluto y los valores intermedios corresponden a distintos tonos de gris [72].

- **Imágenes RGB:**

  En imágenes a color, cada píxel está compuesto por tres valores correspondientes a los canales de Rojo (R), Verde (G) y Azul (B), cada uno con un rango de 0 a 255 [72].

- **Tipo de Formatos:** Existen diversos formatos de archivo para almacenar imágenes, cada uno optimizado para diferentes aplicaciones [72]:

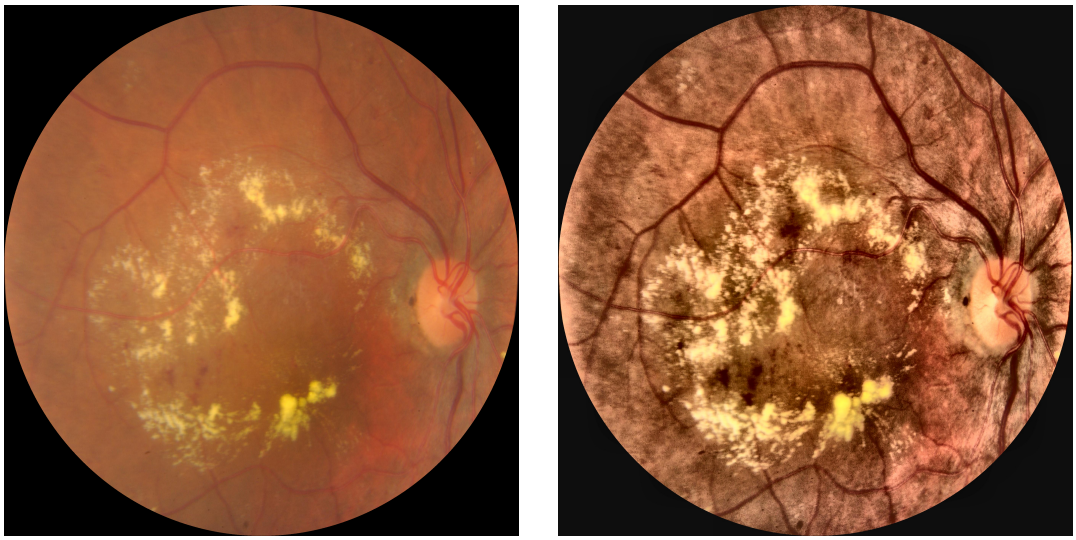
  - **JPEG:** Formato comprimido con pérdida, ampliamente utilizado en fotografía digital y almacenamiento eficiente de imágenes.
  - **PNG:** Formato sin pérdida que permite transparencia y es ideal para imágenes con detalles nítidos y colores planos.
  - **TIFF:** Usado en aplicaciones médicas y científicas debido a su alta fidelidad y almacenamiento sin pérdida.
  - **DICOM:** Específico para imágenes médicas, permitiendo almacenar metadatos relevantes junto con la imagen.

3.5.4. Preprocesamiento de Imágenes

El preprocesamiento de imágenes es una etapa en el análisis de imágenes con modelos de aprendizaje profundo. Consiste en transformar las imágenes antes de ser utilizadas en la red neuronal [73].

Filtrado

- **CLAHE:**
  El método CLAHE [74] es una técnica de mejora del contraste utilizada en procesamiento de imágenes. Se basa en el algoritmo de Histogram Equalization, pero con modificaciones que permiten evitar los problemas de amplificación excesiva del ruido en regiones homogéneas de la imagen así como se ilustra en la Figura 3.13. Su aplicación es especialmente útil en imágenes médicas y en imágenes con bajo contraste.

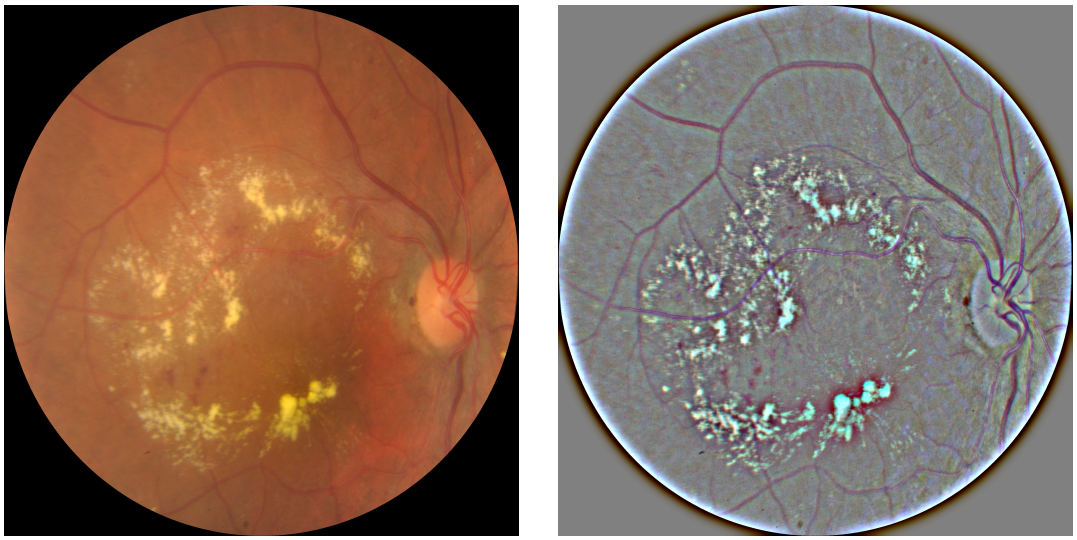


(a) (b)

Figura 3.13: Comparación de una imagen de retinopatía diabética. (a) Imagen original de retinopatía diabética. (b) Misma imagen con la aplicación del filtro CLAHE.

- **Filtro de Ben Graham:**
  El filtro de Ben Graham[8] constituye un método de procesamiento digital de imágenes que combina la idea de reducir selectivamente el ruido y, a la vez, resaltar estructuras de interés. Este enfoque parte de la superposición ponderada entre la imagen original y una versión suavizada por un desenfoque Gaussiano, introduciendo un término de compensación para mantener los valores de brillo e intensidad dentro de rangos adecuados tal como se ilustra en la Figura 3.14.

[8]Filtro de Ben Graham. https://www.kaggle.com/code/ratthachat/aptos-eye-preprocessing-in-diabetic-retinopathy



(a) (b)

Figura 3.14: Comparación de una imagen de retinopatía diabética. (a) Imagen original de retinopatía diabética. (b) Misma imagen con la aplicación del filtro de Ben Graham.

Normalización

La normalización en el procesamiento de imágenes es una técnica importante para estandarizar los valores de los píxeles y mejorar la estabilidad del entrenamiento en modelos de aprendizaje profundo. Su objetivo es ajustar la distribución de los valores de intensidad de los píxeles dentro de un rango específico, permitiendo una mejor convergencia en los modelos al reducir la variabilidad no deseada en las imágenes de entrada [73].

Aumento de Imágenes

El aumento de imágenes es una técnica en el preprocesamiento de datos para visión por computadora. Su objetivo es generar nuevas variaciones de una imagen original aplicando transformaciones con el fin de mejorar la robustez y generalización de los modelos de aprendizaje profundo. Esta estrategia es especialmente útil en conjuntos de datos limitados, ya que permite incrementar artificialmente la cantidad de ejemplos sin necesidad de recolectar nuevas muestras. Dentro de estas técnicas, el volteo horizontal y la rotación son estrategias fundamentales en este proyecto [73].

- **Volteo horizontal:**
  Esta técnica refleja la imagen con respecto al eje vertical, intercambiando el lado izquierdo con el derecho.

- **Rotación:**
  Consiste en girar la imagen en torno a su centro un ángulo θ, generando vistas desde diferentes perspectivas sin modificar la estructura interna de los objetos.

One-Hot Encoding

OHE (de sus siglas en inglés, One-Hot Encoding) es un método de codificación que transforma etiquetas categóricas en vectores binarios, donde cada posición representa una categoría específica [73].

El procedimiento de OHE consiste en mapear cada clase a un vector de longitud equivalente al número total de clases. En este vector, un único elemento es activado (valor 1) y los demás permanecen inactivos (valor 0). A diferencia de la clasificación multiclase, donde una muestra solo pertenece a una única categoría, en los problemas multi-etiqueta una misma imagen puede estar asociada a múltiples etiquetas simultáneamente. En este caso, el vector one-hot puede tener múltiples valores activados (1).

Por ejemplo, si una imagen presenta hemorragias y exudados duros, la codificación resultante sería:

$$[0, 1, 1, 0, 0] \tag{3.1}$$

donde cada posición del vector representa un biomarcador específico dentro del modelo multietiqueta.

Desbalance de Datos

En problemas de clasificación, especialmente en el ámbito médico, es común que algunas clases estén representadas con muchas más muestras que otras. Este fenómeno, conocido como desbalance de datos [75], puede sesgar el modelo hacia la clase mayoritaria, reduciendo su capacidad de generalización y disminuir su rendimiento en la detección de clases minoritarias. Para mitigar este problema, se emplean estrategias como el submuestreo y el sobremuestreo, las cuales se describen a continuación:

- **Submuestreo:**
  Consiste en reducir el número de muestras de la clase mayoritaria para equilibrar la distribución de datos [75] [76].

- **Sobremuestreo:**
  Aumenta artificialmente la cantidad de muestras de la clase minoritaria para balancear el conjunto de datos [75] [76].

El uso de estas técnicas depende del contexto del problema y de la cantidad de datos disponibles. Un balance adecuado entre clases permite que el modelo aprenda de manera más equitativa y mejore su desempeño en la predicción de todas las categorías.

3.5.5. Tareas de Visión por Computadora

La visión por computadora permite la automatización de diversas tareas de análisis e interpretación de imágenes [77]. A continuación, se describen las principales tareas en las que se emplean modelos de aprendizaje profundo.

Clasificación de Imágenes en Multiclase

La clasificación de imágenes en multiclase es una tarea en la que el modelo asigna cada imagen a una única categoría dentro de un conjunto de clases. Se representa mediante una distribución de probabilidad sobre todas las categorías posibles, donde la clase con mayor probabilidad se asigna como la predicción final [73].

Clasificación de Imágenes en MultiEtiqueta

A diferencia de la clasificación multiclase, en la clasificación multietiqueta una imagen puede pertenecer a múltiples categorías simultáneamente. En este caso, la salida del modelo es un vector de etiquetas binarias, donde cada entrada indica la presencia o ausencia de una categoría en la imagen [73]. En el caso de las imágenes de fondo de ojo, una misma imagen puede contener múltiples biomarcadores de la retinopatía diabética, como la presencia simultánea de exudados duros y hemorragias.

Segmentación de Imágenes

La segmentación de imágenes es una técnica que permite dividir una imagen en regiones basadas en características específicas, asignando a cada píxel una categoría determinada. En este proceso, los píxeles que pertenecen a una región de interés, como lesiones en imágenes médicas, se diferencian del fondo y de otras estructuras. Dependiendo del enfoque utilizado, la segmentación puede generar una máscara binaria donde los píxeles de la región segmentada se representan en blanco y el resto en negro, o una segmentación multicategoría donde cada región se asigna a un valor diferente según su clase [73].

Detección de Objetos

La detección de objetos es una combinación de clasificación y localización, donde el modelo identifica la presencia de objetos en la imagen y dibuja una caja delimitadora alrededor de cada instancia detectada [73].

Generación de Imágenes

La generación de imágenes se centra en la creación de imágenes sintéticas a partir de modelos de aprendizaje profundo, permitiendo mejorar, restaurar o crear nuevas representaciones visuales [73].

3.6. Computación Acelerada

El entrenamiento de modelos de aprendizaje profundo requiere una gran cantidad de operaciones matemáticas, cálculos matriciales y operaciones con tensores. Para optimizar estos procesos, se utilizan diferentes tipos de hardware que permiten acelerar las operaciones y reducir el tiempo de cómputo. En este contexto, se destacan tres tipos de procesadores:

3.6.1. CPU

La CPU (de sus siglas en inglés, Central Processing Unit) es el procesador principal de un sistema. Su arquitectura está diseñada para ejecutar instrucciones de propósito general con gran flexibilidad. Las CPUs modernas cuentan con múltiples núcleos, lo que permite realizar varias tareas en paralelo mediante el procesamiento multinúcleo. Sin embargo, debido a su diseño orientado a la ejecución secuencial y a la optimización de tareas generales, las CPUs suelen ser menos eficientes que las GPUs y TPUs para cálculos intensivos en redes neuronales, donde se requieren miles de operaciones matemáticas en paralelo [73].

3.6.2. GPU

La GPU (de sus siglas en inglés, Graphics Processing Unit) fue originalmente diseñada para el procesamiento de gráficos y renderizado de imágenes. No obstante, su arquitectura altamente paralelizable la hace ideal para tareas de aprendizaje profundo, donde miles de operaciones pueden ejecutarse simultáneamente [78] [79] [73]. A diferencia de la CPU, que posee un número reducido de núcleos de propósito general, la GPU cuenta con miles de núcleos pequeños especializados en cálculos matriciales y operaciones sobre tensores. Esto permite acelerar significativamente el entrenamiento de modelos de aprendizaje profundo. Frameworks como TensorFlow y PyTorch permiten aprovechar la aceleración mediante GPUs a través de bibliotecas como CUDA y cuDNN de NVIDIA.

3.6.3. TPU

La TPU (de sus siglas en inglés, Tensor Processing Unit) es un procesador desarrollado por Google específicamente para acelerar tareas de inteligencia artificial y aprendizaje profundo. A diferencia de las CPUs y GPUs, las TPUs están diseñadas exclusivamente para operar con tensores de manera eficiente, lo que permite ejecutar redes neuronales con un consumo energético mucho menor y a una gran velocidad [80]. Son utilizadas en infraestructuras en la nube como Google Cloud y optimizadas para frameworks como TensorFlow.

3.7. Aprendizaje Profundo

El aprendizaje profundo es una subdisciplina del aprendizaje automático que emplea redes neuronales artificiales de múltiples capas para modelar representaciones complejas de los datos. Estos métodos han mejorado drásticamente el estado del arte en el reconocimiento de voz, el reconocimiento visual de objetos, la detección de objetos. El aprendizaje profundo descubre estructuras intrincadas en grandes conjuntos de datos mediante el uso del algoritmo de retropropagación para indicar cómo debe cambiar sus parámetros internos que se utilizan para calcular la representación en cada capa a partir de la representación en la capa anterior [81] [73].

3.7.1. Redes Neuronales Convolucionales

Las CNN son un tipo de arquitectura especializada en el procesamiento de datos en forma de imágenes. Su principal ventaja es la capacidad de detectar patrones locales mediante el uso

de filtros convolucionales, permitiendo extraer características relevantes [73]. Las CNN reducen la cantidad de parámetros al compartir pesos entre regiones de la imagen, lo que las hace más eficientes en el procesamiento de imágenes a gran escala.

Kernel

El kernel en una CNN es un pequeño filtro matricial que se desliza sobre la imagen de entrada para extraer características específicas, como bordes, texturas y estructuras. También conocido como filtro convolucional, el kernel aplica una operación de convolución sobre la imagen, generando un mapa de características que resalta patrones importantes para la tarea de aprendizaje [73].

Los tamaños de kernel más comunes son 2×2 o 3×3, dependiendo de la granularidad de las características a extraer. Kernels más pequeños capturan detalles finos, mientras que los más grandes permiten detectar patrones más globales en la imagen. En las primeras capas de una CNN, los kernels suelen detectar bordes y contornos, mientras que en capas más profundas capturan representaciones como formas y texturas complejas.

Por ejemplo, la Figura 3.15 muestra dos posibles conjuntos de pesos, llamados kernels. El primero se representa como un cuadrado negro con una línea blanca vertical en el medio (es una matriz de 7×7 llena de 0, excepto la columna central, que está llena de 1); las neuronas que usan estos pesos ignorarán todo lo que se encuentre en su campo receptivo, excepto la línea vertical central (ya que todas las entradas se multiplicarán por 0, excepto las ubicadas en la línea vertical central). El segundo filtro es un cuadrado negro con una línea blanca horizontal en el medio. Una vez más, las neuronas que usan estos pesos ignorarán todo lo que se encuentre en su campo receptivo, excepto la línea horizontal central.

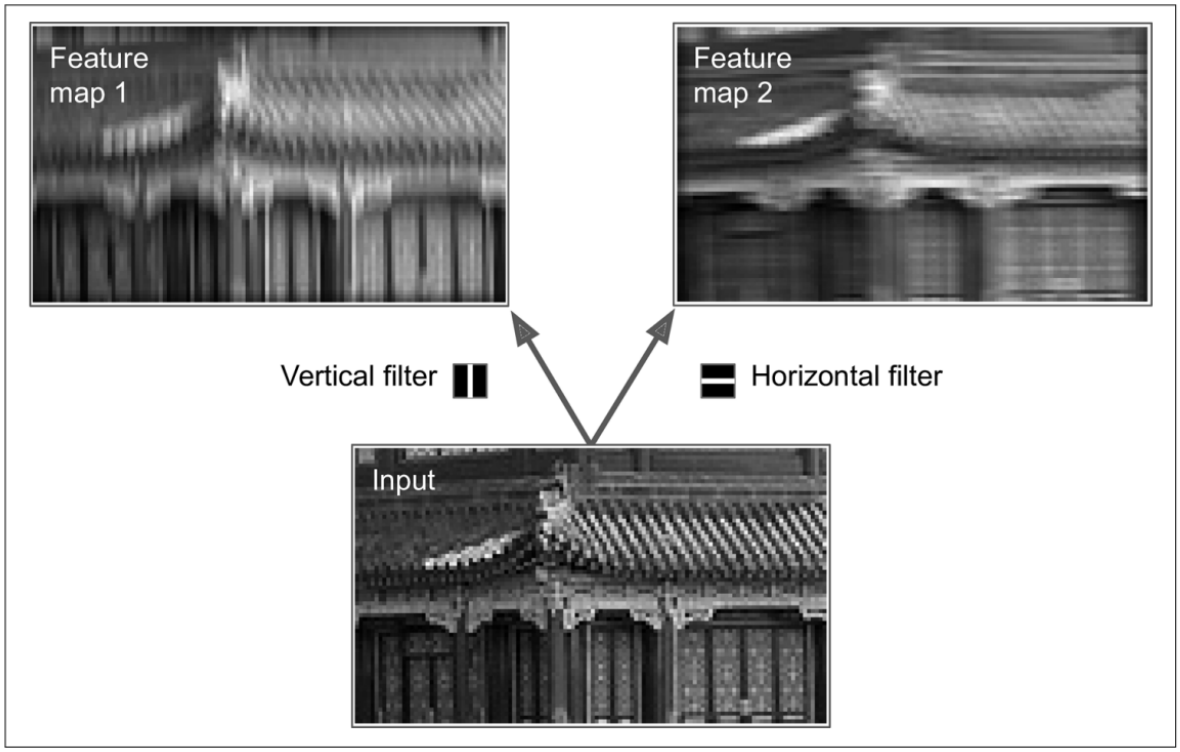


Figura 3.15: Proceso de aplicación de dos filtros diferentes para obtener dos mapas de características.

Stride

El stride determina el desplazamiento del filtro sobre la imagen en cada iteración [73]. Un stride de 1 implica que el filtro se mueve un píxel a la vez, mientras que un stride de 2 o mayor hace que el filtro avance más rápidamente, reduciendo el tamaño espacial de la salida.

El stride afecta directamente la resolución de las características extraídas. Valores pequeños 1×1 preservan más detalles pero aumentan el costo computacional, mientras que valores mayores reducen el tamaño de la representación y pueden perder información.

Capas de Convolución

Las capas de convolución son el componente central de las CNN, responsables de la extracción de características a partir de imágenes [73]. Cada capa aplica un conjunto de kernels que recorren la imagen de entrada mediante operaciones de convolución, produciendo mapas de características que resaltan patrones específicos como bordes, texturas y formas tal como se ilustra en la Figura 3.16.

La operación de convolución se expresa mediante la siguiente ecuación:

$$O(i,j) = \sum_m \sum_n I(i+m, j+n) \cdot K(m,n) \tag{3.2}$$

donde:

- $I(i,j)$ es el valor del píxel en la posición (i,j) de la imagen de entrada.

- $K(m, n)$ representa el filtro de convolución de tamaño $k \times k$.
- $O(i, j)$ es el valor resultante en la salida.

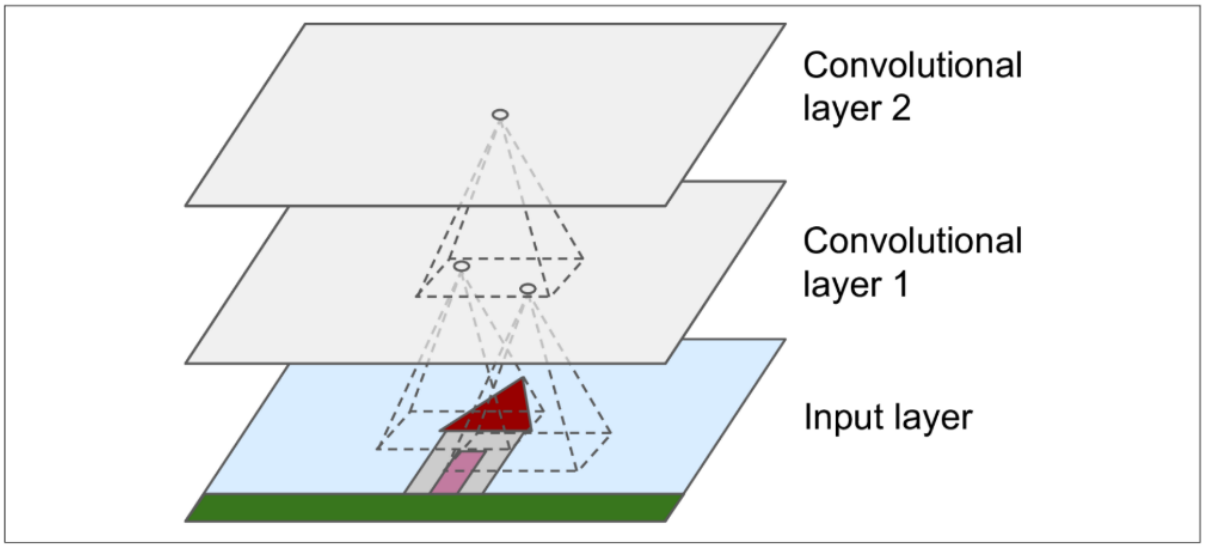


Figura 3.16: Proceso de extracción de características en capas de convolución.

Capas de Zero-Padding

En redes neuronales convolucionales, las capas de Zero-Padding consisten en la adición de ceros alrededor de los bordes de la imagen de entrada antes de aplicar la convolución. Al aplicar una convolución con un filtro de tamaño $k \times k$, la imagen de salida se reduce en dimensiones así como se muestra en la Figura 3.17. Para evitar esta reducción y mantener el tamaño original, se añaden ceros en los bordes, permitiendo que la convolución se aplique sin pérdida de información espacial. Sin el uso de zero-padding, los píxeles en los bordes de la imagen se verían menos veces que los píxeles centrales, lo que podría provocar un sesgo en el aprendizaje del modelo. La adición de padding ayuda a que todas las regiones de la imagen sean tratadas de manera más uniforme [73].

El tamaño del zero-padding se define como p, y su efecto en la dimensión de salida de la convolución puede calcularse con la ecuación:

$$O = \frac{(I - k + 2p)}{s} + 1 \tag{3.3}$$

donde:

- O es la dimensión de salida.
- I es el tamaño de la imagen de entrada.
- k es el tamaño del kernel.
- p es la cantidad de padding aplicado.

- s es el stride.

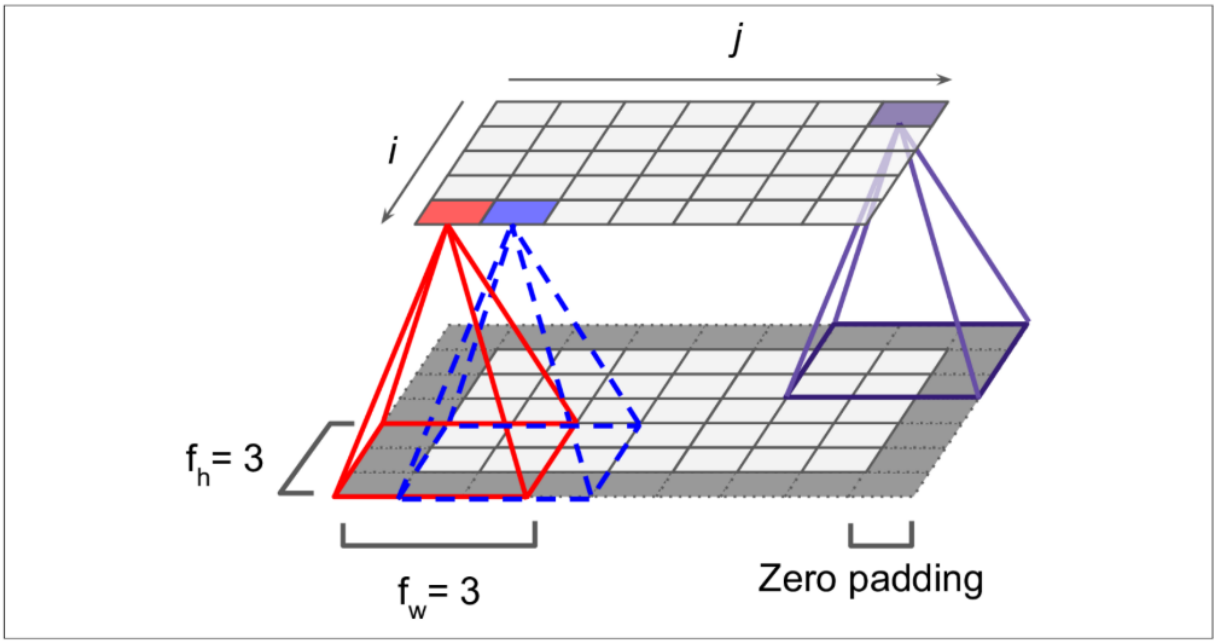


Figura 3.17: Proceso de conexión entre capas de convolución y zero padding.

Capas de MaxPooling

Las capas de MaxPooling son un tipo de operación en redes neuronales convolucionales utilizadas para reducir la dimensionalidad espacial de las imágenes manteniendo las características más relevantes [73]. Su función principal es realizar un submuestreo espacial mediante la selección del valor máximo dentro de una ventana de tamaño $k \times k$ que se desliza sobre la imagen con un determinado stride así como se ilustra en la Figura 3.18. Al disminuir la cantidad de datos en la imagen, se optimiza el rendimiento computacional. A diferencia de otros métodos de reducción de tamaño, el MaxPooling conserva las características dominantes en cada región analizada.

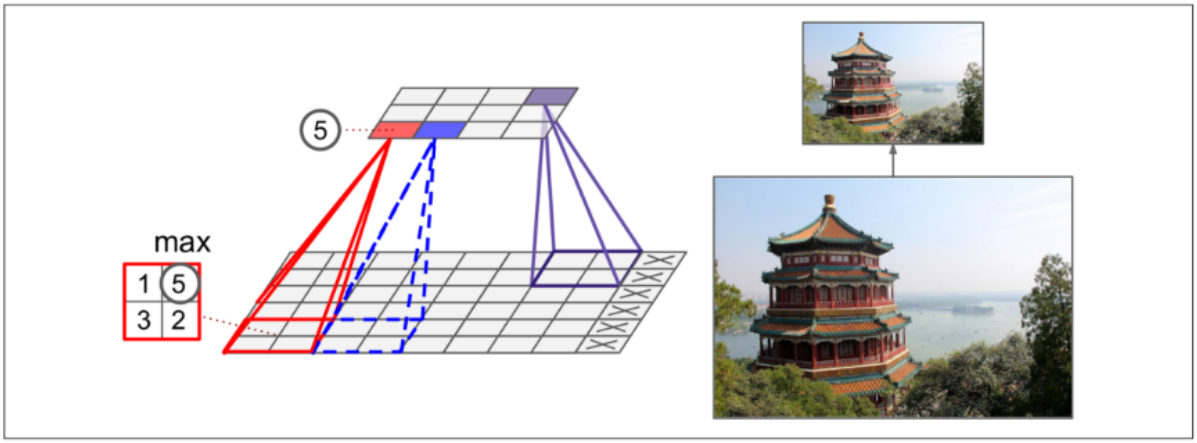


Figura 3.18: Proceso de la capa de max pooling en la extracción de características relevantes (2 × 2 pooling kernel, stride 2, no padding).

Capas de Average Pooling

Las capas de Average Pooling son un tipo de operación en CNN utilizadas para reducir la dimensionalidad espacial de las imágenes mientras se preserva la información relevante. A diferencia de MaxPooling, que selecciona el valor máximo en una ventana determinada, Average Pooling calcula el promedio de los valores en la región de la ventana deslizante. Al promediar los valores dentro de cada región, la capa ayuda a reducir la variabilidad en la activación de los píxeles [73].

Capas de Flatten

Las capas de Flatten se utilizan en CNN para convertir una matriz multidimensional de características en un vector unidimensional. Esta transformación es necesaria cuando se conectan las capas convolucionales con las capas densas o fully connected layers en una red neuronal [73]. La salida de una capa convolucional es un tensor tridimensional con dimensiones $(H \times W \times C)$, donde H y W representan la altura y anchura de la imagen, y C es el número de canales. La capa Flatten convierte este tensor en un vector de dimensión $H \times W \times C$. La capa Flatten permite que la representación extraída por las capas convolucionales sea interpretada por las capas densas.

Capas Densas

Las capas densas en una red neuronal corresponden a la etapa final del procesamiento de características extraídas por capas convolucionales o de pooling y aprenden patrones complejos a través de pesos entrenables. En estas capas, cada neurona está conectada con todas las neuronas de la capa anterior, permitiendo una combinación global de la información [73]. Cada neurona de una capa densa realiza una transformación lineal seguida de una función de activación.

Este proceso completo define el flujo de trabajo de una CNN, donde cada etapa cumple una función específica tal como se ilustra en la Figura 3.19.

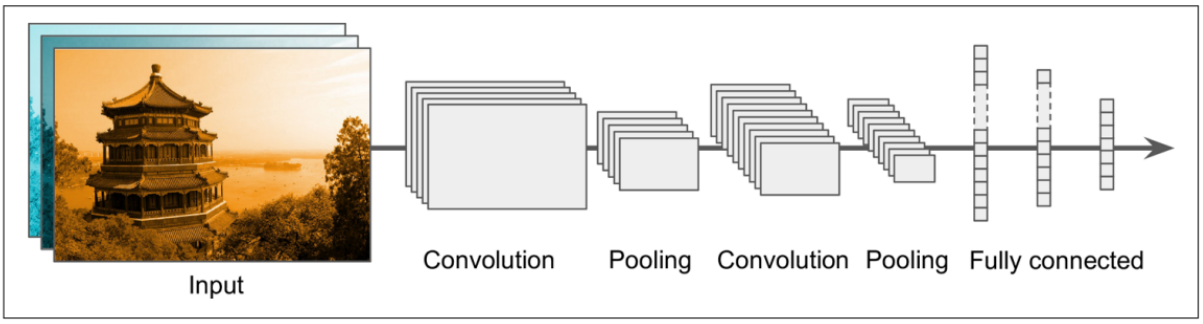


Figura 3.19: Flujo de trabajo en la creación de una arquitectura típica de una CNN.

3.7.2. Regularización

La regularización en redes neuronales es una técnica utilizada para reducir el sobreajuste y mejorar la generalización del modelo.

Validación Cruzada K-Fold

La validación cruzada K-Fold es una técnica utilizada para evaluar el rendimiento de un modelo dividiendo el conjunto de datos en K subconjuntos o folds. El objetivo es mitigar el sesgo introducido por la selección del conjunto de entrenamiento y prueba, asegurando que el modelo se evalúe en diferentes particiones de los datos [82]. El proceso de validación cruzada K-Fold sigue los siguientes pasos:

1. Dividir el conjunto de datos en K subconjuntos de tamaño similar.
2. Entrenar el modelo K veces, utilizando en cada iteración $K-1$ subconjuntos para entrenamiento y el restante para prueba.
3. Calcular la métrica de evaluación en cada iteración y promediar los resultados obtenidos para obtener una medida más robusta del desempeño del modelo.

Este método reduce la varianza en la evaluación del modelo, proporcionando una estimación más confiable del desempeño general.

L1 y L2

Existen dos formas principales de regularización aplicadas a los pesos de la red. **L1**: También conocida como Lasso, esta técnica agrega una penalización proporcional al valor absoluto de los pesos del modelo [73]. Su función de pérdida se expresa como:

$$J(\theta) = J_{\text{original}} + \lambda \sum_i |\theta_i| \tag{3.4}$$

donde λ es el factor de regularización que controla la intensidad del castigo aplicado a los pesos. Esta regularización tiende a generar modelos más esparsos, eliminando pesos innecesarios al forzarlos a cero.

L2: Conocida como Ridge, esta técnica agrega una penalización proporcional al cuadrado de los pesos [73]. Su función de pérdida es:

$$J(\theta) = J_{\text{original}} + \lambda \sum_i \theta_i^2 \tag{3.5}$$

En este caso, en lugar de forzar pesos a cero, los reduce en magnitud, lo que ayuda a controlar su impacto en el modelo y a estabilizar el entrenamiento.

Dropout

Dropout es una de las técnicas de regularización más utilizadas en redes neuronales profundas. Fue propuesta por Geoffrey Hinton en 2012 [83] y detallada más a fondo por Nitish Srivastava et al. en 2014 [84]. Se ha demostrado que agregar dropout a un modelo puede mejorar su precisión en un 1–2 %, lo cual puede reducir drásticamente la tasa de error en modelos con alta precisión.

El principio detrás de dropout es que durante cada paso de entrenamiento, cada neurona en la red (excepto las de la capa de salida) tiene una probabilidad p de ser desactivada temporalmente. Es decir, su contribución es ignorada en ese paso de entrenamiento, pero en el siguiente puede volver a estar activa. Este hiperparámetro p se conoce como la **tasa de dropout** y suele establecerse entre el 10 % y el 50 %, dependiendo del tipo de red. En redes neuronales recurrentes (RNNs), se usa típicamente entre 20 %-30 %, mientras que en redes convolucionales (CNNs) se emplean valores más altos, entre 40 %-50 %.

El mecanismo de dropout tiene un efecto similar al de entrenar múltiples redes neuronales con diferentes arquitecturas y promediar sus predicciones, lo que lo convierte en una forma de ensamblado eficiente. Esta técnica evita la coadaptación excesiva entre neuronas, obligándolas a ser más robustas y a no depender demasiado de ciertas entradas. Como resultado, la red se vuelve menos sensible a cambios leves en los datos de entrada y generaliza mejor a datos no vistos [73].

3.7.3. Funciones de Activacion

Las funciones de activación son componentes fundamentales en las redes neuronales artificiales, ya que introducen no linealidad en el modelo y permiten aprender relaciones complejas en los datos. Su propósito es transformar la suma ponderada de las entradas de una neurona en una salida que pueda ser utilizada en la siguiente capa de la red [73].

ReLU

Las ReLU (de sus siglas en inglés, Rectified Linear Units) representan una de las funciones de activación más utilizadas en redes neuronales profundas debido a su simplicidad y eficiencia en la propagación del gradiente [85]. Su expresión matemática está dada por:

$$f(x) = \text{máx}(0, x) \tag{3.6}$$

Esta función actúa como un filtro que anula los valores negativos y deja pasar los positivos de manera lineal. A diferencia de funciones sigmoide o tanh, ReLU no satura los valores positivos, permitiendo que el gradiente fluya de manera efectiva durante la retropropagación.

Softmax

La función Softmax es una función de activación ampliamente utilizada en problemas de clasificación multiclase. Su principal objetivo es convertir un vector de valores reales en una distribución de probabilidad sobre diferentes clases [73].

Dada una entrada $\mathbf{z} = [z_1, z_2, \ldots, z_K]$ correspondiente a las activaciones de la capa final de una red neuronal, la función Softmax se define como:

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}, \quad \text{para } i = 1, 2, ..., K. \tag{3.7}$$

Esta ecuación garantiza que:

- Cada valor de salida $\sigma(z_i)$ está en el rango (0,1), lo que lo hace interpretable como una probabilidad.
- La suma de todas las probabilidades es igual a 1, es decir, $\sum_{i=1}^{K} \sigma(z_i) = 1$.
- La clase con la mayor probabilidad $\sigma(z_i)$ se considera la predicción final del modelo.

Sigmoide

La función Sigmoide es una función de activación utilizada en redes neuronales, especialmente en problemas de clasificación binaria. Su principal característica es que convierte cualquier valor real en un rango comprendido entre 0 y 1, lo que facilita la interpretación como una probabilidad [73].

Dada una entrada z, la función sigmoide se define como:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{3.8}$$

Esta ecuación garantiza que:

- El valor de salida siempre estará en el intervalo $(0, 1)$, lo que permite interpretarlo como una probabilidad.
- Si z es grande y positivo, $\sigma(z)$ se aproxima a 1.
- Si z es grande y negativo, $\sigma(z)$ se aproxima a 0.
- Para $z = 0$, la función retorna un valor de 0.5.

Tanh

La función Tanh (de sus siglas en inglés, Hyperbolic Tangent) es una función de activación ampliamente utilizada en redes neuronales. A diferencia de la función sigmoide, cuya salida está en el rango $(0, 1)$, la función $\tanh$ produce valores en el intervalo $(-1, 1)$, lo que la hace más adecuada para ciertas aplicaciones de aprendizaje profundo [73].

Dada una entrada z, la función tangente hiperbólica se define como:

$$\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \tag{3.9}$$

Esta ecuación garantiza que:

- Para valores grandes y positivos de z, $\tanh(z)$ se aproxima a 1.
- Para valores grandes y negativos de z, $\tanh(z)$ se aproxima a -1.
- Para $z = 0$, la función retorna un valor de 0.

En la Figura 3.20 se ilustra las derivadas de las funciones de activaciones mencionadas.

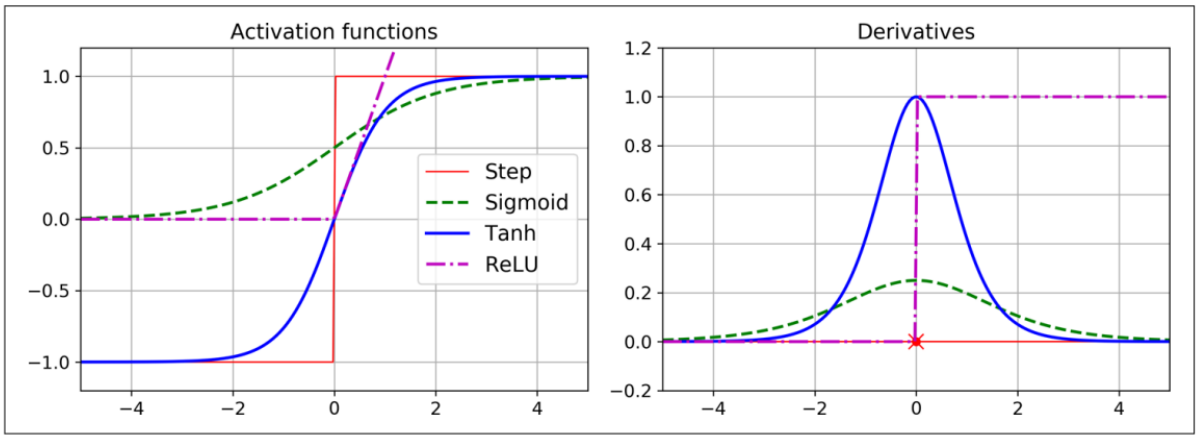


Figura 3.20: Funciones de activación y sus derivadas.

3.7.4. Funciones de Pérdida

Las funciones de pérdida son métricas matemáticas que evalúan qué tan bien está aprendiendo un modelo de redes neuronales al comparar sus predicciones con los valores reales. Su objetivo es proporcionar una medida del error para que el algoritmo pueda ajustar sus parámetros a través de la optimización.

Entropía Cruzada

La entropía cruzada es una función de pérdida ampliamente utilizada en problemas de clasificación, ya que mide qué tan bien las probabilidades estimadas por el modelo se alinean con las clases objetivo. Su minimización penaliza al modelo cuando asigna bajas probabilidades a la clase correcta, promoviendo predicciones más precisas [73].

Dada una clasificación con K clases y un conjunto de m muestras, la función de pérdida de entropía cruzada se define como:

$$J(\Theta) = -\frac{1}{m} \sum_{i=1}^{m} \sum_{k=1}^{K} y_i^k \log p_i^k \tag{3.10}$$

donde:

- y_i^k es la probabilidad objetivo de que la i-ésima muestra pertenezca a la clase k (en one-hot encoding, es 1 si la muestra pertenece a la clase k, y 0 en caso contrario).
- p_i^k es la probabilidad estimada por el modelo de que la i-ésima muestra pertenezca a la clase k.
- m es el número total de muestras en el conjunto de entrenamiento.

Para minimizar la función de entropía cruzada mediante gradiente descendente, la derivada con respecto a los parámetros θ_k es:

$$\nabla_{\theta_k} J(\Theta) = \frac{1}{m} \sum_{i=1}^{m} (p_i^k - y_i^k) x_i \tag{3.11}$$

Este gradiente se utiliza en algoritmos de optimización como descenso de gradiente para actualizar los parámetros Θ y mejorar las predicciones del modelo.

Dice Loss

La función de pérdida Dice Loss se basa en el Coeficiente de Dice, una métrica ampliamente utilizada para evaluar la superposición entre una segmentación generada por el modelo y el estándar de referencia (ground truth). Fue propuesta como función de pérdida por Milletari et al. [86] y es particularmente efectiva en tareas de segmentación con desequilibrio de clases.

Dado un conjunto de datos de segmentación, consideremos:

- R como la segmentación de referencia (ground truth), con valores de vóxeles r_n.
- P como el mapa de probabilidades predicho para la clase de interés (foreground), con valores p_n para cada vóxel.
- N como el número total de elementos en la imagen.

La variante de Dice Loss para dos clases se expresa como:

$$DL_2 = 1 - \frac{\sum_{n=1}^{N} p_n r_n + \epsilon}{\sum_{n=1}^{N} p_n + r_n + \epsilon} - \frac{\sum_{n=1}^{N} (1 - p_n)(1 - r_n) + \epsilon}{\sum_{n=1}^{N} 2 - p_n - r_n + \epsilon} \tag{3.12}$$

donde:

- ϵ es un pequeño valor positivo agregado para evitar problemas de división por cero en caso de que R y P sean conjuntos vacíos.

La Dice Loss se basa en el coeficiente de Dice, que mide la similitud entre dos conjuntos. Su función es maximizar la superposición entre la segmentación predicha y la de referencia. Esto se traduce en minimizar la pérdida DL_2, lo que impulsa la red neuronal a mejorar la segmentación generada.

3.7.5. Hiperparámetros

Los hiperparámetros son configuraciones predefinidas que controlan el proceso de entrenamiento de un modelo de aprendizaje profundo. A diferencia de los parámetros aprendidos por la red neuronal, como los pesos y sesgos, los hiperparámetros deben ajustarse manualmente o mediante técnicas de optimización [73].

Optimizador

El entrenamiento de una red neuronal se basa en minimizar una función de pérdida mediante un algoritmo de optimización, el cual ajusta los pesos de la red iterativamente. Un optimizador es responsable de calcular los gradientes y actualizar los parámetros del modelo de manera eficiente [73].

Momentum

El algoritmo de optimización por Momentum fue propuesto por Boris Polyak [87] y se inspira en la física del movimiento de un objeto con inercia. La idea es similar a la de una bola de boliche rodando por una pendiente: al principio se mueve lentamente, pero con el tiempo gana velocidad hasta alcanzar una velocidad terminal. Este concepto se aplica a la optimización en redes neuronales para acelerar la convergencia y evitar que el modelo quede atrapado en mínimos locales [73].

A diferencia del Descenso del Gradiente estándar, que ajusta los pesos θ restando el gradiente de la función de costo $J(\theta)$ multiplicado por la tasa de aprendizaje η:

$$\theta \leftarrow \theta - \eta \nabla_\theta J(\theta), \tag{3.13}$$

la optimización por momentum introduce un vector de momento m, el cual acumula una fracción del gradiente de las iteraciones anteriores. Su actualización se define de la siguiente manera:

$$m_t = \beta m_{t-1} - \eta \nabla_\theta J(\theta), \tag{3.14}$$

$$\theta_t = \theta_{t-1} + m_t. \tag{3.15}$$

Donde:

- m_t es el vector de momento en la iteración t.
- β es el coeficiente de momentum, generalmente fijado en $0{,}9$.
- η es la tasa de aprendizaje.
- $\nabla_\theta J(\theta)$ es el gradiente de la función de pérdida.
- θ_t representa los parámetros del modelo en la iteración t.

El parámetro β actúa como una forma de fricción que controla la velocidad de convergencia. Si $\beta = 0$, el algoritmo se reduce al Descenso del Gradiente estándar. En cambio, si β es cercano

a 1, el algoritmo acumula más información de gradientes previos, lo que permite superar valles en la función de costo más rápidamente y evitar oscilaciones.

RMSProp

El algoritmo RMSProp (de sus siglas en inglés, Root Mean Square Propagation) [9] soluciona su principal problema: la rápida disminución de la tasa de aprendizaje, lo que puede evitar que el modelo converja al óptimo global. Para evitar esto, RMSProp acumula solo los gradientes de las iteraciones más recientes en lugar de considerar todos los gradientes desde el inicio del entrenamiento. Esto se logra mediante el uso de un promedio móvil exponencialmente decaído, evitando que la magnitud de los gradientes se reduzca demasiado rápido.

La actualización de los parámetros en RMSProp se realiza con las siguientes ecuaciones:

$$s_t = \beta s_{t-1} + (1-\beta)\nabla_\theta J(\theta) \odot \nabla_\theta J(\theta), \tag{3.16}$$

$$\theta_t = \theta_{t-1} - \frac{\eta}{\sqrt{s_t} + \epsilon}\nabla_\theta J(\theta). \tag{3.17}$$

Donde:

- s_t es el promedio móvil de los cuadrados de los gradientes.
- β es el factor de decaimiento exponencial, generalmente fijado en 0.9.
- $\nabla_\theta J(\theta)$ representa el gradiente de la función de pérdida con respecto a los parámetros.
- η es la tasa de aprendizaje.
- ϵ es un pequeño valor de estabilidad para evitar divisiones por cero.

El uso de β permite que RMSProp siga ajustando los pesos de manera eficiente a lo largo del entrenamiento, sin que la tasa de aprendizaje se reduzca drásticamente como en AdaGrad. Esta técnica es particularmente efectiva en problemas con datos no estacionarios y en redes profundas donde los gradientes pueden tener escalas muy diferentes en distintas capas.

Adam

El algoritmo Adam (de sus siglas en inglés, Adaptive Moment Estimation) [88] combina las ventajas de momentum optimization y RMSProp. Al igual que la optimización por momentum, mantiene un promedio exponencialmente decreciente de los gradientes pasados, y al igual que RMSProp, conserva un promedio exponencialmente decreciente de los gradientes al cuadrado. Esto permite una actualización más eficiente de los pesos al adaptarse dinámicamente a los gradientes y sus magnitudes [73].

La actualización de parámetros en Adam sigue la siguiente formulación matemática:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\nabla_\theta J(\theta) \tag{3.18}$$

[9] RMSProp. https://paperswithcode.com/method/rmsprop

$$s_t = \beta_2 s_{t-1} + (1 - \beta_2)\nabla_\theta J(\theta) \odot \nabla_\theta J(\theta) \tag{3.19}$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \tag{3.20}$$

$$\hat{s}_t = \frac{s_t}{1 - \beta_2^t} \tag{3.21}$$

$$\theta_t = \theta_{t-1} - \eta \frac{\hat{m}_t}{\sqrt{\hat{s}_t} + \epsilon} \tag{3.22}$$

Donde:

- t es el número de iteración.
- m_t y s_t representan los promedios exponenciales de los gradientes y los gradientes al cuadrado, respectivamente.
- β_1 y β_2 son hiperparámetros de decaimiento, típicamente $\beta_1 = 0{,}9$ y $\beta_2 = 0{,}999$.
- $\hat{m}_t$ y $\hat{s}_t$ son las versiones corregidas de m_t y s_t para contrarrestar el sesgo inicial.
- η es la tasa de aprendizaje, normalmente $\eta = 0{,}001$.
- ϵ es un término de suavizado para evitar divisiones por cero, generalmente $\epsilon = 10^{-7}$.

Adam es un optimizador adaptativo que requiere menos ajuste manual de la tasa de aprendizaje. Su combinación de técnicas lo hace robusto para diferentes tipos de problemas y facilita la convergencia estable durante el entrenamiento.

Tasa de Aprendizaje

La tasa de aprendizaje (η) es uno de los hiperparámetros más importantes en el entrenamiento de redes neuronales, ya que controla el tamaño de los pasos que da el algoritmo de optimización en la actualización de los pesos. Si se establece un valor demasiado alto, el entrenamiento puede divergir, mientras que un valor demasiado bajo puede hacer que el modelo tarde mucho en converger al óptimo global. En algunos casos, si la tasa de aprendizaje es ligeramente alta, el modelo puede avanzar rápido al inicio, pero terminar oscilando alrededor del óptimo sin estabilizarse [73].

Batch Size

El tamaño del lote o batch size es un hiperparámetro fundamental en el entrenamiento de redes neuronales que afecta tanto el rendimiento del modelo como el tiempo de entrenamiento. Se refiere al número de muestras procesadas antes de actualizar los pesos del modelo durante el entrenamiento. En la práctica, una estrategia recomendada es intentar un tamaño de lote grande para aprovechar la eficiencia computacional, y si el entrenamiento es inestable o el rendimiento del modelo no es el esperado, reducir el tamaño del lote [73].

Épocas

Una época es un ciclo completo en el que el modelo procesa todo el conjunto de datos de entrenamiento una vez. Durante cada época, los pesos de la red neuronal se actualizan iterativamente mediante el algoritmo de optimización elegido, con el objetivo de minimizar la función de pérdida [73].

3.7.6. Transfer Learning

Transfer Learning es una técnica en la que un modelo preentrenado en un gran conjunto de datos se reutiliza como base para una nueva tarea. Esta estrategia es especialmente útil cuando el conjunto de datos disponible para la nueva tarea es pequeño o cuando se desea acelerar el proceso de entrenamiento aprovechando características previamente aprendidas por redes profundas. El aprendizaje por transferencia puede implementarse de diferentes formas, siendo la congelación de capas una de las más comunes [73] [89].

Congelación de Capas

La congelación de capas se refiere a la técnica de fijar los pesos de ciertas capas de una red neuronal para evitar su actualización durante el entrenamiento. Esto permite que el modelo aproveche representaciones de alto nivel aprendidas previamente en un conjunto de datos grande sin necesidad de reajustarlas [73] [89].

Fine-Tuning

Después de entrenar el modelo con capas congeladas, es posible mejorar su desempeño descongelando algunas capas superiores y ajustándolas con una tasa de aprendizaje baja. Esto permite que el modelo adapte mejor sus representaciones a la nueva tarea. El ajuste fino permite mejorar el rendimiento del modelo sin alterar en exceso las características aprendidas en las capas inferiores [73] [89].

3.7.7. Arquitecturas para Clasificación

Inception

El módulo Inception es una arquitectura avanzada dentro de las CNN, diseñada para capturar patrones en diferentes escalas de manera eficiente. Su estructura se basa en aplicar múltiples convoluciones en paralelo con distintos tamaños de kernel, lo que permite extraer características de niveles variados dentro de la imagen [73] [90].

En la Figura 3.21 se ilustra como cada módulo Inception comienza con una copia de la entrada, que es procesada simultáneamente por cuatro rutas diferentes:

- **Convolución 1 × 1:** Reduce la dimensionalidad de los canales antes de aplicar convoluciones más grandes, disminuyendo la carga computacional.
- **Convolución 3 × 3:** Captura patrones de tamaño intermedio, extrayendo detalles más finos.

- **Convolución 5 × 5:** Detecta características de mayor escala en la imagen.
- **Max Pooling 3 × 3:** Resalta las características más relevantes en una región, contribuyendo a la invariancia a pequeñas transformaciones.

Todas las capas convolucionales utilizan la función de activación ReLU y emplean same-padding, lo que garantiza que la salida tenga la misma altura y ancho que la entrada. Posteriormente, los mapas de características generados en cada una de estas rutas se concatenan en profundidad, formando una única representación enriquecida de la imagen.

Aunque una convolución 1 × 1 opera sobre un solo píxel a la vez y no captura relaciones espaciales, permite modelar interacciones entre distintos canales. Además, disminuye el número de mapas de características antes de convoluciones más costosas, optimizando el procesamiento. Por último, al combinarla con convoluciones más grandes, simula una red neuronal de dos capas deslizándose sobre la imagen, mejorando la capacidad de representación.

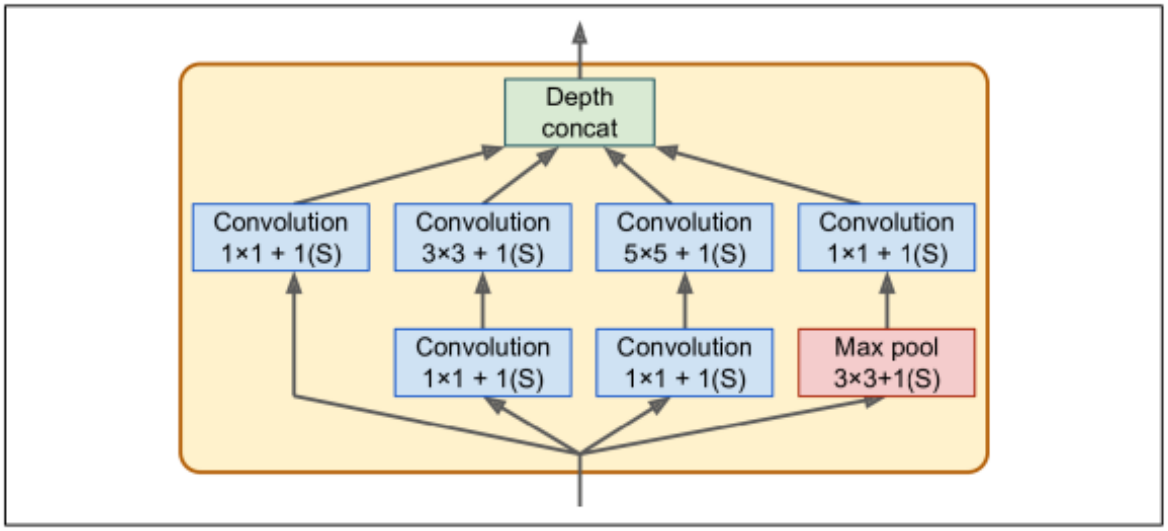


Figura 3.21: Arquitectura de Inception.

ResNet

Las ResNet representan una de las arquitecturas más influyentes en el aprendizaje profundo, introducidas por Kaiming He et al [91]. Su principal innovación radica en el uso de conexiones de acceso directo o skip connections, permitiendo la construcción de redes extremadamente profundas sin los problemas de degradación del gradiente [73]. El objetivo de entrenar una red neuronal es modelar una función objetivo $h(x)$. Sin embargo, en una ResNet, en lugar de aprender directamente $h(x)$, la red aprende una función residual $f(x)$, tal que:

$$f(x) = h(x) - x \tag{3.23}$$

Esta estrategia se denomina aprendizaje residual, donde la entrada x se suma directamente a la salida de ciertas capas, como se ilustra en la Figura 3.22. Gracias a esta conexión, la red inicializa modelando la función identidad, lo que acelera el entrenamiento y evita la desaparición del gradiente en redes profundas.

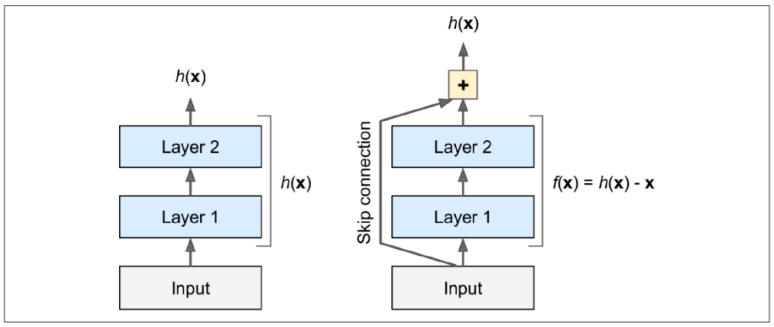


Figura 3.22: Esquema del aprendizaje residual en ResNet.

A diferencia de arquitecturas previas como VGG, que apilan capas convolucionales de manera secuencial, ResNet introduce unidades residuales, en las cuales la señal de entrada se suma a la salida de un bloque convolucional, como se muestra en la Figura 3.23. Cada unidad residual está compuesta por:

- Dos capas convolucionales con kernels de 3×3, activación ReLU y normalización por lotes (*Batch Normalization*).
- Una conexión de acceso directo que permite la propagación eficiente del gradiente a través de la red.
- En ciertas capas, una convolución 1×1 para adaptar la dimensión de los mapas de características antes de realizar la suma.

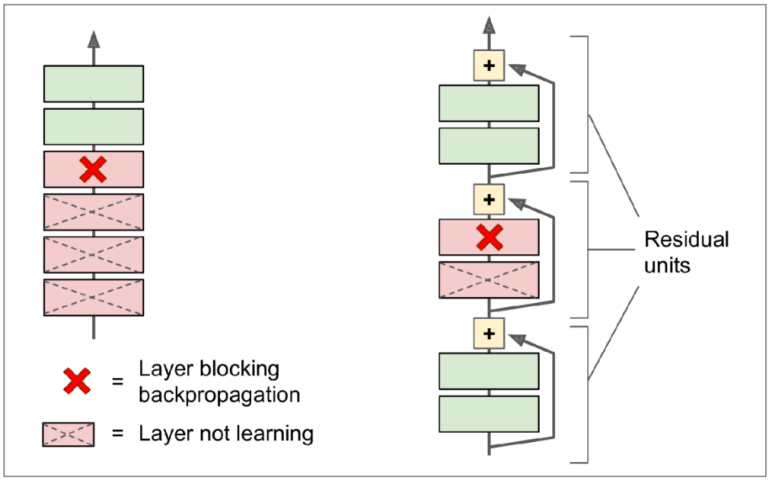


Figura 3.23: Unidad residual en ResNet.

El uso de conexiones residuales evita el problema de degradación del gradiente, al permitir el flujo directo de la señal, las capas más profundas pueden seguir aprendiendo sin que el gradiente se atenúe significativamente. Además, facilita el entrenamiento de redes más profundas gracias a las conexiones de acceso directo, ResNet permite el entrenamiento de modelos con más de 100 capas sin pérdida de precisión. Finalmente, reduce la complejidad computacional.

La inclusión de convoluciones 1×1 en ciertas unidades residuales actúa como un cuello de botella, disminuyendo el número de parámetros y mejorando la eficiencia del modelo.

ResNet está formada por múltiples unidades residuales apiladas, donde el número de mapas de características se duplica progresivamente mientras que la resolución espacial se reduce a la mitad. La arquitectura sigue un patrón general en el cual:

- Se inicia con una convolución 7×7 seguida de una capa de max pooling.
- Se incluyen múltiples unidades residuales organizadas en bloques.
- Se finaliza con una capa densa conectada a la salida del modelo.

En la Figura 3.24 se ilustra la estructura general de una red residual.

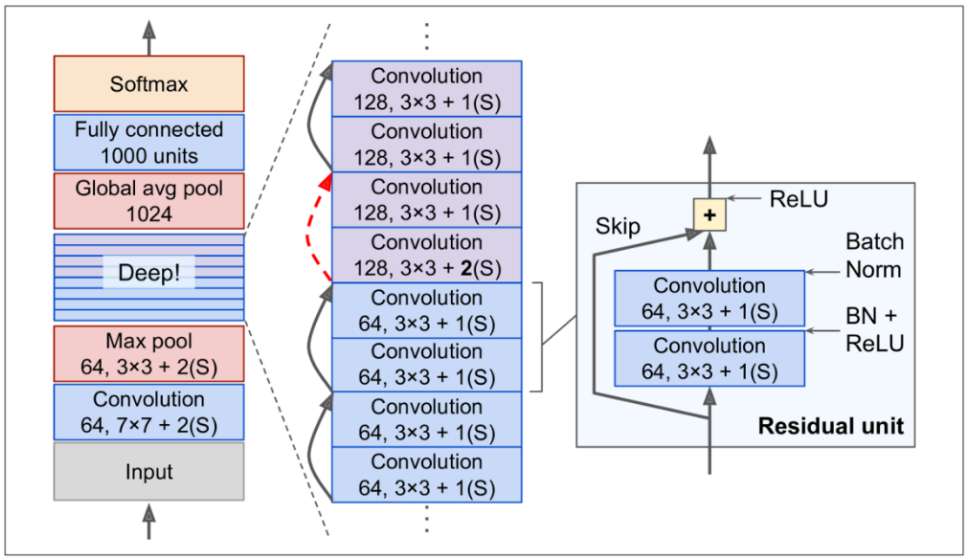


Figura 3.24: Arquitectura general de ResNet.

DenseNet

Las DenseNet representan una evolución en la arquitectura de redes profundas, introduciendo una conectividad más eficiente entre capas. En lugar de las conexiones secuenciales tradicionales en redes convolucionales o las conexiones de acceso directo en ResNet, DenseNet conecta cada capa con todas las capas anteriores, facilitando el flujo de información y reduciendo la redundancia en los parámetros [92].

A diferencia de las arquitecturas tradicionales en las que cada capa recibe únicamente la salida de la capa anterior, en DenseNet la entrada de cada capa se define como:

$$x_\ell = H_\ell([x_0, x_1, ..., x_{\ell-1}]) \tag{3.24}$$

donde $H_\ell(\cdot)$ representa una transformación no lineal compuesta por normalización por lotes (*Batch Normalization*), una activación ReLU y una convolución 3×3. El término $[x_0, x_1, ..., x_{\ell-1}]$ denota la concatenación de las salidas de todas las capas anteriores, como se muestra en la Figura 3.25.

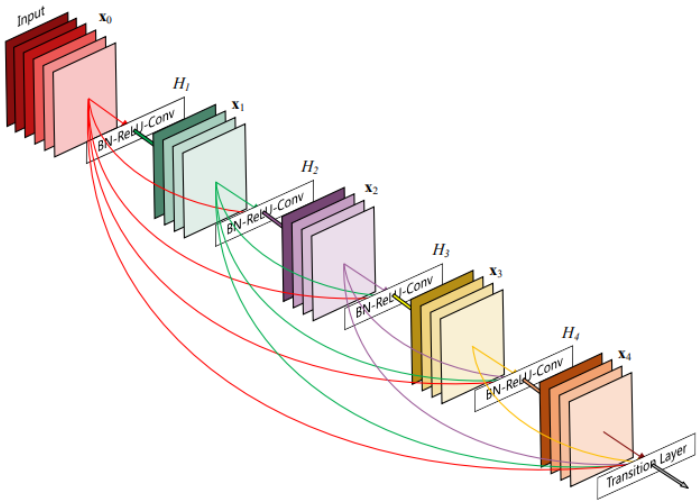


Figura 3.25: Esquema de la conectividad densa en DenseNet.

Esta estrategia permite un mejor flujo del gradiente en la red, mitigando problemas de desaparición del gradiente y promoviendo un mejor aprendizaje de características a diferentes niveles de abstracción.

Para manejar eficientemente la concatenación progresiva de características, DenseNet se organiza en bloques densos, separados por capas de transición. Estas capas de transición incluyen:

- Una convolución 1×1 para reducir la cantidad de mapas de características.
- Una capa de *pooling promedio* (*Average Pooling*) 2×2 para disminuir la resolución espacial.

Cada bloque denso está compuesto por múltiples capas convolucionales interconectadas, como se ilustra en la Figura 3.26.

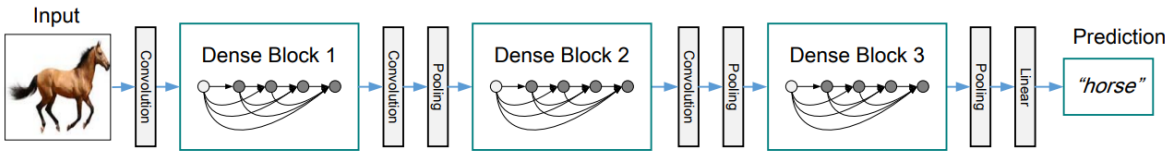


Figura 3.26: Estructura de los bloques densos en DenseNet.

Una característica clave de DenseNet es el parámetro de crecimiento k, que define cuántos mapas de características adicionales produce cada capa. La cantidad de entradas en la capa ℓ está dada por:

$$k_0 + k \times (\ell - 1) \tag{3.25}$$

donde k_0 es el número inicial de canales. Esto implica que la red no solo reutiliza información de capas previas, sino que también evita el crecimiento excesivo del número de parámetros.

DenseNet introduce varias mejoras con respecto a arquitecturas anteriores, una de ellas es que mejora la propagación del gradiente, gracias a la conectividad densa, el gradiente puede fluir más fácilmente a través de la red. En lugar de recalcular características en cada capa, DenseNet reutiliza activaciones previas, promoviendo representaciones más eficientes.

DenseNet sigue una arquitectura estructurada en bloques, donde cada uno tiene una serie de capas convolucionales densamente conectadas. Entre bloques, las capas de transición reducen la dimensionalidad para mejorar la eficiencia computacional. La Figura 3.27 ilustra la estructura de una DenseNet.

Layers	Output Size	DenseNet-121	DenseNet-169	DenseNet-201	DenseNet-264
Convolution	112 × 112	7 × 7 conv, stride 2			
Pooling	56 × 56	3 × 3 max pool, stride 2			
Dense Block (1)	56 × 56	[1 × 1 conv; 3 × 3 conv] × 6	[1 × 1 conv; 3 × 3 conv] × 6	[1 × 1 conv; 3 × 3 conv] × 6	[1 × 1 conv; 3 × 3 conv] × 6
Transition Layer (1)	56 × 56	1 × 1 conv			
	28 × 28	2 × 2 average pool, stride 2			
Dense Block (2)	28 × 28	[1 × 1 conv; 3 × 3 conv] × 12	[1 × 1 conv; 3 × 3 conv] × 12	[1 × 1 conv; 3 × 3 conv] × 12	[1 × 1 conv; 3 × 3 conv] × 12
Transition Layer (2)	28 × 28	1 × 1 conv			
	14 × 14	2 × 2 average pool, stride 2			
Dense Block (3)	14 × 14	[1 × 1 conv; 3 × 3 conv] × 24	[1 × 1 conv; 3 × 3 conv] × 32	[1 × 1 conv; 3 × 3 conv] × 48	[1 × 1 conv; 3 × 3 conv] × 64
Transition Layer (3)	14 × 14	1 × 1 conv			
	7 × 7	2 × 2 average pool, stride 2			
Dense Block (4)	7 × 7	[1 × 1 conv; 3 × 3 conv] × 16	[1 × 1 conv; 3 × 3 conv] × 32	[1 × 1 conv; 3 × 3 conv] × 32	[1 × 1 conv; 3 × 3 conv] × 48
Classification Layer	1 × 1	7 × 7 global average pool			
		1000D fully-connected, softmax			

Figura 3.27: Arquitectura general de DenseNet.

3.7.8. Arquitecturas para Detección de Objetos

YOLO

El modelo YOLO (de sus siglas en ingles, You Only Look Once) [93] revolucionó la detección de objetos al formularla como un problema de regresión unificado en lugar de utilizar métodos tradicionales basados en clasificadores o propuestas de regiones. YOLO predice todas las cajas delimitadoras y las clases asociadas en una única evaluación de red.

YOLO divide la imagen en una rejilla de $S \times S$ celdas. Cada celda es responsable de predecir B cajas delimitadoras y sus puntuaciones de confianza, además de C probabilidades de clase condicionadas a la existencia de un objeto en la celda. La arquitectura del modelo se representa en la Figura 3.28.

La arquitectura completa de YOLO, ilustrada en la Figura 3.29, está basada en una red neuronal convolucional profunda con la siguiente estructura:

- **Extracción de características:** 24 capas convolucionales para aprender representaciones visuales de alto nivel.

- **Reducción dimensional:** Capas 1×1 para reducir la dimensionalidad antes de convoluciones 3×3.
- **Capas completamente conectadas:** Dos capas densas finales para predecir las cajas delimitadoras y las clases.

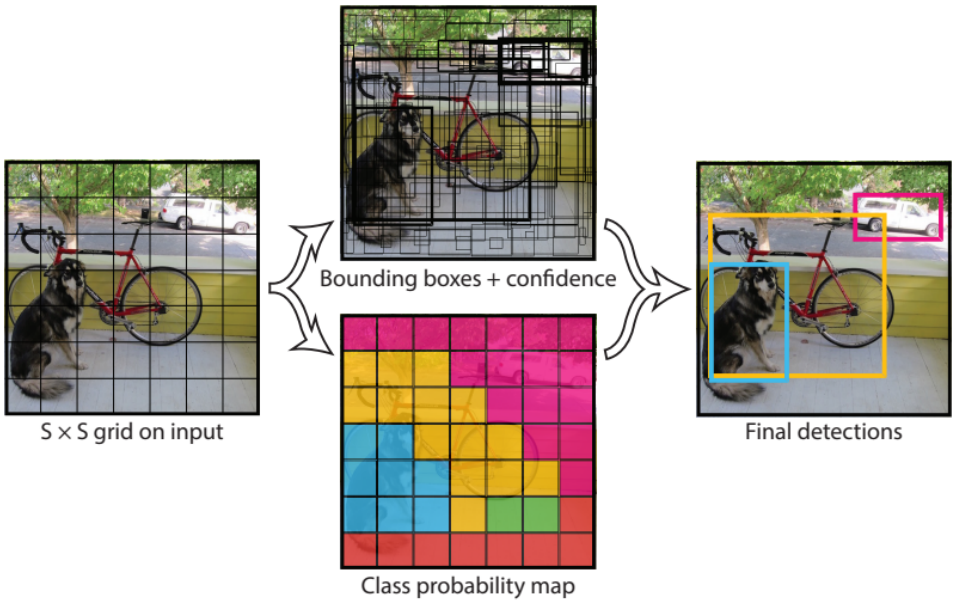


Figura 3.28: Esquema del modelo YOLO para detección de objetos. La imagen se divide en una rejilla de $S \times S$ celdas, donde cada celda predice B cajas delimitadoras, sus puntajes de confianza y C probabilidades de clase. Las predicciones se codifican en un tensor de tamaño $S \times S \times (B \times 5 + C)$, permitiendo una detección rápida y eficiente en una sola evaluación de la red.

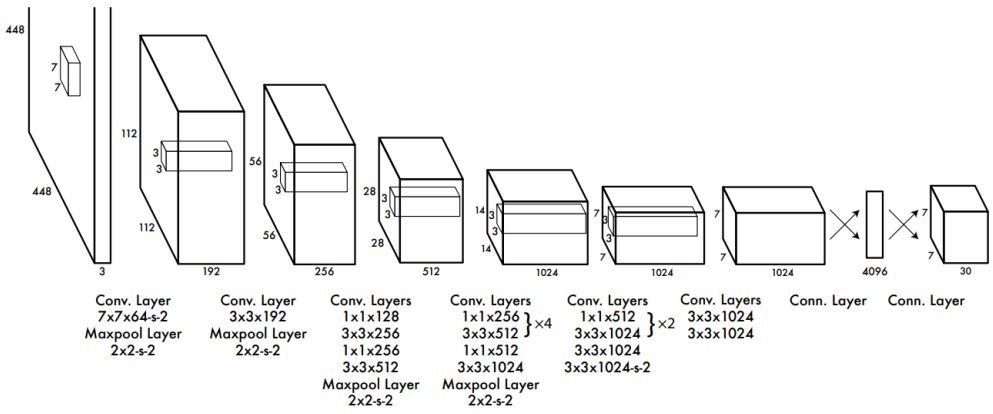


Figura 3.29: Estructura de la red YOLO. Alterna convoluciones 1×1 y 3×3 para extracción de características antes de la detección final.

3.7.9. Arquitecturas para Segmentacion

SAM

El modelo SAM [11] representa un enfoque innovador para la segmentación de imágenes, ofreciendo una arquitectura flexible y adaptable a múltiples aplicaciones mediante el uso de prompts. SAM consta de tres componentes principales, ilustrado en la Figura 3.30: un codificador de imágenes, un codificador de indicaciones y un decodificador de máscaras. El codificador de imágenes, basado en un Vision Transformer ViT [8] entrenado previamente, procesa la imagen de entrada antes de aplicar ninguna indicación. El codificador de indicaciones admite indicaciones dispersas (como puntos, cuadros y texto) e indicaciones densas (como máscaras), utilizando codificaciones posicionales para indicaciones dispersas y convoluciones para las densas. El decodificador de máscaras combina la imagen y las incrustaciones de indicaciones para generar una máscara.

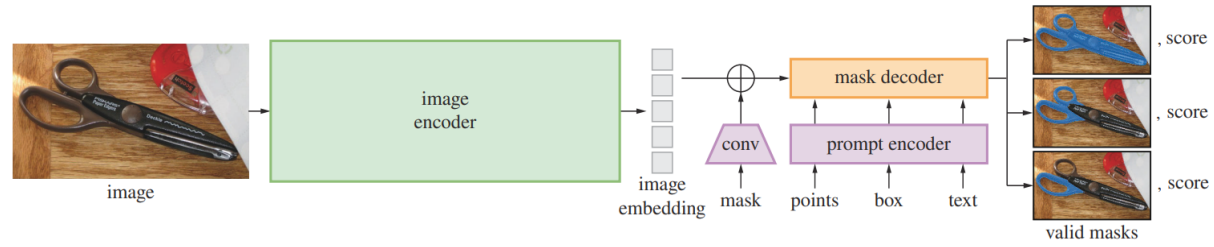


Figura 3.30: Descripción general del modelo Segment Anything (SAM). Un codificador de imágenes de alto rendimiento genera una incrustación de imágenes que luego se puede consultar de manera eficiente mediante una variedad de indicaciones de entrada para producir máscaras de objetos a una velocidad en tiempo real amortizada. Para indicaciones ambiguas que corresponden a más de un objeto, SAM puede generar múltiples máscaras válidas y puntajes de confianza asociados.

3.7.10. Arquitecturas para Generación de Imágenes

Neural Style Transfer

NST es una técnica que permite generar una imagen I_G combinando la estructura de una imagen de contenido I_C y el estilo de una imagen I_S. Fue introducida por Gatys et al. [13] y se basa en el uso de redes neuronales convolucionales preentrenadas (típicamente VGG-19) para extraer representaciones de contenido y estilo.

El principio subyacente es que las capas profundas de una CNN capturan la información estructural de una imagen (contenido), mientras que las correlaciones entre las respuestas de los filtros de diferentes capas encapsulan la información de textura y color (estilo). La generación de I_G se formula como un problema de optimización de la siguiente manera:

$$I_G \leftarrow I_G - \eta \cdot \alpha \frac{\partial L_{\text{content}}}{\partial I_G} + \eta \cdot \beta \frac{\partial L_{\text{style}}}{\partial I_G} \tag{3.26}$$

donde η es la tasa de aprendizaje, y α y β son factores de ponderación que regulan la contribución del contenido y el estilo, respectivamente.

NST explota el poder de las CNNs para modelar de manera efectiva los patrones visuales de una imagen. Para lograrlo, se utilizan dos funciones de pérdida:

- **Pérdida de contenido** L_{content}: Mide la diferencia entre las representaciones de contenido de I_G y I_C en una capa profunda de la CNN.
- **Pérdida de estilo** L_{style}: Evalúa las diferencias entre las matrices de Gram de las activaciones de la red para I_G y I_S, en múltiples capas.

La matriz de Gram G^l en la capa l se define como:

$$G_{ij}^l = \sum_k F_{ik}^l F_{jk}^l \tag{3.27}$$

donde F^l representa las activaciones de la capa l para una imagen dada.
El proceso de transferencia de estilo se optimiza mediante descenso de gradiente sobre la imagen generada I_G, ajustándola iterativamente para minimizar L_{total}:

$$L_{\text{total}} = \alpha L_{\text{content}} + \beta L_{\text{style}} \tag{3.28}$$

NST se basa en arquitecturas profundas preentrenadas como VGG-19, que extraen representaciones jerárquicas de la imagen. La configuración típica incluye:

- Uso de capas profundas (ej. `conv4_2`) para la representación de contenido.
- Uso de múltiples capas (ej. `conv1_1`, `conv2_1`, `conv3_1`, `conv4_1`, `conv5_1`) para la representación de estilo.
- Inicialización de I_G con ruido aleatorio o con I_C.
- Uso de optimización basada en Adam o L-BFGS.

Redes Generativas Adversarias (GANs)

Las GANs fueron introducidas por Goodfellow et al. [94], y representan un avance clave en la generación de datos sintéticos. Estas redes consisten en un enfrentamiento entre dos modelos:

- Un **generador** $G(z)$ que aprende a generar datos sintéticos $x \sim p_g(x)$ a partir de una distribución latente $p_z(z)$.
- Un **discriminador** $D(x)$, encargado de distinguir entre datos reales $x \sim p_{\text{data}}(x)$ y los generados por G.

Ambos modelos se entrenan mediante un juego minimax, donde el generador intenta engañar al discriminador, mientras que este ´ltimo mejora su capacidad de distinguir datos reales de falsos. La función objetivo que define esta competencia se expresa como:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{\text{data}}}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))]. \tag{3.29}$$

El proceso de entrenamiento de las GANs sigue un esquema iterativo:

1. Se entrena $D(x)$ para maximizar la probabilidad de clasificar correctamente las muestras reales y generadas.

2. Se actualiza $G(z)$ para minimizar la capacidad de $D(x)$ de distinguir los datos sintéticos de los reales.

3. Se repiten los pasos anteriores hasta alcanzar un equilibrio donde $G(z)$ genera muestras indistinguibles de los datos reales.

CycleGAN

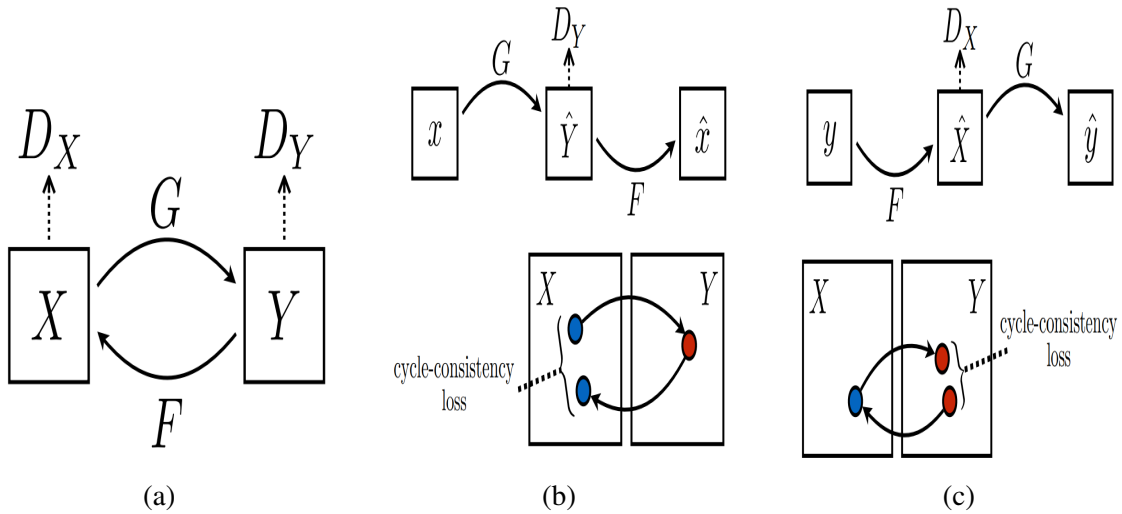
CycleGAN es un tipo de GAN diseñada para la traducción de imágenes entre dominios sin necesidad de datos de entrenamiento emparejados. En este estudio, se utiliza para traducir características entre las etapas de la retinopatía diabética y las imágenes de fondo de ojo normales. Esto permite la síntesis de representaciones realistas de la progresión de la enfermedad retinal al transferir características patológicas a imágenes reales de retina.

Como se muestra en la Figura 3.31, CycleGAN consta de dos generadores, $G : X \rightarrow Y$ y $F : Y \rightarrow X$, que aprenden las correspondencias entre el dominio de origen X (por ejemplo, imágenes de fondo de ojo normales) y el dominio objetivo Y (por ejemplo, imágenes de fondo de ojo con retinopatía diabética). Además, incluye dos discriminadores, D_X y D_Y, que intentan distinguir entre imágenes reales y generadas en cada dominio, mediante una función objetivo completa que combina la pérdida adversarial y la pérdida de consistencia cíclica (Ecuación 3.30).

En esta función, $\mathcal{L}_{\text{GAN}}(G, D_Y, X, Y)$ y $\mathcal{L}_{\text{GAN}}(F, D_X, Y, X)$ representan la pérdida adversarial, que incentiva a los generadores a producir imágenes indistinguibles de las imágenes reales en los dominios objetivo. Mientras que $\mathcal{L}_{\text{cyc}}(G, F)$ es la pérdida de consistencia cíclica, que garantiza que las transformaciones G y F sean inversas entre sí, preservando el contenido de las imágenes durante la traducción [95].

$$\begin{aligned}\mathcal{L}(G, F, D_X, D_Y) = \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y)+ \\ \mathcal{L}_{\text{GAN}}(F, D_X, Y, X)+ \\ \lambda\mathcal{L}_{\text{cyc}}(G, F)\end{aligned} \tag{3.30}$$

Aquí, λ es un factor de ponderación que equilibra la importancia de la pérdida de consistencia cíclica en relación con las pérdidas adversariales.



(a) (b) (c)

Figura 3.31: (a) Cyclegan contiene dos funciones de mapeo $G : X \to Y$ y $F : Y \to X$, junto con los discriminadores adversariales asociados D_Y y D_X. El discriminador D_Y incentiva a G a traducir X en salidas indistinguibles del dominio Y, y de manera análoga, D_X incentiva a F a traducir Y al dominio X. Para regularizar aún más los mapeos, introducimos dos pérdidas de consistencia cíclica que capturan la intuición de que si traducimos de un dominio a otro y luego regresamos al original, deberíamos obtener el mismo punto de partida: (b) pérdida de consistencia cíclica hacia adelante: $x \to G(x) \to F(G(x)) \approx x$, y (c) pérdida de consistencia cíclica hacia atrás: $y \to F(y) \to G(F(y)) \approx y$.

3.7.11. Callbacks

ReduceLROnPlateau

El `ReduceLROnPlateau` es un callback utilizado en el entrenamiento de redes neuronales que ajusta dinámicamente la tasa de aprendizaje cuando el modelo deja de mejorar. Su objetivo principal es evitar estancamientos en el proceso de optimización y mejorar la convergencia del modelo [73]. Durante el entrenamiento, se monitorea una métrica específica, generalmente la pérdida de validación o la precisión. Si esta métrica no mejora después de un número determinado de épocas (patience), el algoritmo reduce la tasa de aprendizaje multiplicándola por un factor predefinido. Esto permite que la red realice ajustes más finos en sus pesos, lo que puede mejorar su desempeño y evitar quedar atrapada en mínimos locales.

Early Stopping

Early Stopping es una técnica de regularización que detiene el entrenamiento de una red neuronal cuando el error en el conjunto de validación deja de mejorar. Esto evita que el modelo siga entrenando innecesariamente, reduciendo el riesgo de sobreajuste [73].

Durante el entrenamiento, se monitorea una métrica, generalmente la pérdida de validación. Si esta métrica deja de mejorar después de un número determinado de épocas, el entrenamiento se detiene y se carga el mejor modelo encontrado hasta ese punto. De esta manera, se evita

que el modelo siga aprendiendo patrones específicos del conjunto de entrenamiento en lugar de generalizar bien a datos no vistos.

3.7.12. Evaluación del Modelo

Matriz de Confusión

La matriz de confusión es una herramienta fundamental para evaluar el desempeño de un clasificador. Su propósito es comparar las predicciones del modelo con las etiquetas reales, proporcionando una visión detallada de los errores cometidos. Cada fila de la matriz representa las instancias de una clase real, mientras que cada columna representa las instancias de una clase predicha [73].

Para calcular la matriz de confusión, primero se obtienen las predicciones del modelo y luego se comparan con las etiquetas verdaderas:

$$\text{Matriz de Confusión} = \begin{bmatrix} TN & FP \\ FN & TP \end{bmatrix} \tag{3.31}$$

donde:

- TN (*True Negatives*): Casos correctamente clasificados como negativos.
- FP (*False Positives*): Casos incorrectamente clasificados como positivos.
- FN (*False Negatives*): Casos incorrectamente clasificados como negativos.
- TP (*True Positives*): Casos correctamente clasificados como positivos.

Un modelo perfecto tendría una matriz de confusión con valores diferentes de cero solo en la diagonal principal, indicando que todas las instancias han sido clasificadas correctamente.

Precisión, Recall y F1-Score

Para evaluar el desempeño de un clasificador, se utilizan diversas métricas que permiten medir su capacidad para realizar predicciones correctas. Entre las métricas más importantes se encuentran precisión, Recall y el F1 Score. Estas métricas proporcionan una evaluación integral del rendimiento de un modelo de clasificación, asegurando que tanto la cantidad de predicciones correctas como la cobertura de las instancias positivas sean consideradas [73].

- **Precisión (precisión, P)**: Mide la proporción de predicciones positivas que realmente son correctas. Se define como:
$$P = \frac{TP}{TP + FP} \tag{3.32}$$
donde TP representa los verdaderos positivos y FP los falsos positivos.

- **Recall o Tasa de Verdaderos Positivos (TPR)**: Mide la capacidad del modelo para detectar correctamente las instancias positivas dentro del conjunto total de instancias positivas reales:
$$TPR = R = \frac{TP}{TP + FN} \tag{3.33}$$
donde FN son los falsos negativos.

- **F1-Score**: Es la media armónica entre la precisión y el recall, proporcionando un balance entre ambas métricas:

$$F1 = 2 \cdot \frac{P \cdot R}{P + R} \tag{3.34}$$

El F1-score es útil cuando se requiere un equilibrio entre precisión y recall, especialmente en casos donde la distribución de clases está desbalanceada.

Dice Score

El Dice Score es una métrica ampliamente utilizada para evaluar el rendimiento en tareas de segmentación de imágenes. Mide el grado de superposición entre la segmentación predicha y la segmentación real (ground truth) [96]. Se define mediante la siguiente ecuación:

$$Dice = \frac{2 \cdot |A \cap B|}{|A| + |B|} \tag{3.35}$$

donde A representa la segmentación predicha y B la segmentación de referencia. La intersección $|A \cap B|$ indica los píxeles correctamente segmentados, mientras que $|A|$ y $|B|$ representan el número total de píxeles en las máscaras predicha y de referencia, respectivamente. Un valor de Dice Score de 1 indica una superposición perfecta, mientras que un valor de 0 indica que no hay coincidencia.

AUC-PR

El AUC-PR (de sus siglas en inglés, Area Under Curve - Precisión-Recall) evalúa el balance entre precisión y recall [73]. Se calcula sumando las AUC-PR construida a partir de múltiples puntos discretos:

$$AUC - PR = \sum_{i=1}^{n-1} \frac{P_{i+1} + P_i}{2} \cdot (R_{i+1} - R_i) \tag{3.36}$$

donde P_i y R_i representan los valores de precisión y recall en cada punto de la curva. La integral se aproxima utilizando la regla del trapecio. Un AUC-PR alto indica que el modelo tiene una buena capacidad de distinguir los verdaderos positivos minimizando los falsos positivos.

Curva ROC

La curva ROC (de sus siglas en inglés, Receiver Operating Characteristic) es una herramienta utilizada para evaluar el desempeño de clasificadores binarios [73]. Es similar a la curva de precisión/recall, pero en lugar de representar la precisión frente al recall, la curva ROC grafica la TPR (de sus siglas en inglés, True Positive Rate) contra la FPR (de sus siglas en inglés, False Positive Rate).

$$TPR = \frac{TP}{TP + FN} \tag{3.37}$$

$$FPR = \frac{FP}{FP + TN} \tag{3.38}$$

El FPR mide la proporción de instancias negativas que han sido incorrectamente clasificadas como positivas. Es complementario a la TNR (de sus siglas en inglés, True Negative Rate), también llamada especificidad, la cual se define como:

$$TNR = \frac{TN}{TN + FP} = 1 - FPR \tag{3.39}$$

Por lo tanto, la curva ROC representa recall frente a $1 -$ especificidad.

AUC-ROC

El AUC-ROC (de sus siglas en inglés, Area Under Curve - Receiver Operating Characteristic) mide la capacidad del modelo para distinguir entre clases al graficar la TPR contra la FPR:

$$AUC - ROC = \sum_{i=1}^{n-1} (FPR_{i+1} - FPR_i) \cdot \frac{TPR_{i+1} + TPR_i}{2} \tag{3.40}$$

donde TPR es la sensibilidad o recall, y FPR representa la tasa de falsos positivos. La curva ROC se construye evaluando diferentes umbrales de decisión, y su área bajo la curva proporciona una medida del desempeño global del clasificador [73]. Un AUC-ROC de 1 indica un modelo perfecto, mientras que un valor de 0.5 sugiere un desempeño aleatorio.

CAPÍTULO 4

METODOLOGÍA DE LA EXPERIMENTACIÓN

Esta sección describe los métodos y herramientas empleados para la clasificación, detección y segmentación de biomarcadores, así como la generación de imágenes sintéticas.

4.1. Datasets Utilizados

Se emplearon los siguientes conjuntos de datos para las diversas tareas de clasificación, generación y segmentación, como se describe en la Tabla 4.1, y se amplía su explicación en las siguientes subsecciones:

Tabla 4.1: Conjuntos de datos utilizados en las tareas de clasificación, segmentación y generación.

Modelo	Dataset	Cantidad de Imágenes Utilizadas
CycleGAN	APTOS	3031
CNN	APTOS	3031
CycleGAN	EyePACS	3500
CNN	E-Optha	47
Detección de Objetos	E-Optha	47
Segmentación	E-Optha	47
Segmentación	IDRID	82
Segmentación	Messidor-2	197
CNN	ODIR-5K	2600
CycleGAN	ODIR-5K	1500
NST	ODIR-5K	2
YOLOv8	ODIR-5K	200
Segmentación	REFUGE	400
Segmentación	RITE	80

4.1.1. APTOS

El conjunto de datos APTOS [97] incluye CFI categorizadas en cinco clases, que incluyen el fondo de ojo normal y cuatro etapas de retinopatía diabética. Para la experimentación, se utilizaron 3031 imágenes. Este conjunto de datos fue la fuente principal para los modelos CNN, CycleGAN y proporcionó todas las imágenes para el entrenamiento.

4.1.2. ODIR-5K

El conjunto de datos ODIR-5K [98] incluye 8000 imágenes de fondo de ojo que cubren una variedad de enfermedades de la retina. Para este estudio, se seleccionaron 1347 imágenes correspondientes a diferentes etapas de retinopatía diabética y 1253 imágenes con fondo de ojo normal. Este conjunto de datos se utilizó para la clasificación de múltiples etiquetas en una misma imagen, centrándose en la detección de biomarcadores. Se empleó una estrategia de vector one-hot, donde las etiquetas se crearon desde cero. Se agregaron columnas para biomarcadores específicos, incluidos microaneurismas, hemorragias, exudados duros y exudados suaves. Al vector one-hot de cada imagen se le asignó un "1" o "0" según la presencia o ausencia de estos biomarcadores. Para la detección de objetos, se seleccionaron 200 imágenes y los cuadros delimitadores para los biomarcadores de retinopatía diabética se anotaron manualmente utilizando la herramienta makesense.ai[10], con anotaciones guardadas en formato de texto. Finalmente, estas imágenes se utilizaron en los experimentos NST y CycleGAN.

4.1.3. EyePacs

El conjunto de datos EyePACS [99] incluye una colección de 30,262 CFI categorizadas en cinco clases: fondo de ojo normal y las cuatro etapas de la retinopatía diabética (leve, moderada, severa y proliferativa). Además, este conjunto de datos se utilizó para entrenar CycleGAN.

4.1.4. IDRID

El conjunto de datos IDRID [100] proporciona 81 imágenes detalladas del fondo de ojo que contienen retinopatía diabética, junto con anotaciones para localizar biomarcadores. Este conjunto de datos se utilizó específicamente para tareas de segmentación a través del modelo de SAM.

4.1.5. Messidor-2

El conjunto de datos Messidor-2 [101] contiene originalmente 1200 imágenes de fondo de ojo de retinopatía diabética. Sin embargo, para este experimento, solo se utilizaron 197 imágenes, ya que las máscaras solo estaban disponibles para estas imágenes. Estas máscaras, proporcionadas por Gabriel Lepetit-Aimon [102], contienen anotaciones detalladas de biomarcadores de retinopatía diabética. El conjunto de datos se utilizó específicamente para tareas de segmentación en este experimento a través de SAM.

[10]makesense.ai. https://www.makesense.ai/

4.2. Preprocesamiento de Imágenes

Antes de entrenar los modelos de clasificación (multiclase y multietiqueta), de generación de imágenes sintéticas, y de detección y segmentación de objetos, se realizaron los métodos de preprocesamiento (a, b,.., j) de la Tabla 4.2 como se describe a continuación para eliminar el ruido y mejorar la eficacia del entrenamiento posterior del modelo. Las imágenes preprocesadas se almacenaron en una carpeta individual para una mejor organización y eficiencia del flujo de trabajo.

a) Proceso de recorte de imagen para eliminar bordes negros irrelevantes y resaltar la región central de la imagen.

b) Conversión al espacio de color RGB para estandarizar el formato.

c) Redimensionamiento de todas las imágenes a un tamaño uniforme de 224×224 píxeles.

d) Redimensionamiento de todas las imágenes a un tamaño uniforme de 256×256 píxeles.

e) Redimensionamiento de todas las imágenes a un tamaño uniforme de 400×400 píxeles.

f) Redimensionamiento de todas las imágenes a un tamaño uniforme de 640×640 píxeles.

g) Normalización de los valores de los píxeles al rango [0, 1].

h) Normalización de los valores de los píxeles al rango [-1, 1].

i) Los valores de los píxeles se normalizaron utilizando una media de [0.485, 0.456, 0.406] y una desviación estándar de [0.229, 0.224, 0.225].

j) Aplicación de técnicas de aumento de datos como el volteo horizontal y la rotación de hasta 20 grados.

Cada uno de estos pasos se aplicó a las imágenes dependiendo del modelo y la tarea específica a realizar. A continuación se detalla cada uno de los preprocesamientos 4.2.

Tabla 4.2: Pasos de preprocesamiento aplicados a los diferentes modelos.

Modelo	a	b	c	d	e	f	g	h	i	j
Multi-clase CNN	●	●	●	-	-	-	●	-	-	●
Multi-etiqueta CNN	●	●	●	-	-	-	●	-	-	●
Multi-clase Vision Transformer	●	●	●	-	-	-	●	-	-	-
Segmentación	●	●	-	●	-	-	-	●	-	-
Detección de Objetos	●	●	-	-	-	●	-	-	-	-
Neural Style Transfer	●	●	-	-	●	-	-	-	●	-
CycleGAN	●	●	●	-	-	-	-	●	-	-

4.2.1. Recorte de Imágenes

El primer paso del preprocesamiento consiste en eliminar los bordes negros de las imágenes, asegurando que solo se mantengan las regiones relevantes. Para ello, se implementó la función crop_image_from_gray, desarrollada en Python y utilizando las librerías NumPy y OpenCV, que recorta las áreas sin información visual en función de un umbral, tal como se ilustra en la Listing 4.1.

```
def crop_image_from_gray(img, tol=7):
    if img.ndim == 2:
        mask = img > tol
        return img[np.ix_(mask.any(1), mask.any(0))]
    elif img.ndim == 3:
        gray_img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        mask = gray_img > tol
        check_shape = img[:, :, 0][np.ix_(mask.any(1), mask.any(0))].shape
    [0]

        if check_shape == 0:
            return img
        else:
            img1 = img[:, :, 0][np.ix_(mask.any(1), mask.any(0))]
            img2 = img[:, :, 1][np.ix_(mask.any(1), mask.any(0))]
            img3 = img[:, :, 2][np.ix_(mask.any(1), mask.any(0))]
            img = np.stack([img1, img2, img3], axis=-1)
    return img
```

Listing 4.1: Implementación de la función crop_image_from_gray para recortar bordes irrelevantes en imágenes.

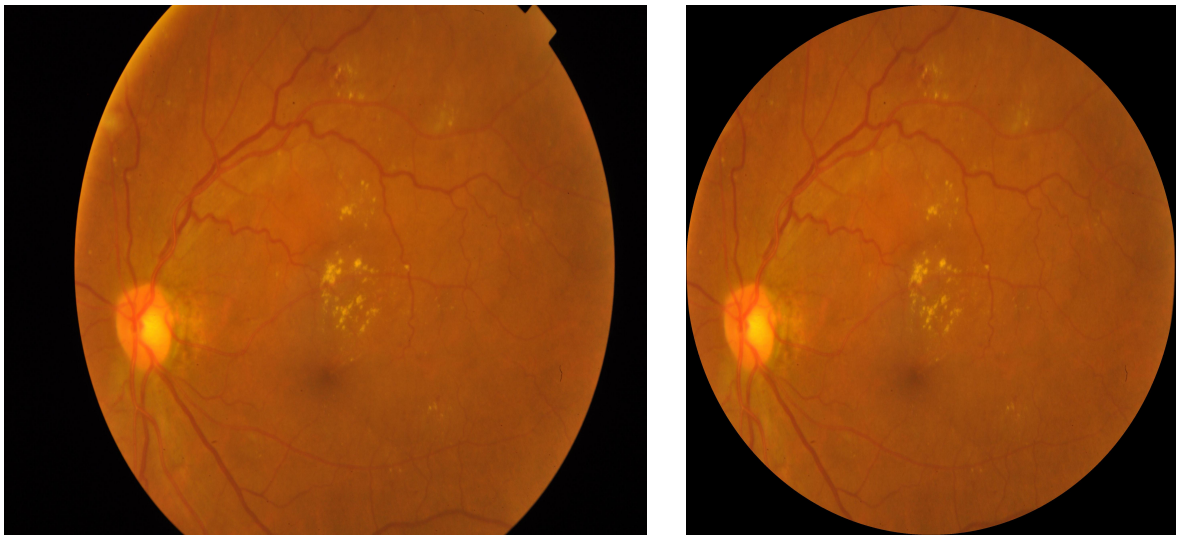
Después del recorte inicial, se aplica un recorte circular centrado en las imágenes para destacar únicamente la región de interés. Este proceso también asegura que las dimensiones de la imagen sean uniformes. La función circle_crop realiza este procedimiento al calcular el radio más pequeño desde el centro de la imagen y generar una máscara circular que elimina las áreas externas, como se ilustra en el Listing 4.2.

```
def circle_crop(img):
    img = cv2.imread(img)
    img = crop_image_from_gray(img)
    height, width, depth = img.shape
    largest_side = np.max((height, width))
    img = cv2.resize(img, (largest_side, largest_side))
    height, width, depth = img.shape

    x = int(width / 2)
    y = int(height / 2)
    r = np.amin((x, y))

    circle_img = np.zeros((height, width), np.uint8)
    cv2.circle(circle_img, (x, y), int(r), 1, thickness=-1)
    img = cv2.bitwise_and(img, img, mask=circle_img)
    img = crop_image_from_gray(img)

    return img
```

Listing 4.2: Implementación de la función circle_crop para recortar imágenes en una región

circular central.

El preprocesamiento descrito se aplicó iterativamente a todas las imágenes de un directorio, garantizando que cada archivo fuera transformado de manera uniforme. Como resultado, las imágenes procesadas presentan las áreas de interés recortadas, eliminando elementos irrelevantes. Este cambio se evidencia en la Figura 4.1, donde se muestra un ejemplo visual comparativo del estado original y el resultado final tras el preprocesamiento.



(a) (b)

Figura 4.1: Proceso de recorte circular: (a) Imagen original sin modificaciones, (b) Imagen resultante tras aplicar el recorte circular

4.2.2. Normalización

1. **Normalización al rango [0, 1]**
   Esta técnica divide los valores de los píxeles entre 255.0, convirtiendo cada valor a un rango entre 0 y 1 así como se muestra en el Listing 4.3.

```
from PIL import Image
import numpy as np

image_path = "imagen.jpg"
image = Image.open(image_path)
image = np.array(image, dtype=np.float32)
image = image / 255.0
print(image.shape)
```

Listing 4.3: Normalización de imágenes en NumPy dividiendo los valores de los píxeles entre 255.

2. **Normalización al rango [-1, 1]**
   En este método, los valores de los píxeles se ajustan al rango [-1, 1]. Esto se logra dividiendo entre 127.5 y restando 1, así como se ilustra en el Listing 4.4.

```python
from PIL import Image
import numpy as np
import tensorflow as tf

image_path = "imagen.jpg"
image = Image.open(image_path)
image = np.array(image, dtype=np.float32)
image = (tf.cast(image, tf.float32) / 127.5) - 1
print(image.shape)
```

Listing 4.4: Normalización de imágenes en TensorFlow al rango de valores entre -1 y 1.

3. **Normalización basada en la media y la desviación estándar**
   En esta etapa del preprocesamiento, los valores de los píxeles de la imagen se ajustan según una media [0.485, 0.456, 0.406] y una desviación estándar de [0.229, 0.224, 0.225]. Estas constantes corresponden a la estadística promedio de la base de datos ImageNet [103], sobre la cual fue entrenada la red neuronal VGG19. De esta forma, cada canal (RGB) se estandariza restando su media y dividiendo por su desviación típica así como se muestra en el Listing 4.5.

```python
from PIL import Image
import numpy as np
from torchvision import transforms

transform = transforms.Compose([
    transforms.Normalize((0.485, 0.456, 0.406), (0.229, 0.224, 0.225))
])

image_path = "imagen.jpg"
image = Image.open(image_path).convert("RGB")
image = transform(image)
print(image.shape)
```

Listing 4.5: Código en PyTorch para normalización utilizando la media y desviación estándar en los valores de los píxeles.

4.2.3. Aumento de Datos

Se aplicaron varias técnicas de aumentación de datos, como el volteo horizontal y rotaciones de hasta 20 grados como se muestra el resultado en la Figura 4.2 y Listing 4.6.

1. **Volteo Horizontal**

   El volteo horizontal invierte la imagen de izquierda a derecha, generando una nueva perspectiva sin alterar el contenido visual.

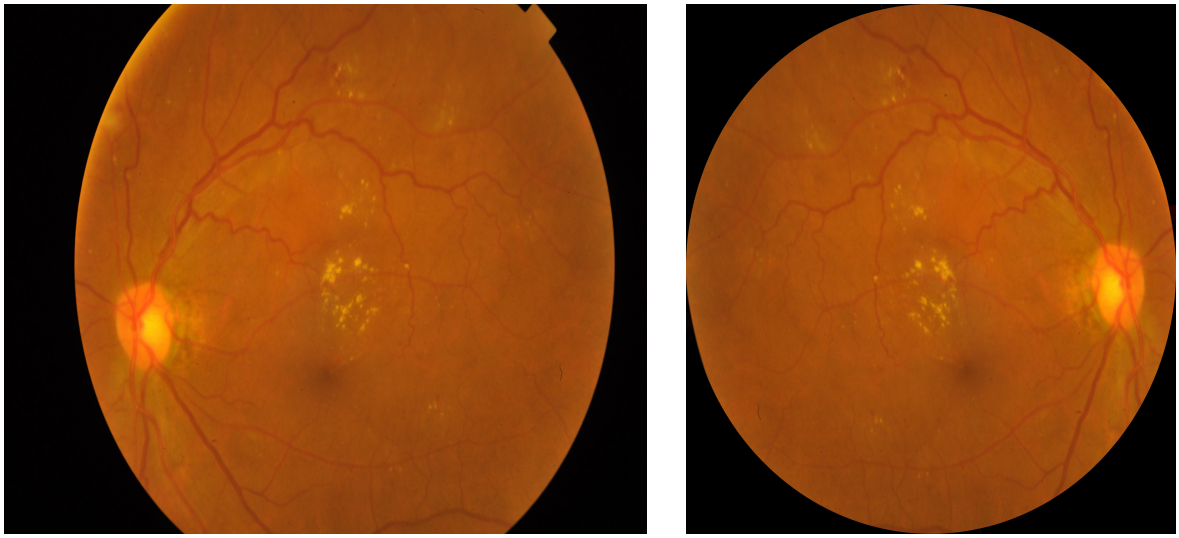
2. **Rotación de 20 Grados**

   Se aplica una rotación aleatoria en el rango de 20 grados.

```python
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import cv2

image_path = "imagen.jpg"
```

```
datagen = ImageDataGenerator(
    horizontal_flip=True,
    rotation_range=20,
)

image = cv2.imread(image_path)
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
image = image.astype('float32')
augmented_image = datagen.random_transform(image)
```

Listing 4.6: Ejemplo de código para aumento de datos utilizando ImageDataGenerator en TensorFlow. Incluye volteo horizontal y rotaciones de hasta 20 grados aplicados a una imagen de entrada.



(a) (b)

Figura 4.2: Proceso de aumento de datos: (a) Imagen original sin modificaciones, (b) Imagen resultante tras aplicar el preprocesamiento de recorte de la imagen, rotación de 20 grados y volteo horizontal.

4.3. Arquitectura de Modelo CNN

En este desarrollo se utilizó InceptionV3 [104] tanto para la clasificación multiclase y multietiqueta en imágenes de retina. La división de los datos en entrenamiento, validación y prueba se realizó mediante validación cruzada de 5 fold, asegurando una evaluación robusta de cada modelo.

Se adaptó para clasificar las etapas de la retinopatía diabética y detectar biomarcadores en imágenes de fondo de ojo. Las capas convolucionales del modelo InceptionV3 fueron inicializadas con pesos de ImageNet y se mantuvieron congeladas para retener la extracción de características, mientras que las últimas 32 capas se descongelaron para ajuste fino (Fine Tuning). Las capas de pooling global promedio y densas se modificaron para satisfacer el requisito

específico mediante normalización por lotes (BatchNormalization); para la arquitectura del modelo de aprendizaje por transferencia, mostrado en la Figura 4.3, considerando el cambio en las capas de salida según los enfoques. En el caso de la clasificación multiclase, la capa de salida utiliza una función softmax para predecir una única clase entre las posibles, mientras que, para la clasificación multietiqueta, la capa de salida emplea una función sigmoid que permite asignar múltiples etiquetas a cada imagen.

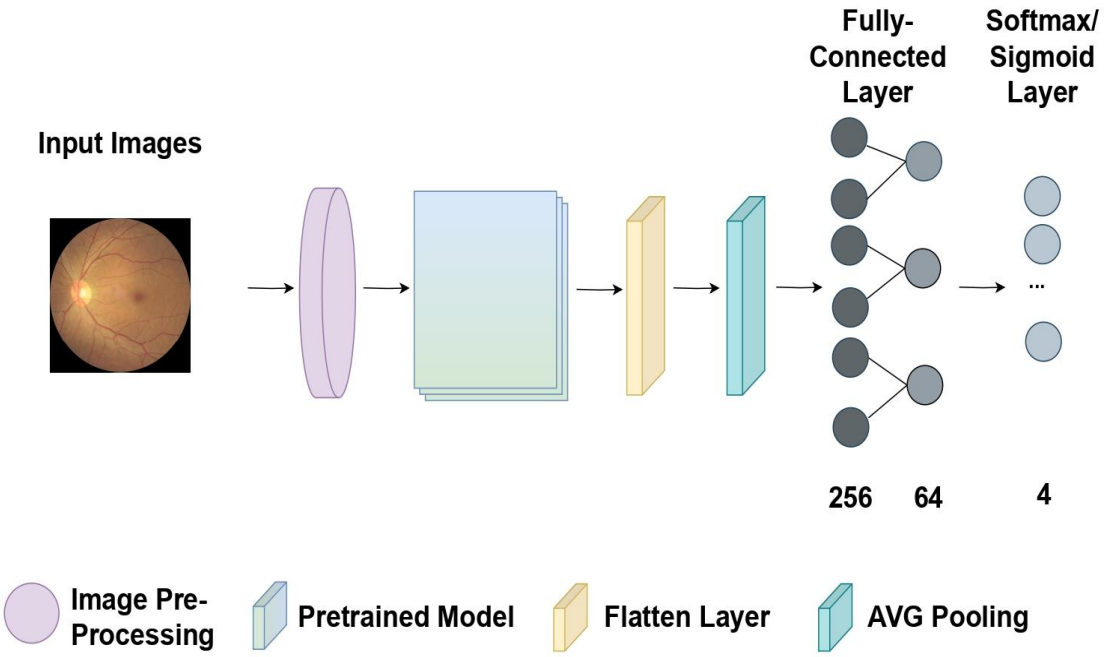


Figura 4.3: Arquitectura para la clasificación multiclase y multietiqueta basada en un modelo preentrenado

Se aplicó el optimizador Adam con una tasa de aprendizaje inicial de 0.0001. Para optimizar el entrenamiento, se implementó un criterio de detección anticipada con una paciencia de 10 épocas, lo que permitió finalizar el entrenamiento si no se observaban mejoras significativas en ese periodo. Además, se utilizó el callback `ReduceLROnPlateau`, que ajusta automáticamente la tasa de aprendizaje reduciéndola en un factor de 0.2 después de 5 épocas consecutivas sin mejora en la métrica de interés. Finalmente, se empleó un tamaño de lote de 16 imágenes para garantizar un equilibrio entre rendimiento y estabilidad durante el proceso de entrenamiento.

La principal diferencia entre los dos tipos de clasificación radica en la función de pérdida y la función de salida de la capa final:

- Para la clasificación multiclase, se utilizó entropía cruzada categórica con una capa de salida softmax, correspondiente a los enfoques mostrados en la Tabla 4.3, para clasificar fondo de ojo normal y las cuatro etapas de la retinopatía diabética.
- Para la clasificación multietiqueta, se utilizó entropía cruzada binaria con una capa de

salida sigmoide, correspondiente a los enfoques mostrados en la Tabla 4.4, para detectar la presencia o ausencia de múltiples biomarcadores en cada imagen.

Ambos modelos fueron entrenados en el entorno de Kaggle con una GPU, utilizando validación cruzada de 5 folds para obtener el mejor rendimiento. Los modelos fueron evaluados utilizando las métricas de precisión, recall y F1-score, permitiendo analizar su desempeño en la clasificación de etapas y biomarcadores.

4.3.1. Modelo de Clasificación de Etapas de Retinopatía Diabética

Este experimento evaluó 4 experimentos para dos enfoques para la clasificación de etapas de la retinopatía diabética utilizando conjuntos de imágenes del fondo de ojo, de las cuales se extrajeron 3031 imágenes en total. Cada estrategia incluye uno o más modelos, que difieren en la cantidad de clases a clasificar y en la distribución de las imágenes entre los conjuntos de entrenamiento y validación:

- **Modelo de 2 Clases:** Este modelo clasificó las imágenes como fondo de ojo normal o retinopatía diabética. Para su entrenamiento se utilizaron 2217 imágenes y 428 para validación.
- **Modelo de 4 Clases:** Una vez detectada la retinopatía diabética mediante el modelo de 2 clases, se aplicó un segundo modelo para clasificar las imágenes en cuatro etapas de la enfermedad (leve, moderada, severa, proliferativa). Este submodelo fue entrenado con 1597 imágenes y validado con 185.
- **Modelo de 5 Clases:** Este modelo abordó directamente la clasificación en cinco clases (fondo de ojo normal y las cuatro etapas de la retinopatía diabética). Al igual que el modelo de 2 clases, se utilizaron 2217 imágenes para entrenamiento y 428 para validación.

La primera estrategia consistió en un proceso en dos pasos. Inicialmente, el modelo de 2 clases determinó si una imagen correspondía a un fondo de ojo normal o a un caso de retinopatía diabética (Tabla 4.3, fila 1). Si la imagen era clasificada como retinopatía diabética, se procesaba en un segundo modelo encargado de determinar su etapa específica entre cuatro clases (Tabla 4.3, fila 2). Por el contrario, si la imagen era clasificada como normal, no se requería procesamiento adicional.

En la segunda estrategia, el modelo de 5 clases permitió realizar la clasificación directa de las imágenes en todas las categorías simultáneamente (Tabla 4.3, fila 4). Finalmente, los resultados de cada estrategia se presentan en la (Tabla 4.3, fila 3), permitiendo analizar y comparar la precisión y el rendimiento de cada método.

Además, se presentan las curvas de entrenamiento y validación (Figura 4.4), que muestran el progreso del modelo de dos clases y del modelo de cuatro clases en términos de precisión y pérdida durante las iteraciones.

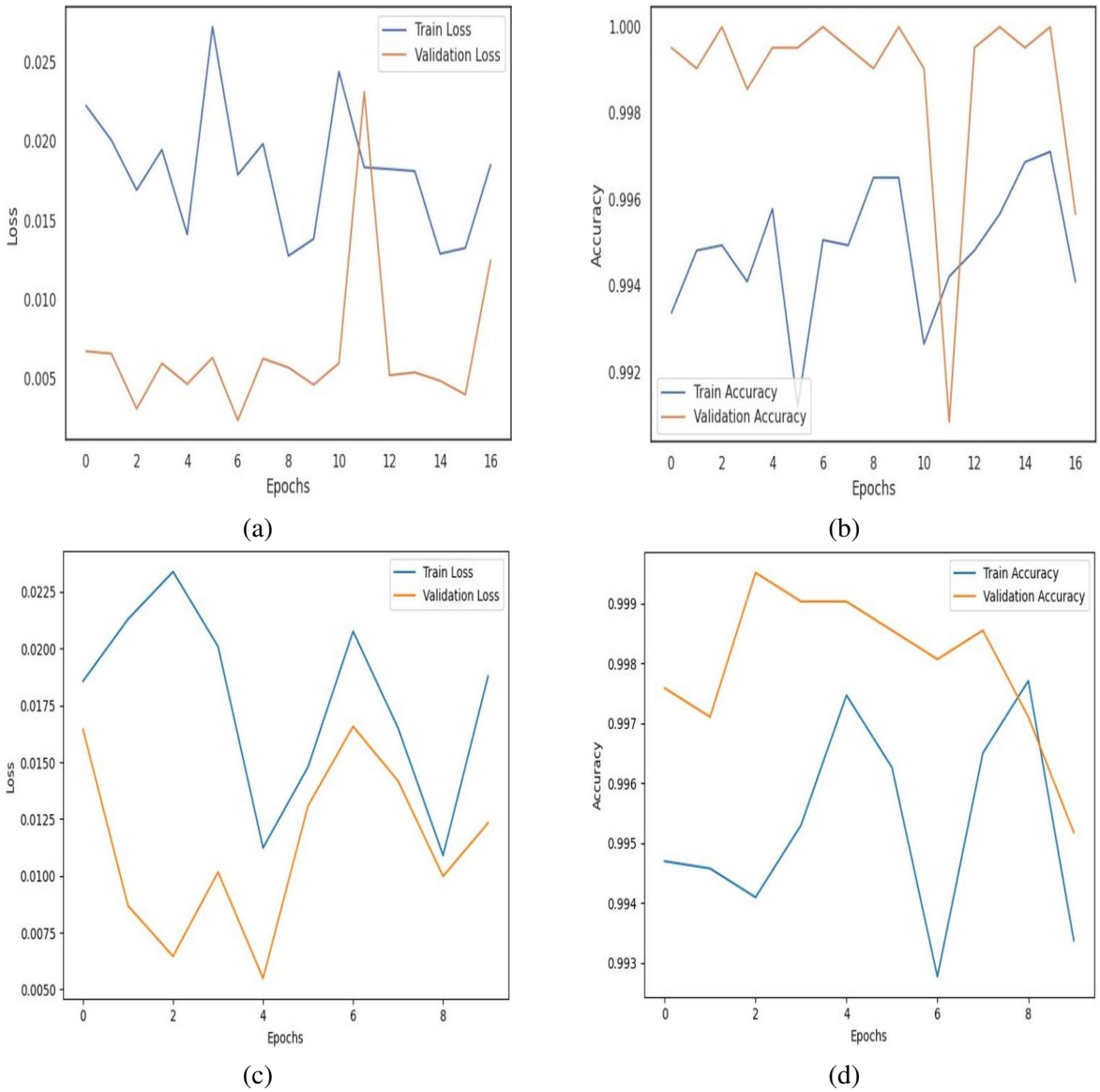
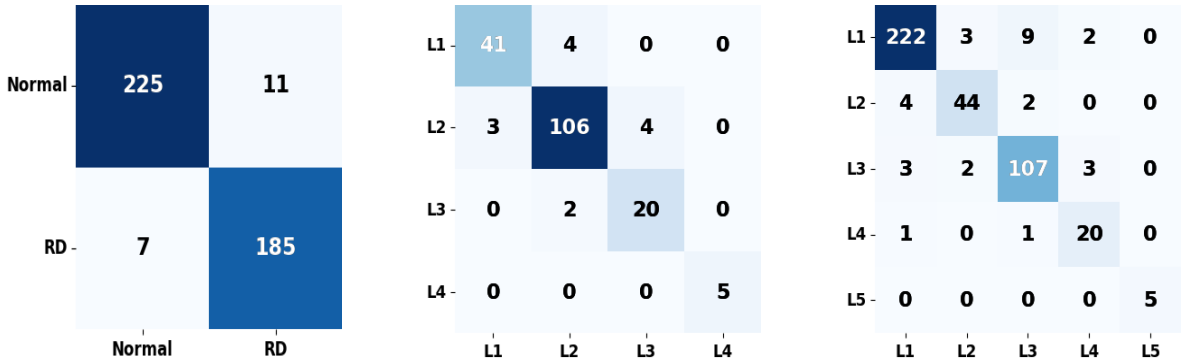


Figura 4.4: Comparación de resultados: (a) Curvas de entrenamiento para el modelo de cuatro clases, (b) Curvas de entrenamiento para el modelo de cinco clases.

El primer enfoque también incluye las matrices de confusión para las predicciones del modelo de dos clases y del modelo de cuatro clases, que proporcionan información detallada sobre la distribución de las predicciones, como se muestra en la Figura 4.5.

En estas matrices, las clases están representadas por las etiquetas L1 a L5, donde L1 corresponde a fondo de ojo normal, L2 a retinopatía diabética no proliferativa leve, L3 a retinopatía diabética no proliferativa moderada, L4 a retinopatía diabética no proliferativa severa y L5 a retinopatía diabética proliferativa.



(a) (b) (c)

Figura 4.5: Comparación de resultados: (a) Matriz de confusión para el modelo de dos clases, (b) Matriz de confusión para el modelo de cuatro clases, (c) Matriz de confusión para el modelo de cinco clases.

El objetivo de comparar estos enfoques fue determinar si el uso de un proceso de dos pasos con una clasificación inicial más simple mejoraría el rendimiento del modelo de clasificación de etapas, concluyendo que el primer enfoque presenta la mayor precisión con un valor de 0.9444.

Tabla 4.3: Resultados de Prueba en Clasificación Multi-Clase

Modelo	Accuracy	precisión	Recall	F1-Score
2 Clases	0.9579	0.9568	0.9584	0.9575
4 Clases	0.9297	0.9278	0.9395	0.9332
2 & 4 Clases	0.9444	0.9035	0.9451	0.9215
5 Clases	0.9299	0.9124	0.932	0.9214

4.3.2. Modelo de Identificación de Biomarcadores de Retinopatía Diabética

Este experimento se centró en la detección de etiquetas clave de la retinopatía diabética, tales como microaneurismas, hemorragias, exudados duros y exudados suaves.

En el modelo de cuatro clases, la detección de etiquetas se aplicó exclusivamente a las imágenes previamente clasificadas como retinopatía diabética mediante el modelo de dos clases (Tabla 4.4, fila 1). Si la imagen se clasificaba como fondo de ojo normal, no se utilizaba el modelo de biomarcadores. Este modelo fue entrenado con 2076 imágenes y validado con 519 imágenes, enfocándose únicamente en las etapas de la retinopatía diabética.

Por otro lado, en el modelo de cinco etiquetas, se incluyeron directamente todas las imágenes en el proceso de clasificación, distribuyéndolas entre fondo de ojo normal y las clases de retinopatía diabética (Tabla 4.4, fila 2). Esto aumentó el tamaño del conjunto de datos para entrenamiento y validación en comparación con el modelo de cuatro clases.

Esto se realizó para determinar si aplicar el modelo de detección de biomarcadores después de la clasificación de dos clases mejora el rendimiento en comparación con aplicarlo directamente en la clasificación de cinco clases. Se evaluaron, además, las matrices de confusión

generadas para cada biomarcador en ambos casos. Estas matrices, presentadas en las Figuras 4.6 y 4.7 (Dado A representa la ausencia del biomarcador, P su presencia, D indica detección correcta y U corresponde a un biomarcador no detectado.) nos muestran que el primer enfoque propuesto en la sección 4.3.1 es la mejor metodología.

Tabla 4.4: Resultados de Prueba en Clasificación Multi-Etiqueta

Modelo	precisión	Recall	F1-Score
4 Clases	0.9770	0.9498	0.9551
5 Clases	0.9173	0.9078	0.9049

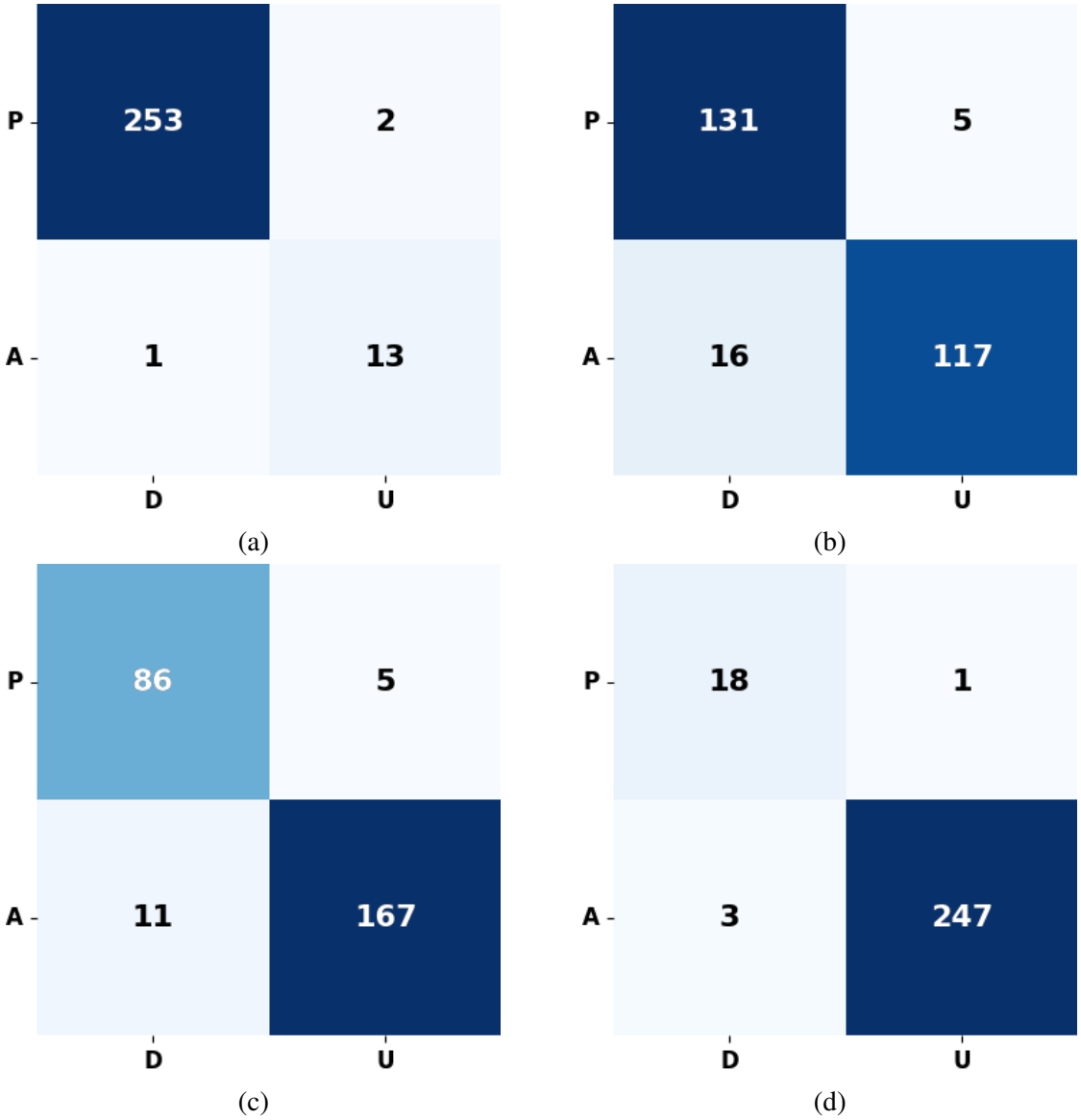


Figura 4.6: Matrices de confusión para el modelo de 4 etiquetas, evaluando la detección de biomarcadores: (a) Microaneurismas, (b) Hemorragias, (c) Exudados Duros y (d) Exudados Suaves.

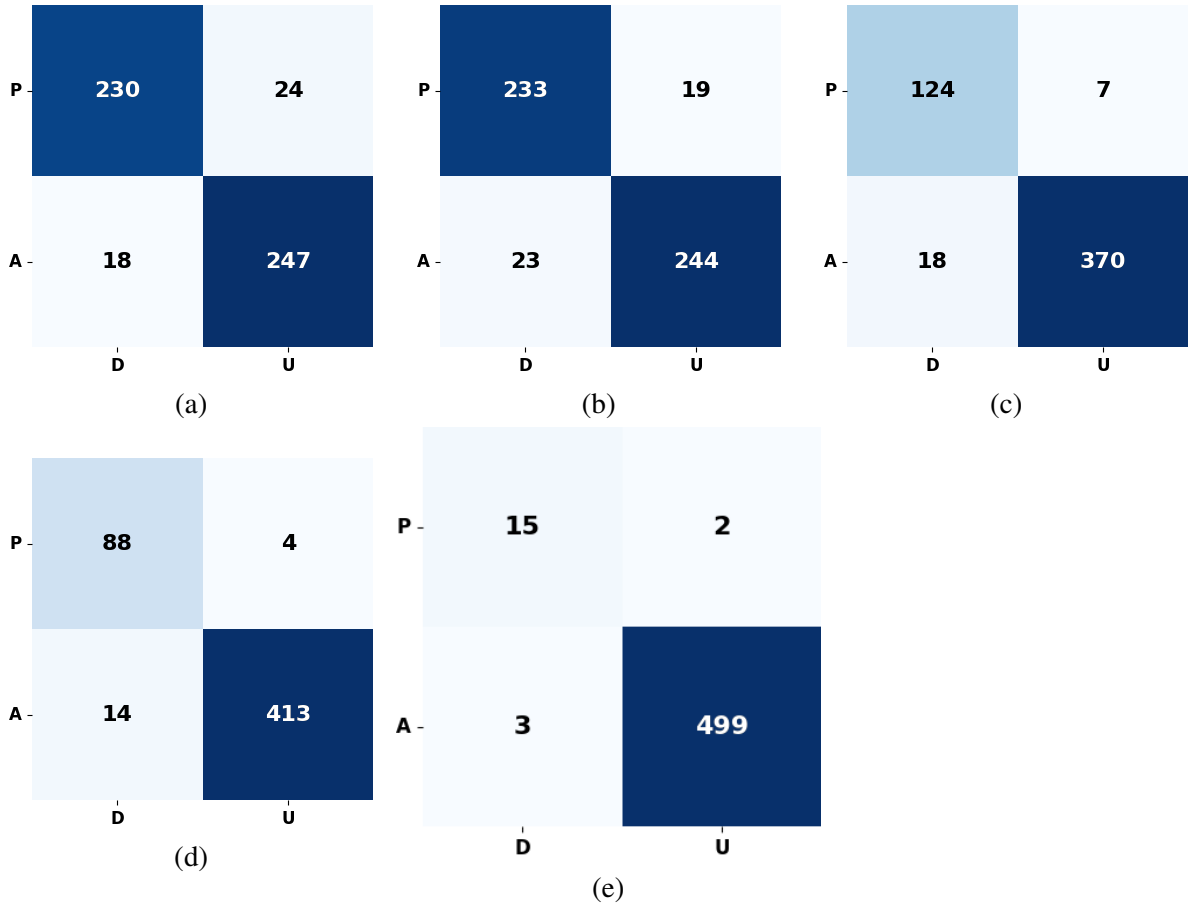


Figura 4.7: Matrices de confusión para el modelo de 5 etiquetas, evaluando la clasificación directa: (a) Fondo de ojo Normal, (b) Microaneurismas, (c) Hemorragias, (d) Exudados Duros y (e) Exudados Suaves.

4.4. Detección y Segmentación de Objetos

La detección y segmentación de objetos en este proyecto es fundamental para identificar y delimitar con precisión biomarcadores en imágenes de fondo de ojo. La detección permite localizar regiones de interés, mientras que la segmentación delimita los contornos exactos de las estructuras relevantes. En este caso, la clasificación por sí sola no es suficiente, ya que no solo es importante determinar la presencia de un biomarcador, sino también su ubicación dentro de la imagen.

4.4.1. Flujo de Trabajo para la Detección de Objetos

El proceso de detección de objetos para los biomarcadores utilizó un modelo YOLOv8 preentrenado de Ultralytics, ejecutado en el entorno de Kaggle con una GPU NVIDIA Tesla P-100 para optimizar el entrenamiento y la inferencia. Como parte de este estudio, se desarrollaron nuevas anotaciones mediante la plataforma de makesense.ai, adicionales a las que provee el conjunto de datos ODIR-5K. A continuación, se describen los pasos realizados:

1. Se cargaron las imágenes oculares en la plataforma de makesense.ai, tal como se muestra en la Figura 4.8, donde se ilustra el diálogo de selección de imágenes por parte del usuario.

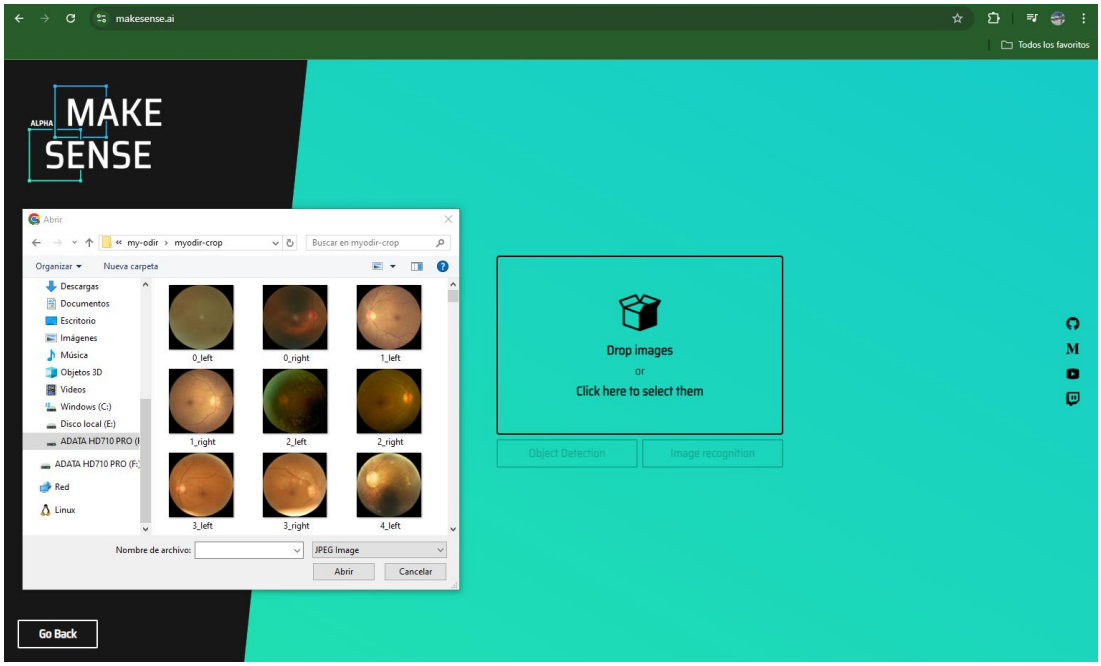


Figura 4.8: Interfaz de la plataforma makesense.ai utilizada para la carga y selección de imágenes oculares por parte del usuario.

2. Se eligió la opción de detección de objetos para definir cuadros delimitadores (bounding boxes) alrededor de los biomarcadores específicos, como se ilustra en la Figura 4.9.

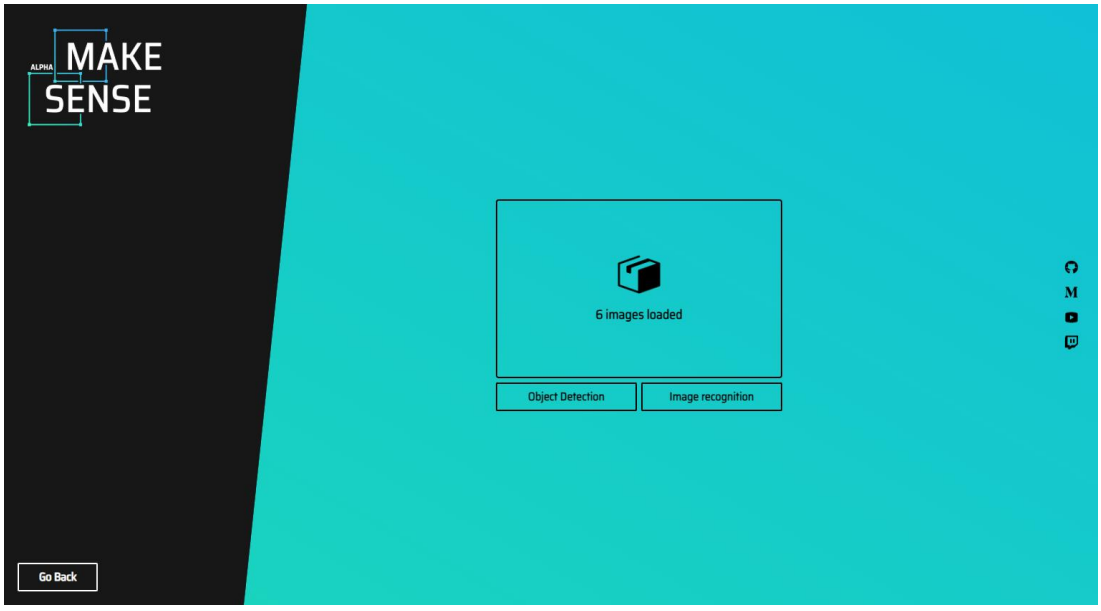


Figura 4.9: Selección de la opción de detección de objetos en la plataforma makesense.ai para definir cuadros delimitadores (bounding boxes) alrededor de biomarcadores específicos.

3. Se definieron etiquetas específicas para cada biomarcador, incluyendo Microaneurismas, Hemorragias, Exudados Duros y Exudados Suaves. Estas etiquetas fueron aplicadas de manera consistente a todas las anotaciones, como se observa en la Figura 4.10.

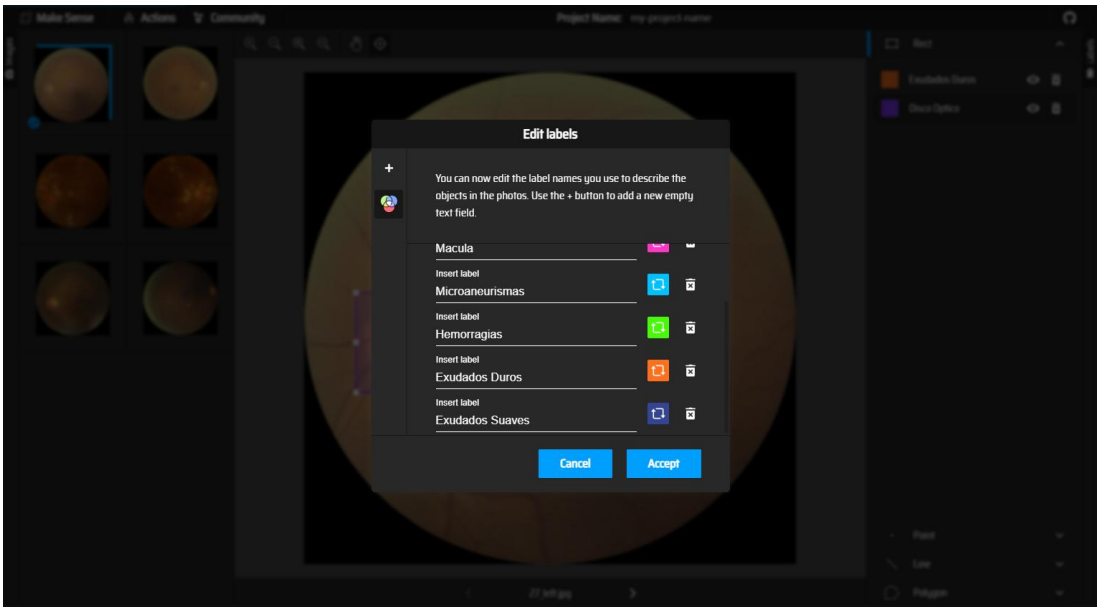


Figura 4.10: Definición y aplicación de etiquetas específicas para los biomarcadores (Microaneurismas, Hemorragias, Exudados Duros y Exudados Suaves) en la plataforma makesense.ai.

4. Para cada imagen, se trazaron manualmente cuadros delimitadores alrededor de las regiones de interés correspondientes a los biomarcadores, como se muestra en la Figura 4.11.

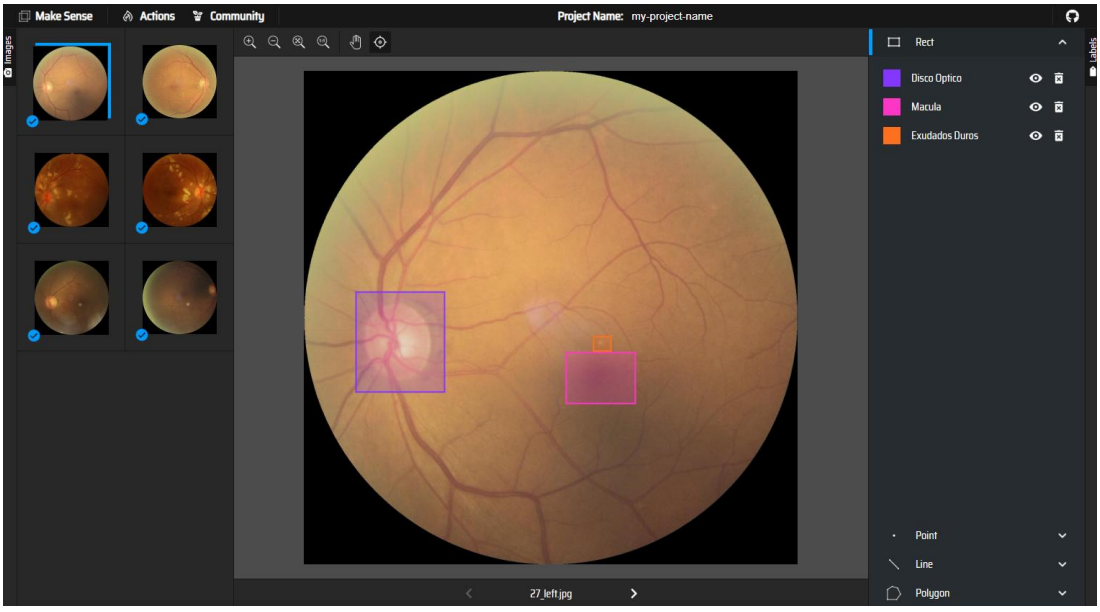


Figura 4.11: Proceso manual de trazado de cuadros delimitadores para identificar las regiones de interés asociadas a los biomarcadores en la plataforma makesense.ai.

5. Una vez finalizada la anotación, los archivos se exportaron en formato YOLO (.txt) para integrarlos en el flujo de trabajo del modelo de detección de biomarcadores, como se evidencia en la Figura 4.12. El archivo correspondiente a cada imagen contiene el tipo de biomarcador anotado, su etiqueta y las coordenadas de los cuadros delimitadores que indican su posición en la imagen.

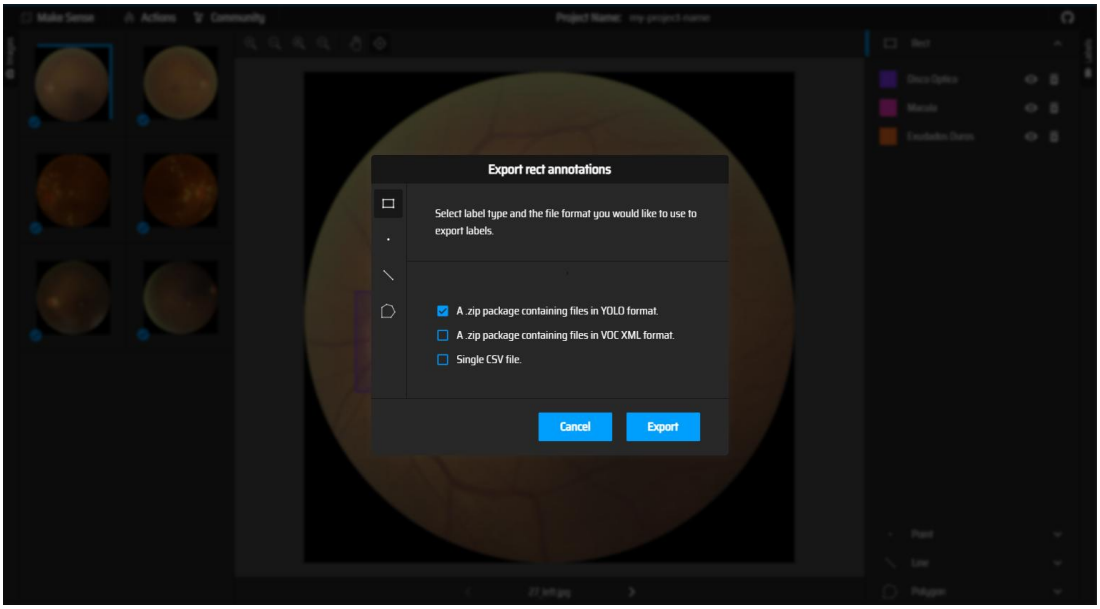


Figura 4.12: Exportación de anotaciones en formato YOLO, incluyendo la clase y las coordenadas de los cuadros delimitadores, para su integración en el modelo de detección.

Después del proceso de anotación, el conjunto de datos se dividió en 80 % de las imágenes (160) para entrenamiento y 20 % (40) para pruebas. Se generó un archivo en formato .yaml, que especifica las rutas de los archivos de las imágenes en el conjunto de datos, el número de imágenes y los tipos de biomarcadores incluidos, como se ilustra en la Figura 4.13.

```yaml
train: ./yolo/train
val: ./yolo/val

nc: 6
names: ["Disco Optico","Fovea", "Microaneurismas", "Hemorragias", "Exudados Duros","Exudados Blandos"]
```

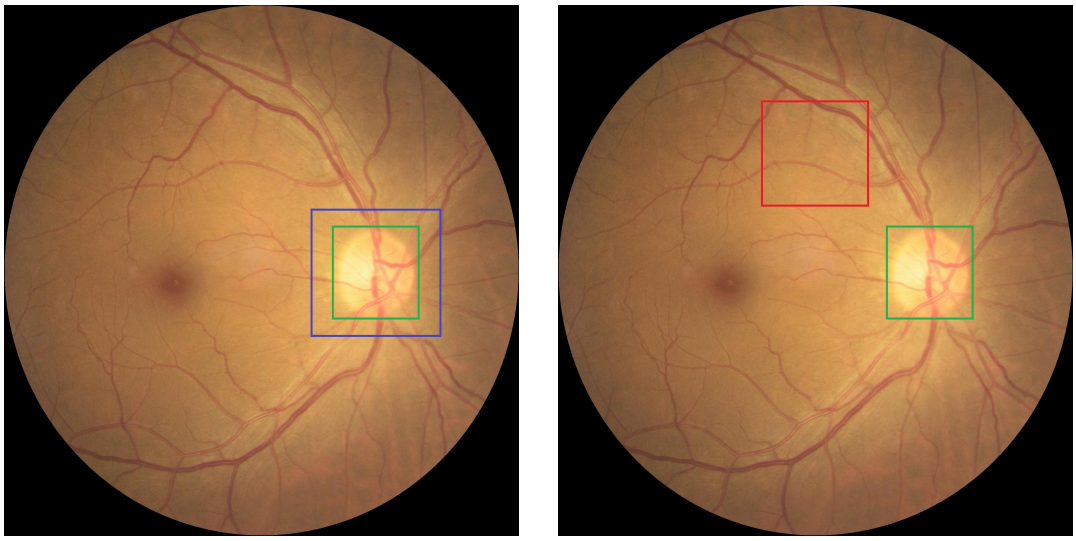
Figura 4.13: Archivo YAML generado para configurar las rutas del conjunto de datos, el número de imágenes y los tipos de biomarcadores utilizados en el modelo de detección.

El modelo se entrenó durante 120 épocas utilizando la arquitectura YOLOv8[11] y se exportó en formato ONNX. En la Figura 4.14 se muestra comó se evaluó el rendimiento comparando los cuadros delimitadores previstos con los cuadros delimitadores conocidos en términos de superposición como se muestra. Las métricas de evaluación utilizadas fueron Precisión, Recall y F1-Score, cuyos resultados se muestran en la Tabla 4.5.

[11] YOLOv8 Architecture. https://yolov8.org/yolov8-architecture/

Tabla 4.5: Resultados de la prueba en detección de objetos

Biomarcadores	precisión	Recall	F1-Score
Disco Óptico	1.0	1.0	1.0
Mácula	0.975	0.975	0.975
Microaneurismas	0.964	0.192	0.321
Hemorragias	0.944	0.670	0.784
Exudados Duros	0.944	0.510	0.662
Exudados Suaves	1.000	0.775	0.873



(a) (b)

Figura 4.14: Ejemplo de evaluación del modelo YOLOv8: (a) el cuadro delimitador azul corresponde a la predicción del modelo y el cuadro verde a la etiqueta real, mostrando una coincidencia correcta. (b) el cuadro delimitador rojo no coincide con el cuadro verde, representando un error en la predicción.

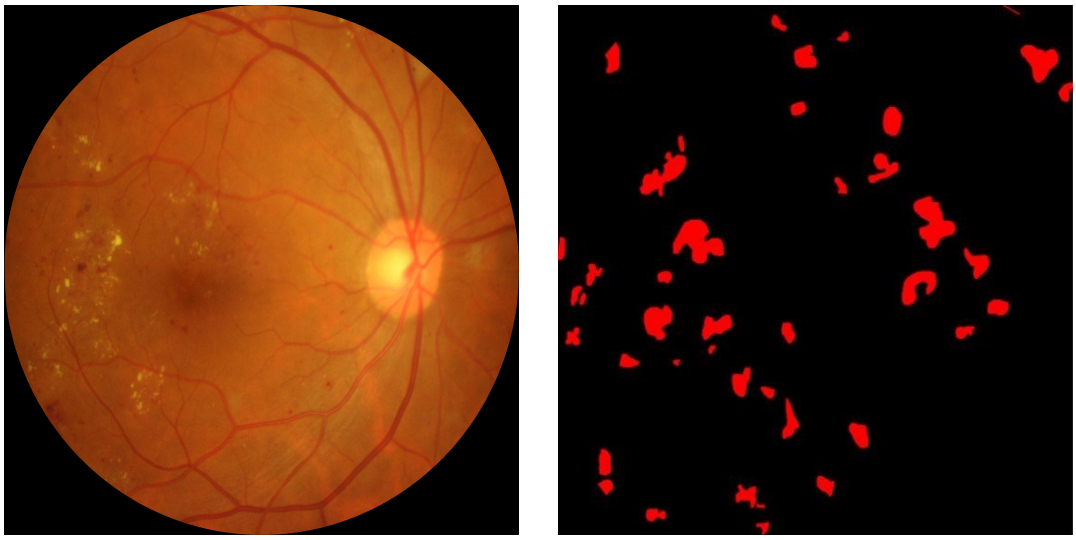
4.4.2. Flujo de Trabajo para la Segmentación

La segmentación se aplicó para delimitar regiones específicas dentro de las imágenes de retina correspondientes a biomarcadores identificados. Esto requirió entrenar siete modelos, uno para cada biomarcador. No se hizo distinción entre exudados suaves y duros, ya que las máscaras del conjunto de datos de Messidor-2 no los diferencian.

Para cada modelo de biomarcadores, las regiones de interés de las máscaras se alinearon con las imágenes de retina correspondientes antes del proceso de recorte. Sin embargo, antes de iniciar este procedimiento, fue necesario organizar y gestionar adecuadamente los datos. Las imágenes de retina y las máscaras correspondientes se agruparon en carpetas separadas, una destinada a las imágenes originales y otra a las máscaras.

Las CFI, en su mayoría, requirieron realizar el preprocesamiento de recorte circular para

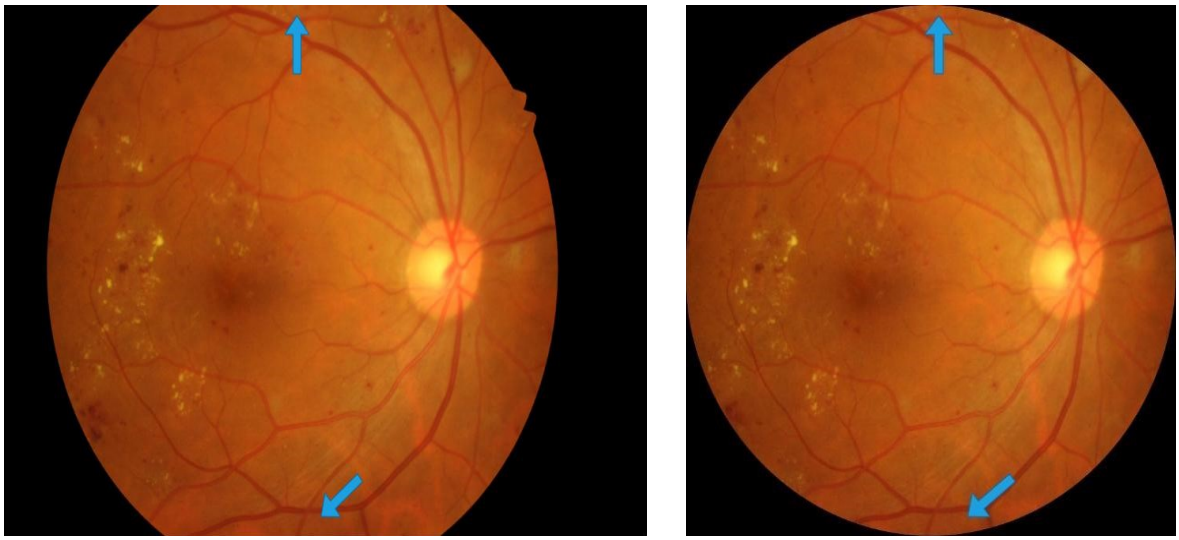
uniformar las regiones de interés y asegurar dimensiones homogéneas entre las muestras. Este proceso funciona correctamente para las CFI, sin embargo, el mismo algoritmo no era efectivo para las máscaras debido a su diseño binario. En la Figura 4.15, se ilustra cómo las máscaras no podían procesarse correctamente con el algoritmo de recorte.



(a) (b)

Figura 4.15: Comparación del algoritmo de recorte: (a) Imagen original del fondo de ojo con curvatura preservada, (b) Máscara binaria correspondiente, donde el algoritmo de recorte falla al no identificar correctamente la curvatura.

Además, en la Figura 4.16 se evidencia otro problema relacionado con la falta de curvatura en las máscaras. Mientras que el algoritmo de recorte puede identificar y preservar la curvatura del fondo de ojo en las imágenes originales, este no es el caso para las máscaras. Dado que las máscaras son imágenes binarias sin información de color o textura, el algoritmo no lograba detectar la curvatura.

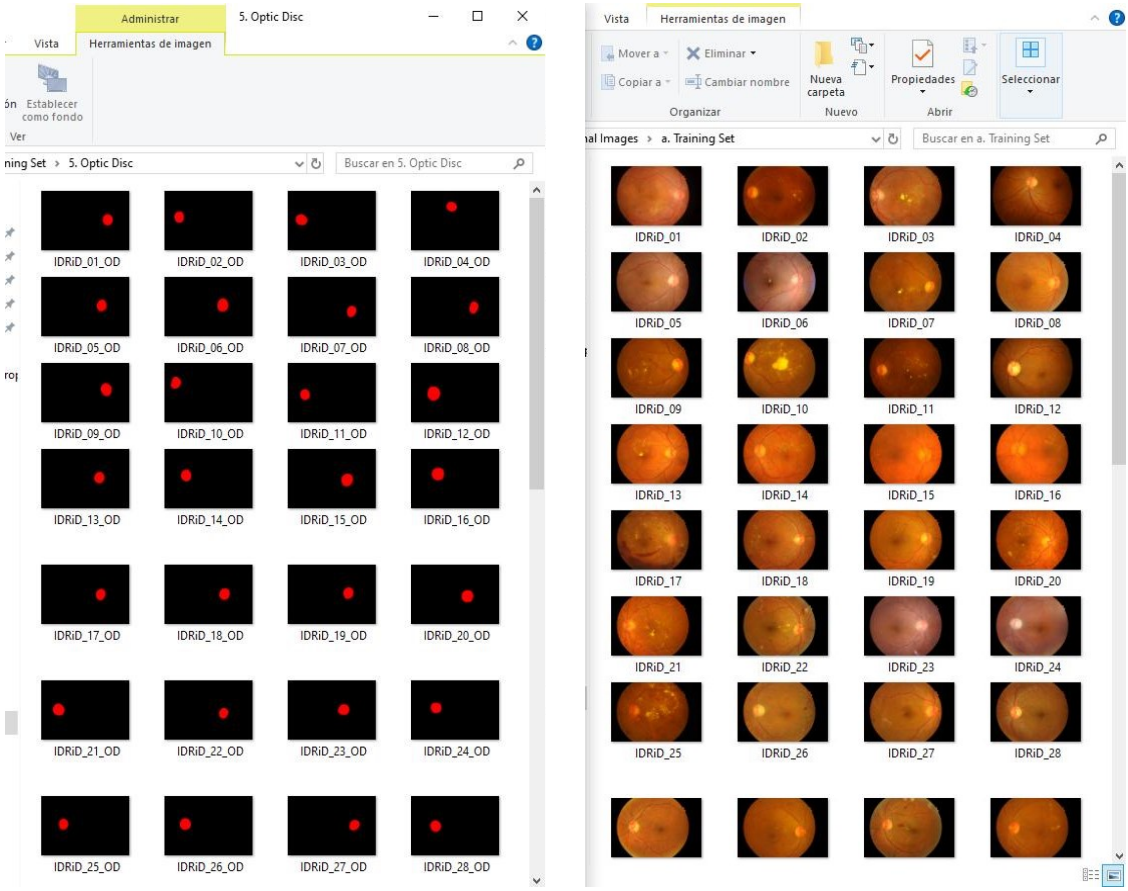


(a) (b)

Figura 4.16: Comparación del algoritmo de recorte: (a) Imagen sin aplicar el algoritmo, con flechas azules señalando la ausencia de curvatura en ciertas áreas, (b) Imagen con el algoritmo aplicado, donde las flechas azules destacan la curvatura creada por el proceso de recorte.

Ante esos problemas, se implementaron los siguientes pasos para garantizar que tanto las imágenes como las máscaras estuvieran adecuadamente alineadas y listas para el entrenamiento del modelo:

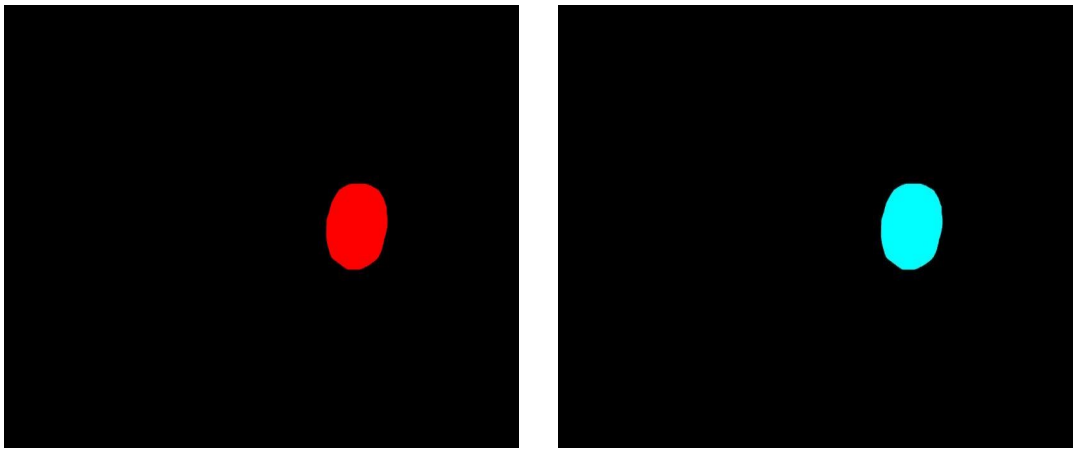
1. Se organizaron las imágenes de fondo de ojo en una carpeta y las máscaras correspondientes en otra como se muestra en la Figura 4.17.



(a) (b)

Figura 4.17: Organización inicial de las imágenes de fondo de ojo y sus máscaras: (a) Ventana que muestra las máscaras correspondientes a las imágenes, (b) Ventana que contiene las imágenes originales sin anotaciones.

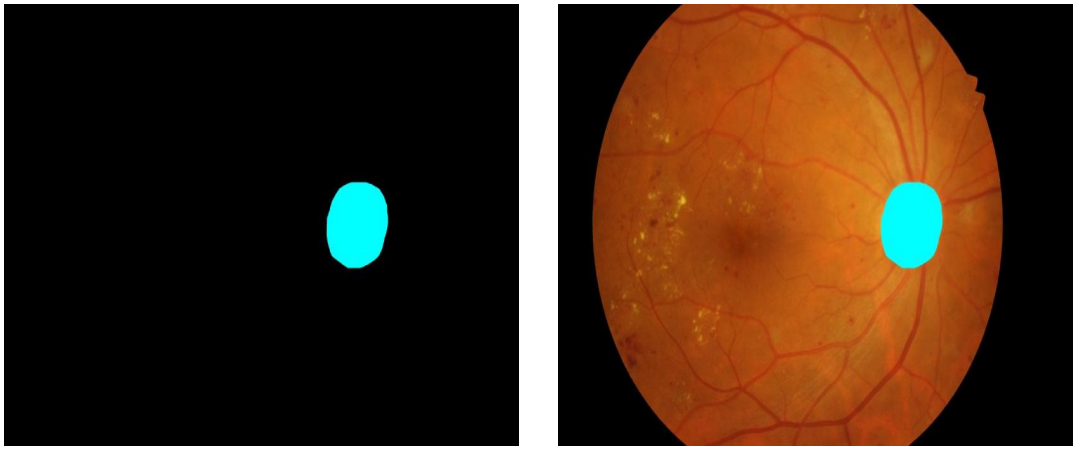
2. Las máscaras binarias fueron transformadas al color cian, ya que este resalta de manera efectiva los píxeles de las regiones de interés, mejorando su visibilidad. Este cambio beneficia los pasos posteriores al facilitar la identificación y procesamiento de estas áreas, tal como se ilustra en la Figura 4.18.



(a) (b)

Figura 4.18: Transformación de máscaras binarias a color cian para resaltar las regiones de interés: (a) Máscara binaria original, (b) Máscara transformada con la región de interés resaltada en color cian.

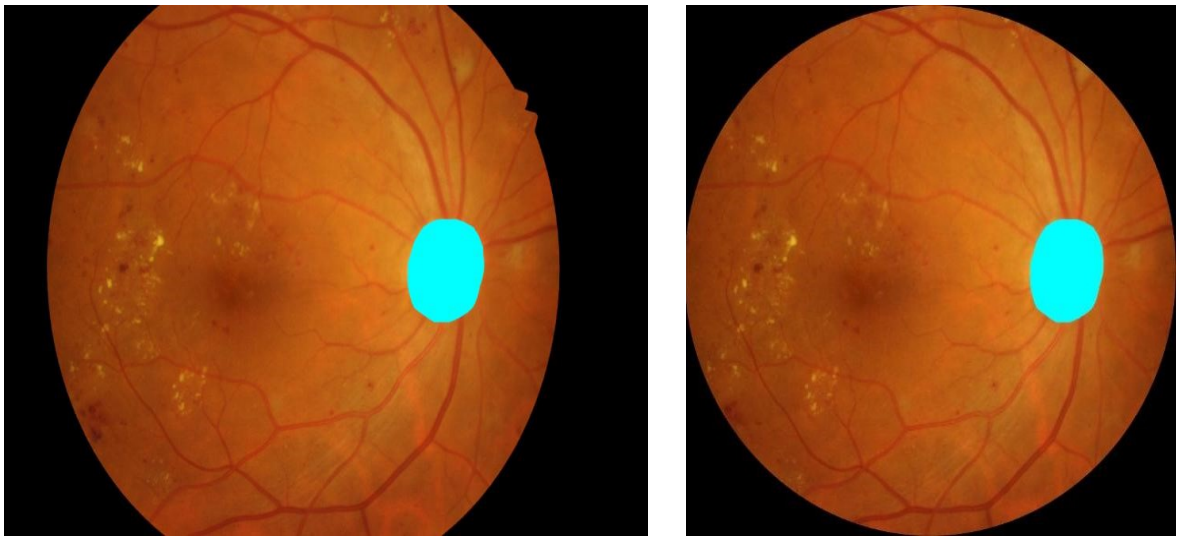
3. Los píxeles de color cian, correspondientes a las regiones de interés, se sobrepusieron en las imágenes originales, manteniendo la misma posición automaticamente, tal como se muestra en la Figura 4.19.



(a) (b)

Figura 4.19: Superposición de píxeles cian sobre las imágenes originales: (a) Máscara con la región de interés resaltada en color cian, (b) Imagen original con la máscara superpuesta para resaltar la región de interés.

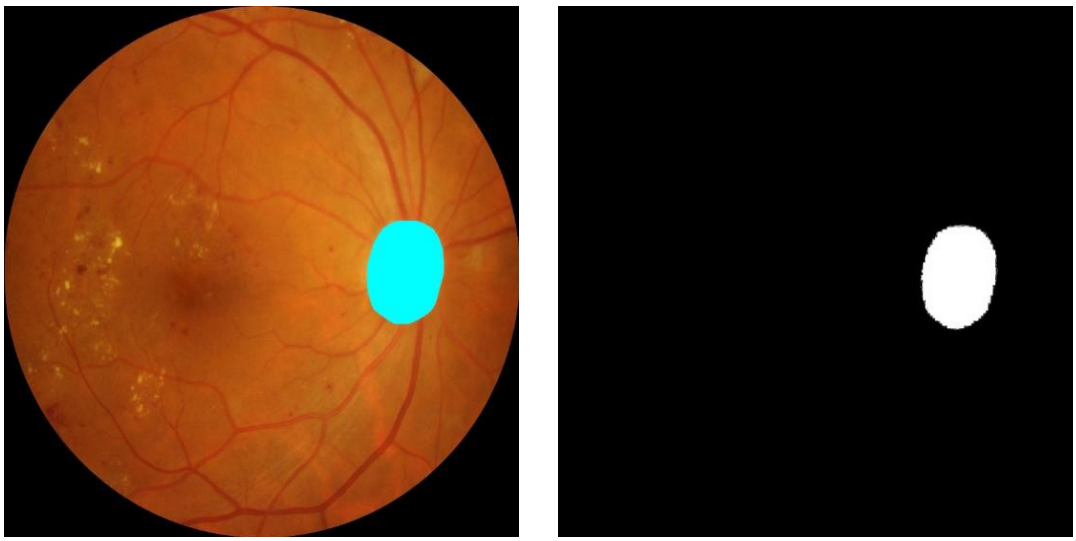
4. Se aplicó el algoritmo de recorte a las imágenes originales y sus máscaras correspondientes para eliminar áreas irrelevantes, tal como se define en la Figura 4.20.



(a) (b)

Figura 4.20: Resultado del recorte de imágenes y máscaras para eliminar áreas irrelevantes: (a) Imagen original sin recorte, (b) Imagen tras aplicar el preprocesamiento de recorte.

5. Se extrajeron los píxeles de color cian y se transformaron a color blanco, generando una máscara binaria final que representa exclusivamente las regiones de interés, como se muestra en la Figura 4.21.



(a) (b)

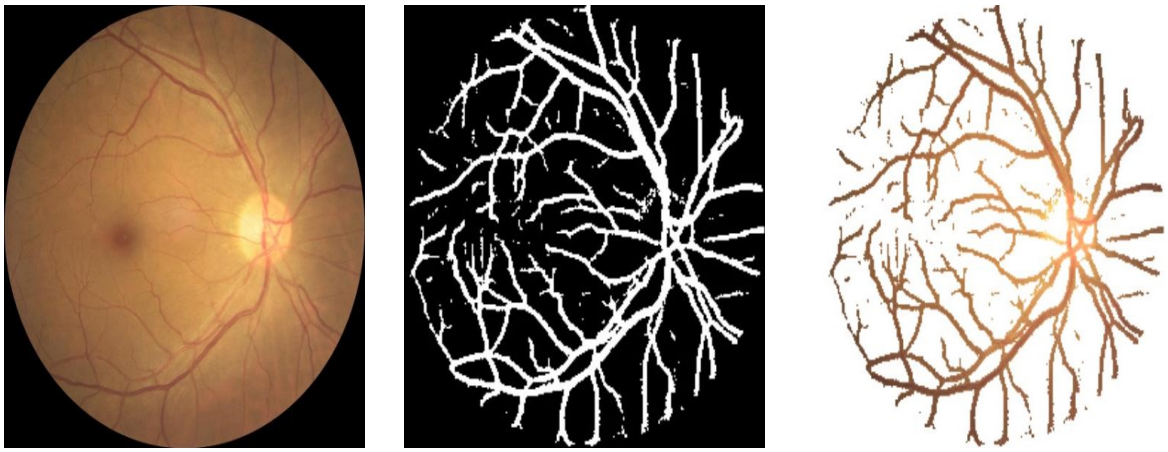
Figura 4.21: Máscara binaria final con regiones de interés en color blanco: (a) Imagen recortada final, (b) Máscara binaria con la región de interés previamente resaltada en color cian.

Se generaron cuadros delimitadores a partir de las máscaras para capturar las regiones de interés. El conjunto de datos se dividió en un 80 % para entrenamiento y un 20 % para pruebas. Los

cuadros delimitadores y las imágenes de retina se usaron como entradas para el entrenamiento del modelo. Se utilizó SAM con una base de Vision Transformer (ViT), inicializada con pesos preentrenados. Durante el ajuste fino, solo se actualizaron los parámetros del decodificador de máscaras, mientras que los codificadores de visión y de prompt permanecieron congelados para preservar sus representaciones aprendidas. El modelo se entrenó durante 30 épocas utilizando el optimizador Adam con una tasa de aprendizaje de 1×10^{-5}. La función de pérdida Dice Cross-Entropy se empleó para equilibrar la precisión a nivel de píxel y el rendimiento de la segmentación basada en regiones. Durante el entrenamiento, las predicciones del modelo se redimensionaron para que coincidieran con las máscaras de referencia, y se calcularon gradientes para actualizar los parámetros del decodificador de máscaras, mejorando así la precisión de la segmentación. El rendimiento del modelo se evaluó utilizando el coeficiente de Dice, que mide la superposición entre las máscaras predichas y las máscaras de referencia. Los resultados, resumidos en la Tabla 4.6, demuestran la efectividad del modelo SAM ViT Base en la segmentación precisa de biomarcadores dentro de imágenes de retina. Adicionalmente, se presentan los resultados individuales de segmentación para cada biomarcador, incluyendo vasos sanguíneos 4.22, disco óptico4.23, copa óptica 4.24, mácula 4.25, microaneurismas 4.26, hemorragias 4.27 y exudados 4.28.

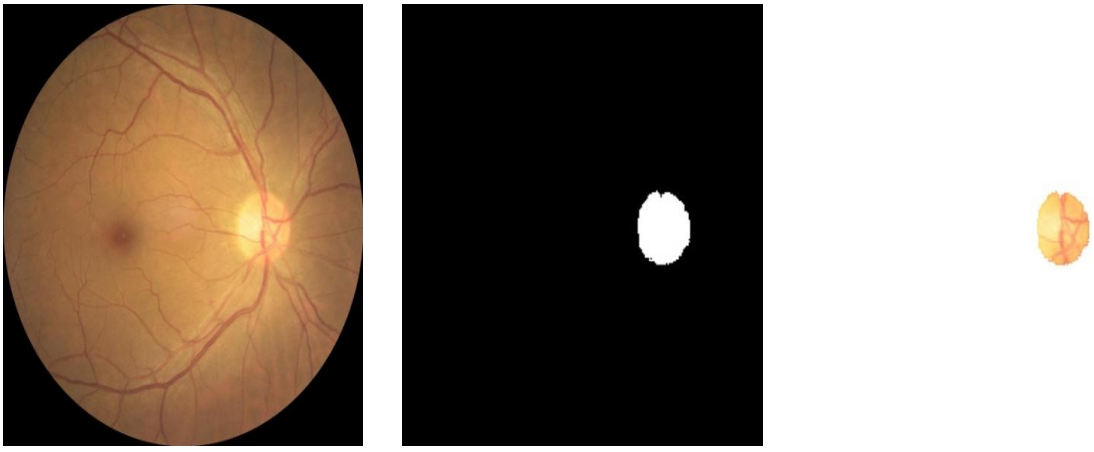
Tabla 4.6: Resultados de Dice Score para la segmentación de biomarcadores de retinopatía diabética

Biomarcador	**Dice**	**AUC-PR**	**AUC-ROC**
Disco Óptico	0.9510	0.9540	0.9662
Copa Óptica	0.8162	0.8381	0.9300
Vasos Sanguineos	0.7076	0.7307	0.8564
Mácula	0.7871	0.8060	0.9017
Microaneurismas (L1)	0.5454	0.5562	0.8033
Hemorragias (L2)	0.6481	0.6761	0.8124
Exudados (L3,L4)	0.6635	0.6890	0.8910



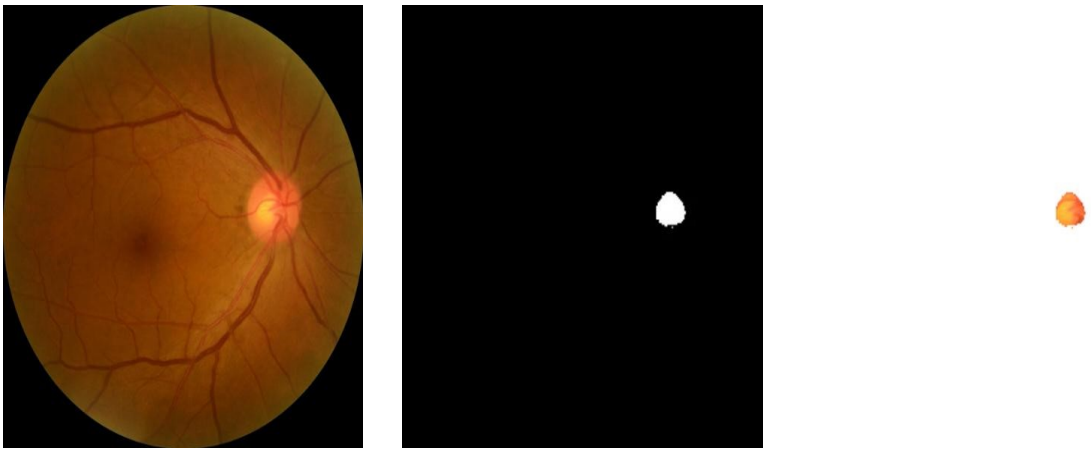
(a) (b) (c)

Figura 4.22: Resultados de segmentación en imágenes originales: (a) Imagen original, (b) Máscara binaria de vasos sanguíneos segmentados, (c) Píxeles extraídos según la región de interés.



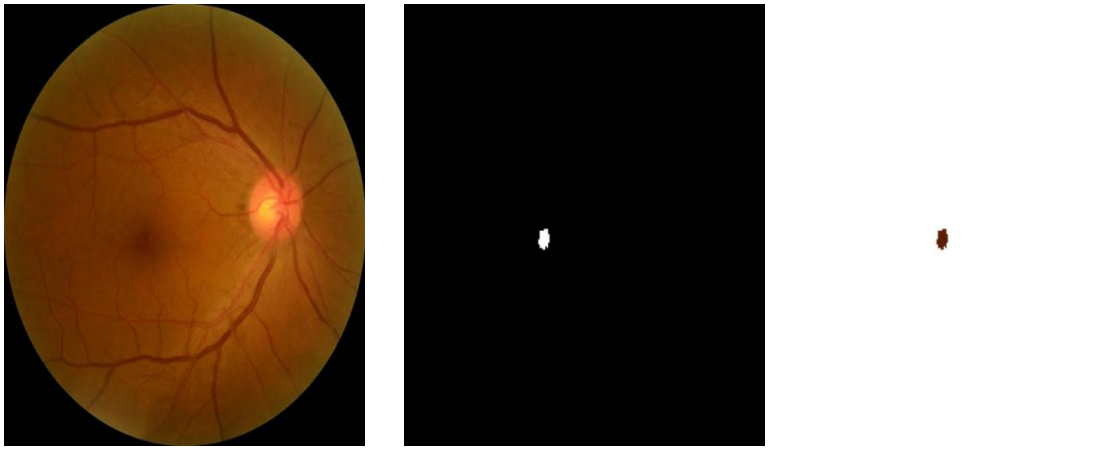
(a) (b) (c)

Figura 4.23: Resultados de segmentación en imágenes originales: (a) Imagen original, (b) Máscara binaria del disco óptico segmentado, (c) Píxeles extraídos según la región de interés.



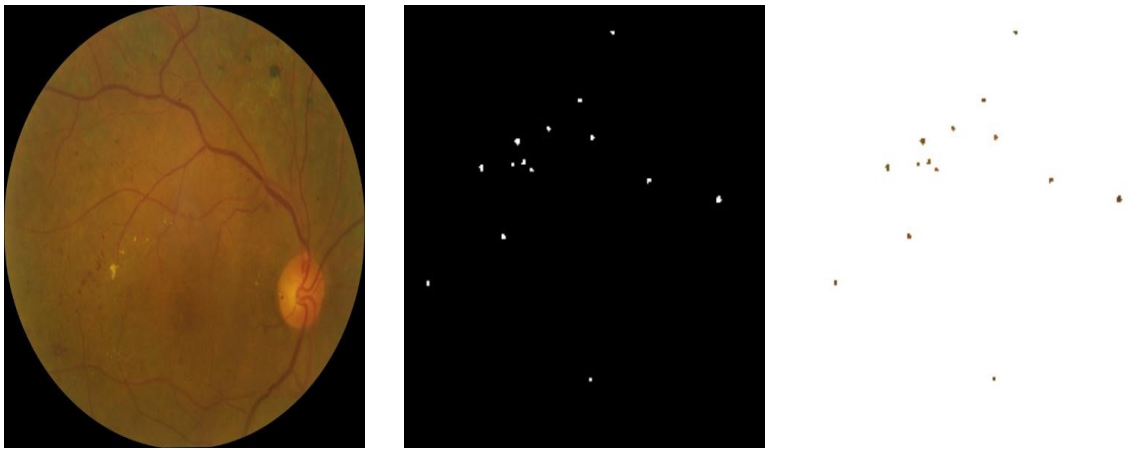
(a) (b) (b)

Figura 4.24: Resultados de segmentación en imágenes originales: (a) Imagen original, (b) Máscara binaria de la copa óptica segmentada, (c) Píxeles extraídos según la región de interés.



(a) (b) (b)

Figura 4.25: Resultados de segmentación en imágenes originales: (a) Imagen original, (b) Máscara binaria de la mácula segmentada, (c) Píxeles extraídos según la región de interés.

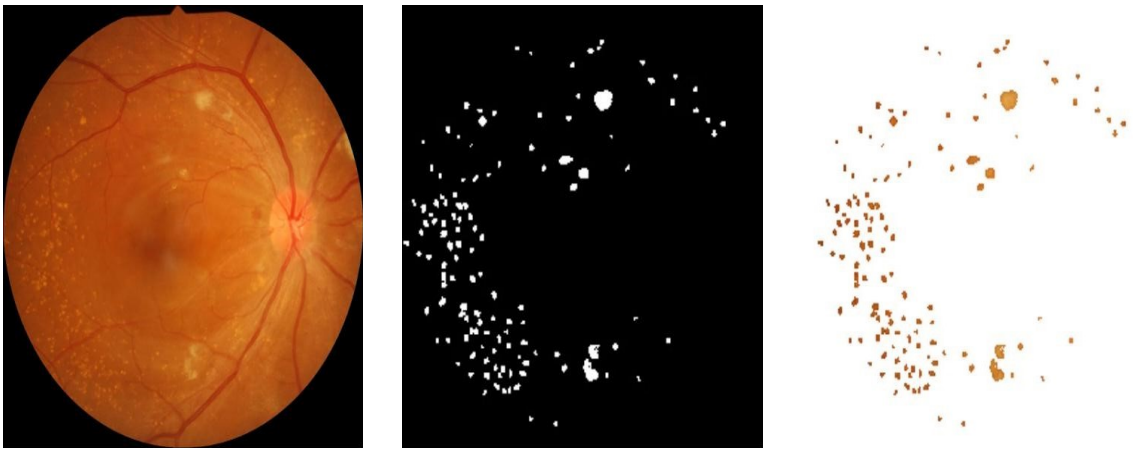


(a) (b) (b)

Figura 4.26: Resultados de segmentación en imágenes originales: (a) Imagen original, (b) Máscara binaria de microaneurismas segmentadas, (c) Píxeles extraídos según la región de interés.

(a) (b) (b)

Figura 4.27: Resultados de segmentación en imágenes originales: (a) Imagen original, (b) Máscara binaria de hemorragias segmentadas, (c) Píxeles extraídos según la región de interés.



(a) (b) (b)

Figura 4.28: Resultados de segmentación en imágenes originales: (a) Imagen original, (b) Máscara binaria de exudados segmentados, (c) Píxeles extraídos según la región de interés.

4.5. Generación de Imágenes Sintéticas

La generación de imágenes sintéticas en este proyecto tiene como objetivo aumentar la diversidad del conjunto de datos y mejorar la capacidad del modelo para generalizar. Se generaron imágenes sintéticas para complementar las imágenes reales y abordar posibles sesgos en la distribución de datos. Esta estrategia es importante para mejorar el rendimiento del modelo en la detección de biomarcadores, permitiendo entrenarlo en escenarios más variados y representativos. Además, al generar datos adicionales, se reduce la dependencia de conjuntos de datos médicos limitados, fortaleciendo la robustez del modelo en condiciones reales.

4.5.1. Transferencia de Estilo Neuronal

Se utilizó la técnica de NST para combinar el contenido de imágenes de fondo de ojo normal con el estilo de imágenes de la retinopatía diabética. Este proceso permitió generar imágenes sintéticas que preservan las características estructurales de las imágenes base mientras adoptan los estilos definidos por las distintas etapas de la Retinopatía Diabética.

El procedimiento inició con la selección de un modelo preentrenado VGG19, cuyos parámetros se congelaron para preservar las características previamente aprendidas. Se calcularon las características de contenido y estilo en capas específicas, las representaciones del contenido se extrajeron de la capa `conv4_2`, mientras que las características de estilo se obtuvieron de las capas `conv1_1`, `conv2_1`, `conv3_1`, `conv4_1` y `conv5_1`. Estas características se representaron mediante matrices de Gram, que describen las correlaciones entre las activaciones de las características de estilo.

El procedimiento se llevó a cabo con las siguientes ponderaciones asignadas a las capas de estilo:

- conv1_1: 1.0

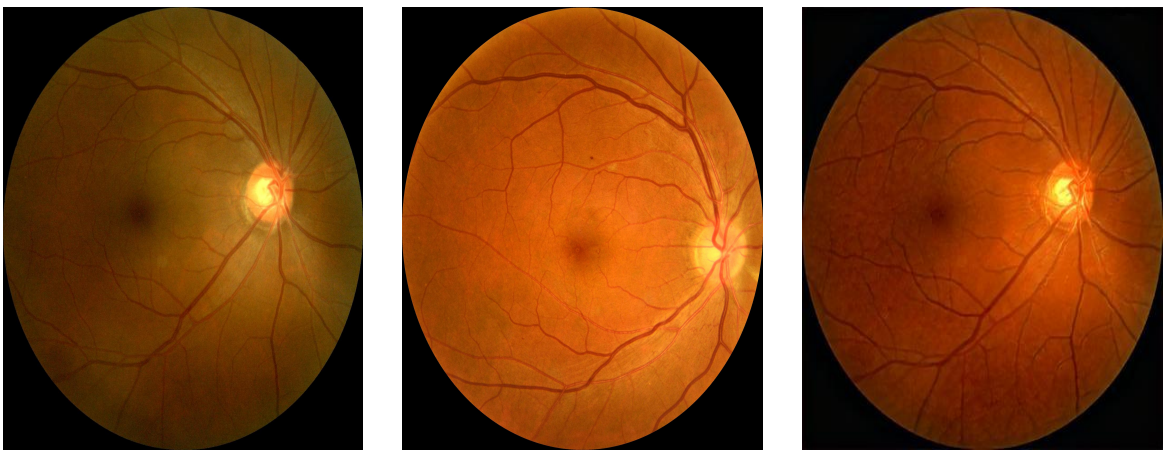
- conv2_1: 0.75
- conv3_1: 0.2
- conv4_1: 0.2
- conv5_1: 0.2

El proceso de optimización comenzó con la imagen objetivo, que fue inicializada como una copia de la imagen de contenido. Se calcularon dos tipos de pérdida: la pérdida de contenido, definida como la diferencia cuadrática media entre las activaciones de la imagen objetivo y las de la imagen de contenido en la capa conv4_2, y la pérdida de estilo, obtenida como la diferencia entre las matrices de Gram de la imagen objetivo y las de la imagen de estilo para cada capa seleccionada, ponderada con un factor de 1×10^9.

La pérdida total se definió como la combinación ponderada de ambas pérdidas, utilizando un factor de ponderación de 1 para la pérdida de contenido y 1×10^9 para la pérdida de estilo. La imagen objetivo fue optimizada durante 5000 iteraciones utilizando el optimizador Adam con una tasa de aprendizaje inicial de 0.003. En cada iteración, el modelo calculó las características de contenido y estilo a partir de las capas seleccionadas del modelo, actualizó las matrices de Gram para las características de estilo, evaluó la pérdida total combinando las pérdidas de contenido y estilo, y finalmente ajustó la imagen objetivo mediante descenso de gradiente.

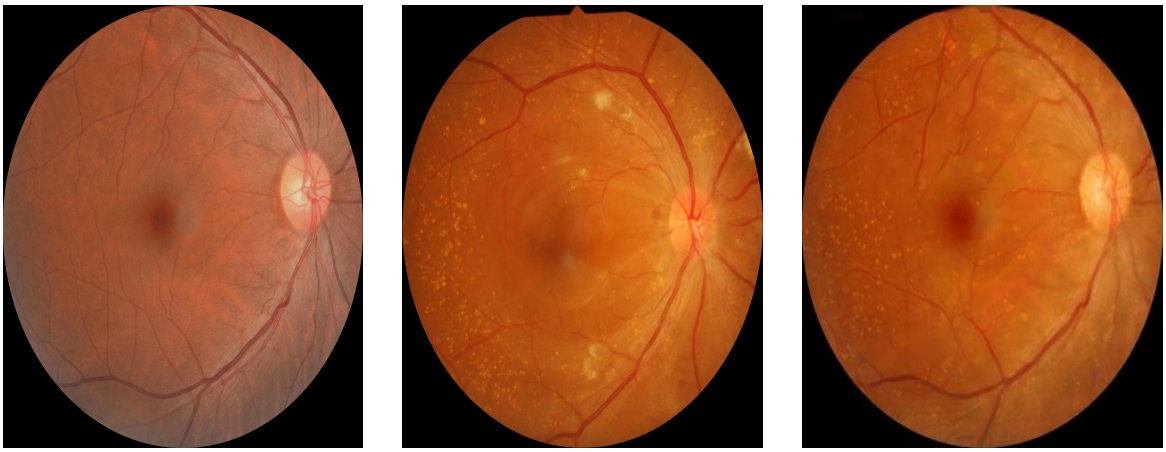
El progreso de la imagen generada fue monitoreado cada 500 iteraciones para evaluar cómo el contenido estructural se combinaba progresivamente con las características estilísticas. En las Figuras 4.29, 4.30, 4.31 y 4.32 se presenta el resultado final para cada etapa de la Retinopatía Diabética, donde se observa cómo la imagen sintetizada logra capturar la esencia de la retinopatía diabética mientras preserva las características estructurales del fondo de ojo normal.

Asimismo, se empleó el modelo SAM para la segmentación de regiones asociadas a biomarcadores como vasos sanguíneos 4.33, disco óptico 4.34, copa óptica 4.35, mácula 4.36, microaneurismas 4.37, hemorragias 4.38 y exudados 4.39.



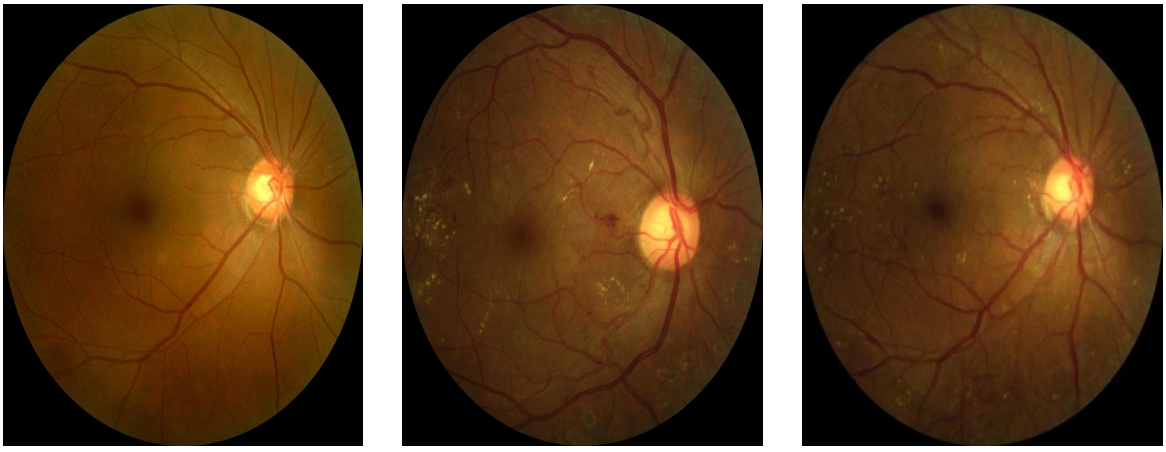
(a) (b) (c)

Figura 4.29: Resultados utilizando Neural Style Transfer: (a) Imagen de Contenido (Fondo de Ojo Normal), (b) Imagen de estilo (retinopatía diabética no proliferativa leve), (c) Imagen sintética creada por NST .



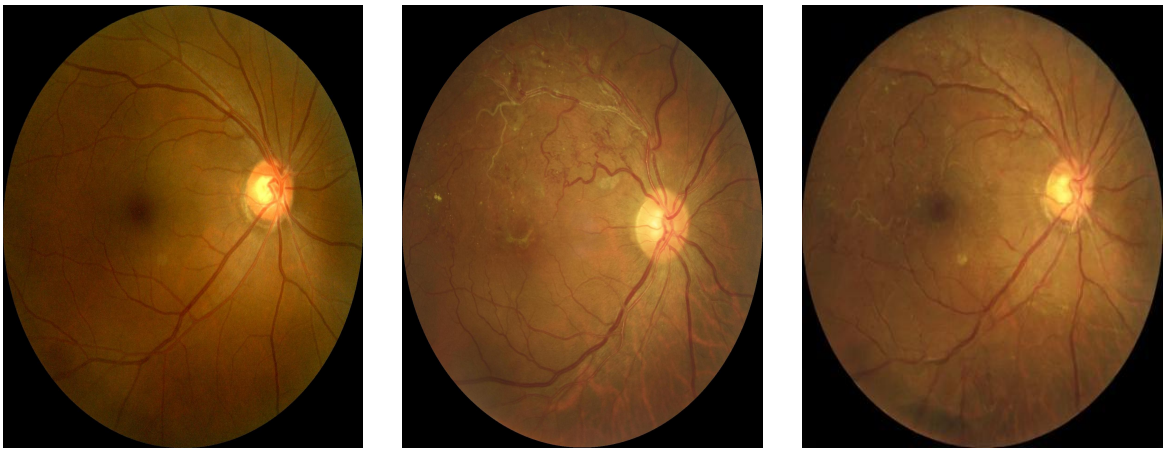
(a) (b) (c)

Figura 4.30: Resultados utilizando Neural Style Transfer: (a) Imagen de Contenido (Fondo de Ojo Normal), (b) Imagen de estilo (retinopatía diabética no proliferativa moderada), (c) Imagen sintética creada por NST .



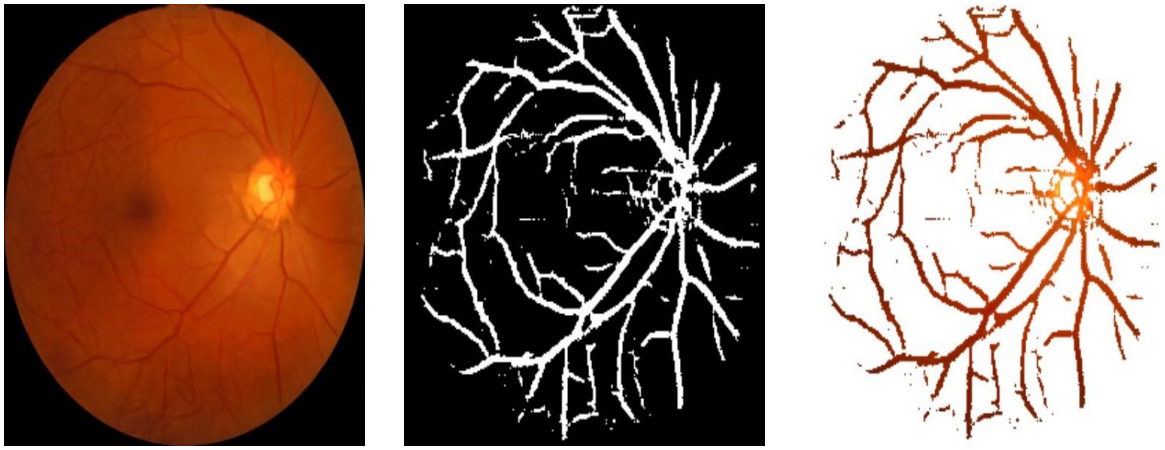
(a) (b) (c)

Figura 4.31: Resultados utilizando Neural Style Transfer: (a) Imagen de Contenido (Fondo de Ojo Normal), (b) Imagen de estilo (retinopatía diabética no proliferativa severa), (c) Imagen sintética creada por NST .



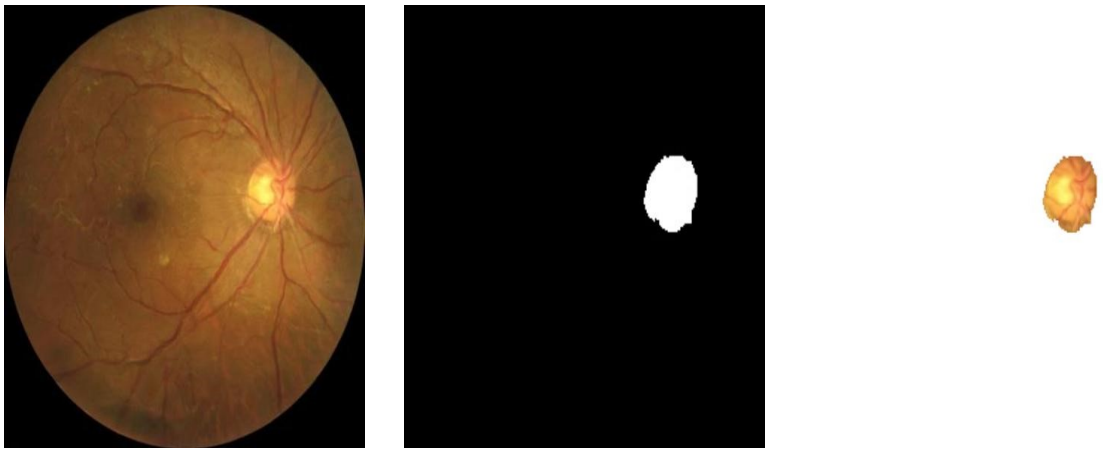
(a) (b) (c)

Figura 4.32: Resultados utilizando Neural Style Transfer: (a) Imagen de Contenido (Fondo de Ojo Normal), (b) Imagen de estilo (retinopatía diabética proliferativa), (c) Imagen sintética creada por NST .



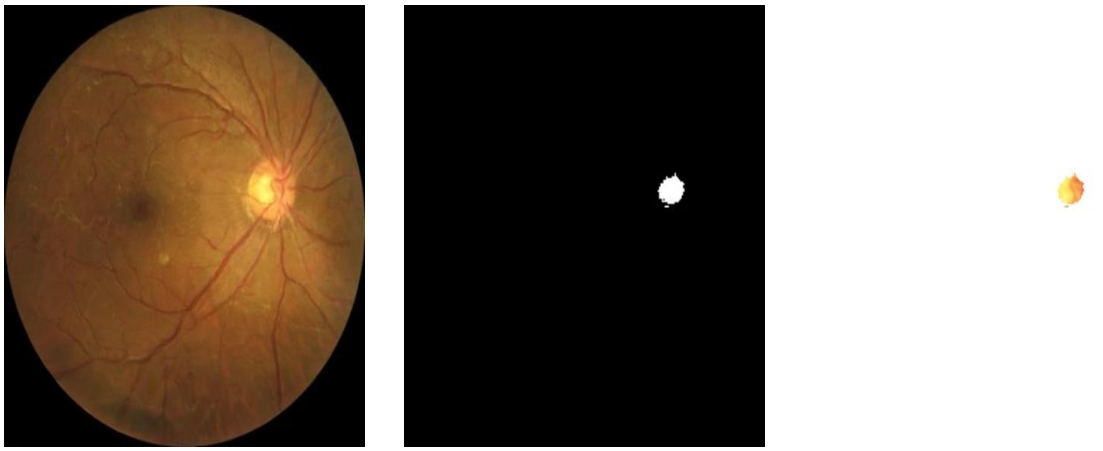
(a) (b) (c)

Figura 4.33: Resultados de segmentación en imágenes NST: (a) Imagen sintética creada por NST, (b) Máscara binaria de vasos sanguíneos segmentados, (c) Píxeles extraídos según la región de interés.



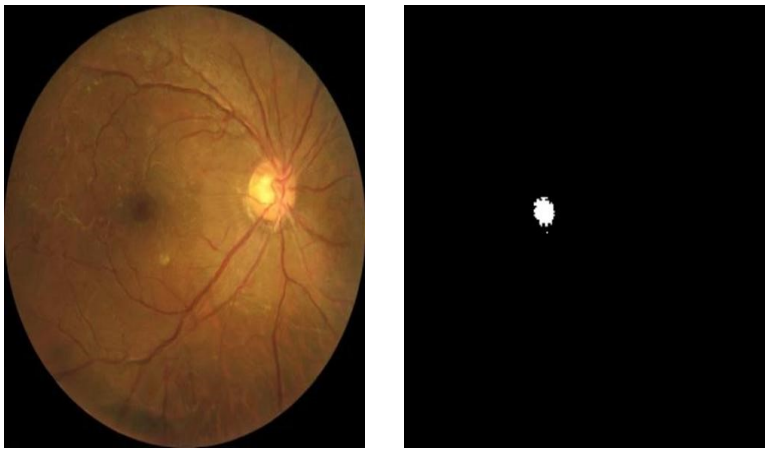
(a) (b) (c)

Figura 4.34: Resultados de segmentación en imágenes NST: (a) Imagen sintética creada por NST, (b) Máscara binaria del disco óptico segmentado, (c) Píxeles extraídos según la región de interés.



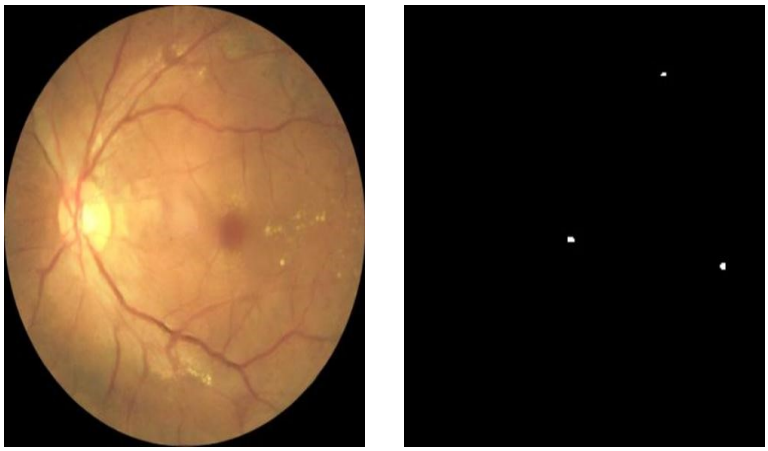
(a) (b) (c)

Figura 4.35: Resultados de segmentación en imágenes NST: (a) Imagen sintética creada por NST, (b) Máscara binaria de la copa óptica segmentada, (c) Píxeles extraídos según la región de interés.



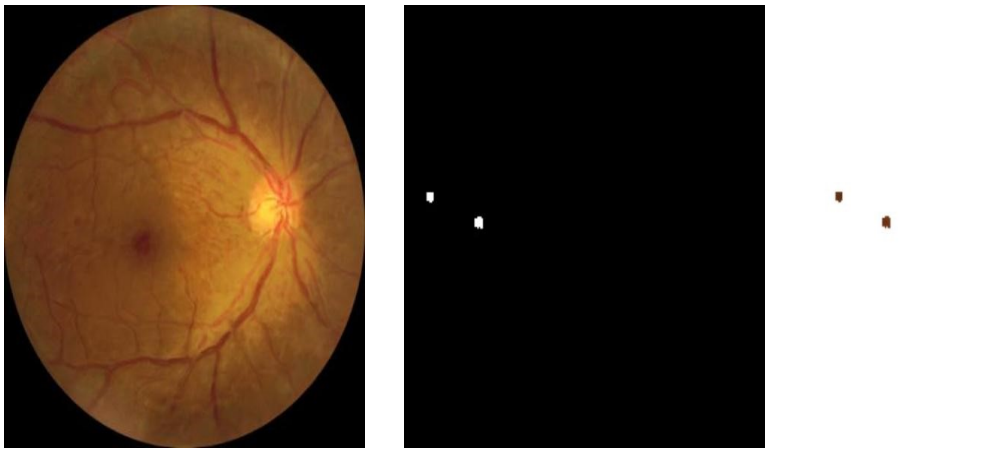
(a) (b) (c)

Figura 4.36: Resultados de segmentación en imágenes NST: (a) Imagen sintética creada por NST, (b) Máscara binaria de la mácula segmentada, (c) Píxeles extraídos según la región de interés.



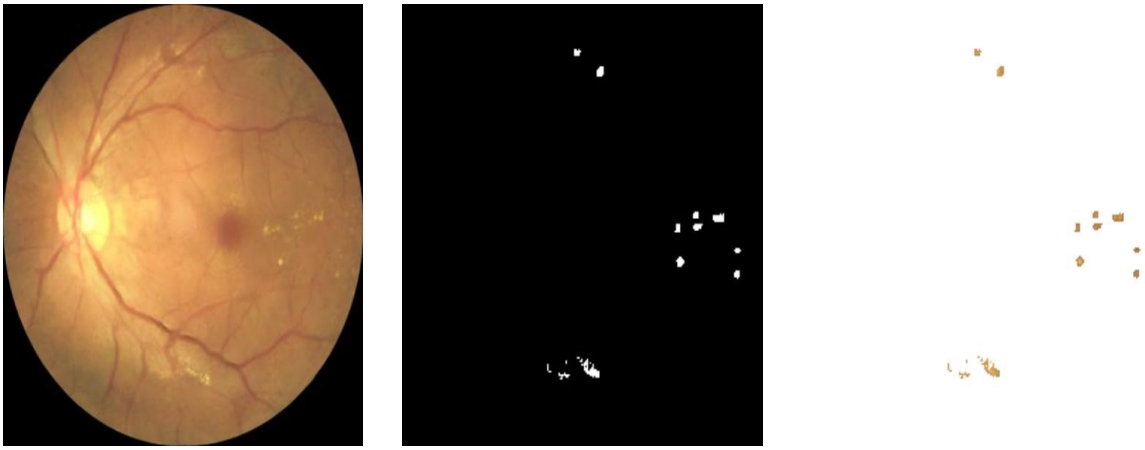
(a) (b) (c)

Figura 4.37: Resultados de segmentación en imágenes NST: (a) Imagen sintética creada por NST, (b) Máscara binaria de microaneurismas segmentadas, (c) Píxeles extraídos según la región de interés.



(a) (b) (c)

Figura 4.38: Resultados de segmentación en imágenes NST: (a) Imagen sintética creada por NST, (b) Máscara binaria de hemorragias segmentadas, (c) Píxeles extraídos según la región de interés.



(a) (b) (c)

Figura 4.39: Resultados de segmentación en imágenes NST: (a) Imagen sintética creada por NST, (b) Máscara binaria de exudados segmentados, (c) Píxeles extraídos según la región de interés.

4.5.2. CycleGAN

Se implementó CycleGAN para la conversión de imágenes entre dos dominios: fondo de ojo normal e imágenes de retinopatía diabética. Se generaron cuatro modelos, uno para cada etapa de la retinopatía diabética. Para cada modelo, se utilizaron 2,200 imágenes correspondientes a la etapa de la retinopatía diabética y 1,500 imágenes de fondo de ojo normal provenientes de los conjuntos de datos APTOS, ODIR-5K y EyePACS.

La arquitectura del modelo se compuso de dos generadores y dos discriminadores. Los generadores transforman imágenes entre dominios utilizando una serie de capas de muestreo

ascendente y descendente, como se puede ver en la Figura 4.40.

Las capas de muestreo ascendente y descendente reducían las dimensiones espaciales al tiempo que aumentaban la profundidad de los mapas de características.
Cada capa de muestreo descendente incluía:

- Una capa convolucional con un tamaño de filtro de 4×4, inicializada utilizando una distribución normal aleatoria.
- Normalización de instancias para estabilizar el entrenamiento y mantener la coherencia entre dominios.
- Una función de activación LeakyReLU para capturar características de la imagen al tiempo que evita la desaparición de gradientes.

Las capas de muestreo ascendente seguieron una estructura complementaria:

- Una capa convolucional transpuesta con un tamaño de filtro de 4×4.
- Normalización de instancias para refinar las características de la imagen reconstruida.
- Una función de activación ReLU para mejorar la nitidez y claridad de la imagen.
- Se incluyeron capas eliminadas en etapas seleccionadas para evitar el sobreajuste durante el entrenamiento.

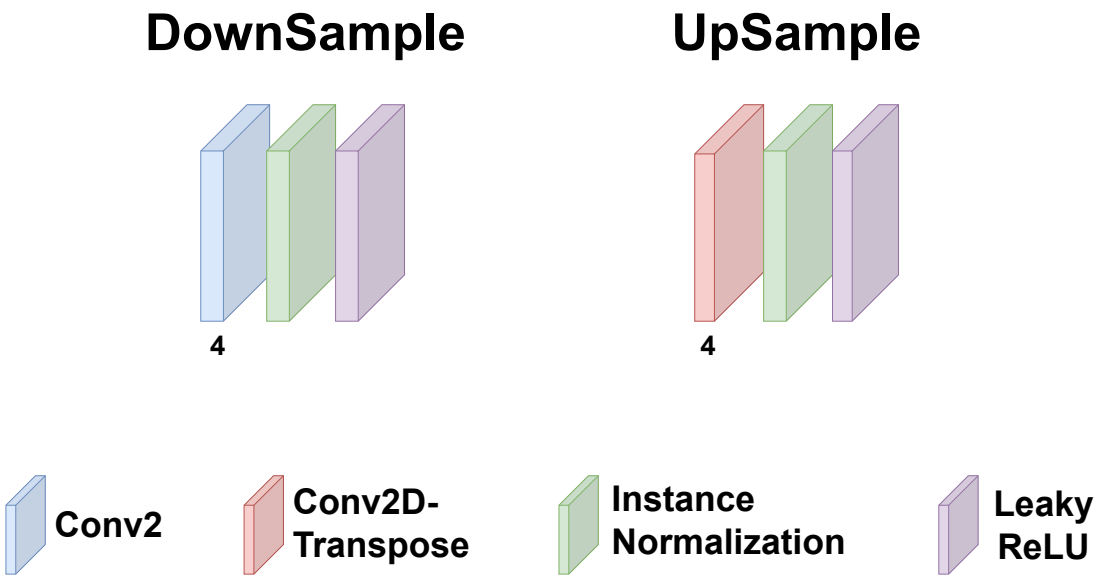


Figura 4.40: Estructura de las capas utilizadas en las operaciones de DownSample y UpSample dentro del modelo.

El modelo del generador utilizó ocho capas de muestreo ascendente y descendente, con conexiones de omisión para mejorar la reutilización de características y preservar los detalles espaciales, como se muestra en la Figura 4.41.

La ruta de muestreo descendente utilizó tamaños de filtro (64, 128, 256 y 512), cada uno con un tamaño de kernel de 4×4. La primera capa omitió la normalización de instancias para capturar las características iniciales, mientras que las capas posteriores aplicaron la normalización de instancias y la activación de LeakyReLU para estabilizar el entrenamiento y extraer representaciones jerárquicas.

La ruta de muestreo ascendente reflejó esta estructura, reduciendo progresivamente los tamaños de filtro de 512 a 64. Se aplicó la eliminación en las dos primeras capas para mitigar el sobreajuste, y la capa final empleó una convolución transpuesta con tres canales de salida, un tamaño de kernel de 4×4 y una función de activación tanh para generar la salida de imagen normalizada.

Las conexiones de salto vincularon directamente cada capa de muestreo descendente con su capa de muestreo ascendente correspondiente, preservando las características espaciales críticas. Para solucionar las dimensiones no coincidentes entre estas capas, se aplicó la interpolación del vecino más cercano, lo que garantizó una integración fluida.

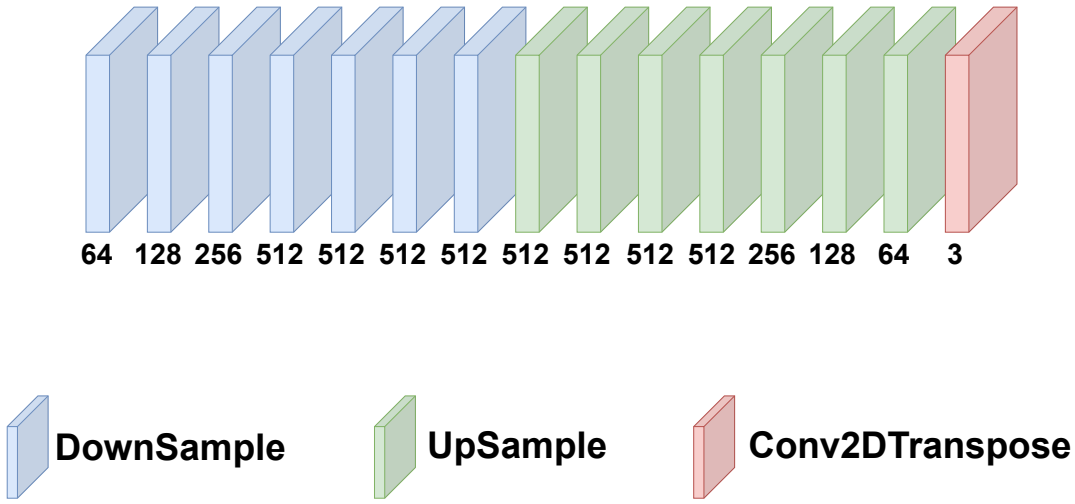


Figura 4.41: Estructura del modelo generador compuesto por capas de DownSample y UpSample.

El modelo discriminador fue diseñado para diferenciar entre imágenes reales y sintéticas, empleando una secuencia de capas de submuestreo para extraer progresivamente características y reducir las dimensiones espaciales como se muestra en la Figura 4.42. La arquitectura incluyó los siguientes componentes:

- **Capas de submuestreo:** Se utilizaron tamaños de filtro de 64, 128, 256 y 512, cada uno con un tamaño de kernel de 4×4. Se aplicó la normalización de instancias para estabilizar el entrenamiento, excepto en la primera capa, que se centró en capturar características iniciales. Se utilizó la activación de LeakyReLU en todo momento para mantener el flujo de gradiente.

- **Capas convolucionales finales:** Después de la ruta de submuestreo, se aplicó una capa de relleno de ceros para preservar las dimensiones espaciales, seguida de una capa con-

volucional con 512 filtros y un tamaño de kernel de 3×3, empleando la normalización de instancias y la activación de LeakyReLU.

- **Capa de salida:** Se utilizó una capa convolucional final con un solo canal de salida para producir una puntuación para cada parche de entrada, indicando si es real o sintético.

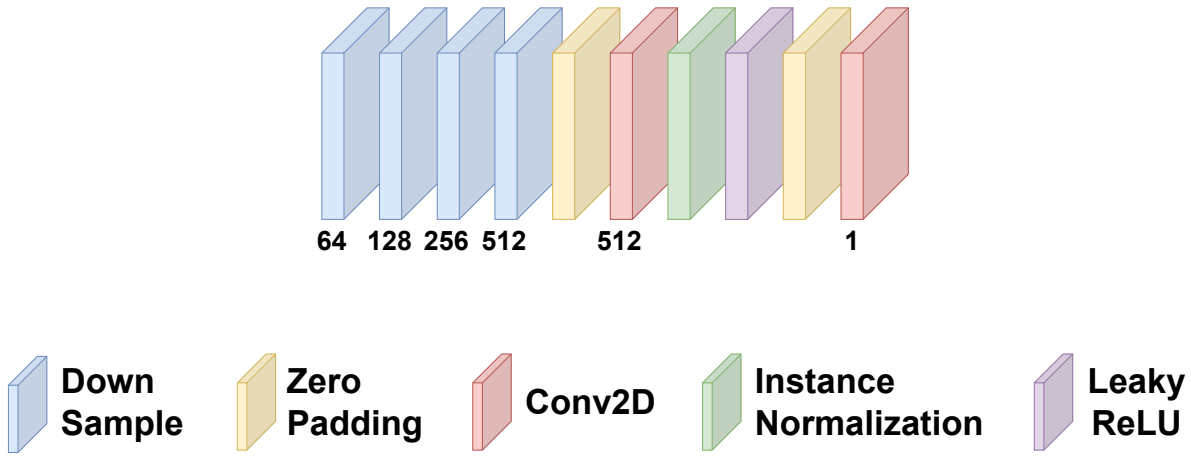
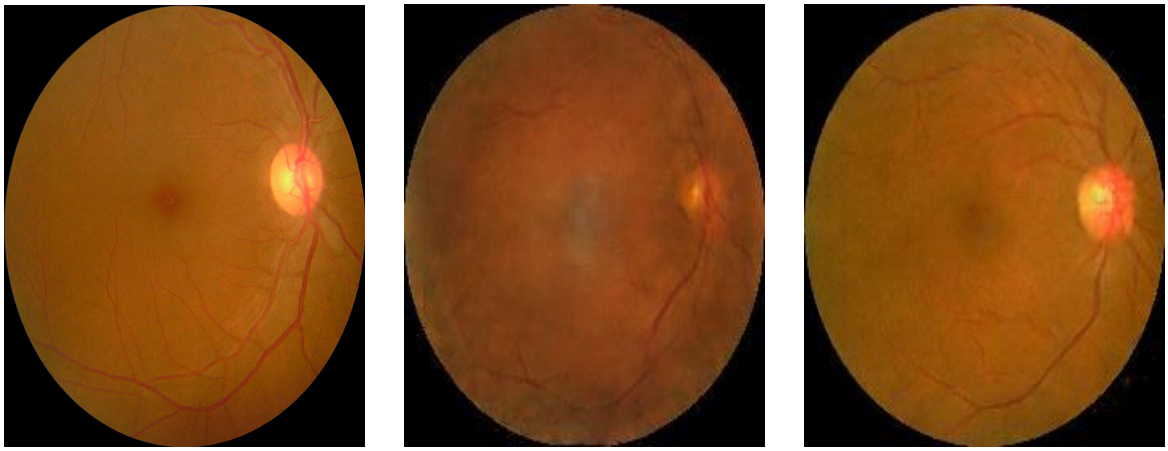


Figura 4.42: Arquitectura del discriminador de CycleGAN.

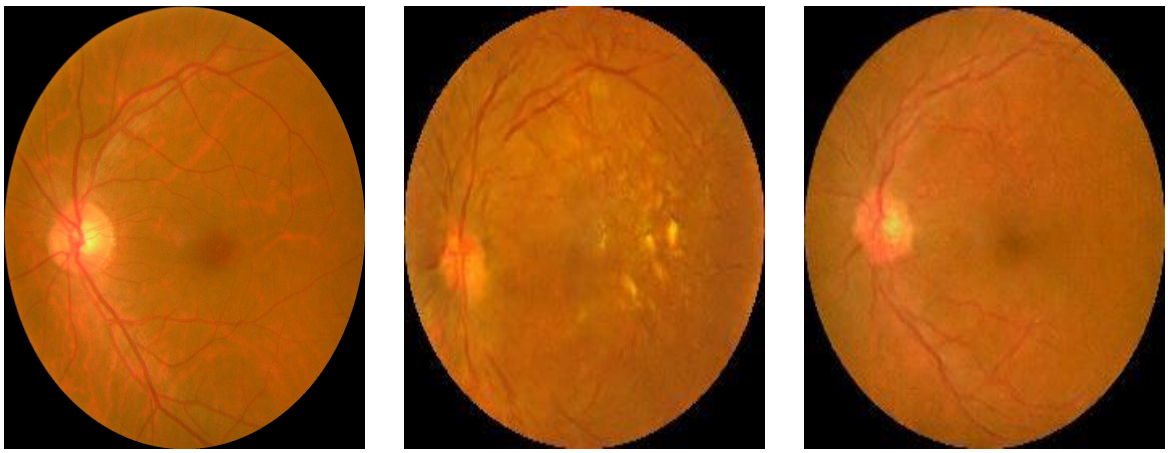
El entrenamiento del modelo utilizó funciones de pérdida para el generador y el discriminador, aplicando una combinación de pérdida adversarial, pérdida de consistencia de ciclo y pérdida de identidad. La pérdida total del generador incorporó la consistencia de ciclo y la identidad ponderadas por un factor $\lambda = 10$. La optimización se realizó con Adam, utilizando una tasa de aprendizaje de 0.0002 para ambos generadores y discriminadores. El modelo fue entrenado durante 120 épocas con un tamaño de lote de 16 imágenes. En las Figuras 4.43, 4.44, 4.45 y 4.46 se presentan los resultados finales, mostrando cómo las imágenes sintetizadas capturan la esencia de la retinopatía diabética, al tiempo que preservan las características estructurales del fondo de ojo normal.

Adicionalmente, se empleó el modelo SAM para realizar segmentaciones precisas en las imágenes generadas, enfocándose en biomarcadores como vasos sanguíneos (Figura 4.47), disco óptico (Figura 4.48), copa óptica (Figura 4.49), mácula (Figura 4.50), microaneurismas (Figura 4.5.2), hemorragias (Figura 4.52) y exudados (Figura 4.53). Estas segmentaciones destacan la utilidad de las imágenes generadas por CycleGAN para aplicaciones clínicas y de investigación.



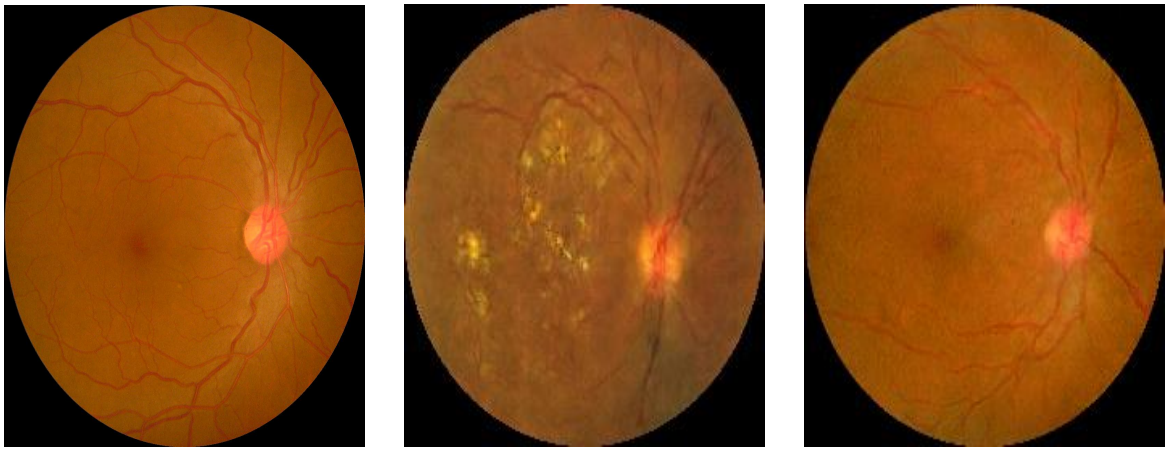
(a) (b) (c)

Figura 4.43: Resultados utilizando CycleGAN: (a) Imagen de fondo de ojo normal, (b) Imagen generada a partir del modelo de retinopatía diabética no proliferativa leve, (c) Aplicación de consistencia de ciclo.



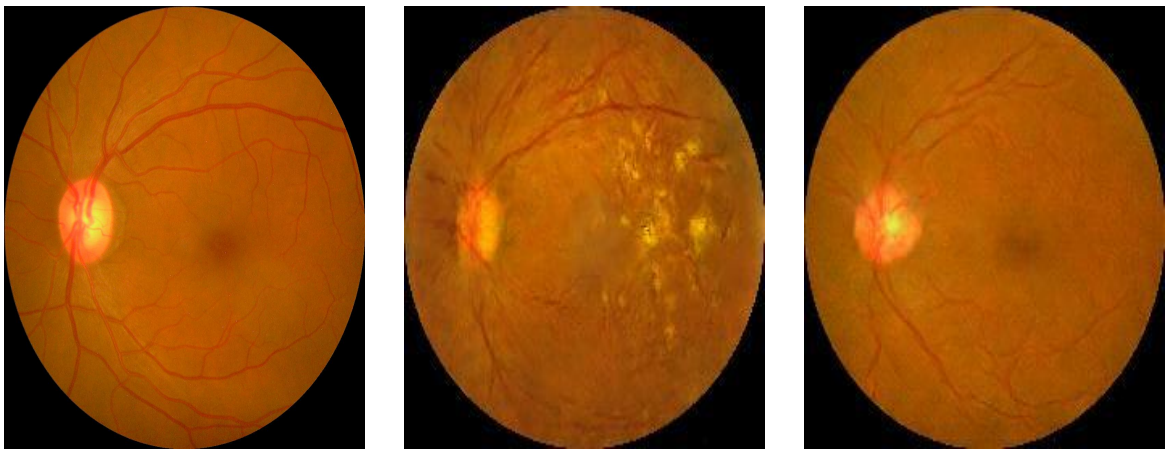
(a) (b) (c)

Figura 4.44: Resultados utilizando CycleGAN: (a) Imagen de fondo de ojo normal, (b) Imagen generada a partir del modelo de retinopatía diabética no proliferativa moderada, (c) Aplicación de consistencia de ciclo.



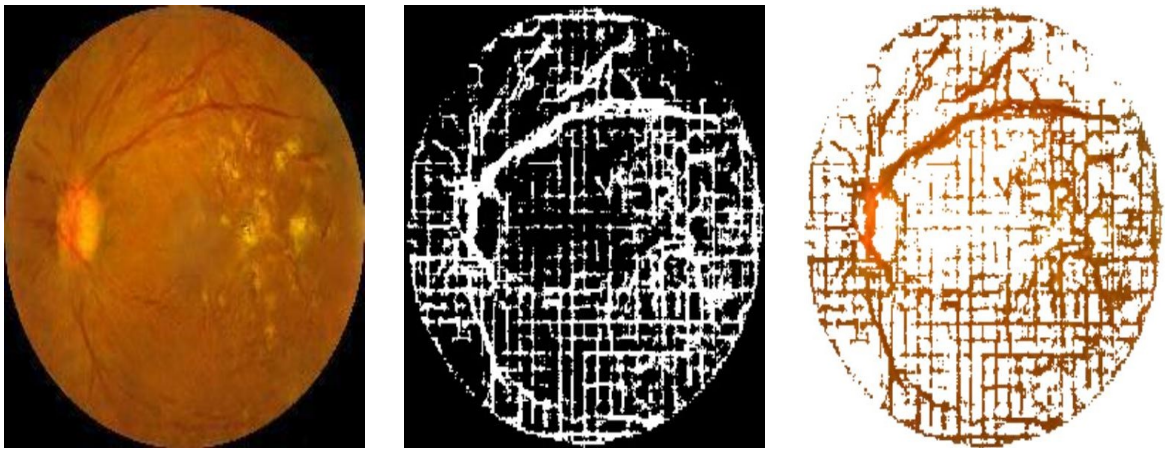
(a) (b) (c)

Figura 4.45: Resultados utilizando CycleGAN: (a) Imagen de fondo de ojo normal, (b) Imagen generada a partir del modelo de retinopatía diabética no proliferativa severa, (c) Aplicación de consistencia de ciclo.



(a) (b) (c)

Figura 4.46: Resultados utilizando CycleGAN: (a) Imagen de fondo de ojo normal, (b) Imagen generada a partir del modelo de retinopatía diabética proliferativa , (c) Aplicación de consistencia de ciclo.

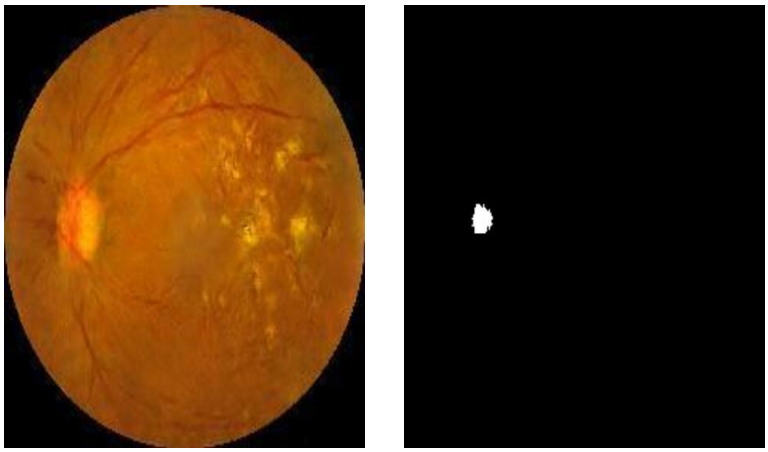


(a) (b) (b)

Figura 4.47: Resultados de segmentación en imágenes CycleGan: (a) Imagen sintética creada por CycleGan, (b) Máscara binaria de vasos sanguíneos segmentados, (c) Píxeles extraídos según la región de interés. La cuadrícula visible en los resultados se debe principalmente a la inconsistencia en la generación de la imagen por parte del modelo generador, lo que provocó variaciones no deseadas en la textura.

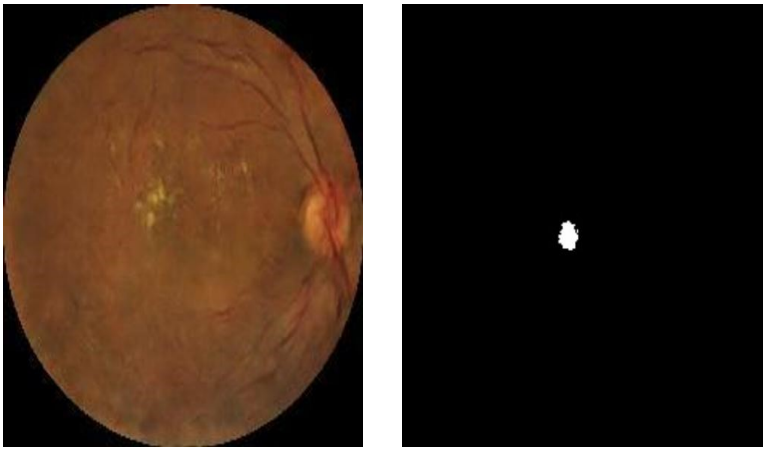
(a) (b) (b)

Figura 4.48: Resultados de segmentación en imágenes CycleGan: (a) Imagen sintética creada por CycleGan, (b) Máscara binaria del disco óptico segmentado, (c) Píxeles extraídos según la región de interés.



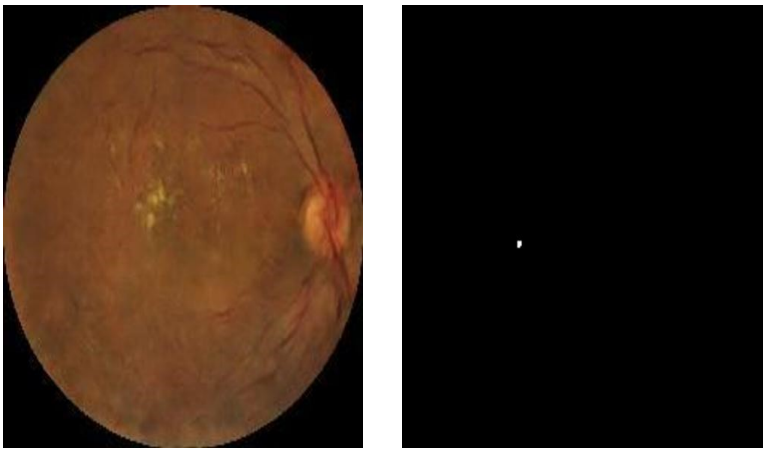
(a) (b) (b)

Figura 4.49: Resultados de segmentación en imágenes CycleGan: (a) Imagen sintética creada por CycleGan, (b) Máscara binaria de la copa óptica segmentada, (c) Píxeles extraídos según la región de interés.



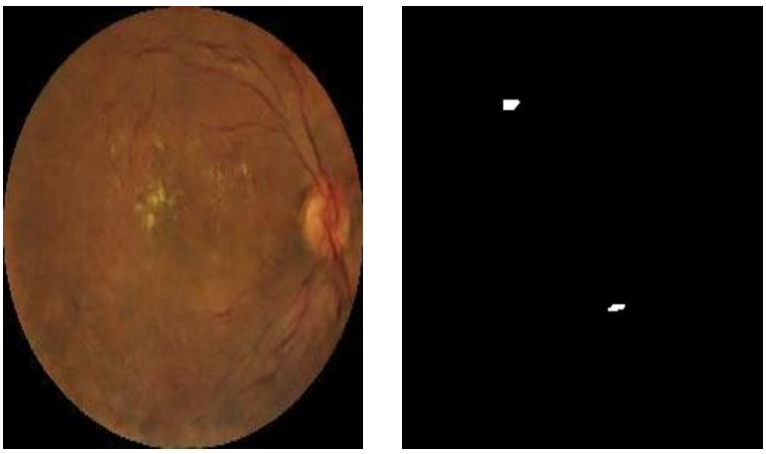
(a) (b) (b)

Figura 4.50: Resultados de segmentación en imágenes CycleGan: (a) Imagen sintética creada por CycleGan, (b) Máscara binaria de la mácula segmentada, (c) Píxeles extraídos según la región de interés.



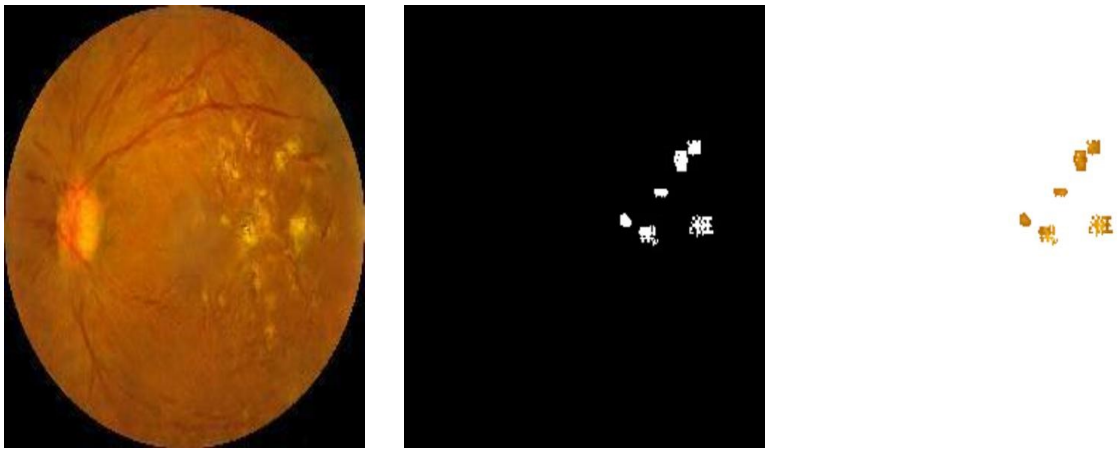
(a) (b) (b)

Figura 4.51: Resultados de segmentación en imágenes CycleGan: (a) Imagen sintética creada por CycleGan, (b) Máscara binaria de microaneurismas segmentadas, (c) Píxeles extraídos según la región de interés.



(a) (b) (b)

Figura 4.52: Resultados de segmentación en imágenes usando CycleGan: (a) Imagen sintética creada por CycleGan, (b) Máscara binaria de hemorragias segmentadas, (c) Píxeles extraídos según la región de interés.



(a) (b) (b)

Figura 4.53: Resultados de segmentación en imágenes CycleGan: (a) Imagen sintética creada por CycleGan, (b) Máscara binaria de exudados segmentados, (c) Píxeles extraídos según la región de interés.

CAPÍTULO 5

RESULTADOS

En esta sección se presentan los resultados obtenidos a partir de las pruebas realizadas con los modelos desarrollados para la clasificación multiclase, detección de objetos y segmentación, empleando CNN. Estas pruebas incluyeron datasets adicionales no utilizados en la metodología, con el objetivo de evaluar la robustez. Asimismo, se implementaron estos resultados en una aplicación web como una demostración práctica de su utilidad en un entorno clínico.

5.1. Flujo de Trabajo

Se empleó el dataset E-OPTHA (40 imágenes), específicamente diseñado para la detección y clasificación de exudados duros. En el caso de la segmentación y detección de objetos, los modelos YOLOv8 y SAM fueron evaluados con el mismo dataset, asegurando una comparación consistente en el análisis de biomarcadores y su desempeño en distintas tareas.

Se utilizó el dataset REFUGE para la segmentación del disco óptico y el dataset RITE para evaluar la segmentación de vasos sanguíneos. Estas pruebas tuvieron como objetivo validar la capacidad de los modelos en tareas específicas relacionadas con biomarcadores importantes en la retinopatía diabética.

Por último, se incorporó el dataset de ODIR-5k para la clasificación de etapas de la retinopatía diabética. Para cada modelo y enfoque, se siguió el mismo preprocesamiento y metodología descritos en las secciones anteriores, garantizando consistencia en las configuraciones y pruebas realizadas.

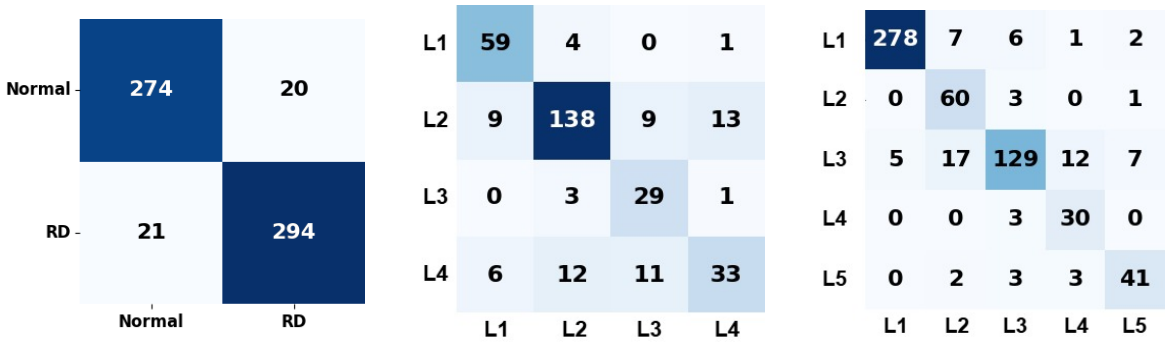
5.2. Modelo de Predicción de Retinopatía Diabética

En esta sección se presentan los resultados obtenidos a partir de los modelos de clasificación multiclase aplicados a la detección de las etapas de la retinopatía diabética, utilizando CNN. En la Tabla 5.1 se resumen los resultados de las clasificaciones de dos clases, cuatro clases y cinco clases, junto con las métricas clave como precisión, recall y F1-score para ambos modelos.

Además, las matrices de confusión, mostradas en la Figura 5.1, proporcionan un análisis visual detallado del rendimiento de los modelos.

Tabla 5.1: Resultados en Clasificación Multi-Clase Usando CNN.

Modelo	Accuracy	precisión	Recall	F1-Score
2 Clases	0.9300	0.9300	0.9300	0.9300
4 Clases	0.7900	0.7390	0.7870	0.7520
5 Clases	0.8820	0.8060	0.8780	0.8330



(a) (b) (c)

Figura 5.1: Matrices de confusión para la clasificación de 2,4 y 5 clases: (a) Matriz de confusión del modelo CNN de 2 clases, (b) Matriz de confusión del modelo CNN de 4 clases, y (c) Matriz de confusión del modelo CNN de 5 clases.

5.3. Detección de Objetos

Los resultados de detección de biomarcadores mediante el modelo YOLOv8 se exponen en esta sección. En la Figura 5.2 se presentan ejemplos de detección, los cuales evidencian un desempeño sólido del modelo en la identificación de biomarcadores clave como el disco óptico y los exudados duros. Aunque los resultados son precisos en muchos casos, aún presentan margen de mejora en la detección de biomarcadores, lo que indica un potencial para mejorar el modelo en futuros proyectos.

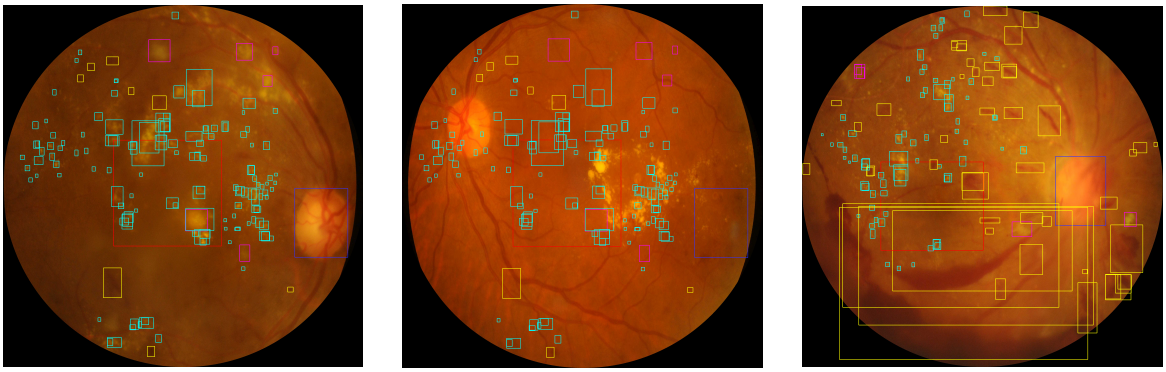


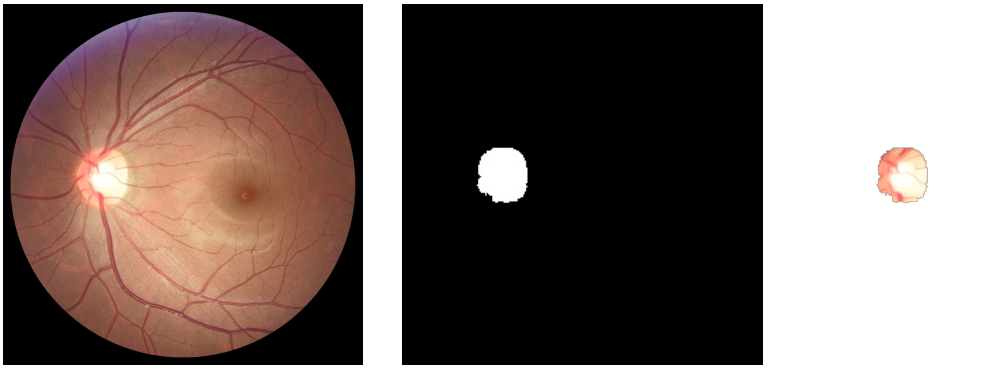
Figura 5.2: Ejemplos de detección de biomarcadores de la retinopatía diabética mediante el modelo YOLOv8. Las imágenes muestran cuadros delimitadores que identifican distintas regiones de interés: el disco óptico en azul, la fóvea en rojo, los exudados duros en cian, los exudados suaves en magenta y las hemorragias en amarillo.

5.4. Segmentación

En esta sección se describen los resultados obtenidos con el modelo SAM en tareas de segmentación. En la Tabla 5.2 se presentan las métricas obtenidas, como dice, AUC-PR y AUC-ROC y ademas en la Figura 5.3 5.4 5.5 se muestra ejemplares del resultado en el uso del modelo SAM en cada biomarcador.

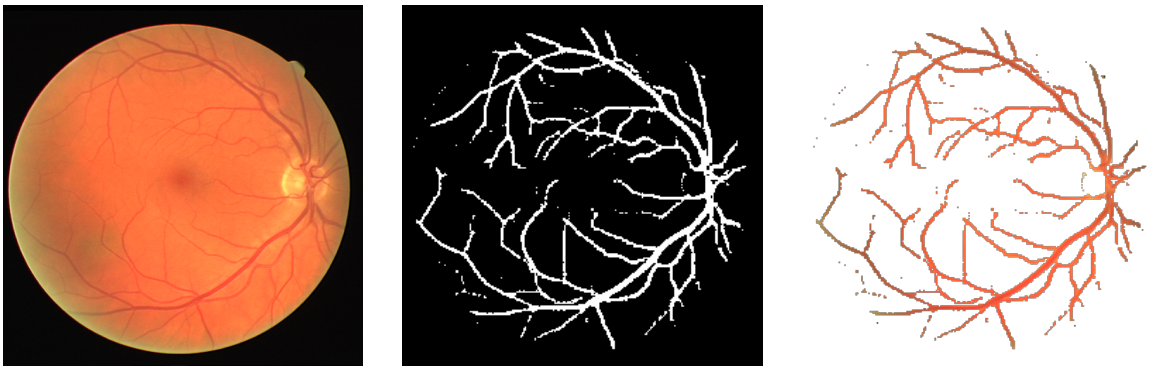
Tabla 5.2: Resultados en la Segmentación de Biomarcadores.

Modelo	Dice	AUC-PR	AUC-ROC
Disco Óptico	0.7925	0.8084	0.8716
Vasos Sanguíneos	0.6857	0.7825	0.7825
Exudados Duros	0.3576	0.4379	0.67646



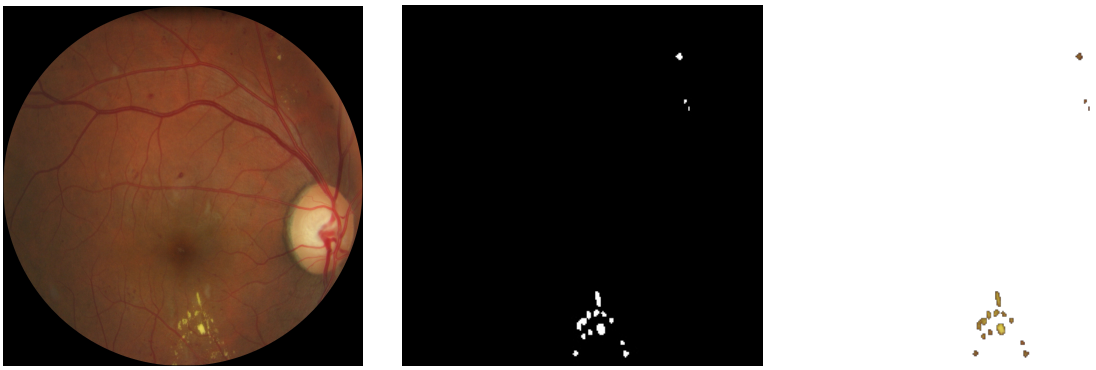
(a) (b) (c)

Figura 5.3: Resultados de segmentación del disco óptico en imágenes del fondo de ojo: (a) Imagen original, (b) Máscara binaria del disco óptico segmentada, (c) Píxeles extraídos según la región de interés.



(a) (b) (c)

Figura 5.4: Resultados de segmentación de los vasos sanguíneos en imágenes del fondo de ojo: (a) Imagen original, (b) Máscara binaria de los vasos sanguíneos segmentados, (c) Píxeles extraídos según la región de interés.



(a) (b) (c)

Figura 5.5: Resultados de segmentación de los exudados duros en imágenes del fondo de ojo: (a) Imagen original, (b) Máscara binaria de los exudados duros segmentados, (c) Píxeles extraídos según la región de interés.

5.5. Implementación de los Modelos en un Software

Para facilitar la interacción con los modelos desarrollados en este estudio, se implementó una aplicación web que permite a los usuarios cargar imágenes y recibir resultados de predicción en tiempo real. Esta plataforma fue desarrollada con una arquitectura frontend-backend, donde el frontend fue construido con React.js para la interfaz de usuario y Node.js para la gestión de la comunicación, mientras que el backend fue implementado en Python con Django, encargado del procesamiento de imágenes y la ejecución de los modelos de aprendizaje profundo.

A continuación, se presentan los pasos de interacción con la aplicación, ilustrados con capturas de pantalla del sistema en funcionamiento.

5.5.1. Interfaz de la Aplicación

La Figura 5.6 muestra la interfaz inicial de la aplicación web. En esta pantalla, el usuario puede cargar una imagen desde su dispositivo para ser analizada por el modelo. La interfaz ha sido diseñada para ser intuitiva y accesible, facilitando la selección de imágenes con un botón central.

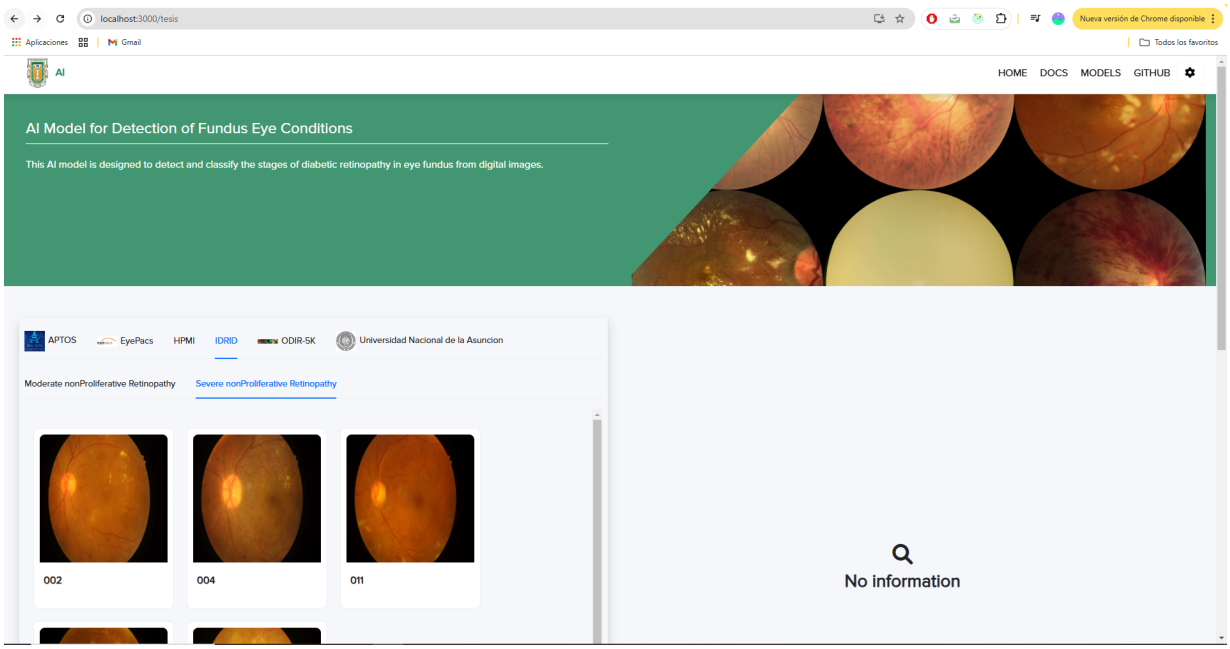


Figura 5.6: Interfaz principal de la aplicación web.

5.5.2. Procesamiento de la Imagen y Espera de Resultados

Tras confirmar la imagen, la aplicación envía la solicitud al servidor backend para que el modelo procese la imagen. Durante este tiempo, se muestra un indicador de carga para informar al usuario que el análisis está en progreso. La Figura 5.7 muestra esta etapa.

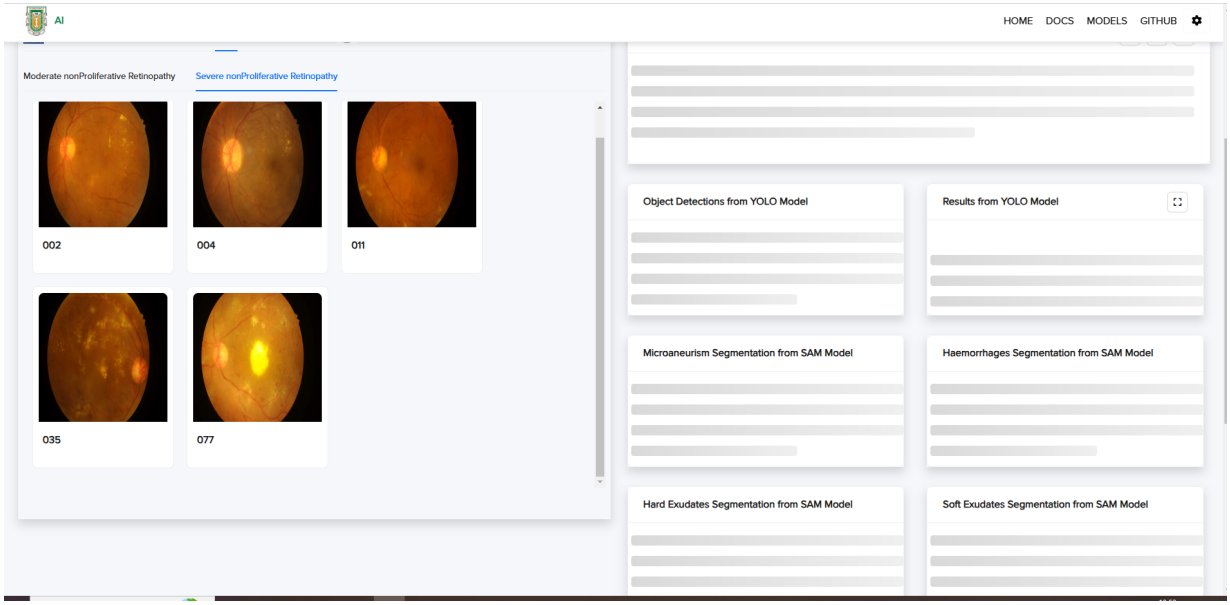


Figura 5.7: Indicador de procesamiento en la aplicación mientras el modelo analiza la imagen.

5.5.3. Resultados de Clasificación de Etapas de Retinopatía Diabética

La primera funcionalidad de la aplicación es la clasificación de imágenes de fondo de ojo en una de las cuatro etapas de la retinopatía diabética. En la Figura 5.8, se muestra un ejemplo donde el modelo ha clasificado la imagen como etapa severa. Además del resultado de la predicción, la interfaz presenta una gráfica de barras que representa la distribución de probabilidades para cada clase, proporcionando una mejor interpretación de la confianza del modelo.

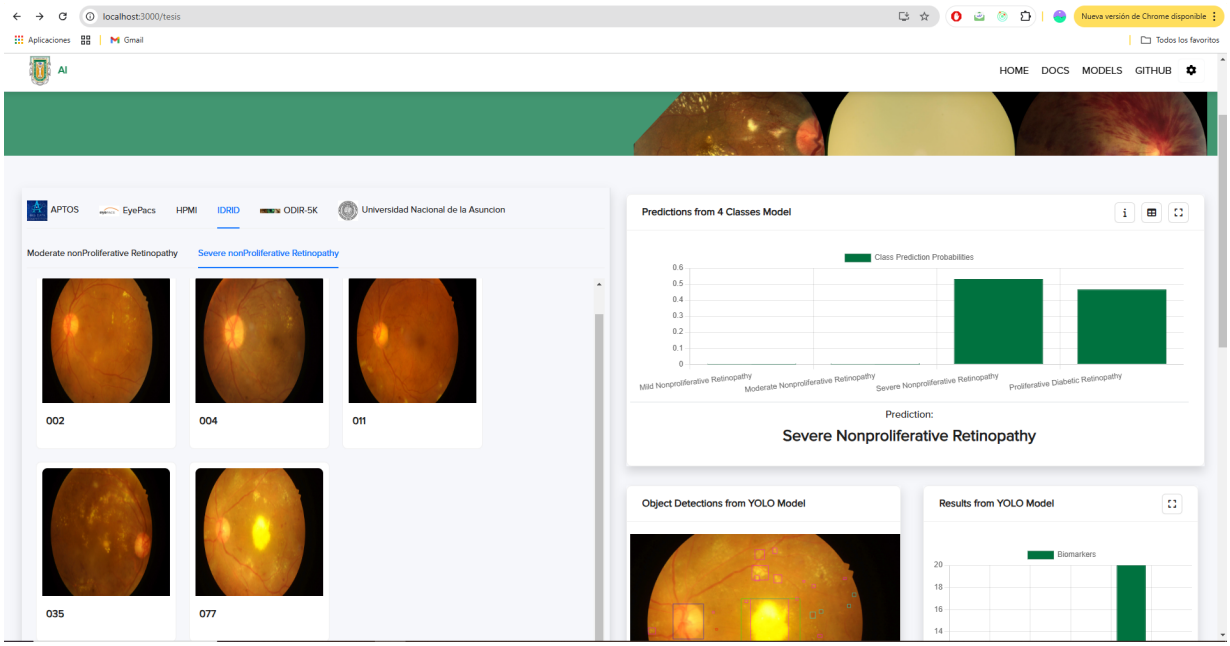


Figura 5.8: Resultados de clasificación de las etapas de retinopatía diabética.

5.5.4. Resultados de Detección de Biomarcadores con YOLO

En la segunda funcionalidad, se implementó un modelo YOLO para la detección de biomarcadores en las imágenes de fondo de ojo. Como se observa en la Figura 5.9, el modelo identifica y delimita los biomarcadores con cuadros delimitadores, permitiendo visualizar con precisión las áreas donde se encuentran.

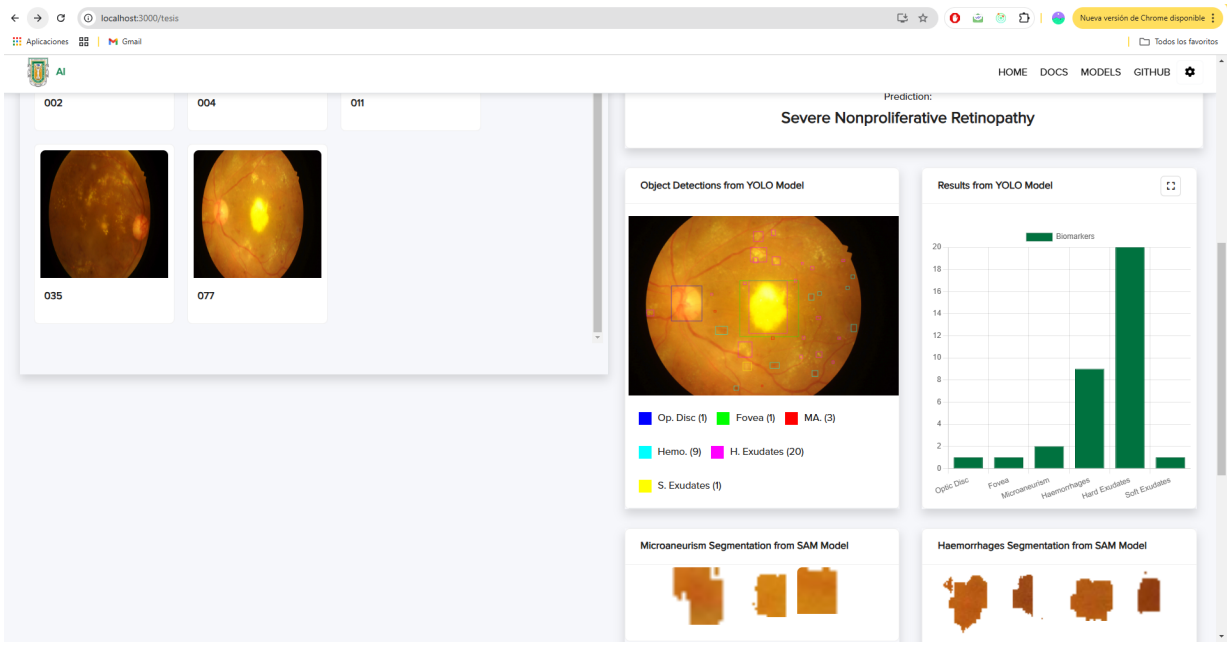


Figura 5.9: Detección de biomarcadores utilizando el modelo YOLO. Se muestran las regiones donde el modelo ha identificado las características relevantes en la imagen.

5.5.5. Visualización Ampliada de los Resultados de Detección

Para mejorar la interpretación de los resultados obtenidos con el modelo de detección, la aplicación permite al usuario hacer clic en la imagen para ampliar la región detectada y analizarla con mayor detalle. La Figura 5.10 ilustra esta funcionalidad, en la que se muestra una imagen ampliada resaltando las regiones detectadas.

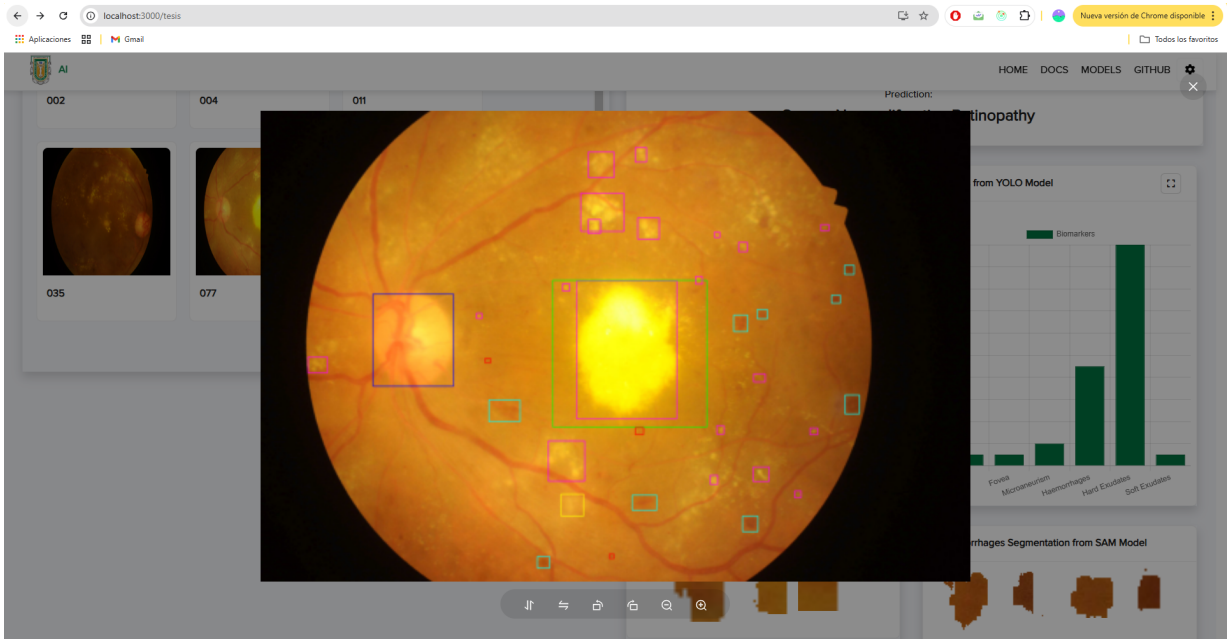


Figura 5.10: Funcionalidad de zoom en la detección de biomarcadores con YOLO. Al hacer clic en la imagen, se amplía la región detectada para un análisis más detallado.

5.5.6. Resultados de Segmentación de Biomarcadores con el Modelo SAM

Finalmente, se implementó la segmentación de biomarcadores mediante el modelo SAM. A diferencia del modelo YOLO, que proporciona únicamente cajas delimitadoras, SAM genera una segmentación más precisa de las estructuras detectadas. En la Figura 5.11, se observa el resultado del modelo, donde los biomarcadores han sido segmentados y resaltados en la imagen.

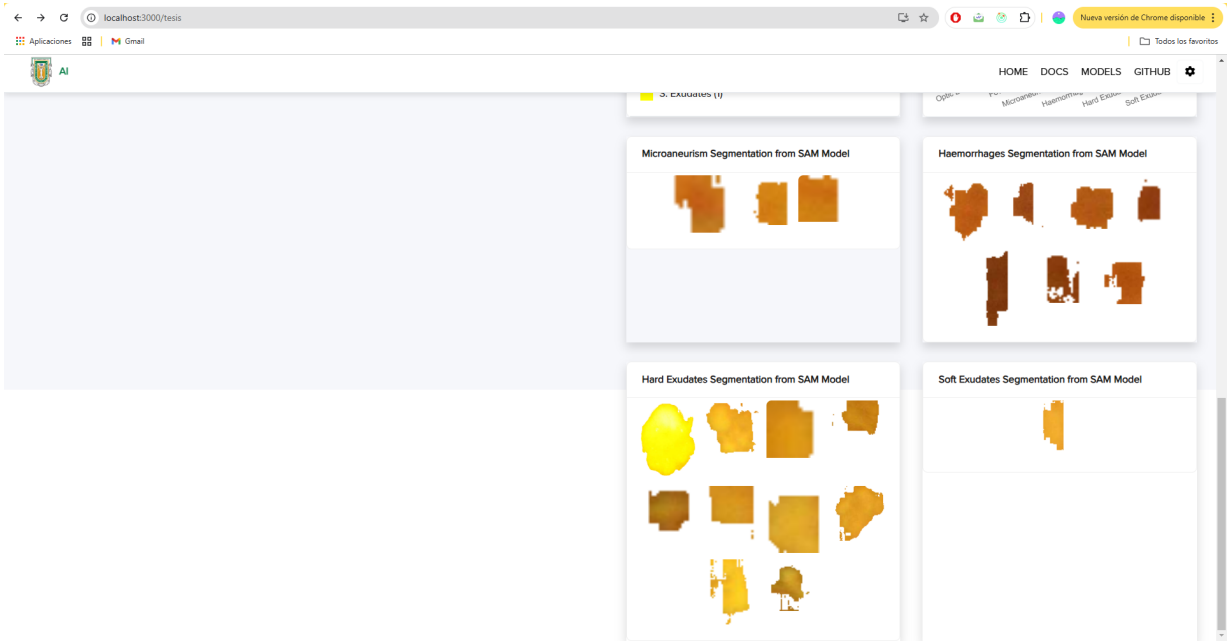


Figura 5.11: Resultados de segmentación de biomarcadores utilizando el modelo SAM. Se observa cómo la segmentación ofrece un detalle más preciso de la forma y extensión de los biomarcadores.

CAPÍTULO 6

CONCLUSIONES

En este trabajo se presentó un enfoque integral para la clasificación multiclase y multietiqueta, detección de objetos y segmentación de biomarcadores relacionados con la retinopatía diabética, empleando modelos de aprendizaje profundo como CNN, CycleGAN, YOLOv8 y SAM. Estas metodologías fueron evaluadas tanto en datasets tradicionales como en conjuntos no utilizados durante el entrenamiento, con el objetivo de validar la robustez y la generalización de los modelos en distintos escenarios clínicos.

Se implementaron modelos de clasificación multiclase utilizando CNN para detectar retinopatía diabética en dos, cuatro y cinco clases. Los resultados evidenciaron un desempeño moderado del modelo CNN en tareas de clasificación binaria, así como en 4 y 5 clases, en comparación con los experimentos reportados en la literatura. No obstante, se obtuvieron métricas superiores de precisión y recall, tal como se muestra en la Tabla 6.1 Por otro lado, el modelo YOLOv8 fue empleado para la detección de objetos clave como el disco óptico, copa óptica, vasos sanguíneos, mácula, microaneurismas, hemorragias, exudados duros y exudados suaves en imágenes del fondo de ojo. Los resultados mostraron una precisión alta y consistencia en los biomarcadores más representados, aunque áreas menos frecuentes como los microaneurismas y hemorragias aún presentan un margen de mejora.

Tabla 6.1: Desempeño del modelo CNN en clasificación de 5 clases en comparación con trabajos previos.

Autor	Dataset	Accuracy	precisión	Recall	F1-Score
Nuestro Trabajo	APTOS	0.9299	0.9124	0.9320	0.9214
Shanshan Zhu et al. [105]	APTOS	0.9700	?	0.9700	?
Thanikachalam et al. [106]	MESSIDOR	0.9791	0.9700	0.9782	0.9800
Cao et al. [107]	DIARETDBI	0.9610	0.9970	0.8780	0.9340
Menaouer et al. [108]	Kaggle	0.9060	0.9460	0.9500	0.9400
Qummar et al. [109]	Kaggle	0.9060	0.9460	0.9500	0.9400

En cuanto a segmentación, el modelo SAM fue utilizado para segmentar biomarcadores como el disco óptico, los vasos sanguíneos y los exudados duros. Los resultados, evaluados con métricas como Dice, AUC-PR y AUC-ROC, demostraron que el modelo proporciona segmentaciones confiables, destacándose el desempeño en la segmentación del disco óptico. Asimismo,

se integraron datasets adicionales, como REFUGE, RITE, E-Optha y IDRID, lo que permitió evaluar la generalización de los modelos en escenarios clínicos más diversos.

Finalmente, se implementaron los resultados en una aplicación web como prueba de concepto, demostrando su utilidad potencial en un entorno clínico. Esta plataforma busca facilitar la integración de los modelos en la práctica médica, apoyando a los profesionales en el análisis e identificación de biomarcadores de la retinopatía diabética. Los resultados obtenidos confirman el potencial de los modelos de aprendizaje profundo en el análisis de imágenes médicas, aunque se reconoce la necesidad de mejorar los modelos y ampliar los datasets para alcanzar niveles óptimos de precisión y confiabilidad. Este trabajo representa un avance significativo en el uso de herramientas de inteligencia artificial en el ámbito de la oftalmología y sienta las bases para futuras investigaciones y desarrollos clínicos más avanzados.

Referencias

[1] International Diabetes Federation. Diabetes facts figures, 2024. Último acceso: 11 de diciembre de 2024.

[2] GE Umpierrez, GM Davis, NA ElSayed, GP Fadini, RJ Galindo, IB Hirsch, DC Klonoff, RG McCoy, S Misra, RA Gabbay, RR Bannuru, and KK Dhatariya. Hyperglycaemic crises in adults with diabetes: a consensus report. *Diabetologia*, 67(8):1455–1479, Aug 2024. Epub 2024 Jun 22.

[3] D Aliseda and L Berástegui. Retinopatía diabética. In *Anales del sistema sanitario de Navarra*, volume 31, pages 23–34. SciELO Espana, 2008.

[4] F. Tan, Q. Chen, X. Zhuang, et al. Associated risk factors in the early stage of diabetic retinopathy. *Eye and Vision*, 6:23, 2019.

[5] F M Javed Mehedi Shamrat, Rashiduzzaman Shakil, Sharmin, Nazmul Hoque ovy, Bonna Akter, Md Zunayed Ahmed, Kawsar Ahmed, Francis M. Bui, and Mohammad Ali Moni. An advanced deep neural network for fundus image analysis and enhancing diabetic retinopathy detection. *Healthcare Analytics*, 5:100303, 2024.

[6] Yinlin Cheng, Mengnan Ma, Xingyu Li, and Yi Zhou. Multi-label classification of fundus images based on graph convolutional network. *BMC Medical Informatics and Decision Making*, 21:1–9, 2021.

[7] Leyuan Fang, Liumao Yang, Shutao Li, Hossein Rabbani, Zhimin Liu, Qinghua Peng, and Xiangdong Chen. Automatic detection and recognition of multiple macular lesions in retinal optical coherence tomography images with multi-instance multilabel learning. *Journal of biomedical optics*, 22(6):066014–066014, 2017.

[8] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[9] Zongyun Gu, Yan Li, Zijian Wang, Junling Kan, Jianhua Shu, and Qing Wang. Classification of diabetic retinopathy severity in fundus images using the vision transformer and residual attention. *Computational Intelligence and Neuroscience*, 2023(1):1305583, 2023.

[10] Lisa Toto, Anna Romano, Marco Pavan, Dante Degl'Innocenti, Valentina Olivotto, Federico Formenti, Pasquale Viggiano, Edoardo Midena, and Rodolfo Mastropasqua. A deep learning approach to hard exudates detection and disorganization of retinal inner layers identification on oct images. *Scientific Reports*, 14(1):16652, 2024.

[11] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything, 2023.

[12] Wenxue Li, Xinyu Xiong, Peng Xia, Lie Ju, and Zongyuan Ge. Tp-drseg: improving diabetic retinopathy lesion segmentation with explicit text-prompts assisted sam. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 743–753. Springer, 2024.

[13] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. A neural algorithm of artistic style, 2015.

[14] Karanveer Singh and Daniel Drzewicki. Neural style transfer for medical image augmentation, 2019.

[15] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks, 2020.

[16] Zineb Sadok, Mohamed Akil, Rostom Kachouri, and Ali Ahaitouf. Diabetic retinopathy screening within unlabeled dataset based on least squares cycle-gan domain transfer. In *2024 IEEE Thirteenth International Conference on Image Processing Theory, Tools and Applications (IPTA)*, pages 1–7, 2024.

[17] Ruiming Zhu, Xinliang Liu, Mingrui Li, Wei Qian, and Yueyang Teng. A novel perceptual constrained cyclegan with attention mechanisms for unsupervised mr-to-ct synthesis. *International Journal of Imaging Systems and Technology*, 34(5):e23169, 2024.

[18] GS Martinez, AJ Campbell, J Reinken, and BC Allan. Prevalence of ocular disease in a population study of subjects 65 years old and older. *American journal of ophthalmology*, 94(2):181–189, 1982.

[19] Liran Tiosano, Ron Abutbul, Rivkah Lender, Yahel Shwartz, Itay Chowers, Yedid Hoshen, and Jaime Levy. Anomaly detection and biomarkers localization in retinal images. *Journal of Clinical Medicine*, 13(11):3093, 2024.

[20] Rui Li, Ying Hui, Xiaoyue Zhang, Shun Zhang, Bin Lv, Yuan Ni, Xiaoshuai Li, Xiaoliang Liang, Ling Yang, Han Lv, et al. Ocular biomarkers of cognitive decline based on deep-learning retinal vessel segmentation. *BMC geriatrics*, 24(1):28, 2024.

[21] Ethan Waisberg, Joshua Ong, Sharif Amit Kamran, Mouayad Masalkhi, Phani Paladugu, Nasif Zaman, Andrew G Lee, and Alireza Tavakkoli. Generative artificial intelligence in ophthalmology. *Survey of ophthalmology*, 2024.

[22] Jeremy N Shapiro, Rebhi O Abuzaitoun, Yue Pan, Maria A Woodward, Daniel J Clauw, Paul P Lee, Roni M Shtein, and Lindsey B De Lott. Health care use for eye pain. *JAMA ophthalmology*, 2024.

[23] Chu-Ting Wu, Ting-Yi Lin, Cheng-Jun Lin, and De-Kuang Hwang. The future application of artificial intelligence and telemedicine in the retina: A perspective. *Taiwan Journal of Ophthalmology*, 13(2):133–141, 2023.

[24] Xinpeng Zhang, Congcong Wang, Yilin Han, and Shengyong Chen. Apnet: Adaptive patch-based network for microaneurysm detection in fundus images. In *2024 IEEE 33rd International Symposium on Industrial Electronics (ISIE)*, pages 1–6, 2024.

[25] Mufassir Matloob Abbasi, Shahzaib Iqbal, Khursheed Aurangzeb, Musaed Alhussein, and Tariq M Khan. Lmbis-net: A lightweight bidirectional skip connection based multi-path cnn for retinal blood vessel segmentation. *Scientific Reports*, 14(1):15219, 2024.

[26] Caroline Morais, Ka Lai Yung, Karl Johnson, Raphael Moura, Michael Beer, and Edoardo Patelli. Identification of human errors and influencing factors: A machine learning approach. *Safety science*, 146:105528, 2022.

[27] Yeji Hong, Somin Park, Hongjo Kim, and Hyoungkwan Kim. Synthetic data generation using building information models. *Automation in Construction*, 130:103871, 2021.

[28] Jian Lian and Tianyu Liu. Lesion identification in fundus images via convolutional neural network-vision transformer. *Biomedical Signal Processing and Control*, 88:105607, 2024.

[29] Asifullah Khan, Zunaira Rauf, Anabia Sohail, Abdul Rehman Khan, Hifsa Asif, Aqsa Asif, and Umair Farooq. A survey of the vision transformers and their cnn-transformer based variants. *Artificial Intelligence Review*, 56(Suppl 3):2917–2970, 2023.

[30] Sachin Panchal and Manesh Kokare. Resmu-net: Residual multi-kernel u-net for blood vessel segmentation in retinal fundus images. *Biomedical Signal Processing and Control*, 90:105859, 2024.

[31] Jinchen Yu, Yongwei Nie, Fei Qi, Wenxiong Liao, and Hongmin Cai. Fundusam: A specialized deep learning model for enhanced optic disc and cup segmentation in fundus images. In *2024 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 3955–3960, 2024.

[32] Guilherme C Oliveira, Gustavo H Rosa, Daniel CG Pedronette, João P Papa, Himeesh Kumar, Leandro A Passos, and Dinesh Kumar. Robust deep learning for eye fundus images: Bridging real and synthetic data for enhancing generalization. *Biomedical Signal Processing and Control*, 94:106263, 2024.

[33] Henry Shen-Lih Chen, Guan-An Chen, Jhen-Yang Syu, Lan-Hsin Chuang, Wei-Wen Su, Wei-Chi Wu, Jian-Hong Liu, Jian-Ren Chen, Su-Chen Huang, and Eugene Yu-Chuan Kang. Early glaucoma detection by using style transfer to predict retinal nerve fiber layer thickness distribution on the fundus photograph. *Ophthalmology Science*, 2(3):100180, 2022.

[34] Lixue Liu, Jiaming Hong, Yuxuan Wu, Shaopeng Liu, Kai Wang, Mingyuan Li, Lanqin Zhao, Zhenzhen Liu, Longhui Li, Tingxin Cui, et al. Digital ray: enhancing cataractous fundus images using style transfer generative adversarial networks to improve retinopathy detection. *British Journal of Ophthalmology*, 2024.

[35] Ghazal Bargshady, Xujuan Zhou, Prabal Datta Barua, Raj Gururajan, Yuefeng Li, and U Rajendra Acharya. Application of cyclegan and transfer learning techniques for automated detection of covid-19 using x-ray images. *Pattern Recognition Letters*, 153:67–74, 2022.

[36] Zheng Huang, Ming Sun, Yuxin Liu, and Jiajun Wu. Csaunet: A cascade self-attention u-shaped network for precise fundus vessel segmentation. *Biomedical Signal Processing and Control*, 75:103613, 2022.

[37] Shreeram Athreya, Ashwath Radhachandran, Vedrana Ivezić, Vivek Sant, Corey W Arnold, and William Speier. Ultrasound image enhancement using cyclegan and perceptual loss. *arXiv preprint arXiv:2312.11748*, 2023.

[38] Ali Jamali, Swalpa Kumar Roy, and Pedram Ghamisi. Wetmapformer: A unified deep cnn and vision transformer for complex wetland mapping. *International Journal of Applied Earth Observation and Geoinformation*, 120:103333, 2023.

[39] A. R. Fernandes, E. Sanchez-Lopez, T. d. Santos, M. L. Garcia, A. M. Silva, and E. B. Souto. Development and characterization of nanoemulsions for ophthalmic applications: Role of cationic surfactants. *Materials*, 14:7541, 2021.

[40] Derek W. DelMonte and Terry Kim. Anatomy and physiology of the cornea. *Journal of Cataract Refractive Surgery*, 37(3):588–598, 2011.

[41] Andrew J Rózsa and Roger W Beuerman. Density and organization of free nerve endings in the corneal epithelium of the rabbit. *Pain*, 14(2):105–120, 1982.

[42] B.K. Leung, J.A. Bonanno, and C.J. Radke. Oxygen-deficient metabolism and corneal edema. *Progress in Retinal and Eye Research*, 30(6):471–492, 2011.

[43] Ashim K Mitra and Ashim K Mitra. Ophthalmic drug delivery systems. 2003.

[44] Martina Brandner, Sarah Thaler-Saliba, Sophie Plainer, Bertram Vidic, Yosuf El-Shabrawi, and Navid Ardjomand. Retropupillary fixation of iris-claw intraocular lens for aphakic eyes in children. *PLoS One*, 10(6):e0126614, 2015.

[45] Christina J Lee, Jonathan A Vroom, Harvey A Fishman, and Stacey F Bent. Determination of human lens capsule permeability and its feasibility as a replacement for bruch's membrane. *Biomaterials*, 27(8):1670–1678, 2006.

[46] Cleveland Clinic. Optic nerve, n.d. Accessed: 2024-08-30.

[47] M Asejczyk-Widlicka and BK Pierscionek. The elasticity and rigidity of the outer coats of the eye. *British Journal of Ophthalmology*, 92(10):1415–1418, 2008.

[48] Poonam Chopra, Jinsong Hao, and S Kevin Li. Iontophoretic transport of charged macromolecules across human sclera. *International journal of pharmaceutics*, 388(1-2):107–113, 2010.

[49] Sam P Most, Steven Ross Mobley, and Wayne F Larrabee. Anatomy of the eyelids. *Facial plastic surgery clinics*, 13(4):487–492, 2005.

[50] PE Miller. Chapter 11—uvea. *Slatter's Fundamentals of Veterinary Ophthalmology, 4ed.; Maggs, DJ, Miller, PE, Ofri, R., Eds*, pages 203–229, 2008.

[51] Mike Boulton and Pierrette Dayhaw-Barker. The role of the retinal pigment epithelium: topographical variation and ageing changes. *Eye*, 15(3):384–389, 2001.

[52] K. H. Nguyen, B. C. Patel, and P. Tadi. *Anatomía de la cabeza y el cuello: retina del ojo*. StatPearls Publishing, Treasure Island (FL), 2024. Updated: 2023-08-08.

[53] R Michel, C Wilkinson, and T Rice. Retinal detachment. *St Louis: CV Mosby*, 117, 1990.

[54] E Dhiravidachelvi and V Rajamani. A survey on anatomical structures: In fundus retinal images. *Red*, 1:3, 2014.

[55] JOST B JONAS, ANSELM E GRÜNDLER, and KONSTANTINOS I PAPASTATHOPOULOS. Optic disc dimensions, body length, and body weight. *British journal of ophthalmology*, 82(2):196–196, 1998.

[56] Ahmed Almazroa, Ritambhar Burman, Kaamran Raahemifar, and Vasudevan Lakshminarayanan. Optic disc and optic cup segmentation methodologies for glaucoma image detection: a survey. *Journal of ophthalmology*, 2015(1):180972, 2015.

[57] Muhammad Moazam Fraz, Paolo Remagnino, Andreas Hoppe, Bunyarit Uyyanonvara, Alicja R Rudnicka, Christopher G Owen, and Sarah A Barman. Blood vessel segmentation methodologies in retinal images–a survey. *Computer methods and programs in biomedicine*, 108(1):407–433, 2012.

[58] Navjot K Gahir and Mirriam Shah. Anatomy of the eye and the healthy fundus. *Diabetic Retinopathy: Screening to Treatment 2E (ODL)*, page 13, 2020.

[59] Lawrence A Yannuzzi, Michael D Ober, Jason S Slakter, Richard F Spaide, Yale L Fisher, Robert W Flower, and Richard Rosen. Ophthalmic fundus imaging: today and beyond. *American journal of ophthalmology*, 137(3):511–524, 2004.

[60] S Ramkumar, G Sasi, et al. Detection of diabetic retinopathy using oct image. *Materials Today: Proceedings*, 47:185–190, 2021.

[61] Nikos Tsiknakis, Dimitris Theodoropoulos, Georgios Manikis, Emmanouil Ktistakis, Ourania Boutsora, Alexa Berto, Fabio Scarpa, Alberto Scarpa, Dimitrios I Fotiadis, and Kostas Marias. Deep learning for diabetic retinopathy detection and classification based on fundus images: A review. *Computers in biology and medicine*, 135:104599, 2021.

[62] Tien-En Tan and Tien Yin Wong. Diabetic retinopathy: Looking forward to 2030. *Frontiers in Endocrinology*, 13:1077669, 2023.

[63] Jing Zhou and Bo Chen. Retinal cell damage in diabetic retinopathy. *Cells*, 12(9):1342, 2023.

[64] Nihat Sayin, Necip Kara, and Gökhan Pekel. Ocular complications of diabetes mellitus. *World journal of diabetes*, 6(1):92, 2015.

[65] J Moore, S Bagley, G Ireland, D McLeod, and ME Boulton. Three dimensional analysis of microaneurysms in the human diabetic retina. *The Journal of Anatomy*, 194(1):89–100, 1999.

[66] Baljit Kaur and David Taylor. Retinal haemorrhages. *Archives of disease in childhood*, 65(12):1369, 1990.

[67] Wynne Hsu, PMDS Pallawala, Mong Li Lee, and Kah-Guan Au Eong. The role of domain knowledge in the detection of retinal hard exudates. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 2, pages II–II. IEEE, 2001.

[68] Atif Bin Mansoor, Zohaib Khan, Adil Khan, and Shoab Ahmad Khan. Enhancement of exudates for the diagnosis of diabetic retinopathy using fuzzy morphology. In *2008 IEEE International Multitopic Conference*, pages 128–131. IEEE, 2008.

[69] U Rajendra Acharya, Eddie Yin-Kwee Ng, Jen-Hong Tan, S Vinitha Sree, and Kwan-Hoong Ng. An integrated index for the identification of diabetic retinopathy stages using texture parameters. *Journal of medical systems*, 36:2011–2020, 2012.

[70] Asharul Islam Khan and Salim Al-Habsi. Machine learning in computer vision. *Procedia Computer Science*, 167:1444–1451, 2020.

[71] Pattara Leelaprute, Bodin Chinthanet, Supatsara Wattanakriengkrai, Raula Gaikovina Kula, Pongchai Jaisri, and Takashi Ishio. Does coding in pythonic zen peak performance? preliminary experiments of nine pythonic idioms at scale, 2022.

[72] Vipin Tyagi. *Understanding digital image processing*. CRC Press, 2018.

[73] Aurélien Géron. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. O'Reilly Media, Inc., Beijing, Boston, Farnham, Sebastopol, Tokyo, second edition, September 2019.

[74] Akshansh Mishra. Contrast limited adaptive histogram equalization (clahe) approach for enhancement of the microstructures of friction stir welded joints, 2021.

[75] Justin M Johnson and Taghi M Khoshgoftaar. Survey on deep learning with class imbalance. *Journal of big data*, 6(1):1–54, 2019.

[76] Kushankur Ghosh, Colin Bellinger, Roberto Corizzo, Paula Branco, Bartosz Krawczyk, and Nathalie Japkowicz. The class imbalance problem in deep learning. *Machine Learning*, 113(7):4845–4901, 2024.

[77] Junfeng Gao, Yong Yang, Pan Lin, and Dong Sun Park. Computer vision in healthcare applications. *Journal of healthcare engineering*, 2018, 2018.

[78] Zhe Jia, Marco Maggioni, Benjamin Staiger, and Daniele P Scarpazza. Dissecting the nvidia volta gpu architecture via microbenchmarking. *arXiv preprint arXiv:1804.06826*, 2018.

[79] Jack Choquette, Wishwesh Gandhi, Olivier Giroux, Nick Stam, and Ronny Krashinsky. Nvidia a100 tensor core gpu: Performance and innovation. *IEEE Micro*, 41(2):29–35, 2021.

[80] Kiran Seshadri, Berkin Akin, James Laudon, Ravi Narayanaswami, and Amir Yazdanbakhsh. An evaluation of edge tpu accelerators for convolutional neural networks, 2022.

[81] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[82] Davide Anguita, Luca Ghelardoni, Alessandro Ghio, Luca Oneto, Sandro Ridella, et al. The'k'in k-fold cross validation. In *ESANN*, volume 102, pages 441–446, 2012.

[83] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors, 2012.

[84] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[85] Abien Fred Agarap. Deep learning using rectified linear units (relu), 2019.

[86] Carole H. Sudre, Wenqi Li, Tom Vercauteren, Sebastien Ourselin, and M. Jorge Cardoso. *Generalised Dice Overlap as a Deep Learning Loss Function for Highly Unbalanced Segmentations*, page 240–248. Springer International Publishing, 2017.

[87] Feihu Huang. On momentum-based gradient methods for bilevel optimization with nonconvex lower-level, 2023.

[88] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[89] Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big data*, 3:1–40, 2016.

[90] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision, 2015.

[91] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[92] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks, 2018.

[93] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection, 2016.

[94] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[95] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

[96] Lee R. Dice. Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302, 1945.

[97] Maggie Karthik and Sohier Dane. Aptos 2019 blindness detection. https://kaggle.com/competitions/aptos2019-blindness-detection, 2019. Accessed: 2025-01-23.

[98] Ltd. Shanggong Medical Technology Co. Odir-2019: Ophthalmic database for intelligent recognition. ODIR-2019 Grand Challenge, 2019.

[99] Emma Dugas, Jared, Jorge, and Will Cukierski. Diabetic retinopathy detection. https://kaggle.com/competitions/diabetic-retinopathy-detection, 2015. Kaggle.

[100] Prasanna Porwal, Samiksha Pachade, Ravi Kamble, Manesh Kokare, Girish Deshmukh, Vivek Sahasrabuddhe, and Fabrice Meriaudeau. Indian diabetic retinopathy image dataset (idrid), 2018.

[101] Etienne Decencière, Xiwei Zhang, Guy Cazuguel, Bob Lay, Béatrice Cochener, Claire Trone, Philippe Gain, René Ordonez, Pascale Massin, André Erginay, Bertrand Charton, and Jean-Claude Klein. Feedback on a publicly distributed image database: The messidor database. *Image Analysis and Stereology*, 33(3):231–234, 2014.

[102] G Lepetit-Aimon, C Playout, MC Boucher, R Duval, MH Brent, and F Cheriet. Maplesdr: Messidor anatomical and pathological labels for explainable screening of diabetic retinopathy (jan 2024).

[103] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[104] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. 2014.

[105] Shanshan Zhu, Changchun Xiong, Qingshan Zhong, and Yudong Yao. Diabetic retinopathy classification with deep learning via fundus images: A short survey. *IEEE Access*, 12:20540–20558, 2024.

[106] V Thanikachalam, K Kabilan, and Sudheer Kumar Erramchetty. Optimized deep cnn for detection and classification of diabetic retinopathy and diabetic macular edema. *BMC Medical Imaging*, 24(1):227, 2024.

[107] Kedir M Adal, Peter G Van Etten, Jose P Martinez, Kenneth W Rouwen, Koenraad A Vermeer, and Lucas J van Vliet. An automated system for the detection and classification of retinal changes due to red lesions in longitudinal fundus images. *IEEE transactions on biomedical engineering*, 65(6):1382–1390, 2017.

[108] Brahami Menaouer, Zoulikha Dermane, Nour El Houda Kebir, and Nada Matta. Diabetic retinopathy classification using hybrid deep learning approach. *SN Computer Science*, 3(5):357, 2022.

[109] Suvajit Dutta, BC Manideep, Syed Muzamil Basha, Ronnie D Caytiles, and NCSN Iyengar. Classification of diabetic retinopathy images by using deep learning models. *International Journal of Grid and Distributed Computing*, 11(1):89–106, 2018.