Detección y Prevención de Riesgo de Enfermedades en Plantaciones de Café mediante Inteligencia Artificial

Luis Esteban Rosas Ruiz, Andrey Fernando Salom Medina

Trabajo de Grado para Optar al Título de ingeniero electrónico

Director

Jaime Guillermo Barrero Pérez

Magíster en Potencia Eléctrica

Universidad Industrial de Santander

Facultad de ingenierías fisicomecánicas

Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones

Ingeniería electrónica

Bucaramanga

2024

Dedicatoria

Este trabajo es la culminación de muchos años de trabajo duro, sacrificios económicos, mentales y físicos no solo míos sino de mi familia. Doy gracias a Dios, mi Padre que siempre me acompañó desde el amor más grande y me abrazó como el hijo pródigo que soy.

Esto va dedicado a mi papa, Merardo, mi ejemplo más grande y líder de mi familia que me enseñó el valor de trabajar por mi propio esfuerzo. También va por mi mamá, Esperanza, ella siempre me dio todo su amor incondicional y siempre creyó en mí, incluso cuando ni yo mismo lo hacía. Va por Sergio, mi hermano, mi mejor amigo y mayor orgullo, quien siempre supo escucharme, apoyarme y sobre todo entenderme.

Gracias a mi tía Mónica que me acogió en su hogar, me tendió la mano cuando nadie más lo hizo y me convirtió en un hijo más, gracias a Diego, a Mónica, a Marcela y Daniel por darme aliento, apoyo incondicional y mucho amor de familia.

Y a mis amigos: mi familia escogida que siempre estuvieron a mi lado, a Yan y Juli por ser mis compañeros de risas, lágrimas e historias; a Eve que siempre estuvo orgullosa de esto que hago y me apoyó desde el primer momento; a José, Zuley, Yerle, Diana y Luisa quienes supieron ser amigos verdaderos donde encuentro paz para mi corazón. a Nini, con quién siempre encuentro el lado chistoso de la vida; a Yerferson, con quien compartimos la dificultad de la vida universitaria, pero entendiendo que al final siempre vale la pena. Gracias al movimiento Juan 23 donde encuentro un pedacito de cielo aquí en la tierra. Y a tantos más:
¡GRACIAS INFINITAS!

- Esteban Rosas

Quisiera tomar este breve momento para expresar mi más sincero agradecimiento a las personas que han hecho posible la realización de este proyecto. En especial, quiero agradecerles a mis padres, su apoyo incondicional y sacrificio han sido la piedra angular de mi viaje. Además, quiero agradecer a mi familia y amigos cercanos, quienes han sido mi fuente de motivación constante. Este proyecto no solo representa mis esfuerzos individuales, sino también el resultado del respaldo de aquellos que han estado siempre a mi lado. Estoy agradecido por tenerlos en mi vida y lo tendré siempre marcado en mi memoria.

- Andrey Salom

Agradecimientos

Expresamos un sincero agradecimiento a todas las personas que contribuyeron de manera significativa a la realización de este trabajo de grado.

En primer lugar, agradecemos a nuestro director, el profesor Jaime Barrero, por su orientación experta, paciencia, y apoyo constante a lo largo de este proceso. Sus valiosas sugerencias y comentarios han sido fundamentales para dar forma y mejorar este trabajo.

Hay que agradecer a la Universidad Industrial de Santander que ha proporcionado herramientas, conocimientos y contactos adecuados los cuales han contribuido a la formación integral de ingenieros como individuos íntegros, así como profesionales capacitados para el servicio de la sociedad.

Mi reconocimiento especial a Martin Javier, ingeniero agrónomo que trabaja en el Comité Nacional de Cafeteros, quien dedicó su tiempo para revisar y ofrecer valiosos comentarios sobre este trabajo, así como proporcionó valiosa ayuda en el enfoque del proyecto.

Agradecemos a Eliana Salom, ingeniera agroindustrial, quien proporcionó valiosa ayuda en el complicado proceso de etiquetado de las imágenes, gracias por tanto apoyo y tiempo.

También queremos agradecer a los compañeros de clase y a la comunidad académica de la Universidad Industrial de Santander por su colaboración, intercambio de ideas, y el ambiente enriquecedor que han proporcionado.

Tabla de contenido

Lista de Figuras

Lista de Tablas

Resumen

Título: Detección y prevención de riesgo de enfermedades en plantaciones de café mediante Inteligencia Artificial[*]

Autor: Luis Esteban Rosas Ruiz, Andrey Fernando Salom Medina[**]

Palabras Clave: Inteligencia artificial, Café, Agroindustria, Imágenes, Enfermedad, Prevención, Detección, Segmentación

Descripción: El cultivo del café es una fuente de ingresos de aproximadamente 550 mil familias en Colombia y específicamente en Santander hay más de 30 mil cafeteras. Dado que es un cultivo masivo y que representa los ingresos económicos de muchas personas, se investiga sobre las enfermedades en hojas de café, centrando el proyecto en la detección de las enfermedades de roya, el bicho minador y el insecto conocido como "coco". La base de datos se obtuvo en un cultivo en El Socorro, Santander, garantizando la autenticidad de las muestras, así como de otras dos fuentes que se revisaron rigurosamente para la aplicación del proyecto.

La calidad de los datos es crucial para este tipo de proyectos, para lo cual se hicieron procesos de limpieza en la base de datos donde se eliminaron imágenes redundantes, incoherentes e innecesarias, lo que asegura una base de datos lo suficientemente diversa para el proceso de entrenamiento. También se etiquetó cada una de las imágenes resultantes de la limpieza, primero usando la herramienta en línea VGG, pero dada la complejidad y el demorado proceso de etiquetar, se decidió usar SAM, una innovadora IA lanzada por el equipo de META en el 2023 que facilitó dicho proceso y proporcionó una base de datos con buena calidad de etiquetas. En el entrenamiento se usó la arquitectura MASK R-CNN, respaldada por un Backbone de RESNET 50 y 101. Además del uso de Transfer Learning con Dataset de “COCO". Luego se creó una aplicación móvil donde se puede tomar una imagen del teléfono, se envía a un servidor web alojado en Google Cloud y desde allí se procesa dicha imagen, hace una inferencia y envía el resultado de la inferencia de nuevo a la aplicación, donde el usuario puede observar las afectaciones de la imagen que previamente envió.

---

[*] Trabajo de grado

[**] Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones. Director: Jaime Guillermo Barrero Pérez, Magíster en Potencia Eléctrica

Abstract

Title: Detection and prevention of disease risks in coffee plantations through artificial intelligence[*]

Authors: Luis Esteban Rosas Ruiz, Andrey Fernando Salom Medina[**]

Key Words: Artificial intelligence, Coffee, Agribusiness, Images, Disease, Prevention, Detection, Segment

Description: Coffee cultivation is a source of income for approximately 550,000 families in Colombia, and specifically in Santander, there are more than 30,000 coffee plants. Given that it is a massive cultivation and represents the economic livelihood of many people, research is conducted on coffee leaf diseases, focusing the project on detecting rust, leaf miner, and the insect known as "coco". The database was obtained from a plantation in El Socorro, Santander, ensuring the authenticity of the samples, as well as from two other sources that were rigorously reviewed for the project's application.

Data quality is crucial for such projects, for which data cleaning processes were performed in the database, removing redundant, inconsistent, and unnecessary images, ensuring a sufficiently diverse database for the training process. Each of the resulting images from the cleaning process was also labeled, initially using the VGG online tool, but given the complexity and time-consuming nature of labeling, SAM, an innovative AI launched by the META team in 2023, was decided to be used, which facilitated the process and provided a database with good label quality. MASK R-CNN architecture, backed by a RESNET 50 and 101 backbone, was used in training, in addition to Transfer Learning with the COCO dataset. Then, a mobile application was created where users can take a picture with their phone, send it to a web server hosted on Google Cloud, process the image there, make an inference, and send the inference result back to the application, where the user can observe the effects of the image they previously sent.

---

[*] Degree Work

[**] Faculty of Physical-Mechanical Engineering. School of Electrical, Electronic, and Telecommunications Engineering. Director: Jaime Guillermo Barrero Pérez, Master's in Electrical Power

Introducción

El cultivo del café es sustento en la economía de cerca de 550 mil familias en Colombia, actualmente enfrenta desafíos críticos que amenazan su productividad y sostenibilidad. Este trabajo aborda uno de los problemas más importantes de los caficultores: la detección y prevención eficaz de enfermedades que afectan la salud de las plantas, centrándose en la implementación de tecnologías avanzadas, específicamente en el ámbito de la inteligencia artificial.

A pesar de que la mayoría de los cultivos de café en Colombia se consideran resistentes a enfermedades, la adaptación constante de hongos, enfermedades y plagas a las condiciones específicas de las plantas ha generado un entorno desafiante. La roya del café, con su capacidad para reducir la eficiencia de cada cosecha, se destaca como uno de los mayores contratiempos en la producción cafetalera latinoamericana.

Este proyecto se apoya en antecedentes proporcionados por el Comité Nacional de Cafeteros de Colombia (CNC) y el Centro Nacional de Investigación del Café (CENICAFE). Estudios previos resaltan la influencia de factores genéticos, ambientales y de gestión en la salud del cultivo, subrayando la importancia de un manejo adecuado para prevenir y tratar enfermedades.

El propósito fundamental de esta investigación es abordar las enfermedades del café mediante un enfoque tecnológico innovador basado en inteligencia artificial. La fundamentación se sustenta en la necesidad de mejorar el manejo fitosanitario para garantizar la productividad y competitividad del sector cafetalero colombiano.

La justificación de este proyecto radica en la capacidad de ofrecer una solución tecnológica avanzada a los desafíos recurrentes en el cultivo del café. La aplicación de inteligencia artificial, específicamente a través de un sistema de detección y clasificación de enfermedades, se presenta como una respuesta innovadora para el diagnóstico temprano, prevención y tratamiento eficiente

de enfermedades como la roya o el minero. Este enfoque busca no solo preservar la principal exportación agrícola del país, sino también mejorar las condiciones laborales de los caficultores.

La investigación se enfoca en la implementación de inteligencia artificial avanzada, empleando la arquitectura MASK R-CNN con un Backbone de RESNET101. La elección de esta arquitectura se fundamenta en su capacidad para realizar segmentación de instancias, lo que permite la identificación precisa de áreas afectadas por enfermedades en las hojas de café. El uso de Transfer Learning con el conjunto de datos "COCO" proporciona a la red neuronal conocimientos previos, mejorando así la eficiencia y precisión del modelo.

1. Objetivos

1.1 Objetivo general

Crear un sistema basado en inteligencia artificial con el propósito de identificar, clasificar y prevenir las distintas clases de enfermedades que afectan a las plantaciones de café.

1.2 Objetivos específicos

En desarrollo del objetivo general del trabajo de grado comprende:

- Diseñar un aplicativo móvil que permita a los usuarios capturar y enviar imágenes de hojas de café a un servidor para su análisis.
- Crear una base de datos de imágenes de enfermedades del café lo suficientemente amplia, como para lograr que la inteligencia artificial pueda tener un entrenamiento optimo.
- Diseñar la red neuronal con modelos óptimos con una precisión alta para las imágenes ingresadas.
- Implementar la red neuronal entrenada en un servidor capaz de recibir imágenes desde la aplicación móvil, procesarlas y proporcionar resultados relevantes sobre el estado de las hojas de café y posibles enfermedades presentes.

2. Marco teórico

En este capítulo se presenta una introducción al cultivo del café en Colombia, resaltando los desafíos enfrentados por los caficultores y sus implicaciones. A continuación, se detallan las enfermedades abordadas en el proyecto, seguidas de una discusión sobre las bases de datos y sus metodologías de diseño relevantes. Se profundiza en el aprendizaje automático, explicando las arquitecturas y conceptos clave para comprender las herramientas utilizadas en el entrenamiento y evaluación del modelo. Por último, se exploran conceptos básicos de programación para su implementación en dispositivos móviles.

2.1 Café de Colombia

Colombia es reconocida como uno de los principales productores y exportadores de café, posee una industria con aproximadamente 930,000 hectáreas dedicadas al cultivo del café, el país se destaca por la calidad y diversidad de sus granos, posicionándose en el mercado internacional como uno de los cafés más apetecidos por su sabor suave, acidez equilibrada y notas frutales; La altitud y los microclimas específicos de las regiones cafetaleras contribuyen a la singularidad de estos granos, haciendo del café colombiano un producto codiciado en el mercado global. El café es entonces una de las principales fuentes de ingresos para miles de personas, sin embargo, el cultivo de café es susceptible a muchos riesgos como las fluctuaciones en los precios internacionales del café que afectan en gran medida los ingresos de los caficultores del país.

2.1.1 Problemas actuales

Aunque hoy por hoy la industria cafetalera en Colombia es sólida, sigue enfrentando desafíos que la aquejan desde hace mucho tiempo, ejemplo de ello son las enfermedades recurrentes de las plantaciones en todo el país, como es la roya del café, que representa una amenaza constante en toda la región cafetalera del país. Es por ello por lo que la industria siempre

está en busca de soluciones duraderas y efectivas para el tratamiento fitosanitario de dichas enfermedades que aseguren cosechas sanas y de calidad.

El café en Colombia es un tesoro nacional que ha forjado la identidad del país y ha dejado una huella profunda en su economía y cultura lo que nos hace sentir orgullosos al hablar del café de Colombia.

2.1.2 Enfermedades.

En la investigación previa al proyecto ese encontraron muchas enfermedades y aunque se cuentan por decenas de ellas, se decidió enfocar el proyecto en 3 de ellas que son las que más aquejan a los campesinos de la región.

2.1.2.1 Roya. La roya del café (Hemileia vastatrix) es un hongo que afecta las hojas, disminuyendo la capacidad fotosintética y, en consecuencia, la calidad y cantidad de la cosecha. Investigaciones destacan la importancia de estrategias de control eficaces para combatir la roya, como la aplicación de fungicidas específicos y la promoción de variedades resistentes (Avelino et al., 2015; Perfecto et al., 2014). La gestión integrada, combinando prácticas agronómicas y tecnológicas, emerge como clave para mitigar los impactos de esta enfermedad (Boshier et al., 2018). En la imagen 1 se evidencia el daño que puede causar las hojas afectadas dicho hongo.

Figura 1
Hoja de café con roya (imagen tomada del dataset Rocole).

2.1.2.2 Coco. El insecto de coco abarca varios tipos de mariquita (Coccinellidae**)**, son un tipo de especies que pueden causar daño al consumir las hojas de café. Las afectaciones incluyen

daño mecánico directo a las hojas, dejando perforaciones y desfigurando su apariencia. Tales afectaciones generan estrés en la planta, lo que la hace más susceptible a otras enfermedades. Es crucial reconocer que no todas las especies de mariquitas causan daño, y la identificación precisa es esencial para tomar decisiones de manejo en la agricultura.

Figura 2.
Hoja afectada por el bicho de coco

2.1.2.3 Bicho minador. El *Leucoptera coffeella* es una pequeña polilla que ataca principalmente a las hojas del cafeto. Su ciclo de vida comprende cuatro etapas principales: huevo, larva, pupa y adulto. Las larvas son las causantes del mayor daño en las hojas pues se alimentan del tejido interno de las mismas lo que crea el característico túnel por cual recibe el nombre su nombre: “minador”.

El ataque del bicho minador tiene varias consecuencias negativas en los cultivos de café como el daño directo a las hojas que impacta negativamente la capacidad fotosintética de la planta, resultando en una disminución de la producción de carbohidratos y, en última instancia, una reducción en la cantidad de granos de café. Las plantas afectadas experimentan un debilitamiento general debido a la pérdida de nutrientes. Además, que el estrés en las plantas compromete la calidad del café producido, pudiendo generar cambios en el sabor, aroma y perfil sensorial de la taza. Detectar y monitorear la presencia del bicho minador es crucial para implementar estrategias de manejo efectivas. Se han desarrollado diversas técnicas, incluyendo trampas de feromonas y evaluación visual de hojas minadas.

Figura 3

Hoja afectada por el bicho Minador (imagen tomada de rust and leaf miner)

2.2 Machine Learning

El aprendizaje automático, también conocido como Machine Learning, es un campo dentro de la inteligencia artificial que consiste en enseñar a las máquinas a identificar patrones y tomar decisiones por sí mismas, sin necesidad de intervención humana directa. Funciona mediante algoritmos que analizan datos, descubren tendencias y se ajustan automáticamente a medida que reciben nueva información. En lugar de seguir reglas predefinidas, estas máquinas aprenden y mejoran con el tiempo. Hay tres enfoques principales: el aprendizaje supervisado, que utiliza datos previamente etiquetados; el no supervisado, que busca patrones en datos sin etiquetar; y el por refuerzo, en el que la máquina toma decisiones basadas en su interacción con el entorno. Desde personalizar recomendaciones en plataformas en línea hasta realizar diagnósticos médicos avanzados, el aprendizaje automático está impulsando la innovación y transformando la manera en que se interactúa con la tecnología y se procesa la información.

2.2.1 Deep learning

El aprendizaje profundo, conocido como Deep Learning, es una rama avanzada del aprendizaje automático que emula la complejidad del cerebro humano mediante el uso de redes neuronales profundas. Estas estructuras computacionales consisten en múltiples capas interconectadas, permitiendo la extracción automática de características y la representación

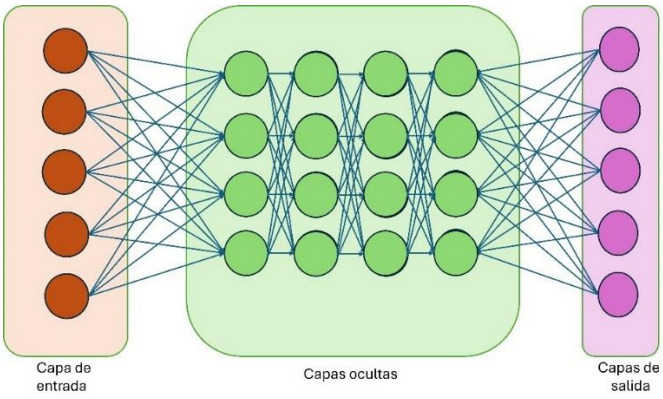
jerárquica de datos. En el corazón del Deep Learning se encuentran los algoritmos que aprenden automáticamente a discernir patrones complejos a medida que se exponen a datos, adaptándose y mejorando con el tiempo.

Lo distintivo del Deep Learning radica en su capacidad para abordar problemas altamente no lineales y capturar relaciones intrincadas en grandes conjuntos de datos. Esta tecnología ha revolucionado áreas como el reconocimiento de imágenes, el procesamiento de lenguaje natural y la toma de decisiones autónoma. Al incorporar capas profundas, las redes neuronales profundas pueden discernir características más abstractas y realizar tareas sofisticadas.

Este enfoque, introducido por autores como Goodfellow, Bengio y Courville (2016), ha transformado significativamente la inteligencia artificial, impulsando avances notables en diversas disciplinas. En la figura 4 se muestra cómo se compone una red neuronal profunda con sus conexiones.

Figura 4

Arquitectura básica de una Red neuronal Profunda



2.3 Arquitectura Mask R-CNN para Resnet50

La Segmentación de Instancia en Visión Artificial, también conocida como Redes Neuronales Convolucionales Basadas en Regiones (Mask R-CNN), se ha convertido en un

componente esencial para la identificación y delimitación de objetos en imágenes. Este enfoque representa una evolución significativa de la Red Neuronal Convolucional Basada en Regiones (R-CNN), centrándose específicamente en la detección de objetos.

Mask R-CNN, introducido por Kaiming He, Georgia Gkioxari, Piotr Dollár y Ross Girshick en 2017, ha demostrado ser una arquitectura efectiva para la detección de objetos y la segmentación de instancias en imágenes. El surgimiento de R-CNN marcó un hito al introducir la idea de regiones de interés (RoI) y propuestas de regiones, permitiendo una mayor eficiencia en la detección de objetos. Posteriormente, Mask R-CNN amplió esta capacidad al incorporar la segmentación de instancias, brindando una comprensión más detallada de la ubicación y forma de cada objeto en una imagen.

2.3.1 Detalles del algoritmo

Base: La base se establece utilizando la sección convolucional de ResNet50, reconocida por su profundidad de 50 capas. En la implementación específica de Mask R-CNN, se eligen solo los cuatro bloques convolucionales, identificados como ResNet50-C4.

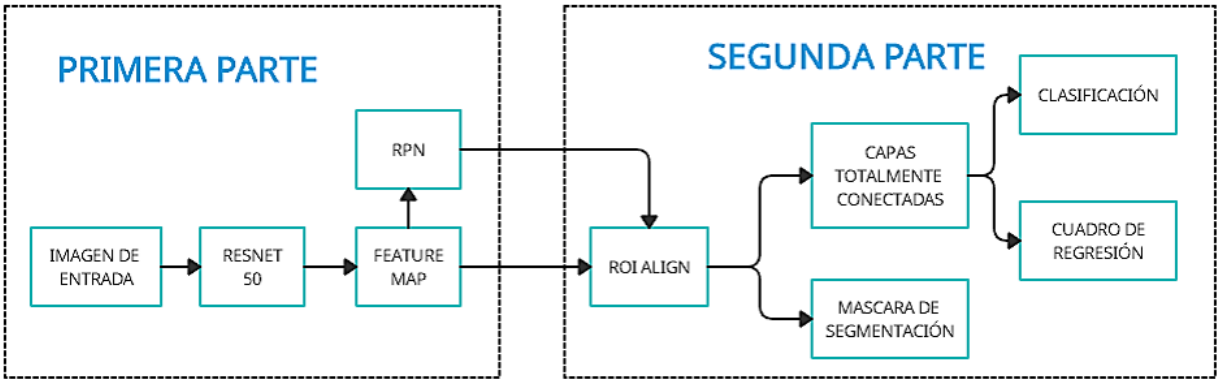
Sección Detectora: Esta sección consta de dos componentes esenciales: una red auxiliar llamada RPN (Red de Propuestas de Regiones) y una red principal que se encarga de analizar las regiones propuestas. Ambas redes tienen la misma entrada derivada de la estructura convolucional.

Red de Propuestas de Regiones (RPN): La RPN utiliza cuadros redimensionados como entrada y pasa por una tercera red de detección con tres salidas: un clasificador que se encarga de determinar la probabilidad de que la región propuesta tenga un objeto una clase específica, un regresor que se encarga de ajustar las coordenadas de la región de interés, y por último un mapa binario de dimensiones n×n.

Todo el funcionamiento de la arquitectura se explica en la Figura 5.

Figura 5

Funcionamiento de una red Mask R-CNN (Resnet50)



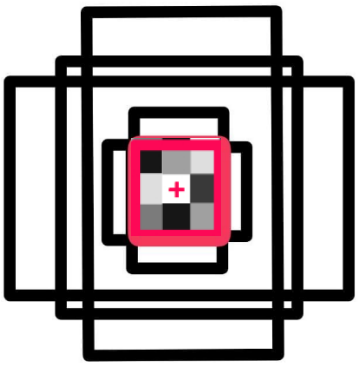
Definición de Ventanas: En Mask R-CNN se emplean 15 ventanas prototipo llamadas anclas, con proporciones de 1:1, 2:1 y 1:2, cubriendo cinco escalas predeterminadas. Estas anclas, estratégicamente centradas en el núcleo del retículo 3×3, son fundamentales en la fase de definición de ventanas.

Capa de Anclaje: En esta capa, 60 neuronas (15 por 4 anclas) desempeñan un papel clave, ajustando y desplazando cada una de las ventanas con respecto a su ancla por medio de una función regresora. Este proceso mejora la alineación y facilita el análisis posterior.

Clasificación Binaria: La siguiente capa alberga un clasificador binario simple, encargado de evaluar si la ventana propuesta contiene un objeto. La función de pérdida utilizada es la entropía binaria cruzada, que mide la discrepancia entre la predicción y la realidad. Si la probabilidad calculada supera el umbral de 0,5, la ventana se envía a la red detectora para un análisis más exhaustivo.

Figura 6

Se utilizan seis de capas de anclaje de dimensión 3x3 dentro de la red con diferentes tamaños



2.3.2 Fase de detección

El detector examina las propuestas generadas, mostrando las coordenadas sobre la última capa convolucional del armazón ResNet50 y creando así la Región de Interés, luego de ello se ajusta la región a un cuadrado de 7x7 pixeles mediante Pooling., este proceso se llama RoIAlign, hecho esto pasa a través de la red detectora, que incluye el quinto bloque del armazón, produciendo 2048 mapas de características, entonces se hace una convolución 3×3 con Stride 1 y una sección de salida.

2.3.3 Creación de la máscara de segmentación

La sección de salida se divide en dos ramas: una que emplea procesos de desconvolución para reducir el número de mapas de características a 80 y aumentar su resolución a 14×14 píxeles, generando máscaras binarias para cada clase; y otra que utiliza Average-Pooling seguido de dos capas de salida, una para clasificación y otra para regresión. En todas las capas se aplica la función de activación ReLU, excepto en la salida clasificadora, donde las neuronas utilizan activación sigmoide junto con la función Softmax.

2.3.4 ¿Por qué se escogió MASK R-CNN sobre otras arquitecturas?

Para responder a esta pregunta se observa la Tabla 1, donde se comparan las 3 arquitecturas bastante conocidas y utilizadas en la segmentación semántica, En los cuadros de color se representan las virtudes y falencias de cada arquitectura con respecto a ciertos aspectos. Aquí, el

color verde indica la mejor opción, el amarillo indica una opción intermedia y el rojo indica una opción menos adecuada para el propósito.

Tabla 1
Comparativa entra las arquitecturas Mask R-CNN vs U-Net vs YOLO v8

Aspecto	Mask R-CNN	U-Net	YOLO v8
Arquitectura	Basada en Faster R-CNN, combina detección y segmentación	Arquitectura de encoder-decoder para segmentación semántica	Basada en una única red neuronal convolución
Precisión	Alta precisión en la detección y segmentación de objetos pequeños	Buena precisión, especialmente en segmentación de instancias	Buena precisión en la detección de objetos, pero menos en segmentación
Eficiencia	Relativamente más lento debido a su complejidad	Eficiente en términos de memoria y tiempo de entrenamiento	Rápido en la detección en tiempo real, YOLOv8 se centra en mejorar la precisión y velocidad
Requiere más datos	Necesita más datos etiquetados para un rendimiento óptimo	Puede funcionar bien con menos datos etiquetados	Puede requerir menos datos etiquetados para un buen rendimiento
Segmentación precisa	Proporciona una segmentación precisa de objetos	Proporciona una buena segmentación de bordes y contornos	Menos preciso en la segmentación de instancias
Flexibilidad	Más flexible en la detección y segmentación de objetos	Menos flexible, diseñado específicamente para segmentación	Menos flexible, diseñado principalmente para detección
Implementación	Disponible en bibliotecas populares como TensorFlow y PyTorch	Implementaciones disponibles, como en TensorFlow y PyTorch	Implementaciones disponibles, como Darknet y TensorFlow
Segmentación de Instancias	Especialmente diseñado para separar objetos individuales, incluso si pertenecen a la misma clase	Puede segmentar objetos en imágenes, pero no es la mejor opción para diferenciar instancias superpuestas de la misma clase	No está diseñado específicamente para la segmentación de instancias

Al utilizar la segmentación de instancias para encontrar el área de afectación en las hojas de café, se cumple los objetivos del proyecto de manera efectiva. Esta técnica permite evaluar con precisión el grado de propagación de las enfermedades y su impacto en la salud de las plantas al calcular el área de afectación. Esto no solo contribuye al objetivo general de identificar y prevenir enfermedades en las plantaciones de café, sino que también proporciona una comprensión detallada del estado de las hojas y permite tomar decisiones informadas sobre la gestión de las plantaciones. Comparado con otras arquitecturas como U-Net o YOLO, la segmentación de instancias, especialmente con Mask R-CNN, destaca por su buena capacidad para separar objetos

individuales, lo que facilita un análisis más preciso de las áreas afectadas por enfermedades. Este enfoque específico es esencial para lograr los objetivos del proyecto y mantener la salud de las plantaciones de café.

Para detectar enfermedades en las hojas de café en el campo, se busca aplicar un enfoque que minimice el contacto directo con las hojas para evitar la propagación de enfermedades. Utilizando Mask R-CNN, que destaca por su capacidad de segmentación de instancias, se segmenta cada hoja como un objeto individual. Esta capacidad permite distinguir cada hoja de café en la imagen, lo que facilita un análisis más detallado de su estado de salud. Se centra en la hoja más grande, que se considera la protagonista, para examinar con detalle el área afectada por la enfermedad. Con la información de la región de la hoja y la región de la enfermedad detectada, se puede calcular el área de afectación de la enfermedad en la hoja principal. Este enfoque integral proporciona una comprensión más completa de la salud de la plantación y ayuda a tomar decisiones informadas sobre su gestión.

$$Area\ de\ afectacion\ = \frac{Area\ de\ la\ enfermedad}{Area\ de\ la\ hoja} * 100 \qquad (1)$$

2.4 Transfer learning 'COCO'

El Conjunto de Datos 'COCO' (Common Objects in Context) desarrollado por Microsoft Research en colaboración con otras instituciones, 'COCO' se presentó por primera vez en 2014 como una respuesta a las limitaciones de los conjuntos de datos existentes en visión por computadora (Lin et al., 2014). Es una colección de imágenes que va más allá de la detección básica de objetos. Contiene información detallada sobre segmentación semántica, relaciones entre objetos y descripciones, convirtiéndolo en un recurso integral para la visión por computadora. Es esencial para entrenar modelos de inteligencia artificial que no solo identifican objetos, sino que también comprenden la complejidad de los entornos del mundo real. En el contexto de la

identificación de enfermedades en hojas de café, ‘COCO’ se destaca como un recurso valioso, proporcionando diversidad y complejidad para adaptar modelos a las condiciones cambiantes y variadas de las plantaciones de café.

2.5 Resnet 50 y Resnet 101

ResNet-50 y ResNet-101 son variantes de la arquitectura ResNet, desarrolladas por He et al. en 2015. Se caracterizan por su profundidad excepcional y su capacidad para aprender representaciones complejas de datos visuales. ResNet-50 consta de 50 capas, mientras que ResNet-101 consta de 101 capas. Estas redes utilizan conexiones residuales para mitigar el problema de la degradación del rendimiento que suele ocurrir con el aumento de la profundidad de la red, permitiendo así el entrenamiento de modelos mucho más profundos.

En la aplicación especifica del proyecto ofrecen una solución efectiva para este desafío al proporcionar capacidades de aprendizaje profundo que permiten identificar patrones sutiles asociados con diversas enfermedades que se abordaron.

2.6 Validación de modelos

Las métricas para evaluar el desempeño de modelos son importantes ya que proporcionan una medida objetiva y cuantificable del éxito. Estas métricas permiten optimizar modelos, identificar áreas de mejora, tomar decisiones informadas y comunicar eficazmente los resultados entre equipos y partes interesadas.

2.6.1 Métrica IOU (Intersection over Union)

La Métrica IoU (Intersección sobre Unión), también conocida como Índice de Jaccard, es importante en la evaluación de las predicciones, especialmente en modelos como Mask R-CNN.

La métrica se calcula al comparar la región predicha con la región de referencia. La fórmula para calcularla es relación entre el área de intersección y el área de unión de ambas regiones.

$$IoU = \frac{Area\ de\ Intersección}{Area\ de\ unión} \quad (2)$$

- **Área de Intersección:** Región en la que las máscaras predichas y de referencia se superponen.
- **Área de Unión:** Área total cubierta por ambas máscaras, incluyendo las regiones superpuestas y las áreas únicas de cada máscara.

Esta métrica proporciona un valor entre 0 y 1, donde 0 indica ninguna superposición y 1 indica una superposición completa. En términos simples, Un valor alto de IoU indica una superposición significativa y, por lo tanto, una mejor calidad de segmentación.

En Mask R-CNN, IoU se emplea para evaluar la precisión de las máscaras predichas en comparación con las máscaras de referencia (ground truth). Es una métrica crucial porque penaliza las predicciones que no se superponen adecuadamente con las regiones de referencia.

3. Base de datos

3.1 Introducción

En este capítulo, se proporcionará una visión detallada de la base de datos creada para el proyecto de detección de enfermedades en hojas de café. Esta sección comenzará con una breve descripción del propósito fundamental de la base de datos, destacando su papel esencial en el desarrollo y entrenamiento del modelo de inteligencia artificial.

3.1.1 Propósito de la base de datos

La base de datos se hace para proporcionar al modelo de detección de enfermedades en hojas de café un conjunto de datos diverso y representativo. La recolección de imágenes, realizada a través de un teléfono celular, se centra en capturar la variabilidad de condiciones y

manifestaciones de enfermedades en las hojas, así como de la toma de imágenes representativas de otros conjuntos de datos ya establecidos. Este conjunto de datos general servirá como la piedra angular sobre la cual el modelo aprenderá a identificar patrones y características clave asociadas con las afecciones.

3.1.2 Importancia de la recopilación de datos

La recopilación meticulosa de datos desempeña un papel crucial en el éxito de cualquier proyecto de inteligencia artificial. La importancia de esta base de datos radica en su capacidad para proporcionar al modelo una comprensión completa de las diferentes formas en que las enfermedades pueden manifestarse en las hojas de café. Desde condiciones de hojas saludables hasta diversas etapas de desarrollo de la roya, el bicho minador y del bicho coco en cada imagen contribuye al enriquecimiento del conocimiento del modelo, permitiéndole realizar inferencias precisas cuando se enfrenta a nuevas instancias.

3.2 Origen de las imágenes

3.2.1 Imágenes propias

Las imágenes de esta sección de la base de datos se tomaron meticulosamente en la finca "La Aurora" en la vereda "La Honda", Socorro, Santander. La finca, de aproximadamente 1.5 hectáreas a 1360 metros sobre el nivel del mar, alberga principalmente la variedad de café "Variedad Colombia", con algunas muestras de "Cenicafe F1". Se eligió este entorno por su diversidad fitogeográfica y la presencia de diferentes variedades de café, enriqueciendo la representatividad del conjunto de datos.

Utilizando un teléfono Xiaomi Poco X3 Pro con cámara principal de 64 MP, lente macro y ajustes fotográficos personalizables, las imágenes se capturaron a lo largo de diferentes días, desde las 8 a.m. hasta las 5 p.m. Este enfoque temporal garantizó la representatividad y robustez

del conjunto, capturando diversas condiciones de iluminación y estados atmosféricos. Además, para ampliar la variedad de entrenamiento, las fotos incluyen diferentes fondos, como hojas verdes, hojas secas y fondo blanco.

Figura 7
Distintos fondos usados en la toma de fotos para la base de datos

3.2.2 Imágenes de RoCoLe: A robusta coffee leaf images dataset

Este conjunto de datos público consta de imágenes capturadas con una cámara de 5MP y apertura f/2.2 de un teléfono inteligente, en formato JPEG. Las fotos se tomaron a una distancia de 200-300 mm de plantas de café, abarcando diversas condiciones meteorológicas y fondos. Se capturaron imágenes de hojas sanas e infectadas desde la parte superior e inferior. Para evaluar la gravedad de la infección por roya, se aplicó el método del Organismo Internacional Regional de Sanidad Agropecuaria (OIRSA), desarrollado por el Laboratorio de Análisis de Riesgo Epidemiológico Fitosanitario (LANREF). El conjunto de datos se recopiló en la Ciudad de la Investigación, Innovación y Desarrollo Agropecuario (CIIDEA), ESPAM MFL, Calceta, Manabí, Ecuador, con coordenadas aproximadas de Latitud -0.834206 y Longitud -80.177159.

Los datos utilizados en este estudio están disponibles públicamente en el repositorio de datos de Mendeley.

3.2.3 Imágenes de rust (Hemileia vastatrix) and leaf miner (Leucoptera coffeella) in coffee crop (Coffea arabica)

Este conjunto de datos auxiliar fue utilizado en un estudio que buscaba desarrollar un algoritmo y una aplicación móvil para la detección y cuantificación de roya y minador en hojas de café. Las imágenes, recopiladas manualmente en una plantación de café en Brasil, comprenden 285 imágenes de roya y 257 de minador, con una resolución original de 4000x2250 píxeles para conservar la calidad visual. Las capturas se realizaron con una cámara de smartphone convencional en condiciones de laboratorio, utilizando un fondo blanco uniforme. Su integración en el proyecto tiene como objetivo diversificar los casos de roya y minador, proporcionando una perspectiva adicional para evaluar el algoritmo y la aplicación móvil propuestos. Se llevó a cabo correlaciones y comparaciones con otros conjuntos de datos, permitiendo evaluar la robustez del modelo ante variaciones en condiciones de captura y contexto.

El conjunto de datos auxiliar utilizado en este estudio está disponible para la comunidad científica y público en general. Los interesados pueden acceder a los datos a través del repositorio Mendeley en línea.

La diversidad en las fuentes de datos y cámaras utilizadas, junto con los ajustes de contraste y el redimensionamiento de imágenes a 960 píxeles que está adjunto en el Apéndice G, garantiza que el modelo funcione de manera correcta en varios dispositivos con lentes distintas. Así, estará expuesto a una amplia gama de condiciones y variaciones, lo que aumenta su capacidad para generalizar y adaptarse a diferentes entornos de captura de imágenes.

3.3 Selección y filtrado de las imágenes

El proceso de selección y filtrado de imágenes para el modelo de detección se basó en la consolidación de tres conjuntos de datos iniciales mostrados en la Tabla 2.

Tabla 2
Cantidad de imágenes antes de filtrado

DATASET PROPIO	
Hojas sanas	800 imágenes
Roya	400 imágenes
Coco	800 imágenes
DATASET ROCOLE	
Hojas sanas	791 imágenes
Roya	602 imágenes
DATASET RUST	
Roya	285 imágenes
Bicho minador	257 imágenes

3.3.1 Limpieza y consolidación

El primer paso fue crear un conjunto de datos unificado y balanceado, considerando las diferentes clases presentes en los conjuntos iniciales. Esto implicó la creación de una nueva categoría llamada "hojas" que agrupara todas las clases de hojas sanas y enfermas de los distintos Datasets. Además, se llevó a cabo un proceso de reducción en el número de imágenes de las categorías con mayor representación para lograr un equilibrio en el conjunto consolidado.

Dado que el objetivo era desarrollar un modelo que se adaptara a las condiciones típicas de un cultivo de café, se implementaron las siguientes estrategias:

- **Cambios de Iluminación:** Se seleccionaron imágenes con niveles de brillo y contraste para diferentes momentos del día, teniendo en cuenta la variación natural de la luz en un entorno de cultivo de café.
- **Cambios de Posición:** Se hicieron rotaciones aleatorias en las imágenes para simular diversos ángulos de visión, reflejando las distintas posiciones en las que las hojas pueden presentarse.
- **Cambios de Tonos de Color:** Se eligieron distintas imágenes para representar diferentes etapas de crecimiento de las plantas y variaciones estacionales.

- **Cambios de Enfoque y Calidad de Imagen:** Variaciones en el enfoque y la calidad de imagen se incorporaron para reflejar condiciones realistas, donde las imágenes pueden variar en nitidez y claridad.

3.3.2. Ampliación de la base de datos

Luego de seleccionar el conjunto de imágenes más representativas se encontró que los datos estaban claramente desbalanceados en el número de imágenes por categoría, para lo cual se hizo necesario la ampliación del Dataset para ello se tomaron varias imágenes de las diferentes categorías para ello se utilizó un script en Google Colab para aumentar un conjunto de imágenes aplicando transformaciones (estiramiento aleatorio, reflejo horizontal, reflejo vertical, rotación, cambio de luminosidad, contraste y tonos de color) aleatorias para mejorar la diversidad del conjunto de datos y, posteriormente, guarda las imágenes aumentadas en Google Drive, también, con la ayuda de la herramienta de edición Photopea se hizo zoom y recortes tomando atención a los detalles más representativos de la afectación.

3.3.3 Resultados del proceso de limpieza

Tras la implementación de estas estrategias, el conjunto de datos final se muestra en la Tabla 3.

Tabla 3
Imágenes luego del proceso de limpieza

Sanas	1592 imágenes
Coco	415 imágenes
Roya	356 imágenes
Minador	406 imágenes
Total	**2769 imágenes**

Adicionalmente a las estrategias de limpieza mencionadas previamente, fue implementado una categorización detallada agrupando las imágenes según características similares. Esta

categorización no solo facilita el análisis y comprensión del conjunto de datos, sino que también enriquece la capacidad del modelo para reconocer matices específicos, así como la facilidad que se le da a futuros investigadores que quieran consultar dicha base de datos. Las categorías se pueden observar en la Tabla 4.

Tabla 4
Clases especificadas con sus respectivas características

ROYA	**COCO**	**HOJAS**	**MINADOR**
Roya recortes	Coco recortes	Hojas del Dataset socorro	Minador fondo blanco
Roya fondo blanco	Coco muy afectada	Hojas sanas fondo variado socorro	Minador recortes
Roya muy afectada	Coco moderadamente afectada	Hojas sanas Rocole	
Roya moderadamente afectada	Coco poco afectada	Hojas recortadas con brillo	
Roya poco afectada		Hojas enfermas	

Este conjunto de datos resultante no solo intenta buscar un balance, sino que también representa una variabilidad más realista de las condiciones en un cultivo de café. La diversidad introducida asegura que el modelo esté preparado para abordar las complejidades presentes en entornos agrícolas reales.

Figura 8
Ejemplo de variabilidad de datos con los diferentes Datasets tomando la mayor cantidad posible de entornos sacados de las bases de datos Rocole e Imágenes de rust (Hemileia vastatrix) and leaf miner (Leucoptera coffeella) in coffee crop (Coffea arabica)

3.4 División de datos para entrenamiento, validación y test

Una vez consolidado y caracterizado el conjunto de datos, el siguiente paso crucial en el desarrollo de este trabajo fue la adecuada división en conjuntos de entrenamiento, validación y prueba. Puede acceder al código en el Apéndice G.

3.4.1 Proceso de división

Las imágenes fueron distribuidas de manera aleatoria en los siguientes conjuntos:

- **Conjunto de Entrenamiento (training):** Destinado a entrenar el modelo y ajustar parámetros.
- **Conjunto de Validación (validation):** Utilizado para ajustar hiperparámetros y evaluar el rendimiento durante el desarrollo del modelo.
- **Conjunto de Prueba (test):** Reservado para evaluar el rendimiento final del modelo después de haber sido completamente entrenado y validado.

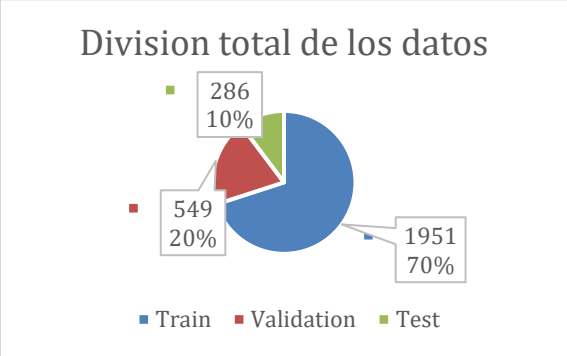
3.4.1.1 Razones para la División. La separación en conjuntos permite una evaluación sistemática del modelo en diversas etapas, desde el entrenamiento hasta la evaluación final. Esto ayuda a evitar el sobreajuste al proporcionar un conjunto de prueba independiente, garantizando que el modelo no esté adaptado excesivamente a los datos de entrenamiento. Además, la presencia de un conjunto de validación facilita la optimización de hiperparámetros, lo que contribuye a

mejorar el rendimiento del modelo. Asimismo, la categorización previa asegura una distribución equitativa de las clases en cada conjunto, eliminando sesgos y promoviendo una evaluación imparcial del modelo.

3.5 Proceso de división aleatoria

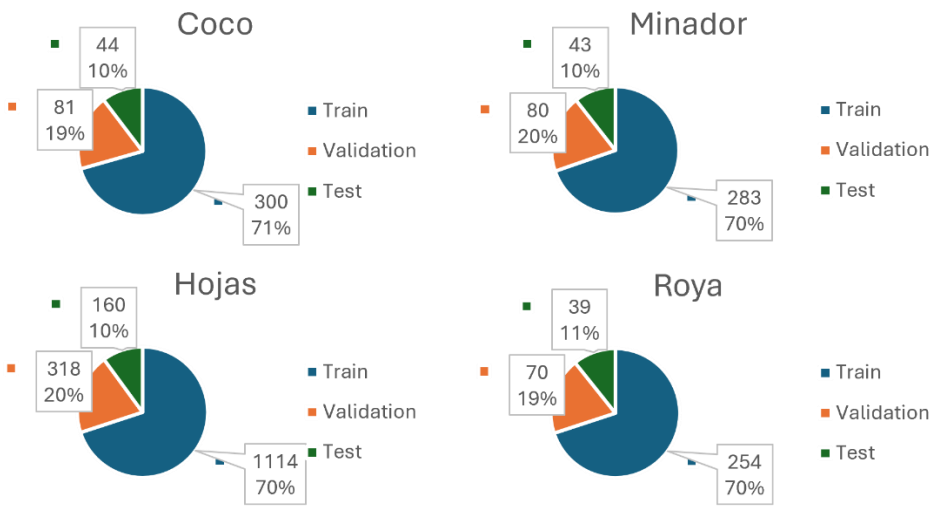
Tomando las recomendaciones de muchos trabajos sobre inteligencia artificial, re realizó una partición de 70% de los datos para entrenamiento, 20% de los datos para validación y 10% de los datos para prueba.

Figura 9
División total de datos entre Train, Test y Validation



De manera más específica los datos según la subdivisión que se hizo anteriormente se evidencian en la Figura 10.

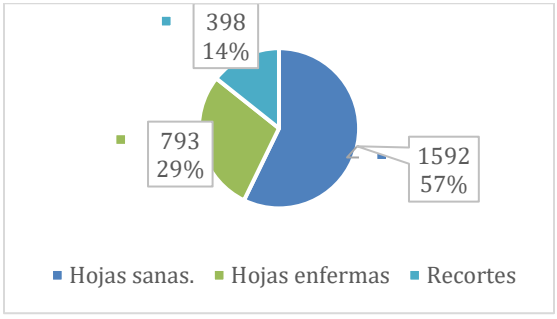
Figura 10
División de los datos de Train, Validation y Test para las categorías de coco, minador, roya y hojas



Ahora teniendo en cuenta que de manera general el dataset se dividió en 2 grandes categorías las cuales son hojas sanas y hojas enfermas (Roya, minador y coco) la división total de ellas se evidencia en la Figura 11.

Figura 11

División de los datos en las categorías Hojas Sanas, Hojas Enfermas y recortes de enfermedades



Después de realizar unas pruebas iniciales de entrenamiento del modelo, se pudo identificar un desequilibrio en el dataset. Aunque se mantenían números de imágenes similares para cada categoría de enfermedad, también es crucial equilibrar el número de etiquetas. Por lo tanto, fue necesario agregar una cantidad significativa de imágenes adicionales de la categoría "hojas" para

intentar balancear clases en medida de las limitaciones. A continuación, se muestra en la Tabla 5, el número total de etiquetas luego de equilibrarlas en número.

Tabla 5
Numero de etiquetas por categoría

Etiquetas de Entrenamiento	
Etiquetas hojas	1673
Etiqueta roya	3663
Etiquetas de coco	4027
Etiquetas de minador	1642
Etiquetas de Validación	
Etiquetas hojas	470
Etiqueta roya	996
Etiquetas de coco	1087
Etiquetas de minador	498
Etiquetas de Prueba	
Etiquetas hojas	245
Etiqueta roya	537
Etiquetas de coco	515
Etiquetas de minador	245
Totales	
Etiquetas totales de hojas	**2388**
Etiquetas totales de roya	**5196**
Etiquetas totales de coco	**5629**
Etiquetas totales de minador	**2385**
Etiquetas totales	**15598**

3.6 Etiquetado de la base de datos

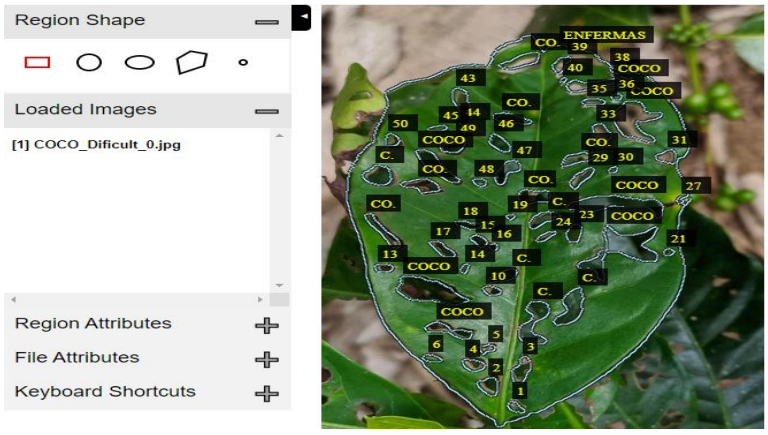
Etiquetar datos es como darle a una computadora los "ojos" para que pueda aprender y entender imágenes. En el caso de la detección de enfermedades en hojas de café, etiquetar significa decirle al modelo qué partes de las fotos son normales y cuáles tienen enfermedades.

Es crucial hacer esto bien porque la precisión de la inteligencia artificial depende de tener datos correctamente etiquetados. Si las etiquetas son confusas o incorrectas, el modelo puede aprender cosas equivocadas. Para ello se usó dos herramientas muy importantes que se explica a continuación.

3.6.1 VGG image annotator

VGG Image Annotator es una herramienta de etiquetado en línea ampliamente utilizada en proyectos de visión por computadora. Desarrollada por el Grupo de Visión por Computadora (VGG) de la Universidad de Oxford, esta plataforma proporciona una interfaz intuitiva que facilita el proceso de etiquetado de imágenes. Allí se pueden etiquetar manualmente las imágenes con la posibilidad de añadir diferentes categorías y exportar los archivos JSON con los datos del etiquetado.

Figura 12
Ejemplo de etiquetado de datos para una hoja de café afectada por "coco" usando VGG



3.6.2 SAM (Segment Anithing Model)

En una fase posterior del proceso de etiquetado de las imágenes, se incorporó SAM, una herramienta que representa un avance significativo en el campo de la segmentación de imágenes; Presentado por el equipo investigación en IA de META (Facebook) en la International Conference on Computer Vision (ICCV) en el 2023. Se trata de un sistema muy sofisticado diseñado para identificar y etiquetar objetos en imágenes y videos. Lo distintivo de SAM radica en su capacidad para realizar esta tarea con precisión y eficiencia, incluso en escenarios donde los objetos no se

encuentran previamente identificados en los datos de entrenamiento del modelo. Esto se logra gracias a su entrenamiento sobre un conjunto de datos de segmentación sin precedentes, compuesto por más de mil millones de máscaras anotadas (el Dataset más grande hasta la fecha). El repositorio facebookresearch/segment-anything es de acceso público en GitHub.

Una de las características destacadas de SAM es su versatilidad en cuanto a la interacción con el usuario. Este modelo puede ser activado mediante una variedad de instrucciones, que van desde simples clics en objetos de interés hasta descripciones textuales de las regiones a segmentar. Esta flexibilidad permite adaptar SAM a una amplia gama de aplicaciones sin necesidad de reentrenamiento.

Hay que destacar que, en el principio del proyecto, en el proceso de etiquetado de las imágenes se optó por la herramienta VGG, pero dado su carácter enteramente manual, etiquetar una sola hoja afectada con muchas manchas solía llevar hasta 20 minutos. Es por ello por lo que se adoptó la herramienta SAM utilizando un cuadernillo en Google Colab, en la cual el proceso de etiquetado con imágenes similares tomaba alrededor solo de 4 minutos y con una calidad de etiqueta superior, lo que permitió acelerar el proceso de construcción de la base de datos.

A continuación, se explica el funcionamiento de la red. Sin embargo, en el apéndice 2 está todo el código del modelo con su explicación detallada.

1 Selección de imágenes para etiquetado:

- Se elige una imagen de una carpeta en Google drive para etiquetar.
- Se especifica la carpeta y el nombre base de las imágenes.
- El sistema busca la imagen dentro del Google drive basándose en el número proporcionado y la estructura del nombre del archivo.

2 Generar cuadro delimitador:

- Permite dibujar cuadros delimitadores alrededor de objetos de interés en la imagen. En este caso las enfermedades del café.
- La imagen se codifica en base64 para ser compatible con el widget interactivo.
- Se muestra el widget al usuario, quien puede dibujar los cuadros delimitadores con el ratón.

3 Acumular máscaras:

- Segmenta las áreas de interés definidas por los cuadros delimitadores.
- Acumula las máscaras resultantes de cada región delimitada por los cuadros.
- La opción OP controla la inclusión de la última máscara en el resultado final. Es decir, que en este caso hacer que OP sea igual a uno, guarda la última máscara en la clase ‘HOJAS’ y al hacer OP igual a cero, guarda la máscara de acuerdo con la elección del usuario.

4 Escoger opciones de máscaras:

- Devuelve tres posibles opciones por región de cuadro delimitador.
- Configura el parámetro para obtener múltiples máscaras como salida para cada región delimitada.

5 Actualizar listas de máscaras:

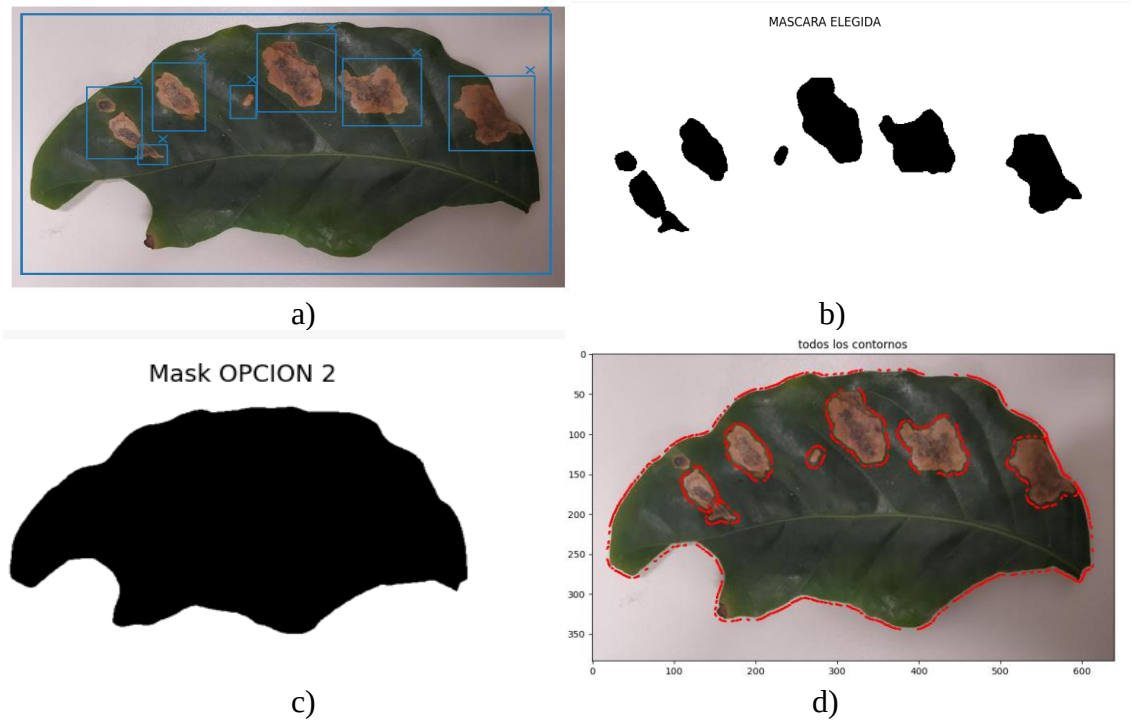
- Actualiza las máscaras asociadas con las regiones de interés según la opción elegida que más se acerque a la delimitación deseada.
- Permite controlar el proceso de actualización, como la inclusión o exclusión de ciertas máscaras en el resultado final.

6 Convertir a JSON:

- Crea un diccionario para almacenar la información necesaria para la anotación de las imágenes, el cual contiene el nombre de la imagen, su clase, y el tamaño del archivo.
- Convierte la imagen binaria a una imagen en escala de grises con valores de 0 a 255, donde el negro representa valores bajos y el blanco representa valores altos.
- Crea polígonos correspondientes a las coordenadas de las máscaras binarias utilizando la biblioteca 'CV2', para luego extraer los contornos de los polígonos dibujados y selecciona el contorno que tiene el área más grande.
- Dibuja objetos poligonales y un objeto principal que se formará alrededor del polígono para aumentar la etiqueta.
- Amplia la imagen con la ayuda de la biblioteca 'shapely geometry', que garantiza que haya un factor de ampliación de píxeles pero que cubra los bordes de la etiqueta necesarios para el entrenamiento.
- Convierte la imagen a formato JSON para su posterior uso.
- Almacena los datos en la ubicación deseada, en este caso una carpeta en Google drive.

Figura 13

Ejemplo de una imagen infectada con Minador segmentada con la herramienta de SAM, personalizada para la identificación de enfermedades en hojas de plantas. a continuación, una breve explicación de su funcionamiento: a) Widget en Python que se utiliza para cargar y mostrar imágenes, así como para realizar anotaciones de bounding boxes en ellas. b) Mascaras resultantes de la selección de los bounding boxes proporcionadas por SAM. c) Máscaras de la hoja elegida proporcionadas por SAM. d) Resultado de la etiqueta en formato JSON



3.6.3 Calidad de las etiquetas

Con la colaboración de un ingeniero agrónomo de la Federación Nacional de cafeteros de Colombia, y en conjunto con una ingeniera agroindustrial, lo que aseguró que las etiquetas individuales generadas fueran adecuadas en función de cada enfermedad abordada en el estudio.

Además, se usó la versión demo de la plataforma oficial de SAM, en particular la herramienta "everything", para discernir qué etiquetas podían segmentarse y cuáles no, lo cual resultó fundamental en el análisis.

Figura 14
Ejemplo de la calidad de las etiquetas proporcionadas por SAM

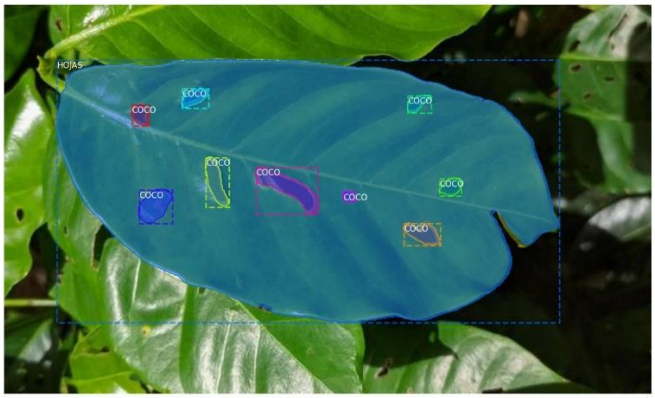


Figura 15
Ejemplo de las etiquetas de las imágenes en el dataset

3.6.4 Experiencias y Lecciones Aprendidas en la Creación de una Base de Datos

El proceso de crear una base de datos de segmentación semántica representó un desafío considerable en términos de investigación, así como de pruebas y errores. Construir una solución que funcione efectivamente en un entorno real de campo, especialmente con segmentación multiclase, presentó diversos obstáculos. Inicialmente, fueron definidas cinco clases (hojas sanas, hojas enfermas, roya, coco, minador), que posteriormente se redujeron a cuatro (hojas, roya, coco,

minador). Un error cometido fue asumir una resolución de 512 píxeles, influenciados por investigaciones previas, aunque la complejidad de esta aplicación requería una resolución mucho mayor para capturar el detalle necesario en las imágenes de las manchas de enfermedades y poder diferenciarlas adecuadamente. A través de iteraciones, llegando a la resolución actual de 960 píxeles.

El aumento en la resolución conlleva un incremento en los parámetros y en los recursos necesarios para el entrenamiento, incluyendo tiempo, dinero y poder de procesamiento. Para abordar estos desafíos, se realizaron diversas pruebas preliminares utilizando estrategias de aprendizaje para optimizar el uso de recursos, incluyendo redimensionamiento de imágenes, combinado con un Batch size de 8 y 16 imágenes por unidad de procesamiento gráfico (GPU), así como ajustes en la tasa de aprendizaje y umbrales de confianza. Se logró reducir significativamente el tiempo de entrenamiento, con épocas de solo 8 minutos para la segmentación semántica cuando los últimos entrenamientos realizados tenían duraciones de hasta 4 horas por épocas.

Durante el proceso, logró desarrollarse diversas bases de datos, experimentando con diferentes tamaños de imágenes y herramientas de etiquetado. La experiencia inicial de etiquetado manual utilizando la herramienta VGG con una resolución de 512 píxeles demostró la tediosidad y el tiempo requerido para esta tarea. Esto llevó a desarrollar una herramienta adaptada de etiquetado para agilizar dicho proceso SAM (Segment Anithing Model). Al experimentar con una resolución de 640 píxeles, se tuvo que enfrentar desafíos para ajustar las etiquetas correctamente dentro del borde interno de las manchas. Tras una investigación exhaustiva, se implementó una solución para ampliar las etiquetas y ajustarlas adecuadamente a los bordes.

Otro desafío fue equilibrar la cantidad de etiquetas por imagen, lo cual implicó métodos de separación de conjuntos de datos para asegurar una distribución adecuada de las características de las imágenes. Aunque la base de datos no está completamente equilibrada en términos generales, se hicieron los ajustes necesarios para adaptarse a limitaciones de entrenamiento, recursos computacionales, tiempo y presupuesto, obteniendo resultados muy satisfactorios.

4. Entrenamiento del modelo

En este capítulo se especifica todos los ajustes que se hicieron al modelo MASK R-CNN durante el proceso de entrenamiento.

Toda la programación del modelo fue hecha en Python usando el cuaderno en línea de Google Colab usando una maquina GPU Nvidia T4 para correr de manera efectiva el modelo de detección. T4 brinda un alto rendimiento en aplicaciones de inteligencia artificial y aprendizaje profundo. Además, se destaca por su capacidad de cómputo optimizada para cargas de trabajo variadas, de forma gratuita, con la opción de una suscripción paga para una mayor capacidad. Aunque inicialmente se pagó una suscripción de US$13 para acceder a la máquina sin limitaciones, debido a dificultades económicas se utilizó principalmente la versión gratuita, aunque hay que mencionar que cada 24 horas se interrumpe la ejecución impidiendo un entrenamiento continuo por lo que se hizo un entrenamiento segmentado.

La implementación original de MASK R-CNN fue creada por Matterport, el cual está disponible el GitHub con uso libre bajo el nombre de "Matterport/Mask_RCNN", dicha implementación en Python usa TensorFlow para detectar y segmentar instanciasen las imágenes.

Este repositorio proporciona código fuente de modelos pre entrenados y ejemplos para ayudar a usuarios con su implementación.

Se usó una versión adaptada del modelo "Mask R-CNN for Object Detection and Instance Segmentation" de Zunayed Mahmud, compatible con TensorFlow 2 y Python 3. Fue necesario adaptar el repositorio de “David Revelo Luna” “DavidReveloLuna/MaskRCNN_Video” para el entrenamiento en Google Colab. Se realizaron ajustes en la librería model.py para solucionar problemas de compatibilidad con las versiones actuales de TensorFlow y Keras con el fin de usar la versión heredada de la función SGD y se ajustó algunas líneas de código en utils.py para corregir errores de sintaxis. Además, se agregó una nueva función en visualice.py para mejorar la visualización de las detecciones y segmentaciones en las imágenes.

4.1 Configuración del entrenamiento en base a pruebas previas

Para la configuración del modelo, se realizaron varios ajustes específicos. Se empleó una GPU, configurando el parámetro "GPU_COUNT" en uno. Para optimizar el proceso de entrenamiento, se ajustó el parámetro "IMAGES_PER_GPU", probando valores de 4 y 16, aunque finalmente se utilizó 1 para entrenamientos con imágenes más grandes.

La resolución de las imágenes se definió en 960 píxeles, considerando una resolución mínima adecuada. El parámetro "STEPS_PER_EPOCH" se calculó utilizando el número total de muestras y el tamaño del lote. Para garantizar una buena precisión estadística en la validación, se configuró el parámetro "VALIDATION_STEPS" en 20.

Inicialmente, se utilizó ResNet-50 como Backbone, pero debido a dificultades en la detección de clases, se cambió a ResNet-101. La media RGB se obtuvo a partir de las imágenes de entrenamiento, resultando en [82.8, 124.3, 99.7]. Se ajustaron las tasas de aprendizaje, observando que el valor predeterminado de 0.001 proporcionaba los mejores resultados.

Para controlar la supresión no máxima, se realizaron pruebas con valores de 0.8 y 0.6 para "RPN_NMS_THRESHOLD", optando por 0.7 debido a la menor función de perdida. El valor de "MAX_GT_INSTANCES" se determinó como 78 etiquetas por imagen en el entrenamiento, y este mismo valor se asignó a "DETECTION_MAX_INSTANCES". Aunque aumentar estos valores mejoraría los resultados, se mantuvieron en su mínimo debido a restricciones computacionales.

Para los demás parámetros no mencionados, se optó por dejarlos en sus valores predeterminados, los cuales están establecidos para configuraciones generalizadas que se ajustan adecuadamente a las necesidades de entrenamiento.

4.2 Ventajas y Limitaciones

4.2.1 Ventajas de Mask R-CNN:

- Segmentación precisa de instancias incluso en objetos superpuestos.
- Arquitectura modular para fácil extensión y personalización.
- Utiliza un backbone poderoso como ResNet para extraer características de alto nivel.

4.2.2 Limitaciones de Mask R-CNN:

- Puede ser computacionalmente intensivo, afectando la velocidad de inferencia.
- Requiere conjuntos de datos grandes y anotados para un entrenamiento efectivo.
- Configuración y entrenamiento pueden ser más complejos en comparación con arquitecturas más simples.

4.3 Técnicas de regularización y callbacks

Para evitar el sobreajuste y mejorar la generalización, se aplicaron las siguientes técnicas:

- **Weight Decay (Decaimiento de Pesos):** Se utilizó la regularización L2 controlada por el hiperparámetro WEIGHT_DECAY.

- **Batch Normalization (BatchNorm):** Se implementó BatchNorm para normalizar activaciones de capas y estabilizar el entrenamiento.

También para mejorar la calidad del entrenamiento se usaron los siguientes Callbacks:

- **ModelCheckpoint:** Se guardaron puntos de control del modelo en intervalos específicos durante el entrenamiento.
- **Early Stopping:** Se detuvo el entrenamiento si el rendimiento en el conjunto de validación dejaba de mejorar.
- **ReduceLROnPlateau:** Se utilizó para ajustar dinámicamente la tasa de aprendizaje.

4.4 Optimizador utilizado

Mask R-CNN utiliza el optimizador SGD (Stochastic Gradient Descent) para la optimización del modelo durante el entrenamiento.

4.5 Procedimientos de entrenamiento

Se realizaron tres tipos diferentes de entrenamientos, guardando puntos de control en cada etapa. En este proceso, el siguiente punto de control cargaba los pesos de entrenamiento del anterior y se ajustaban las configuraciones con el propósito de realizar correcciones basadas en las inferencias de los resultados y ajustes derivados de las observaciones. Los modelos de entrenamiento se encuentran disponibles en el Apéndice N.

4.5.1 Punto de control 1

Cantidad de imágenes de entrenamiento: 1934 y tasa de aprendizaje (lr): 0.001. Se aplicó Transfer Learning utilizando COCO de MASK C-RNN. Los datos proporcionados sugieren que el modelo se entrenó de manera efectiva y generalizó bien a datos no vistos. Se seleccionó el mejor modelo de este entrenamiento tras aplicar Callbacks en el momento en que se empezaba a observar sobreajuste; Sin embargo, al analizar las detecciones en la inferencia del modelo, se identificó una

buena detección de clases, pero se observaron numerosas instancias de falsos positivos debido al fondo y a la sobre iluminación en las hojas, consecuencia de la variabilidad ambiental en donde se tomaron las fotos. La detección de roya y minador fue bastante precisa, pero se presentaron numerosos falsos positivos. La clase "coco" fue la que tuvo peor rendimiento en cuanto a detección de clases. Las configuraciones iniciales se evidencian en la Tabla 6.

Tabla 6
Configuraciones iniciales del punto de control 1

CONFIGURACIÓN	VALOR
STEPS_PER_EPOCH	1934
LEARNING_RATE	0.001
RPN_NMS_THRESHOLD	0.7
DETECTION_MIN_CONFIDENCE	0.7
DETECTION_NMS_THRESHOLD	0.3
ROI_POSITIVE_RATIO	0.3
WEIGHT_DECAY	0.0001

Las correcciones aplicadas fueron las siguientes:

- Corrección y eliminación de etiquetas que confundían al modelo.
- Aumento de imágenes de interés mediante técnicas de aumento de datos (rotación, recortes y ajustes de contraste) para la clase "coco" con 10 imágenes y para la roya con 7 imágenes.
- Modificación de STEPS_PER_EPOCH.

4.5.2 Punto de control 2

Aplicando transfer learning del modelo anterior, se realizaron cambios en los parámetros y se ejecutó el entrenamiento con las siguientes configuraciones evidenciadas en la Tabla 7

Tabla 7
Configuraciones iniciales del punto de control 2

CONFIGURACIÓN	VALOR
STEPS_PER_EPOCH	1951

LEARNING_RATE	0.001
RPN_NMS_THRESHOLD	0.7
DETECTION_MIN_CONFIDENCE	0.74
DETECTION_NMS_THRESHOLD	0.3
ROI_POSITIVE_RATIO	0.33
WEIGHT_DECAY	0.0001

Se observó una mejora significativa en los resultados, aunque aún persistían problemas con los falsos positivos y el sobreajuste. En consecuencia, se decidieron las siguientes acciones:

- Aplicación de aumento de imágenes mediante recortes en la región donde se presentaban los falsos positivos, agregando 40 imágenes.
- Aumento del parámetro WEIGHT_DECAY para abordar el sobreajuste, penalizando los pesos.
- Incremento del umbral DETECTION_MIN_CONFIDENCE.
- Aumento del valor de DETECTION_NMS_THRESHOLD.
- Disminución de ROI_POSITIVE_RATIO.

4.5.3 Punto de control 3

Cantidad de imágenes de entrenamiento: 1989 y Tasa de aprendizaje (lr): 5.0000e-04. Se ajustaron las siguientes configuraciones evidenciadas en la Tabla 8.

Tabla 8
Configuraciones iniciales del punto de control 3

CONFIGURACIÓN	VALOR
STEPS_PER_EPOCH	1989
LEARNING_RATE	0.0005
RPN_NMS_THRESHOLD	0.7
DETECTION_MIN_CONFIDENCE	0.8
DETECTION_NMS_THRESHOLD	0.5
ROI_POSITIVE_RATIO	0.3
WEIGHT_DECAY	0.0005

Los resultados fueron muy satisfactorios al elevar el umbral de inferencia al 80%. Sin embargo, persistían problemas con algunos falsos positivos de roya, por lo que se consideró necesario realizar otro entrenamiento. En este sentido, se añadieron 141 recortes con el objetivo de enseñar al modelo a distinguir entre los reflejos de las hojas y la clase “roya”. También se observó que, para imágenes con muchas instancias o afectaciones, el modelo ignoraba ciertas instancias, lo que indicaba que podría mejorar si se incrementaba la cantidad de regiones propuestas.

5. Resultados y evaluación

En esta sección, se presentan los resultados obtenidos después del entrenamiento del modelo Mask R-CNN más eficiente que se hizo comparándolo con otros modelos previos y se detallarán las evaluaciones realizadas para medir su eficacia en la detección y segmentación de instancias en el conjunto de datos.

5.1 Análisis de resultados.

Figura 16
Comparación de las funciones de costo de los distintos entrenamientos

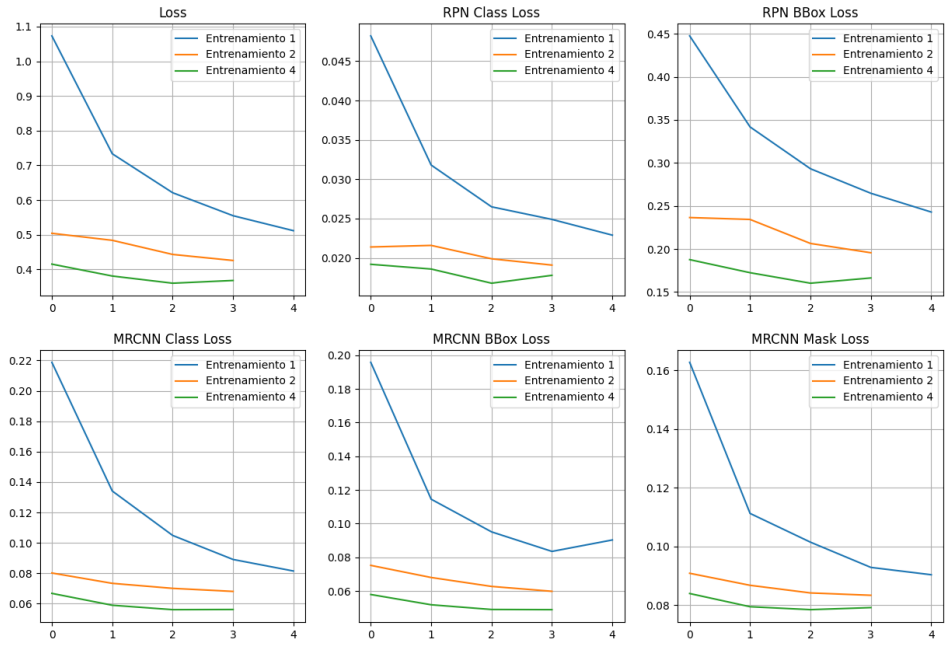


Figura 17
Comparación de las funciones de costo de las distintas validaciones

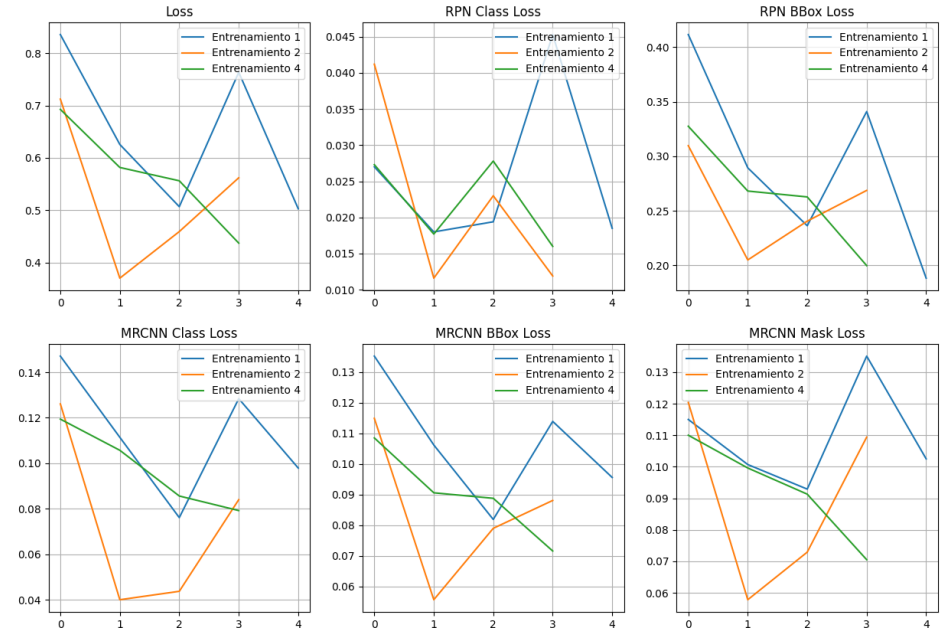


Figura 18
Inferencias en las hojas sobre el conjunto de validación

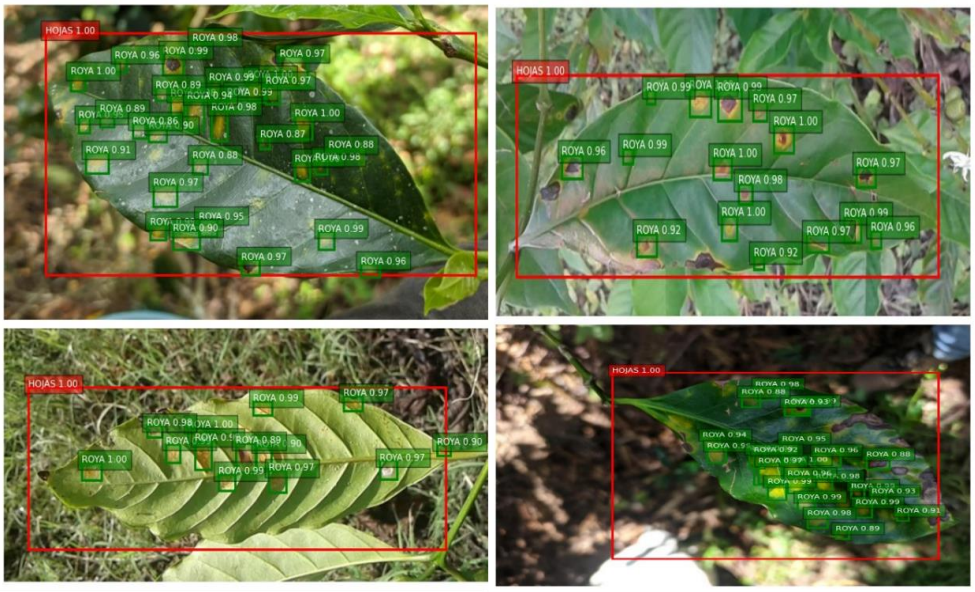


Figura 19
Superposición de máscaras para cada clase que se encuentre en el área de la hoja

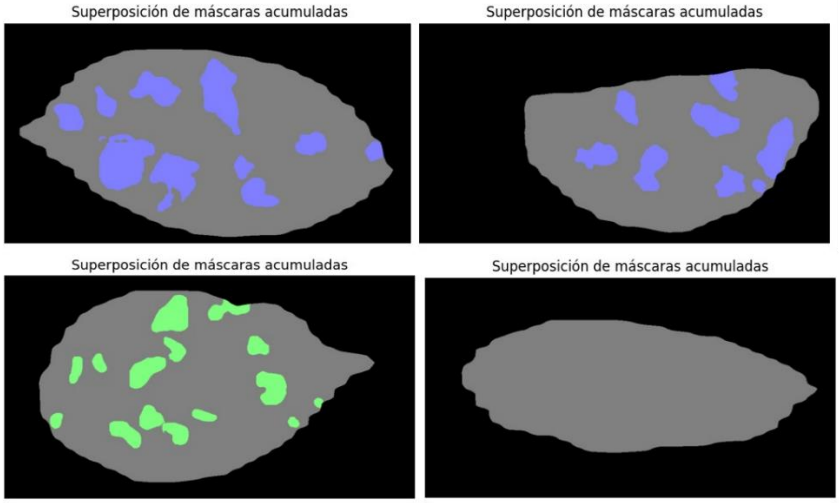


Figura 20
Clasificación, detección y segmentación

Ya hecho el entrenamiento con las respectivas muestras de cómo se comporta el modelo se calcula el Índice de Intersección sobre Unión (IoU) para cada imagen, este se calcula entre las máscaras predichas y las máscaras reales.

5.2 Resultados de validación IOU

5.2.1 Validación

Los resultados de la fase de entrenamiento del modelo con IOU fueron analizados mediante la observación de gráficas de histogramas y Boxplots (representación visual de la dispersión y la simetría de los datos) ubicadas desde la Figura 22 hasta la Figura 26, así como la observación de patrones visuales en las imágenes de inferencia, detección y segmentación semántica. Todo ello se adjunta en el Apéndice J.

Figura 21
Histograma de cada clase en validación con falsos positivos

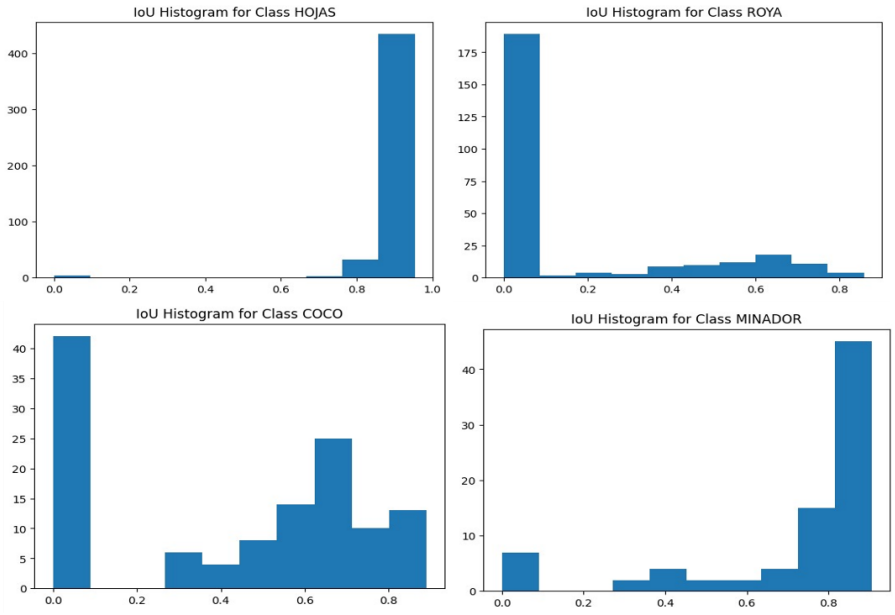


Figura 22

Dispersión y distribución de los valores IOU con falsos positivos (Boxplots de IOU)

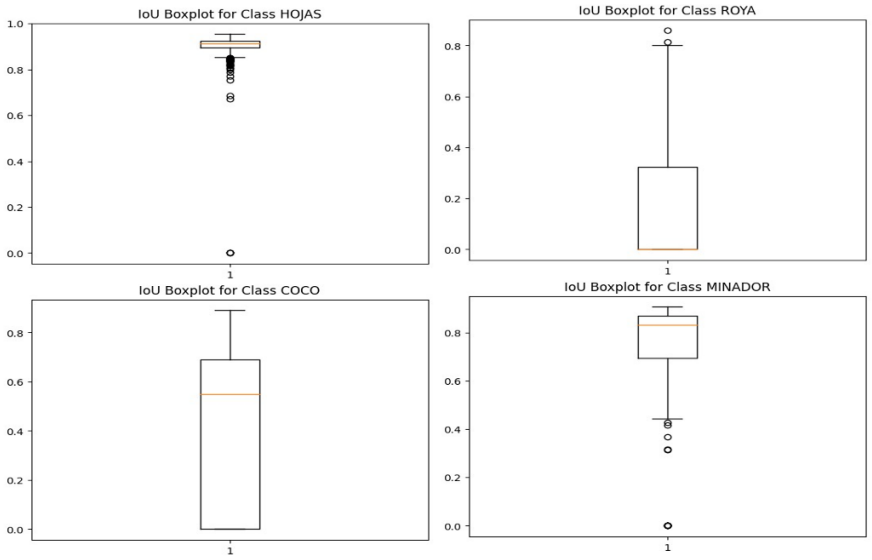


Tabla 9

Resumen de los datos de IOU promedio y desviación estándar de los datos de validación con falsos positivos y negativos

Clase	Evaluaciones	IOU promedio	Desviación estándar
Hojas	474	0.8973	0.0885
Roya	262	0.1534	0.2622
Coco	122	0.4170	0.3235
Minador	74	0.7807	0.1480

Los altos valores de la métrica IOU para los falsos positivos y falsos negativos se deben al enfoque de entrenamiento del modelo, el cual se centró en la detección y segmentación de enfermedades dentro del área de la hoja de café. Sin embargo, la resolución, la limitación de las regiones propuestas por el modelo, junto con la calidad de las imágenes en la base de datos, presentaron desafíos significativos. Esto llevó a dificultades para que el modelo distinga entre las manchas amarillas y blancas del fondo en entornos reales. El bajo índice de intersección se debió a que la métrica se desarrolló acumulando todas las detecciones por clase en una sola máscara por imagen y luego superponiéndola con la máscara real, aplicando el cálculo de IoU. En este proceso, la métrica no tuvo en cuenta completamente la multiplicidad de instancias detectadas dentro de una misma clase. Esta metodología resultó en una acumulación de instancias de una misma clase, lo que llevó a una evaluación alterada de los resultados.

Figura 23

Histograma de cada clase en validación sin falsos positivos y negativos

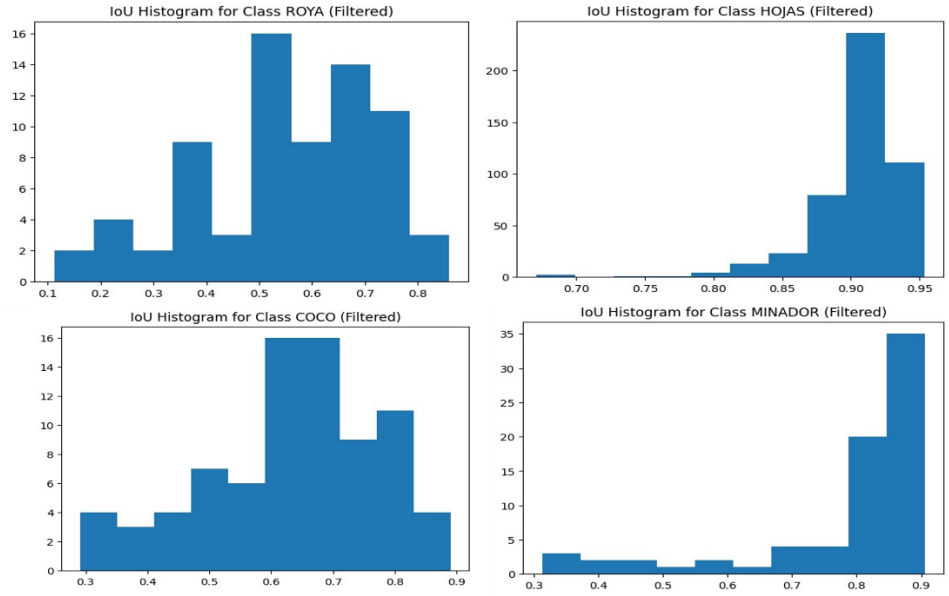


Figura 24

Dispersión y distribución de los valores IOU sin falsos positivos. (Boxplots de IOU)

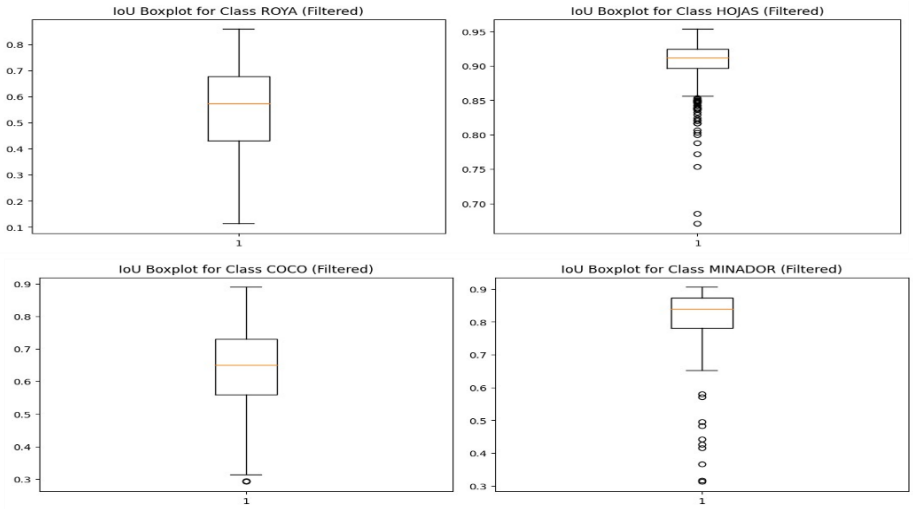


Tabla 10

Resumen de los datos de IOU promedio y desviación estándar de los datos de validación sin falsos positivos

Clase	Evaluaciones	IOU promedio	Desviación estándar
Hojas	474	0.8974	0.0887
Roya	73	0.5494	0.1712
Coco	80	0.6360	0.1427
Minador	74	0.7807	0.1480

5.2.2 Test

Figura 25

Histograma de cada clase en el test

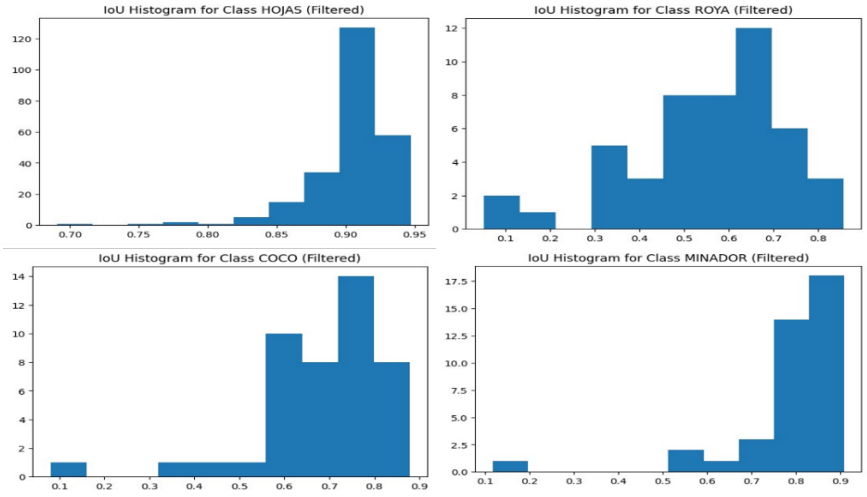


Figura 26
Dispersión y distribución de los valores IOU en el test (Boxplots de IOU)

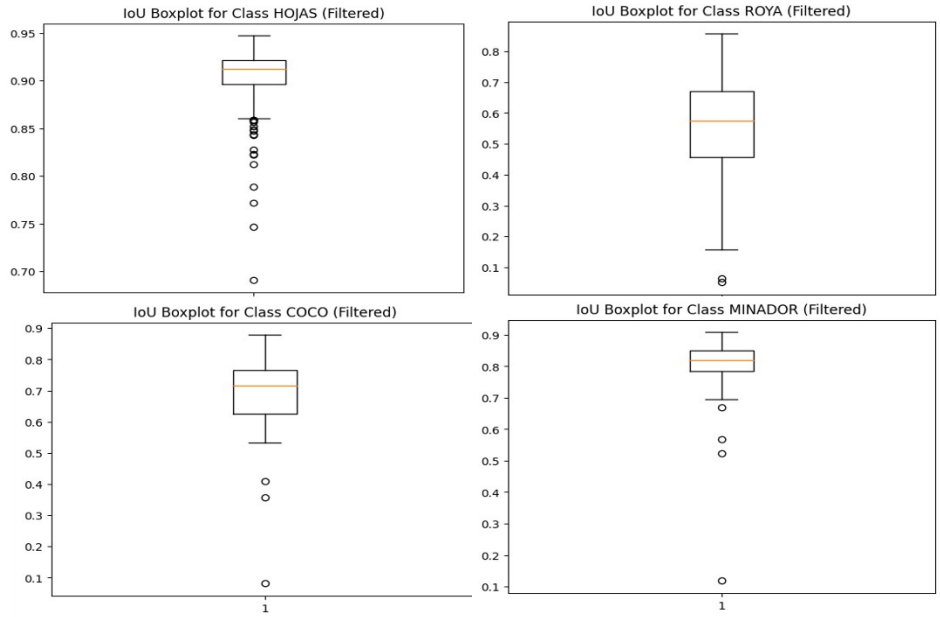


Tabla 11
Resumen de los datos de IOU promedio y desviación estándar de los datos de test

Clase	Evaluaciones	IOU promedio	Desviación estándar
Hojas	244	0.9038	0.00310
Roya	48	0.5496	0.1775
Coco	44	0.6885	0.1485
Minador	39	0.7853	0.1334

Excluyendo los falsos positivos y negativos, principalmente aquellos que pertenecían al área fuera de la hoja se espera un resultado más cercano a la realidad. El modelo se enfoca en la detección dentro de la hoja, lo que arroja mejores resultados, especialmente en la clase roya. Esta última, presenta mayores dificultades debido al entorno complejo en el que se encuentra y al bajo nivel de detalle de las manchas en las fotos, atribuido a la resolución. La segmentación más destacada en ambos casos fue de la hoja, ya que el modelo pudo ignorar hojas del fondo o

incompletas. Se observó que el modelo genera mejores instancias cuando la hoja afectada está en primer plano en la foto.

A pesar de obtener buenos resultados, la clase minador presentó problemas en entornos reales debido a la escasez de imágenes variadas en la base de datos. En el caso de la clase coco, aunque también obtuvo buenos resultados, existe la posibilidad de confundirla con otras perforaciones. Esta situación podría abordarse aumentando la resolución de las imágenes de entrenamiento, sin embargo, esto requeriría un mayor gasto computacional, económico y de tiempo.

6. Implementación en una aplicación móvil

6.1 Introducción a App Inventor

App Inventor es una plataforma web de desarrollo de aplicaciones móviles creada por el MIT, destacada por su enfoque visual basado en bloques, por lo que es muy fácil de aprender sin tener muchos conocimientos en el desarrollo móvil, esta web simplifica el proceso de creación y facilita la ejecución de proyectos simples. Así como su fácil integración con servicios externos como Google Cloud.

6.2 Diseño de la aplicación en App Inventor

El diseño de la aplicación se centró en la selección de una imagen desde la galería del teléfono o directamente desde la cámara de este, luego enviarla a un servidor en Google cloud donde se encuentra alojada una máquina virtual de Ubuntu, en la cual dicha imagen pueda ser procesada, analizada y luego con los resultados del análisis se envía de nuevo a la aplicación donde el usuario va a poder observar las inferencias que el modelo hizo sobre su imagen.

También se decidió crear dentro de la aplicación diferentes menús desplegables donde el usuario podrá interactuar con la aplicación, uno de ellos es el apartado de "¿qué hacer?" en el que presenta al usuario las 3 enfermedades (roya, minador y coco) que se puede clasificar, se enseña sus causas, consecuencias dentro del cultivo y reducciones en la producción, así como posibles tratamientos. Otro menú implementado fue la posibilidad de alojar las imágenes en una carpeta de Google drive y por último se agregó la pestaña de "contacto" donde se encuentra el correo electrónico para posibles quejas sugerencias y reclamos.

Figura 27

Captura de pantalla dentro de aplicación inventor durante el proceso de creación de la aplicación llamada "decafia app"

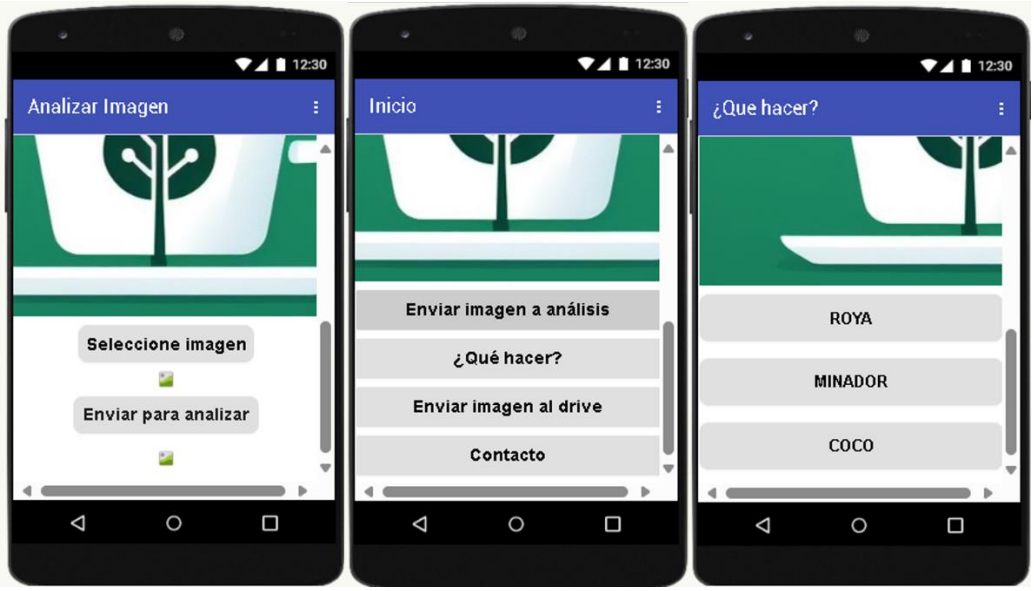
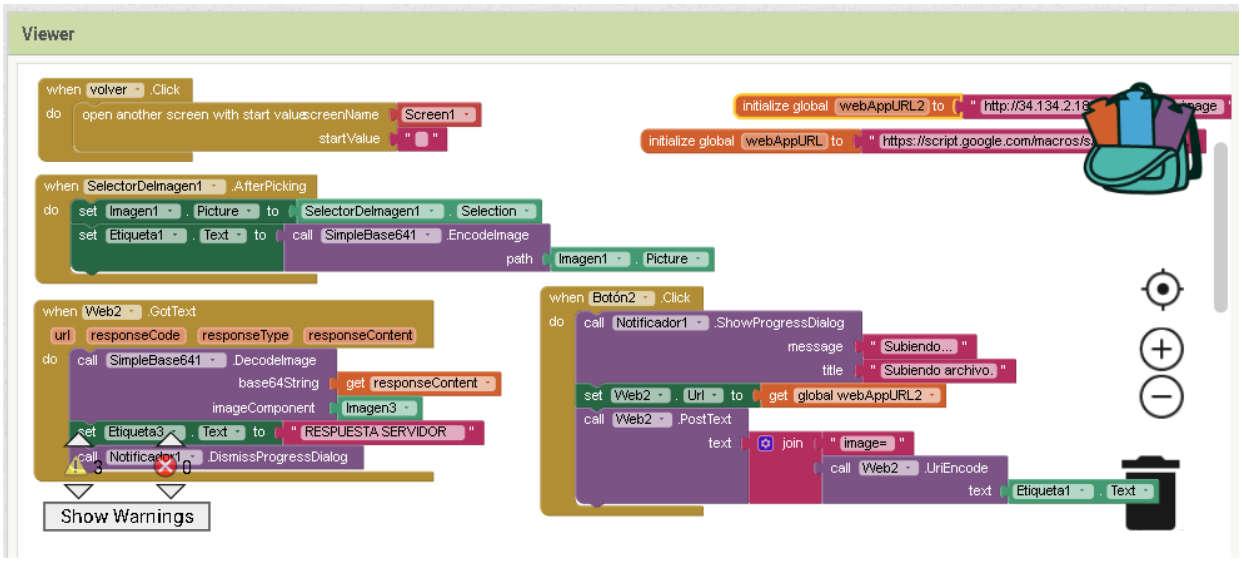


Figura 28

Captura de pantalla dentro de app inventor para la conexión de la aplicación con la máquina virtual de python



6.3 Integración con flask en Python

En la creación de la aplicación, un apartado fundamental es la comunicación de esta con el servidor web donde están alojados los datos para la inferencia del modelo Mask R-CNN. Para ello se usó Flask, que permite desarrollar una API web que recibe las imágenes de las hojas de café enviadas desde la aplicación, donde el modelo hará las inferencias adecuadas. Una vez que la API ha procesado las imágenes, responde a la aplicación móvil con los resultados del análisis.

6.4 Despliegue en Google Cloud Engine

Se usó Google cloud Engine para crear un servidor remoto basado en web que es alojado localmente en el estado de Iowa en Estados Unidos, para el cual se configuró de la siguiente manera:

- Máquina de tipo C2D, especializada en procesamiento intensivo con 8 CPU virtuales, 4 núcleos y 16GB de memoria.
- Se creó una instancia en Ubuntu versión 20240307b.
- 60GB de disco de arranque.

La máquina tipo C2D sirve como el entorno de ejecución para la aplicación, lo que proporciona los recursos necesarios para su funcionamiento. Luego se configuró el entorno para alojar la aplicación, esto implicó la instalación de dependencias y bibliotecas necesarias de Python, así como la configuración de variables de entorno y otros ajustes de configuración. Una vez configurado el entorno, se procedió a alojar el modelo de inferencias que se había previamente entrenado. Todos los archivos para el funcionamiento dentro de la maquina aparecen en Apéndice F.

Durante el proceso de despliegue, es importante tener en cuenta la configuración de firewall para garantizar la seguridad de la aplicación.

Figura 29

Captura de pantalla de la máquina virtual usada llamada "instancia-decafia"

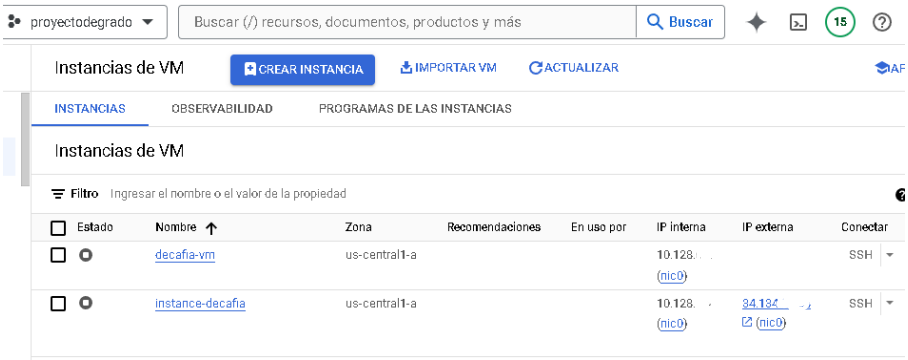
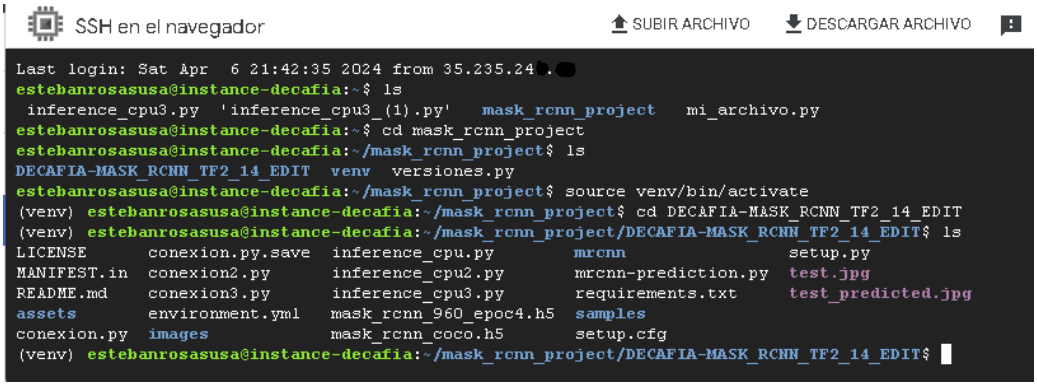


Figura 30

Captura de pantalla dentro de la máquina virtual con sus respectivos archivos



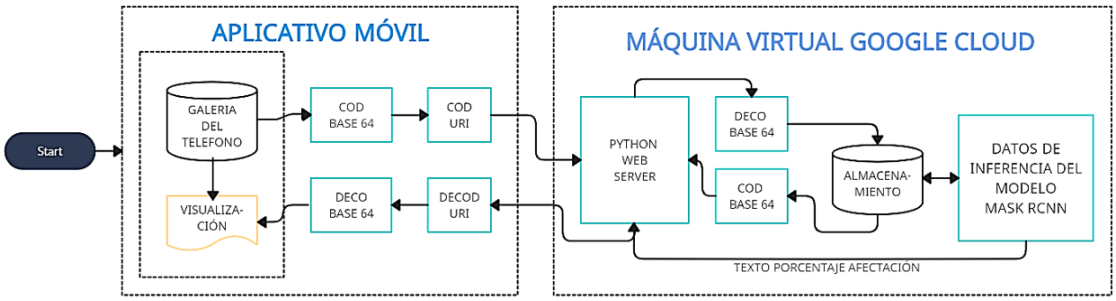
6.5 Conexión de la aplicación

La conexión de la aplicación desarrollada en App Inventor con el servidor Flask desplegado en Google Cloud Engine se realiza a través de solicitudes HTTP. Cuando el usuario interactúa con la aplicación, seleccionándola desde la galería, la aplicación envía una solicitud HTTP al servidor Flask.

Esta solicitud contiene la imagen de la hoja de café, que es recibida por la API Flask. La API procesa la imagen utilizando el modelo de inteligencia artificial para identificar posibles enfermedades, luego filtra excluyendo las etiquetas fuera del área de la hoja y genera una respuesta que incluye los resultados del análisis. Una vez que el servidor Flask ha generado la respuesta, esta se envía de vuelta a la aplicación a través de una respuesta HTTP. La aplicación móvil recibe los resultados del análisis y los presenta al usuario de manera clara y comprensible.

El código para realizar este proceso consta de un Framework Flask para construir una aplicación web que maneja solicitudes Post y Get en diferentes rutas; al recibir una solicitud Post con una imagen codificada en base 64, la aplicación la decodifica y luego la guarda en el sistema de archivos y ejecuta un script externo llamando al archivo “infernces_cpu4.py” que realiza inferencias del modelo con imágenes redimensionadas a un máximo de 960 pixeles. Luego devuelve resultado como una imagen codificada en base 64, también, proporciona rutas para enviar imágenes que han sido almacenadas para obtener el contenido de un archivo texto que tiene la información del área de afectación.

Figura 31
Diagrama del proceso de captura, envío e inferencia de imágenes

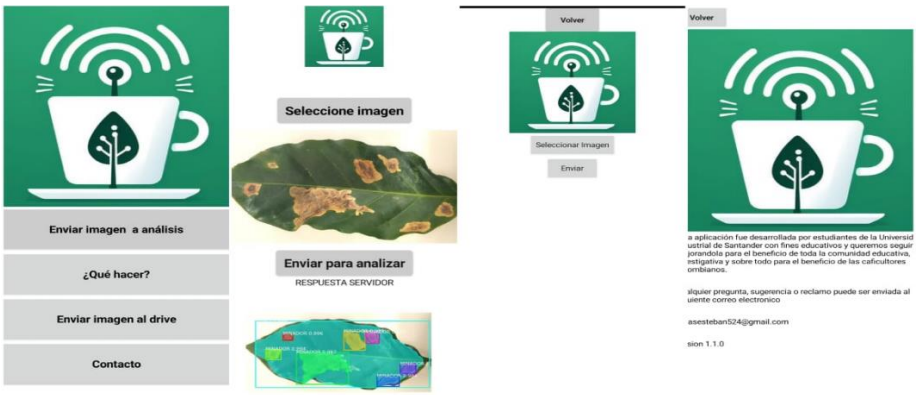


6.6 Pruebas y validación

Luego de realizar el diseño de la aplicación móvil, y su respectiva conexión con el servidor ubicado en la máquina virtual donde se aloja el modelo que hace las interferencias de las imágenes, se realizaron múltiples pruebas enviando una amplia variedad de imágenes representativas de las tres clases de enfermedad de enfermedades, donde se pudo comprobar el correcto funcionamiento de todo el sistema que se había diseñado.

Figura 32

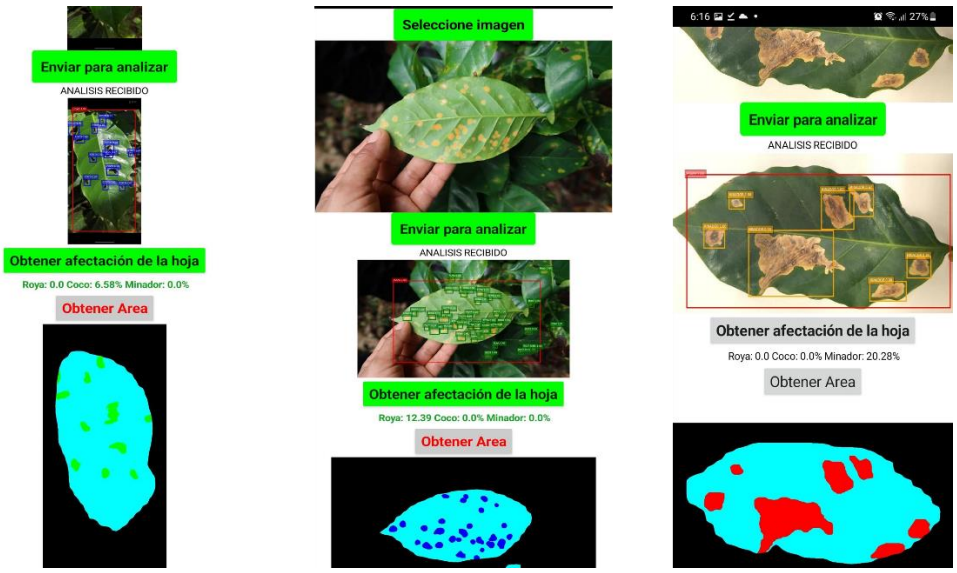
Capturas de pantalla de los diferentes menús de la aplicación móvil



Después de una mejora del diseño de la aplicación, se pudo constatar que funciona correctamente, pudiendo leer no solo el área de afectación y detección, sino también observar la máscara resultante luego del filtrado.

Figura 33

Capturas de pantalla de la aplicación en funcionamiento y posterior inferencia



7. Conclusiones

La presente investigación ha sido un esfuerzo significativo en múltiples frentes, abordando desafíos técnicos, académicos y de infraestructura con el objetivo de desarrollar una solución innovadora para la detección y segmentación de enfermedades en hojas de café. A lo largo del proceso, se han identificado y superado diversos obstáculos, cada uno de los cuales ha contribuido al crecimiento y aprendizaje del equipo investigador.

Una de las primeras dificultades encontradas fue la carencia de bases de datos existentes para la segmentación semántica, solo se encontraron para detección de clases que no eran compatibles con el enfoque del problema. Así que uno de los propósitos de este proyecto fue aportar a la comunidad una base de datos con buena segmentación y de libre acceso al público cuyo link se Encuentra adjunto en el Apéndice K

La formación intensiva en Deep Learning y Python surgió como una necesidad en este proyecto para adquirir habilidades cruciales, abordando con éxito la detección y segmentación de

enfermedades en hojas de café, y dado que la universidad no ofrece una formación suficiente en la carrera de pregrado en Python, fue necesario recurrir a un aprendizaje autodidacta, donde se hizo uso de conceptos y estrategias innovadoras para dar cumplimiento a los objetivos.

Los retos técnicos y las limitaciones de recursos representaron una barrera significativa en el desarrollo del proyecto, la potencia de procesamiento de la máquina de Google Colab fue clave, pero surgieron desafíos técnicos, uno de ellos fue la memoria RAM de la GPU, pues restringía utilizar Batches de manera efectiva para agilizar el entrenamiento debido principalmente a la resolución y al gran número de imágenes e instancias; Lo que, sumado a tiempos de entrenamiento muy prolongados, limitaron el uso de imágenes con mayor resolución para obtener un mejor resultado. Así mismo se usó una versión de pago que incrementa el tiempo de entrenamiento con menos interrupciones, sin embargo, por limitaciones económicas no se pudo hacer un uso intensivo de esta alternativa.

La experiencia en campo resultó invaluable para mejorar la calidad de los datos, especialmente para la clase de enfermedades estudiadas. El contacto directo con las condiciones reales permitió una mejor comprensión de los patrones de enfermedad en las hojas de café, enriqueciendo de este modo el proceso de etiquetado y entrenamiento del modelo.

SAM surgió como un gran descubrimiento en el etiquetado, esta innovadora herramienta que lo segmenta todo fue de gran ayuda pues al adaptarla a las necesidades del proyecto en un cuaderno de Google Colab disminuyó en gran medida los tiempos del proceso de etiquetado en segmentación de máscaras; el trabajo de un mes de etiquetado se reduce a unos pocos días.

Dada la naturaleza altamente específica y personalizada de la arquitectura Mask R-CNN, se enfrentaron diversos desafíos al ajustar los numerosos parámetros necesarios para adaptarla a las necesidades de nuestro proyecto. Este proceso de adaptación resultó ser más prolongado de lo

inicialmente anticipado. Además, aparecieron obstáculos adicionales al emplear la biblioteca Matterport específica para TensorFlow 2 y Keras 3, la cual, lamentablemente, carecía de actualizaciones regulares. Por ende, uno de los logros de este proyecto es la provisión de bibliotecas actualizadas hasta la fecha, lo que facilita el entrenamiento dentro del entorno de Google Colab y en un entorno Python más amplio. Se puede acceder al repositorio en el Apéndice E.

Configurar un servidor supuso un gran desafío debido a la falta de herramientas computacionales adecuadas para realizar inferencias del modelo. Se exploraron varias alternativas, como la creación de un entorno virtual con Anaconda, pero no se contaba con una GPU disponible con tarjeta Nvidia optimizada con CUDA. Se intentó obtener acceso a una computadora con GPU sin éxito. Finalmente, se optó por una máquina virtual en Google Cloud, aprovechando una versión de prueba utilizando un crédito único de $300 US. Aunque la versión de prueba no permitía el uso de GPU, se eligió una máquina virtual con CPU de propósito general enfocada en procesamiento. Esta configuración proporcionó una velocidad de inferencia de aproximadamente 13 segundos.

Finalmente, el proyecto culminó con éxito en la creación de una aplicación funcional que se encuentra en el Apéndice C y ofrece una interfaz bastante intuitiva para el análisis de enfermedades en las hojas de café. Este logro representa el resultado tangible de meses de mucho trabajo y colaboración.

La métrica IOU mostró altos valores para los falsos positivos y falsos negativos, debido al enfoque de entrenamiento, centrado principalmente en la detección de enfermedades en el área de la hoja de café. Sin embargo, la resolución y calidad de las imágenes presentaron complicaciones, dificultando al modelo distinguir entre brillo excesivo, manchas blancas y amarillas del fondo en entornos reales, La acumulación de instancias por clases, resultado de la metodología de

evaluación de la métrica, contribuyó a esta problemática. A pesar de ello, a partir de un filtro se logró enfocar solo en el área dentro de la hoja afectada, dando buenos resultados para la clase roya.

Por otro lado, la escasez de imágenes variadas en la clase minador podría plantear desafíos en entornos reales, mientras que la clase coco podría confundirse con otras afectaciones de tipo similar en la hoja, para lo que se sugiere una mejora en la resolución de las imágenes de entrenamiento, aunque esto implicaría costos adicionales en términos computacionales, económicos y de tiempo.

Gracias a la ayuda y asistencia de un ingeniero experimentado en el campo que es miembro del comité nacional de cafeteros y la colaboración de una ingeniera agroindustrial en el proceso de etiquetado, hemos adquirido experiencia crucial en el reconocimiento y etiquetado para tres enfermedades específicas del café. Esta colaboración ha fortalecido nuestro conocimiento y habilidades en este ámbito.

8. Recomendaciones

Como recomendación para futuras mejoras del proyecto, considerar el uso de Flutter o Android estudio en cuanto al desarrollo de la aplicación móvil, ya que ambas herramientas ofrecen opciones avanzadas y flexibilidad para crear aplicaciones móviles de alto rendimiento y con una interfaz de usuario mucho más atractiva y conveniente. Aunque nuestro proyecto se fundamentó en la actualización de un repositorio GitHub de acceso libre, una opción más sólida es utilizar plantillas basadas en Pythorch ya que cuentan con el respaldo de una amplia comunidad de desarrolladores, lo que garantiza soporte técnico y una evolución continua del proyecto. Para el tema de poca accesibilidad computacional, recomendaría encarecidamente a nuestra institución considerar la implementación de un servidor dedicado para servicios de inferencia, como la prevención de enfermedades en plantas o el diagnóstico médico. Es una inversión estratégica y

rentable. Aunque el retorno de la inversión puede no ser inmediato en términos de ganancias monetarias, los beneficios a largo plazo en desarrollo estudiantil, innovación, investigación y reputación institucional pueden superar los costos asociados. La opción de utilizar Google Cloud Platform se presenta como una solución a corto plazo. La universidad podría ofrecer a los estudiantes acceso o créditos para utilizar máquinas virtuales en Google Cloud. Esto les permitiría desarrollar una amplia gama de aplicaciones basadas en microservicios, aprovechando la flexibilidad y escalabilidad que ofrece la infraestructura en la nube de Google. Esta recomendación preparará a los estudiantes para enfrentar los desafíos del mundo laboral moderno, donde las habilidades en la nube son cada vez más valiosas.

Referencias Bibliográficas

Abril, R. R. (2023, August 27). Segmentación de Instancia I: Mask R-CNN. La Máquina Oráculo. https://lamaquinaoraculo.com/deep-learning/segmentacion-de-instancia-mask-r-cnn/.

Avelino, J., Cristancho, M., Georgiou, S., Imbach, P., Aguilar, L., Bornemann, G., ... & Anzueto, F. (2015). The coffee rust crises in Colombia and Central America (2008–2013): impacts, plausible causes, and proposed solutions. Food Security, 7(2), 303-321.

Boshier, D. H., Gómez, A. C. R., Chaves-Barrantes, N., Estrada, C., & Stevenson, P. R. (2018). Coffee cultivation does not prevent tree regeneration in a neotropical rainforest hotspot.

Brito Silva, Lucas; Cavalcante Carneiro, Álvaro Leandro; Silveira Almeida Renaud Faulin, Marisa (2020), "rust (Hemileia vastatrix) and leaf miner (Leucoptera coffeella) in coffee crop (Coffea arabica).", Mendeley Data, V5, doi: 10.17632/vfxf4trtcg.5.

Carneiro, A. L. C., Silva, L. D. B., & Faulin, M. S. A. R. (2021). Artificial intelligence for detection and quantification of rust and leaf miner in coffee crop. arXiv preprint arXiv:2103.11241.

Facebook Research. (s. f.). segment-anything: The repository provides code for running inference with the SegmentAnything Model (SAM), links for downloading the trained model checkpoints, and example notebooks that show how to use the model [Repositorio de código]. GitHub. https://github.com/facebookresearch/segment-anything.git.

Federación Nacional de Cafeteros de Colombia. (2019). Ensayos sobre Economía Cafetera No. 30 [PDF]. Recuperado de

https://federaciondecafeteros.org/app/uploads/2019/12/Econom%C3%ADa-Cafetera-No.-30_Web.pdf.

Flórez, C. P., Quiroga-Cardona, J., & Arias, J. C. (2021). Variedades del Café. En Centro Nacional de Investigaciones de Café, Guía más agronomía, más productividad, más calidad (3a ed., pp. 11–29). Cenicafé. https://doi.org/10.38141/10791/0014_1.

Gaitán, Á., Rivillas, C. A., Castro Caicedo, B. L., & Cristancho Ardila, M. A. (2013). Manejo integrado de enfermedades. En Federación Nacional de Cafeteros de Colombia, Manual del cafetero colombiano: Investigación y tecnología para la sostenibilidad de la caficultura (Vol. 2, pp. 143–178). Cenicafé; 22. https://doi.org/10.38141/cenbook-0026_22.

Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). Deep Learning (Vol. 1). MIT press Cambridge.

He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask R-CNN. En Proceedings of the IEEE international conference on computer vision (pp. 2961-2969).

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

Hosna, A., Merry, E., Gyalmo, J., Alom, Z., Aung, Z., & Azim, M. A. (2022, October 22). Transfer learning: a friendly introduction. Journal of Big Data. https://doi.org/10.1186/s40537-022-00652-w.

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W.-Y., Dollar, P., & Girshick, R. (2023). "Segment Anything." En Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (pp. 4015-4026).

Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., ... & Zitnick, C. L. (2014). Microsoft. COCO: Common objects in context. En European conference on computervision (pp. 740-755). Springer.

Long, J., Shelhamer, E., & Darrell, T. (2015). Fully Convolutional Networks for Semantic Segmentation. En Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 3431-3440.

Mahmud, Z. (s. f.). Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow 2.14.0 and Python 3.10.12 [Repositorio de Código]. GitHub. https://github.com/z-mahmud22/Mask-RCNN_TF2.14.0.

Matterport. (s. f.). Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow [Repositorio de Código]. GitHub. https://github.com/matterport/Mask_RCNN.

Parraga-Alava, Jorge; Cusme, Kevin; Loor, Angélica; Santander, Esneider (2019), "RoCoLe: A robusta coffee leaf images dataset", Mendeley Data, V2, doi: 10.17632/c5yvn32dzg.2

Perfecto, I., Vandermeer, J., & Philpott, S. M. (2014). Complex ecological interactions in the coffee agroecosystem. Annual Review of Ecology, Evolution, and Systematics, 45, 137-158.

Revelo Luna, D. (s. f.). Segmentación semántica usando MaskRCNN en imágenes y video [Repositorio de código]. GitHub. https://github.com/DavidReveloLuna/MaskRCNN_Video.git.

Ronneberger, O., Fischer, P., & Brox, T. (2015, January 1). U-Net: Convolutional Networks for Biomedical Image Segmentation. Lecture Notes in Computer Science. https://doi.org/10.1007/978-3-319-24574-4_28.

Toka, V. (2023, September 5). Instance Segmentation - MaskRCNN. https://www.linkedin.com/pulse/instance-segmentation-maskrcnn-veeranjaneyulu-toka

Virginio, E., & Astorga, C. (2015). Prevención y control de la roya del café. Manual de buenas prácticas para técnicos y facilitadores. Costa Rica: Centro Agronómico Tropical de Investigación y Enseñanza (CATIE).

Visual Geometry Group - University of Oxford. (n.d.). https://www.robots.ox.ac.uk/~vgg/software/via/.

Apéndices

Apéndice A. Transformaciones de imágenes para aumento de datos de entrenamiento

Apéndice B. Herramienta en Colab para realizar etiquetado con SAM

Apéndice C. Aplicación móvil

Apéndice D. Página web del proyecto

Apéndice E. Repositorio DECAFIA en Github

Apéndice F. Archivos para el funcionamiento del servidor dentro de la vm

Apéndice G. Código hacer aumento, partición y redimensión de imágenes

Apéndice H. Herramienta para realizar ajustes y aumento de etiquetas

Apéndice I. Colab para hacer ajustes de parámetros de entrenamiento

Apéndice J. Colab que realiza inferencia del modelo, métricas y resultados a partir de modelos

Apéndice K. Base de datos de uso libre

Apéndice L. Colab para realizar entrenamientos con Mask Rcnn

Apéndice M. Video explicativo de SAM

Apéndice N. Modelos entrenados en formato HDF5 (.h5)

“Los apéndices están adjuntos”