UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE SOFTWARE

TEMA:

IMPLEMENTACIÓN DE SEGMENTACIÓN SEMÁNTICA CON MASK R-CNN PARA LA CUANTIFICACIÓN DE MALEZAS EN CAMPOS DE PAPA USANDO IMÁGENES AÉREAS DE DRONES.

Trabajo de grado previo a la obtención del título de Ingeniero en Software

AUTOR: Oliver Patricio Zamora Fajardo

DIRECTOR: PhD. Iván Danilo García Santillán

Ibarra-Ecuador

2025

UNIVERSIDAD TÉCNICA DEL NORTE

BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN
A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO			
CÉDULA DE IDENTIDAD:	0803793215		
APELLIDOS Y NOMBRES:	Zamora Fajardo Oliver Patricio		
DIRECCIÓN:	Borbón - Esmeraldas		
EMAIL:	Opzamoraf@utn.edu.ec		
TELÉFONO FIJO:		**TELÉFONO MÓVIL:**	0986255195

DATOS DE LA OBRA	
TÍTULO:	IMPLEMENTACIÓN DE SEGMENTACIÓN SEMÁNTICA CON MASK R-CNN PARA LA CUANTIFICACIÓN DE MALEZAS EN CAMPOS DE PAPA USANDO IMÁGENES AÉREAS DE DRONES.
AUTOR (ES):	ZAMORA FAJARDO OLIVER PATRICIO
FECHA DE APROBACIÓN: DD/MM/AAAA	20/11/2025
PROGRAMA:	■ **PREGRADO** □ **POSGRADO**
TITULO POR EL QUE OPTA:	INGENIERÍA EN SOFTWARE
ASESOR /DIRECTOR:	PhD. Iván Danilo García Santillán

UNIVERSIDAD TÉCNICA DEL NORTE

BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN
A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

2. CONSTANCIAS

El autor manifiesta que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto, la obra es original y que es el titular de los derechos patrimoniales, por lo que asume la responsabilidad sobre el contenido de la misma y saldrá en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 20 días del mes de noviembre de 2025.

EL AUTOR:

(Firma)….. 

……

Nombre: Zamora Fajardo Oliver Patricio.

CERTIFICACIÓN DIRECTOR

Ibarra 20 de noviembre del 2025

CERTIFICACIÓN DIRECTOR DEL TRABAJO DE TITULACIÓN

Por medio del presente yo PhD. Iván Danilo García Santillán, certifico que el Sr. Zamora Fajardo Oliver Patricio portador de la cédula de ciudadanía número 0803793215, ha trabajado en el desarrollo del proyecto de grado "**Implementación de segmentación semántica con Mask R-CNN para la cuantificación de malezas en campos de papa usando imágenes aéreas de drones.",** previo a la obtención del Título de Ingeniero en Software. Este trabajo se ha realizado con responsabilidad, lo cual certifico con honor a la verdad.

Atentamente

IVAN DANILO GARCIA SANTILLAN

Digitally signed by IVAN DANILO GARCIA SANTILLAN
Date: 2025.11.20 16:32:31 -05'00'

PhD. Iván Danilo García Santillán

DIRECTOR DE TRABAJO DE GRADO

DEDICATORIA

A Dios, por ser mi guía, mi fortaleza y mi refugio en los momentos de incertidumbre. Por concederme la sabiduría necesaria para superar cada desafío y por acompañarme en cada paso de este camino.

A mi madre, Rosa Fajardo, la persona que más amo en este mundo. Gracias por ser una madre ejemplar, por brindarme tu amor incondicional y por confiar en mí incluso cuando yo mismo dudaba. Gracias por tu paciencia infinita, por estar siempre dispuesta no solo a escucharme, sino también a comprenderme y reconfortarme con tus palabras de aliento. Sin ti, este logro no habría sido posible.

A mi padre, Cristóbal Zamora, la persona más fuerte que he conocido. Gracias por cada sacrificio realizado por mí y por mis hermanos, por tu esfuerzo constante y tu ejemplo de perseverancia. Espero algún día poder recompensar cada uno de tus desvelos y llegar a ser, al menos, la mitad del padre que tú has sido.

A mis hermanos, Diana, Mónica y Erick, porque cada uno de ustedes me ha hecho sentir el verdadero amor fraternal. Gracias por estar siempre presentes, por sus consejos oportunos y por brindarme su ayuda, incluso sin que yo la pidiera, al notar mis momentos de necesidad. A mis sobrinos, Madeline y Santiago, por ser una fuente constante de alegría. Gracias por llenar mis días de sonrisas y por regalarme momentos de felicidad con sus ocurrencias y ternura.

A mi mejor amiga, Mariuxi, a quien considero como una hermana más. Gracias, porque a pesar de la distancia, siempre has estado presente para escucharme, aconsejarme y acompañarme con tu amistad sincera.

A mi tía Leonor, quien siempre me ha demostrado su apoyo incondicional y su deseo de verme crecer, superarme y alcanzar mis metas en la vida.

A toda mi familia, por estar siempre presente y demostrarme su apoyo. De manera especial, a mi primo Juan Carlos, quien, aunque ya no está físicamente, permanece vivo en mi corazón y en mis recuerdos.

-Zamora Fajardo Oliver Patricio

AGRADECIMIENTOS

Expreso mis más sinceros agradecimientos a la Universidad Técnica del Norte, por abrirme sus puertas y brindarme la oportunidad de formarme como profesional, ofreciéndome una educación de calidad y un espacio para mi desarrollo académico y personal. Agradezco también a cada uno de los docentes con quienes tuve el privilegio de aprender, por compartir sus conocimientos, experiencias y valores que contribuyeron a mi crecimiento durante esta etapa universitaria.

De manera especial, expreso mi más profunda gratitud a mi director, PhD. Iván García, y a mi asesor, PhD. Marco Pusdá, por la confianza depositada en mí para cumplir con este objetivo. Agradezco su guía y paciencia durante el desarrollo de este proyecto, así como sus valiosos aportes que contribuyeron significativamente a la culminación de este trabajo.

A mis padres y hermanos, por estar siempre a mi lado brindándome su apoyo, confianza y sabios consejos cuando más los necesité. Asimismo, a los familiares que me han acompañado y demostrado su apoyo incondicional durante este proceso, gracias por ayudarme a alcanzar esta meta tan importante.

A mis compañeros y amigos con quienes me he formado durante este proceso, y que supieron compartir conmigo sus conocimientos y experiencias. De manera especial, a Henry, Theo, Carlos y Kevin, con quienes pude compartir no solo aprendizajes, sino también momentos que hicieron de este camino una experiencia más amena y llevadera.

-Zamora Fajardo Oliver Patricio

Índice

Índice de figuras.

Índice de tablas

RESUMEN

El presente trabajo de tesis aborda el desarrollo de un sistema inteligente para la segmentación semántica de malezas en cultivos agrícolas. El estudio tiene como finalidad contribuir al monitoreo automatizado de cultivos a partir de imágenes capturadas con vehículos aéreos no tripulados.

La investigación se estructura en torno a dos marcos metodológicos complementarios: el modelo CRISP-DM, aplicado al ciclo de desarrollo del modelo de segmentación, y la metodología Scrum, utilizada para la gestión ágil de los Sprints de implementación del sistema. A lo largo de las fases de comprensión del negocio, preparación de los datos, modelado y despliegue, se construyó un flujo experimental reproducible que integra componentes de backend y frontend en un entorno funcional.

Inicialmente se empleó la arquitectura Mask R-CNN, sin embargo, los resultados obtenidos no alcanzaron los niveles de precisión esperados. Por ello, se adoptó la arquitectura DeepLabV3+ con diferentes backbones, entre ellos EfficientNetV2-B0. Los experimentos se desarrollaron utilizando imágenes UAV de 128×128 píxeles y un conjunto de datos multiclase, dividido en subconjuntos de entrenamiento, validación y prueba, con técnicas de aumento de datos para equilibrar las clases minoritarias.

El sistema resultante fue desplegado como una aplicación web compuesta por un backend en Flask y una interfaz gráfica en Angular, permitiendo procesar imágenes capturadas directamente desde el dron o fragmentadas en parches de 128×128 píxeles. La aplicación genera automáticamente la máscara segmentada y los porcentajes por clase, ofreciendo una herramienta accesible y funcional para investigadores y técnicos agrícolas.

Los resultados obtenidos demuestran la eficacia del modelo DeepLabV3+ con EfficientNetV2-B0 para la segmentación multiclase de malezas, alcanzando una alta precisión y tiempos de inferencia reducidos.

Palabras claves: Segmentación semántica, Mask R-CNN, DeepLabV3+, Vehículos aéreos no tripulado, Imágenes UAV.

ABSTRACT

This thesis addresses the development of an intelligent system for the semantic segmentation of weeds in agricultural crops. The main goal of the study is to contribute to automated crop monitoring through the analysis of images captured by unmanned aerial vehicles (UAVs).

The research is structured around two complementary methodological frameworks: the CRISP-DM model, applied to the segmentation model development cycle, and the Scrum methodology, used for agile management of the implementation Sprints. Throughout the stages of business understanding, data preparation, modeling, and deployment, a reproducible experimental workflow was built, integrating backend and frontend components within a functional environment.

Initially, the Mask R-CNN architecture was employed; however, the results did not reach the expected accuracy levels. Therefore, the DeepLabV3+ architecture was adopted with different backbones, including EfficientNetV2-B0. The experiments were conducted using UAV images of 128×128 pixels and a multiclass dataset divided into training, validation, and test subsets, applying data augmentation techniques to balance minority classes.

The resulting system was deployed as a web application composed of a Flask-based backend and an Angular graphical interface, allowing the processing of either full drone-captured images or 128×128 pixel patches. The application automatically generates the segmented mask and class percentage outputs, providing an accessible and functional tool for researchers and agricultural technicians.

The obtained results demonstrate the effectiveness of the DeepLabV3+ model with EfficientNetV2-B0 for multiclass weed segmentation, achieving high accuracy and reduced inference times.

Keywords: Semantic segmentation, Mask R-CNN, DeepLabV3+, Unmanned aerial vehicles, UAV images.

INTRODUCCIÓN

Tema

IMPLEMENTACIÓN DE SEGMENTACIÓN SEMÁNTICA CON MASK R-CNN PARA LA CUANTIFICACIÓN DE MALEZAS EN CAMPOS DE PAPA USANDO IMÁGENES AÉREAS DE DRONES.

Problema

Las interrupciones de producciones siempre generan pérdidas, y en la agricultura también tenemos un factor que frena la producción de cultivos, llamado maleza (la población no deseada en la agricultura) [1]. Actualmente, la interferencia de las malezas y la infestación de insectos representan el 40% de las pérdidas de productividad en todo el mundo [2], ya que puede reducir la productividad de los cultivos y aumentar la competencia por recursos limitados como el agua, los nutrientes y la luz solar [3].

Tratar de mejorar la producción agrícola requiere manejar de manera efectiva las malezas. Sin embargo, el procedimiento de detección manual de las malezas puede ser costoso y consumir mucho tiempo [2]. Si bien muchas malezas se pueden diferenciar de los cultivos, incluso en sus primeras etapas de plántula, el desmalezado se convierte en un desafío si la maleza tiene un fenotipo similar al del cultivo, en estos casos, es probable que las malezas tengan una ventaja competitiva sobre el cultivo, lo que podría sofocar la producción si no se toman contramedidas adecuadas [4].

Algunas de estas causas también están presentes en la Granja Experimental La Pradera de la Universidad Técnica del Norte (UTN), donde se opta por una estimación subjetiva e imprecisa de las malezas. Esto genera que no se tomen medidas de detección, erradicación y/o eliminación de manera eficiente y en el tiempo oportuno, lo que puede tener un impacto negativo en la salud humana y el medio ambiente debido al uso incorrecto o excesivo de productos químicos. A continuación, se presenta el árbol de problemas en la Figura 1.

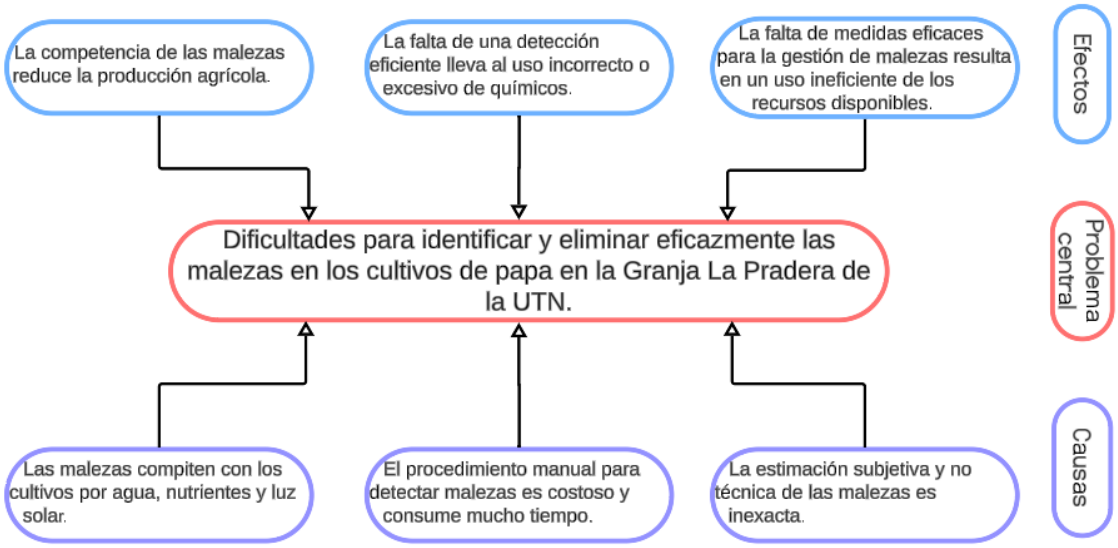

Fig. 1 Árbol de Problemas

Fuente Propia

Objetivos

Objetivo General

Implementar una técnica de segmentación semántica basada en la arquitectura Mask R-CNN para la cuantificación precisa de malezas en campos de papa utilizando imágenes aéreas capturadas por drones.

Objetivos Específicos:

1. Elaborar un marco teórico sobre la detección de malezas basada en segmentación semántica usando la arquitectura Mask R-CNN.
2. Desarrollar una aplicación web que emplee la arquitectura Mask R-CNN para cuantificar las malezas en cultivos de papa utilizando Keras-TensorFlow.
3. Validar los resultados de la solución propuesta utilizando métricas de inteligencia artificial y estadística descriptiva.

Alcance

El presente trabajo tiene como finalidad desarrollar una aplicación web que implemente la red neuronal convolucional Mask R-CNN [5] para la segmentación semántica y cuantificación de malezas en campos de cultivo de papa, utilizando imágenes aéreas capturadas por drones. Durante el desarrollo del proyecto, se aplicará la metodología CRISP-DM [6] para estructurar de manera efectiva el proceso de análisis de datos.

La selección de la arquitectura Mask R-CNN que se muestra en la figura 2, se debe a su capacidad para realizar una segmentación precisa a nivel de instancia, permitiendo no solo la detección de objetos, sino también la delineación exacta de cada objeto dentro de la imagen.

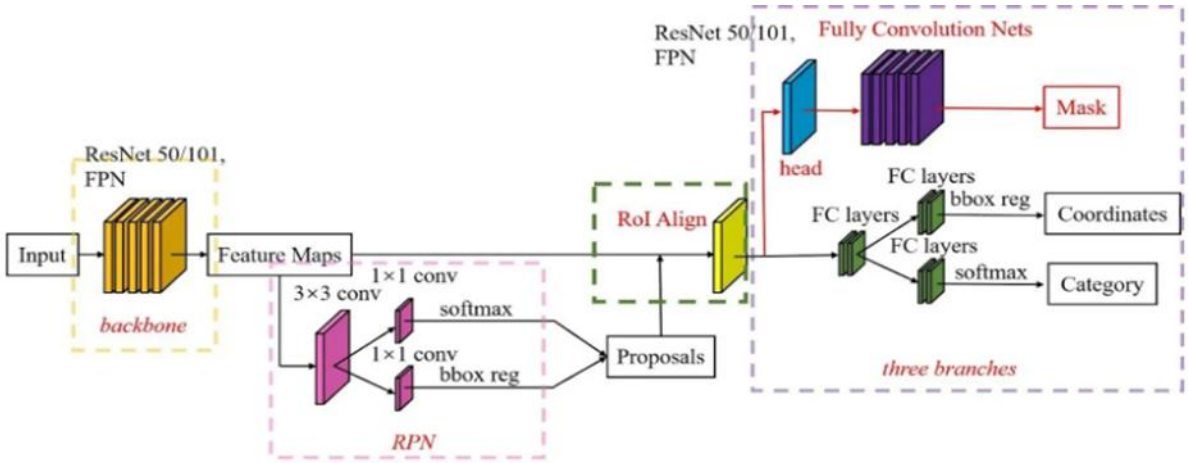

Fig. 2 Framework Mask R-CNN para la segmentación de instancias según [8]

Se utilizará un backbone adecuado que permita extraer características relevantes de las imágenes, optimizando la segmentación y cuantificación de las malezas. El entrenamiento de la red se llevará a cabo utilizando Python como lenguaje de programación, con el apoyo de las librerías TensorFlow y Keras, Pytorch.

La aplicación web desarrollada seguirá una arquitectura Cliente-Servidor. El servidor (Back-End) será implementado utilizando Flask en Python, mientras que la interfaz de usuario (Front-End) se desarrollará con Angular. Docker será utilizado para contenerizar la aplicación, garantizando así la compatibilidad y portabilidad de todo el sistema en diferentes entornos.

Los resultados obtenidos se compararán con aquellos obtenidos utilizando otras arquitecturas de red como ResNet y U-Net, desarrolladas por la carrera CSOFT y otros trabajos de la literatura, para asegurar la precisión y efectividad del modelo propuesto. A continuación, se define la metodología a usar para el proyecto en la figura 3.

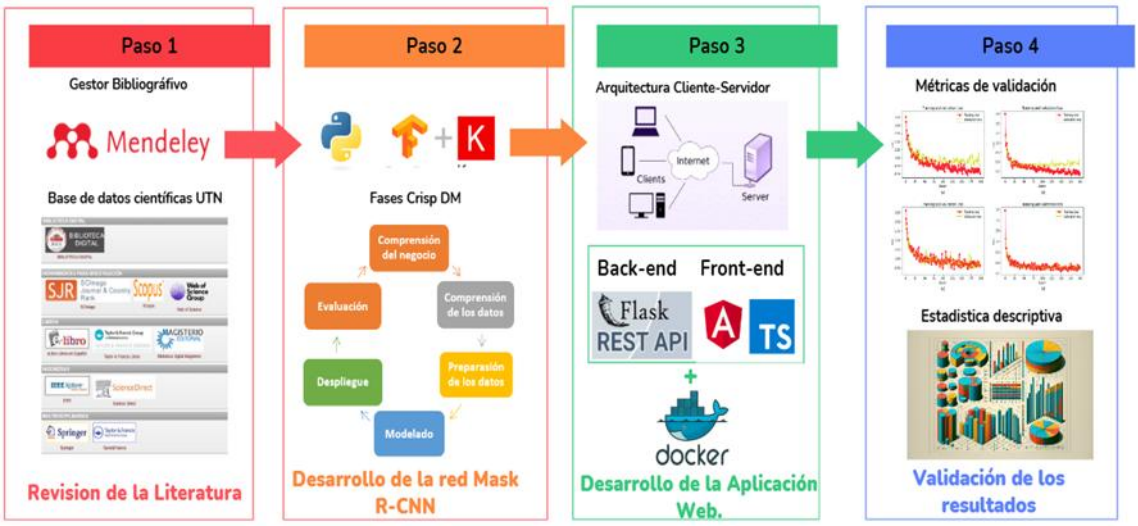

Fig. 3 Proceso del Proyecto

Metodología

El presente trabajo de investigación es de tipo aplicada y tiene como objetivo principal implementar una red neuronal convolucional Mask R-CNN para la segmentación semántica y cuantificación de malezas en campos de papa utilizando imágenes aéreas capturadas por drones. La metodología empleada se basa en una combinación de técnicas y herramientas para abordar el problema de manera efectiva.

Como primer objetivo se llevará a cabo una revisión bibliográfica sobre segmentación semántica y Mask R-CNN. Se utilizarán bases de datos científicas a las que la UTN da acceso tales como IEEE, Scopus, ScienceDirect entre otras, y el gestor bibliográfico Mendeley que se encargará de organizar y gestionar las referencias bibliográficas.

Para cumplir con el segundo objetivo, se adoptará una metodología híbrida que combina CRISP-DM y Scrum [8]. CRISP-DM se empleará en las etapas iniciales para comprender el problema y preparar los datos necesarios para entrenar el modelo Mask R-CNN. Estas tareas se integrarán en el Backlog del Producto de Scrum, donde en cada sprint se desarrollarán actividades específicas como la anotación de datos.

Posteriormente, se gestionará el desarrollo e integración del modelo en la aplicación web mediante Scrum, permitiendo una entrega incremental que asegurará tanto la precisión del modelo como la funcionalidad de la aplicación final.

Para alcanzar el tercer objetivo los resultados serán validados utilizando métricas específicas de inteligencia artificial y segmentación semántica, así como métodos de estadística descriptiva.

Justificación

Esta investigación se alinea con el Objetivo 12 de los Objetivos de Desarrollo Sostenible (ODS), que pretende minimizar la emisión de productos químicos en el aire, suelo y agua, con el fin de reducir su impacto en la salud humana y el medio ambiente [9]. Además, la política 12.1 del Objetivo 12 del Plan de Creación de Oportunidades 2021-2025 resalta la importancia de fortalecer las acciones de mitigación y adaptación frente al cambio climático [10].

Beneficiarios:

- Los agricultores son los principales beneficiarios directos, ya que se verán favorecidos por la reducción en el uso de herbicidas.
- Los consumidores finales serán beneficiarios indirectos al acceder a productos agrícolas más sanos, libres de exceso de herbicidas y producidos de manera más sostenible.

Justificación Tecnológica: Este proyecto respalda el avance tecnológico en la agricultura de precisión. Según el ODS 9, la innovación tecnológica es clave para construir infraestructuras resilientes y fomentar la industrialización inclusiva [9]. Además, el Plan de Creación de Oportunidades destaca la importancia de "modernizar el sector agrícola a través de la adopción de tecnologías avanzadas para mejorar la productividad y la sostenibilidad" [10].

Justificación Social: La investigación apoya el ODS 2, que tiene como meta "poner fin al hambre, lograr la seguridad alimentaria y promover la agricultura sostenible" [9]. La aplicación de tecnologías avanzadas como la Mask R-CNN puede optimizar la gestión agrícola, contribuyendo a la sostenibilidad y aumentando la productividad de los cultivos.

Justificación Ambiental: Este proyecto apoya directamente el ODS 13, que promueve "acciones urgentes para combatir el cambio climático y sus efectos" [9] Al detectar malezas de manera más precisa y temprana, se reduce el uso de herbicidas, minimizando la contaminación del suelo y agua. Esto favorece la salud del ecosistema agrícola y mejora la biodiversidad al evitar la eliminación innecesaria de flora no invasiva.

I. MARCO TEÓRICO

1.1. Cuantificación de Malezas en Cultivos.

1.1.1 Caracterización de malezas en cultivos de papa.

La papa es uno de los principales alimentos básicos a nivel mundial. Sin embargo, su producción enfrenta múltiples desafíos, siendo las malezas una de las más significativas. Estas no solo dificultan el desarrollo adecuado de las plantas, sino que también afectan negativamente tanto la cantidad como la calidad de la cosecha [11].

1.1.2 Impacto de las malezas en la productividad agrícola.

Las malezas representan una amenaza significativa para el rendimiento y la calidad de los cultivos de papa, con pérdidas que pueden alcanzar hasta un 85% en condiciones de alta densidad y competencia [12]. Además, algunas malezas no solo reducen la producción, sino que también afectan la calidad de los tubérculos al rodearlos o penetrarlos físicamente. Un manejo inadecuado o tardío de las malezas durante la temporada de crecimiento incrementa el riesgo de daños.

1.1.3 Técnicas tradicionales para la cuantificación de malezas.

- Método Cuantitativo: El método consiste en contar el número de individuos por unidad de superficie, proporcionando registros precisos. Las muestras se obtienen utilizando uno o varios cuadrados en cada punto seleccionado [13].
- Método semicuantitativo: las malezas no se cuentan individualmente, sino que se clasifican en categorías mediante un sistema visual que requiere práctica. Las categorías son las siguientes: abundancia baja para menos de 4 malezas por m^2, abundancia media para entre 5 y 19 malezas por m^2, y abundancia elevada para más de 20 malezas por m^2 [13].
- Método cualitativo: Este método registra únicamente la presencia o ausencia de malezas en áreas de muestreo, generalmente círculos de aproximadamente 2 m de radio alrededor del observador, distribuidos a lo largo de la transecta o pata. Aunque no mide la severidad de la infestación, es el método más rápido para monitorear un lote. En áreas de la región pampeana con abundancia media o baja, permite calcular la frecuencia de malezas dominantes y detectar especies recientemente introducidas [13].

1.1.4 Fórmulas para la cuantificación de malezas

- Índice de Densidad de Malezas (IDM): El **IDM** mide la cantidad de malezas presentes por unidad de área.

$$IDM = \frac{N_w}{A_c}$$

N_w: Número de malezas detectadas.

A_c: Área del cultivo analizada.

- Índice de Cobertura de Malezas (ICM): Representa el porcentaje del área cubierta por malezas con relación al área total.

$$ICM = \frac{A_w}{A_c} \times 100$$

A_w: Área cubierta por las malezas (en m²).

A_c: Área del cultivo analizada.

- Índice de Distribución de Malezas (IDisM): Mide la dispersión de malezas dentro del campo, utilizando un análisis de cuadrantes.

$$IDisM = \frac{\sigma_w}{\mu_w}$$

σ_w: Desviación estándar del número de malezas por cuadrante.

μ_w: Promedio de malezas por cuadrante.

1.2 Análisis de Imágenes Aéreas en la Agricultura.

1.2.1 Uso de drones en el monitoreo de cultivos

Los drones representan la tercera generación de tecnología de teledetección y destacan por ser más accesibles y económicos en comparación con otras máquinas agrícolas. Su capacidad para capturar imágenes de altísima resolución por debajo de las nubes supera a la de los satélites y los hace ideales para analistas en países en desarrollo [14].

Actualmente, las misiones de mapeo y recopilación de datos se realizan de manera autónoma, ya que los drones vuelan por sí solos, mientras que las herramientas de procesamiento de datos son cada vez más asequibles y fáciles de manejar. Además,

proporcionan información en tiempo real sobre las condiciones de los cultivos, permitiendo a los agricultores optimizar el uso de insumos agrícolas. Esta tecnología también ayuda a reducir la dependencia de la mano de obra, facilitando una gestión más eficiente y sostenible en las granjas.

Los vehículos aéreos no tripulados se clasifican según diversos atributos, tales como el tipo de aeronave, la configuración de las alas, el modo de despegue y aterrizaje, la capacidad de carga útil y la altitud a la que pueden operar, entre otros. Estas categorías permiten diferenciar sus aplicaciones y adaptabilidad a distintas necesidades y entornos [15].

Las plataformas más utilizadas en agricultura de precisión pertenecen a la clase LASE y son sistemas de ala fija o multirotores, como helicópteros, cuadricópteros, hexacópteros, octocópteros, etc.

Aunque los sensores con los que se pueden equipar los drones son numerosos, los más utilizados en plataformas UAV para fines agrícolas son:

- Cámaras visibles, RGB (Rojo, Verde y Azul): El modelo RGB combina valores de rojo, verde y azul, que varían de 0 a 255, para crear colores. Un triplete de (0, 0, 0) produce negro, mientras que (255, 255, 255) genera blanco. Las cámaras RGB capturan y almacenan estos valores como enteros para representar la intensidad de cada color en los píxeles [16].
- Cámaras multiespectrales: Estas cámaras generan imágenes en múltiples bandas del espectro, abarcando comúnmente las regiones visibles (VIS) e infrarroja cercana (NIR) [17]. Su capacidad las hace ideales para calcular índices de vegetación ampliamente empleados en la agricultura [14].
- Cámaras hiperespectrales: La tecnología de imágenes hiperespectrales captura datos tridimensionales que integran información espacial y espectral, permitiendo observaciones extensas, detalladas y repetitivas [18]. Su uso en cultivos facilita análisis profundos, detectando la presencia de diferentes patógenos con alta precisión.
- Cámaras térmicas: Estos sensores generan imágenes térmicas que muestran la temperatura de cada píxel, permitiendo detectar variaciones térmicas en la superficie de las hojas causadas por el estrés hídrico. Esta capacidad los convierte en herramientas valiosas para la gestión eficiente del agua en cultivos, ya que ayudan a identificar zonas con falta de riego o exceso de agua, mejorando el monitoreo y control de los recursos hídricos [19].

- Detección y medición de distancia por imágenes láser (LiDAR): Los sensores LiDAR y RADAR son dispositivos activos que emiten pulsos de luz o microondas y miden el tiempo de regreso del pulso reflejado. Esto permite calcular la distancia y la geometría de los objetos, siendo útiles para medir volúmenes y estructuras como las marquesinas [20].

1.2.2 Características de las imágenes aéreas relevantes para la segmentación semántica.

- Resolución espacial y espectral: La calidad de las imágenes aéreas, determinada por su resolución espacial y la diversidad de bandas espectrales, es fundamental para la segmentación precisa. Imágenes de alta resolución permiten una mejor identificación de objetos y superficies [21].
- Textura y patrones: Las variaciones en la textura y los patrones dentro de una imagen aérea son indicativas de diferentes tipos de terrenos u objetos. Por ejemplo, áreas urbanas presentan texturas y patrones distintos en comparación con zonas agrícolas o forestales. La identificación de estas texturas es esencial para una segmentación precisa [22].
- Anotaciones y datos de entrenamiento: La disponibilidad de conjuntos de datos anotados, donde cada píxel se asigna a una clase correspondiente, es esencial para entrenar modelos de aprendizaje profundo para la segmentación semántica. La calidad y cantidad de tales anotaciones tienen un efecto directo en el rendimiento del modelo [22].
- Variabilidad temporal: Las imágenes aéreas capturadas en diferentes momentos pueden mostrar cambios significativos debido a factores estacionales, condiciones climáticas o desarrollo urbano. Considerar esta variabilidad es importante para desarrollar modelos robustos que puedan generalizar adecuadamente [22].

1.3 Fundamentos de la Segmentación Semántica.

1.3.1 Definición de la segmentación semántica

La segmentación semántica consiste en dividir una imagen en regiones separadas según su clase o significado, asignando a cada píxel una etiqueta correspondiente a la clase del objeto que representa [23]. En comparación con la clasificación de imágenes, que asocia una sola clase al contenido principal de toda la imagen, la segmentación semántica identifica y delimita

detalladamente las regiones correspondientes a múltiples clases dentro de la misma imagen [24].

1.3.2 Redes neuronales profundas y su rol en la segmentación semántica

El aprendizaje profundo ha revolucionado el campo de la inteligencia artificial, ya que las redes neuronales profundas (DNN) han sobrepasado a los modelos convencionales de machine learning, como lo son los árboles de decisión o las máquinas de vectores de soporte, en un sinnúmero de tareas de pronóstico [25].

Las arquitecturas de aprendizaje profundo (DL), como las redes neuronales convolucionales (CNN) y la red completamente convolucional (FCN) son ampliamente utilizadas para la segmentación de imágenes [26].

1.3.3 Mask R-CNN

Las redes neuronales convolucionales basadas en regiones con máscara Mask R-CNN[5] son una popular arquitectura de aprendizaje profundo y una extensión avanzada de Faster R-CNN [27]. Mask R-CNN, además de detectar objetos y generar cuadros delimitadores, permite realizar segmentación a nivel de píxel, logrando una identificación precisa de los contornos de los objetos dentro de una imagen [12]. El gráfico de la arquitectura se lo puede encontrar en el capítulo I.

1.3.4 Definición de segmentación de instancias

La segmentación de instancias es un tipo de técnica dentro de la visión artificial que permite individualmente delimitar e identificar objetos en una imagen al unir las ventajas de lo que es la Detección de objetos y la Segmentación Semántica [28]. Un beneficio clave de la segmentación de instancias en aplicaciones agrícolas es su capacidad para identificar y cuantificar con precisión las estructuras de plantas y cultivos, mejorando el análisis y la gestión agrícola [29].

1.3.5 Segmentación semántica vs segmentación de instancias

Aunque tanto la segmentación semántica como la segmentación de instancias buscan dividir una imagen en regiones relevantes, presentan diferencias fundamentales. La segmentación semántica asigna una clase específica a cada píxel de la imagen sin distinguir entre objetos individuales dentro de la misma clase. En cambio, la segmentación de instancias no solo clasifica, sino que también separa cada objeto individual, incluso si varios pertenecen a la misma clase [5].

La segmentación semántica es útil para identificar áreas generales afectadas por malezas, pero no permite distinguir entre diferentes tipos de malezas ni entre sus instancias individuales. En contraste, la segmentación de instancias no solo identifica las malezas, sino que también permite cuantificar su cantidad y diferenciar entre especies, lo que resulta fundamental para desarrollar estrategias de control más específicas y eficaces [30], [31].

Comparación Principal.

Tabla 1 Comparación entre segmentación semántica y segmentación de instancia

CARACTERÍSTICAS	SEGMENTACIÓN SEMÁNTICA	SEGMENTACIÓN DE INSTANCIAS
SALIDA	Mapas por clase	Mascaras individuales
PRECISIÓN PARA DETALLES	Baja: No distingue objetos individuales	Alta: Analiza cada instancia única
COSTO COMPUTACIONAL	Menor	Mayor
USABILIDAD	Escenarios globales como identificar áreas generales infestadas de malezas.	Análisis detallados como identificar y contar plantas individuales de malezas o incluso diferenciar entre especies.

Fuente: [30], [31]

1.3.6 Componentes de Mask R-CNN

Backbone

La columna vertebral de la arquitectura Mask R-CNN tiene la función principal de extraer mapas de características detallados a partir de las imágenes de entrada, facilitando el análisis y la segmentación [32].

Backbone más comunes:

- Resnet: Es una arquitectura de red neuronal profunda que resuelve el problema del degradado del gradiente en redes muy profundas, que puede causar pérdida de precisión.
- EfficientNet: Es una familia de modelos diseñada para maximizar la eficiencia en términos de memoria y computación, sin sacrificar precisión. Utiliza un enfoque de escalamiento compuesto que ajusta las dimensiones de la red (profundidad, ancho y resolución de entrada) de manera equilibrada para obtener el mejor rendimiento con el menor costo computacional.
- ResNeXt: Combina las ventajas de ResNet e Inception, logrando un desempeño destacado en visión por computadora. Su enfoque se basa en la cardinalidad, que optimiza el aprendizaje al permitir que los distintos caminos dentro de la red utilicen diferentes cantidades de filtros [33].

Red de Propuestas de Región (RPN)

Procesa una imagen de cualquier tamaño como entrada y genera un cuadro delimitador que identifica un objeto, asignándole una puntuación de confianza de 1 [32]. RPN crea cuadros de anclaje en la imagen, que sirven como plantillas de referencia para localizar y detectar elementos de interés en la imagen analizada. Entre todos los cuadros de anclaje generados, aquel con mayor adecuación y mayor nivel de confianza es seleccionado como el cuadro delimitador final [34].

RoI Aling

En esta capa, todas las salidas de las redes RPN se introducen como entrada, También integra los mapas de características originales extraídos por la arquitectura troncal, mejorando la precisión y el detalle en la segmentación y detección de objetos [35].

La alineación de ROI mejora la segmentación al ajustar las regiones de interés (ROI) a una resolución específica mediante interpolación bilineal. Para cada ROI, la red se bifurca en tres ramas: una que genera un cuadro delimitador, la segunda que identifica la clase del objeto y asigna la probabilidad de su existencia, y la tercera que produce una máscara a nivel de píxel [35].

Clasificación y generación de máscaras

En Mask R-CNN, los mapas de características de longitud fija generados por la alineación de ROI se envían a la capa final completamente conectada y a la rama de máscara.

La capa completamente conectada reduce las dimensiones espaciales a 1 × 1, clasifica la imagen mediante un clasificador Softmax y ajusta los cuadros delimitadores usando un regresor. Simultáneamente, estos mapas se procesan en la rama de máscara para realizar la segmentación a nivel de píxel [36].

1.3.7 Otras arquitecturas utilizadas en la segmentación semántica

DeepLab: Este método de segmentación semántica integra una convolución separable profunda con una arquitectura de codificador-decodificador. Inicialmente, utiliza la red liviana para extraer características, seguida del módulo **Atrous Spatial Pyramid Pooling (ASPP)**, que permite capturar información de características a múltiples escalas [37].

Deeplabv3+: La arquitectura DeepLabV3+ es una evolución del modelo DeepLabV3, creada para mejorar la precisión en las tareas de segmentación semántica. Esto se logra a través de una combinación eficiente de módulos encoder-decoder y convoluciones dilatadas.

El modelo se forma de cuatro bloques principales:

Encoder:

El encoder juega un papel crucial en la arquitectura de segmentación semántica, ya que se encarga de extraer las características más importantes de la imagen de entrada. Para lograr esto, utiliza una red base o backbone, lo que le permite captar información jerárquica en diferentes niveles de abstracción. Además, incorpora convoluciones dilatadas (atrous convolutions) que amplían el campo receptivo sin perder la resolución espacial, algo fundamental para mantener los detalles finos y las estructuras locales que se encuentran en la imagen. Así, las capas del encoder producen representaciones profundas y ricas en información, capaces de describir tanto el contexto general de la escena como los contornos precisos de los objetos, proporcionando una base sólida para las etapas posteriores del modelo.

ASPP (Atrous Spatial Pyramid Pooling)

El módulo ASPP (Atrous Spatial Pyramid Pooling) representa el elemento central de la arquitectura DeepLabV3+, ya que su causa principal es intentar capturar información multiescala para hacer frente a la variabilidad en el tamaño y forma de los elementos presentes en la imagen. Este módulo aplica convoluciones dilatadas sobre la misma característica de entrada, variando la tasa de dilatación (rates) con el objetivo de extraer patrones visuales a escala variable, todo ello sin perder resolución espacial.

A su vez, se añade a este módulo un Global Average Pooling, el cual permite obtener una visión contextual de tipo global de la escena, que completará la información local expuesta por las convoluciones. Finalmente, los productos obtenidos de este procedimiento se concatenan y se proyectan a un espacio de menor dimensión mediante la convolución 1x1, logrando de esta manera integrar características relevantes tanto locales como globales que, a su vez, ayudarán a mejorar la forma como el modelo segmenta regiones complejas en la imagen.

Decoder:

El decoder (decodificador) es el que se encargará de reconstruir el mapa de segmentación hasta alcanzar la resolución inicial de la imagen de entrada. Para ello, recibe la salida del módulo ASPP, y fusiona la misma con características de baja profundidad procedente de las primeras capas del encoder, de forma que se permite recuperar los detalles espaciales finos y mejorar la precisión en los bordes de los objetos segmentados.

Con secuencias de convoluciones de 3×3 y operaciones de upsampling bilineal, el decoder irá refinando la predicción a partir de la imagen inicial, adaptando así los contornos y reduciendo la pérdida de información espacial que se produjo durante la codificación. De este modo, el decoder generará un mapa de segmentación de salida que respete la resolución y la coherencia estructural de la imagen de entrada, garantizando así una buena representación de las regiones a segmentar que, a la vez, sea visualmente coherente con la imagen de entrada.

Capa de clasificación:

La capa de clasificación (Output Layer) indica la fase final del modelo que consiste en llevar a cabo la predicción explícita de las clasesque hay en la imagen, para ello se realiza primero una convolución 1×1 que consiste en una reducción del número de dimensiones de las características que se han extraído por parte del decoder seguida de la aplicación de una función de activación softmax Esta acción servirá para construir un mapa de probabilidad por clases, donde cada píxel se le asigna una etiqueta concreta de la que se considera más probable. De esta manera el modelo ofrece la representación semántica de toda la escena, con la que se puede identificar la distribución espacial que tiene cada clase haciendo uso de la imagen.

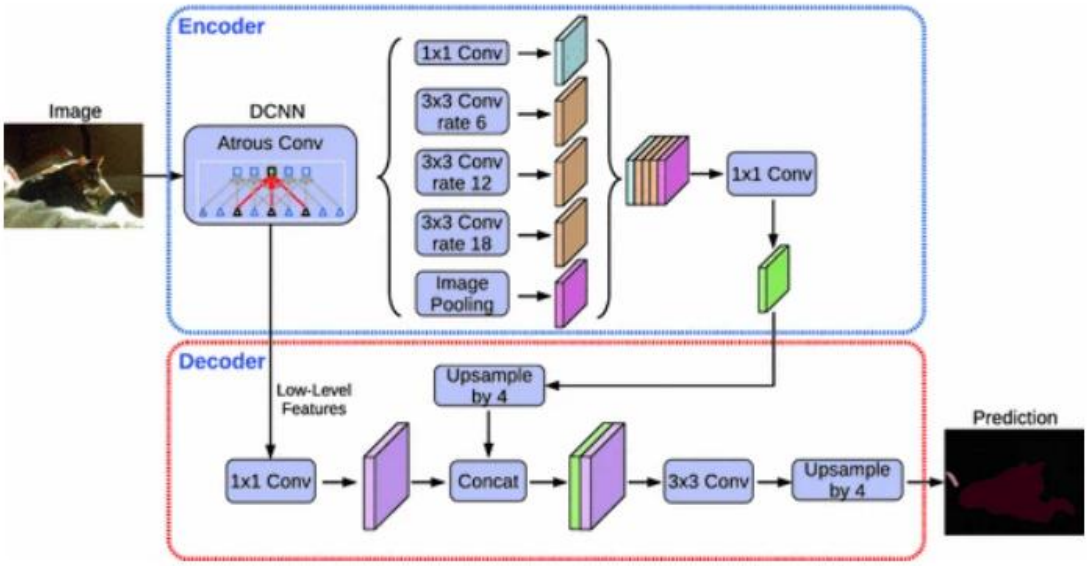

Fig. 4 Arquitectura Deeplab3+ [37]

U-NET: Es una red neuronal convolucional ideada principalmente para tareas de segmentación semántica en imágenes [38]. Su arquitectura se caracteriza por tener una forma de "U", compuesta por dos etapas principales:

Etapa de contracción (encoder): Extrae características mediante una serie de convoluciones y operaciones de pooling, reduciendo progresivamente la resolución de la imagen.

Etapa de expansión (decoder): Restaura la resolución original de la imagen mediante operaciones de upsampling y convoluciones, permitiendo generar un mapa segmentado con la misma dimensión que la entrada.

Comparación entre las arquitecturas:

Tabla 2 Comparación de arquitecturas

ARQUITECTURA	FOCO PRINCIPAL	FORTALEZAS	APLICACIONES AGRÍCOLAS
U-NET	Segmentación semántica en imágenes.	Segmentación precisa de objetos pequeños.	Identificación de malezas y enfermedades.

DEEPLAB3+	Segmentación semántica avanzada.	Captura de contexto multiescala.	Clasificación de cultivos y áreas de interés.
MASK R-CNN	Segmentación de instancias.	Conteo y análisis individual de objetos.	Cuantificación y localización de malezas.

Fuentes: [5], [37], [38]

1.3.8 Métricas de evaluación en segmentación semántica

- Mean IoU (Intersección Over Union):

Es una métrica clave en visión por computadora, empleada para evaluar el desempeño de modelos en detección de objetos y segmentación semántica. IoU calcula la razón entre el área de intersección y el área de unión de las clasificaciones predicha y real, proporcionando una medida cuantitativa de precisión [39]. Su fórmula se expresa como:

$$IoU = \frac{|A \cap B|}{|A \cup B|}$$

A: Representa el conjunto de píxeles previsto.

B: Representa el conjunto de píxeles real.

$|A \cap B|$: El número de píxeles en la intersección de ambos conjuntos.

$|A \cup B|$: El número total de píxeles en la unión de los dos conjuntos.

La puntuación IoU varía entre 0 y 1, donde 1 indica una superposición perfecta entre la predicción y la realidad.

- Mean Dice Coefficient.

La puntuación Dice, o coeficiente Dice, es una métrica utilizada para evaluar la semejanza entre dos conjuntos en diversas áreas, como la segmentación de imágenes [40]. A diferencia de la IoU, la puntuación Dice otorga mayor ponderación a la intersección relativa de los conjuntos. Se calcula como:

$$DICE(A,B) = \frac{2 \cdot |A \cap B|}{|A| + |B|}$$

En este contexto, AAA y BBB representan dos conjuntos, como píxeles o regiones en una imagen. El término |A∩B| indica el número de elementos comunes entre ambos conjuntos, mientras que |A| y |B| representan el número total de elementos en los conjuntos A y B, respectivamente. Este coeficiente varía de 0 a 1, donde un valor de 1 refleja una superposición o similitud perfecta entre los conjuntos.

- Dice Loss

La función de pérdida Dice Loss se basa en el coeficiente Dice y tiene como objetivo optimizar la coincidencia entre las predicciones realizadas y los datos reales de referencia. Su Formula se expresa como:

$$Dice\ Loss(A, B) = 1 - \frac{2 \cdot |A \cap B|}{|A| + |B|}$$

El Mean Dice Coefficient evalúa el grado promedio de superposición entre las predicciones y las etiquetas reales en todas las clases o instancias. Este indicador se utiliza para medir la calidad de la segmentación, donde valores próximos a 1 reflejan una correspondencia sobresaliente [41].

- Índice de Similitud Estructural (SSIM)

En la segmentación semántica, el SSIM evalúa la calidad del mapa segmentado generado por el modelo en comparación con el ground truth [40]. A diferencia de métricas como la IoU o el Dice Score, que se enfocan en el área superpuesta entre segmentos, el SSIM considera la calidad visual y estructural del mapa. La fórmula simplificada en términos de medias (μ_x, μ_y), desviación estándar (σ_x, σ_y) y covarianza (σ_{xy}) es:

$$SSIM(x, y) = \frac{(2\mu_x u_y + C1)(2\sigma_{xy} + C2)}{(\mu_x^2 + \mu_y^2 + C1)(\sigma_x^2 + \sigma_y^2 + C2)}$$

C1 y C2 son constantes para estabilizar la división en caso de valores bajos.

El SSIM tiene un rango de [−1,1] donde:

1: Máxima similitud estructural.

0: Ninguna similitud estructural.

Valores negativos indican estructuras opuestas.

1.4 Desarrollo web.

1.4.1 Metodología Scrum.

Scrum, como el marco ágil más popular, se basa en los principios y valores establecidos en el Manifiesto Ágil. Este enfoque proporciona una estructura flexible y colaborativa para gestionar y organizar trabajos complejos, priorizando la adaptabilidad, la entrega incremental y la mejora continua [42].

Eventos de Scrum

Hay cinco eventos en el marco de Scrum. Estos eventos son oportunidades valiosas para inspeccionar y adaptar el producto:

Sprint: Es el ciclo central de Scrum, de duración limitada (generalmente de una a dos semanas), durante el cual se crean incrementos del producto.

Planificación del sprint: Marca el inicio del ciclo, definiendo el trabajo a realizar durante el Sprint.

Daily Scrum: El objetivo del es evaluar el progreso hacia la meta del Sprint y realizar ajustes en el Sprint Backlog para adaptar las tareas planificadas según las necesidades que surjan.

Revisión del sprint: El propósito es inspeccionar el resultado con las partes interesadas y determinar las adaptaciones necesarias para los próximos Sprints.

Retrospectiva del sprint: El objetivo es diseñar estrategias para mejorar la calidad y la eficiencia.

Artefactos del Scrum

Product backlog: Es un listado priorizado de los requisitos necesarios para optimizar el producto y constituye la única fuente de tareas para el equipo Scrum.

Backlog del sprint: Está formado por el objetivo del Sprint (el "por qué"), los elementos seleccionados del Product Backlog (el "qué") y un plan ejecutable para entregar el Incremento (el "cómo").

Incremento: Cuando un elemento del Product Backlog se completa de acuerdo con los atributos de calidad definidos (generalmente en la definición de terminado) y ofrece valor, se convierte en un "incremento", es decir, una parte funcional del producto que puede ser utilizada.

1.4.2 CRISP DM

Fue elaborado gracias al esfuerzo de un grupo formado en sus inicios por DaimlerChryrler, SPSS y NCR. CRISP-DM sus siglas equivale a Proceso estándar de CRoss-Industry para minería de datos [43].

Es el marco más popular utilizado por equipos de ciencia de datos. Proporciona una descripción clara y comprensible del flujo de trabajo o ciclo de vida de los proyectos de ciencia de datos [44]. CRISP-DM requiere una mayor integración con procesos de gestión, alineándose con metodologías ágiles y de desarrollo de software para optimizar su aplicación.

Etapas:

1. Comprensión del Negocio: Se enfoca en las metas y requerimientos del proyecto desde un punto de vista empresarial, convirtiéndolos en un desafío de ciencia de datos.
2. Comprensión de los Datos: La etapa empieza con la recolección de información y avanza con acciones destinadas a explorar, identificar problemas de calidad, generar ideas preliminares y reconocer patrones clave para desarrollar hipótesis.
3. Preparación de los datos: La etapa abarca todas las tareas requeridas para transformar los datos sin procesar en un conjunto final listo para su uso.

- Modelado: Implica seleccionar y aplicar técnicas de modelado adecuadas, ajustando sus parámetros para alcanzar valores óptimos.
- Evaluación: Los modelos desarrollados se examinan minuciosamente y se revisan los pasos previos para garantizar que cumplen con los objetivos comerciales establecidos.
- Implementación: El desarrollo del modelo generalmente no representa el cierre del proyecto. Incluso cuando la meta es profundizar la comprensión sobre los datos, los resultados deben organizarse y presentarse de manera clara y comprensible para el cliente.

1.4.3 Arquitectura cliente-servidor:

Es un conjunto de tecnologías donde existen dos componentes principales: Cliente y Servidor. El Cliente envía peticiones al Servidor, que las procesa y responde con paquetes de datos. Pueden trabajar de manera conjunta o separada para realizar tareas específicas [45].

1.4.4 Lenguajes de programación:

- Python: Es un lenguaje de programación avanzado, interpretado y con tipado dinámico. Admite la programación orientada a objetos y se aplica en áreas como el desarrollo de aplicaciones web, ciencia de datos, aprendizaje automático, entre otros [46].
- Flask: Es un framework web ligero basado en WSGI, diseñado para que sea rápido y fácil de usar, permitiendo la creación de aplicaciones web sencillas que pueden escalar hasta convertirse en aplicaciones complejas. Es flexible, minimalista y permite al desarrollador decidir cómo estructurar la aplicación [47].
- TypeScript: Es un superset de JavaScript que agrega un sistema de tipos estáticos opcional, lo que permite detectar errores antes de que el código se ejecute. Aunque es completamente compatible con el código JavaScript existente, TypeScript proporciona ventajas adicionales al forzar la consistencia en la asignación de tipos, como string o number. Esto ayuda a identificar posibles errores en tiempo de compilación, mejorando la calidad del código y reduciendo los errores en tiempo de ejecución [48].
- Angular: Es un framework web que facilita a los desarrolladores la creación de aplicaciones rápidas y confiables. Con el respaldo de un equipo especializado de Google, ofrece un conjunto integral de herramientas, API y bibliotecas diseñadas para optimizar y facilitar el proceso de desarrollo. Angular ofrece una plataforma robusta para construir aplicaciones eficientes, adaptándose tanto al tamaño del equipo como a la complejidad de la base de código [49].

1.5 Trabajos relacionados

Goyal et al. [12], desarrolló un sistema para detectar malezas en cultivos de papa usando modelos de aprendizaje profundo, abordando la oclusión densa y la complejidad visual en ambientes post-emergencia. Se recopilaron 452 imágenes de alta resolución en campos de Punjab, India, ampliadas a 1950 mediante aumento de datos. Se compararon dos modelos de segmentación, Mask R-CNN y YOLOv8, entrenados con anotaciones precisas a nivel de píxel. Los resultados mostraron que YOLOv8 logró una precisión del 83.4%, superando a Mask R-CNN (79%), aunque este último destacó en precisión y sensibilidad en ambientes ocluidos. La precisión limitada por las condiciones complejas sugiere integrar datos adicionales de diversas regiones y explorar implementaciones en tiempo real con drones o robots, optimizando así la agricultura de precisión y la gestión eficiente de cultivos.

Guo et al. [50], desarrollaron un modelo de segmentación semántica, CTFFNet, para detectar y segmentar malezas en campos agrícolas usando imágenes capturadas por UAVs. El objetivo fue identificar con precisión diferentes especies de malezas combinando redes neuronales convolucionales (CNN) y transformadores. Se recolectaron imágenes aéreas de campos de arroz, aplicando técnicas avanzadas de etiquetado y aumento de datos para obtener un conjunto robusto. El modelo integró módulos de extracción de características con mecanismos de atención y convoluciones deformables en la decodificación para mejorar la sensibilidad a formas complejas. Los resultados mostraron que CTFFNet superó a modelos basados únicamente en CNN o transformadores, logrando un MIoU del 72.8% en entornos complejos. Sin embargo, las limitaciones incluyen cambios en iluminación y generalización a otros cultivos. Se sugiere ampliar el conjunto de datos y optimizar el modelo para aplicaciones más diversas.

En su investigación, Jiang et al. [51] presentaron SWFormer, una red híbrida CNN-Transformer diseñada para la segmentación precisa de malezas en campos agrícolas. El modelo combina convoluciones para captar dependencias locales y bloques Transformer para modelar relaciones globales. Se introdujeron dos módulos clave: el Scale-Wise Cascade Convolution (SWCC), que captura características multiescala expandiendo el campo receptivo, y el Adaptive Semantic Aggregation (ASA), que fusiona información semántica de manera adaptativa entre mapas de características. El modelo fue evaluado en los conjuntos de datos públicos CropandWeed y SB20, logrando un mIoU del 76.54% y 61.24%, respectivamente, superando a métodos tradicionales como FCN y Swin-Transformer. Sin embargo, el modelo presenta limitaciones al diferenciar malezas de cultivos en condiciones de alta similitud visual y en escenarios con datos limitados. Futuros trabajos se centrarán en mejorar la eficiencia del modelo y su capacidad para adaptarse a condiciones más complejas, optimizando su aplicabilidad en la agricultura de precisión.

Samuel et al. [52], desarrollaron un modelo innovador denominado AGS-MNFELM para la detección y clasificación de malezas en campos agrícolas. Utilizando procesamiento avanzado de imágenes, el modelo incluyó filtración, ecualización adaptativa del histograma y clasificación mediante una red neuronal Mobile Net. Los resultados mostraron que el modelo AGS-MNFELM alcanzó una precisión general del 98.63%, superando significativamente otros enfoques tradicionales en términos de sensibilidad y especificidad. Sin embargo, el estudio

también identificó limitaciones, como la carencia de un grupo de datos más extenso y diverso para optimizar la generalización del modelo en diferentes condiciones de cultivo. Para el futuro, se sugiere la ampliación del entrenamiento del modelo con imágenes en tiempo real y la exploración de su aplicación en la identificación de otras especies de cultivos y malezas, lo que podría contribuir a una agricultura más sostenible y eficiente.

Thiagarajan et al. [53], investigaron la implementación de modelos de segmentación semántica basados en arquitecturas codificador-decodificador para la detección precisa de cultivos y malezas. Se utilizaron imágenes de un conjunto de datos de frijoles, que incluían diversas condiciones ambientales, para entrenar y evaluar la eficacia de diferentes modelos de aprendizaje profundo, como UNet y SegNet, en la clasificación de cada píxel de las imágenes. Los resultados mostraron que los modelos de segmentación semántica lograron una identificación precisa de las malezas, lo que permite desarrollar estrategias de control más efectivas y sostenibles. Sin embargo, el estudio también identificó limitaciones, como la carencia de grandes volúmenes de datos clasificados y el tiempo de entrenamiento prolongado. Para el futuro, se sugiere explorar técnicas de transferencia de aprendizaje y la generación de datos sintéticos para mejorar la eficiencia del entrenamiento y la aplicabilidad de estos modelos en entornos agrícolas diversos.

En el estudio de Xu et al. [54], se desarrolló un método para la detección y segmentación de malezas en campos de soja utilizando imágenes capturadas por drones (UAV). El objetivo fue optimizar la segmentación mediante la integración de un índice de color visible (G/R) con una arquitectura de codificador-decodificador mejorada. Se aplicaron técnicas avanzadas de aprendizaje profundo, como ResNet101_v y módulos DSASPP, para mejorar la extracción de características multiescala y segmentar con precisión los límites de las malezas. El método propuesto superó a modelos convencionales como U-Net, Deeplabv3 y Vision Transformer, logrando una precisión promedio del 90.5% y un IoU del 95.9%. No obstante, la limitación principal fue que el estudio se centró únicamente en campos de soja, lo que podría dificultar su generalización a otros cultivos. Como trabajo futuro, se sugiere integrar imágenes multiespectrales y expandir el enfoque a diversas condiciones ambientales para mejorar su aplicabilidad en la agricultura de precisión.

Khan et al. [55], desarrolló un marco innovador de segmentación de malezas y cultivos utilizando imágenes de drones. El modelo se basó en una arquitectura encoder-decoder mejorada, donde el encoder combinó una red Dense-Inception con un módulo ASPP para

capturar características multiescala, mientras que el decoder utilizó capas de deconvolución y módulos de atención espacial y de canal (CnSAU) para mejorar la precisión. Evaluado con un conjunto de datos público de imágenes de arroz, el modelo alcanzó un mIoU de 0.81, superando a métodos como U-Net y DeepLab. A pesar de su alta precisión en condiciones complejas, el estudio se limitó a un solo tipo de cultivo, restringiendo su generalización. Como trabajo futuro, se propone validar el modelo con otros cultivos, integrar imágenes multiespectrales y aplicar técnicas de aprendizaje por transferencia.

Este estudio Gomroki et al. [56] desarrolla la arquitectura CWRepViT-Net para la segmentación automática de cultivos de soja y malezas en imágenes RGB capturadas por drones en distintas etapas de crecimiento, empleando bloques RepViT en el encoder pre-entrenados mediante transferencia semi-supervisada y bloques MUNet en el decoder. La metodología incluyó aumento de datos mediante rotaciones, entrenamiento por 100 épocas, y evaluación en condiciones variadas, logrando una precisión global del 95.87%, un coeficiente de Kappa de 0.91 y altas métricas en precisión, F1-score e IoU, superando modelos existentes como MobileNetV3 y TinyViT. La red demostró ser efectiva en distinguir diferentes clases de malezas y cultivos incluso en fases tempranas, mostrando potencial para aplicaciones en agricultura de precisión. Para futuras investigaciones, se recomienda ampliar el uso en áreas con diversos cultivos y malezas, incluyendo datos satelitales y multiespectrales, y explorar mejoras en la detección en condiciones complejas para fortalecer la robustez y aplicabilidad del método en escenarios agrícolas reales.

Cheong et al. [57]propone KDOSS-Net, una red de segmentación semántica basada en distilación de conocimiento y outpainting para mejorar la detección de cultivos y malezas fuera del campo de visión (FOV). La metodología combina un modelo maestro (OPOSS-Net) que realiza predicción de objetos y restauración de imagen antes de segmentar, y un modelo estudiante (SSWO-Net) optimizado para dispositivos con baja capacidad, que aprende del maestro mediante distilación canal a canal. Los experimentos en tres bases de datos públicas (Rice seedling, CWFID y BoniRob) mostraron valores mIOU de 0.6315, 0.7101 y 0.7524, superando modelos SOTA con menor carga computacional. Como propuesta futura, se plantea ampliar la predicción de regiones fuera del FOV, optimizar la ligereza del modelo y vincularlo con sistemas de inteligencia artificial agrícola basados en lenguaje como LLaVA para recomendaciones de herbicidas

Silvakumar et al. [58]propone un método de segmentación de cultivos y malezas basado en aprendizaje federado y ensamble de modelos para mejorar la precisión en campos agrícolas con múltiples cultivos. La metodología integra un sistema de aprendizaje distribuido donde los modelos locales entrenan datos específicos de cada región y comparten solo parámetros, evitando la centralización de la información. Se evaluaron distintos esquemas de fusión de pesos y arquitecturas de redes convolucionales, alcanzando un incremento promedio del 7–10 % en precisión y una mejor capacidad de generalización frente a modelos individuales. Los resultados demostraron que el enfoque federado es eficiente y escalable para entornos agrícolas reales. Como propuesta futura, se plantea incorporar modelos ligeros y autoajustables, así como integrar datos multiespectrales y sensores IoT para optimizar el manejo inteligente de cultivos.

II. DESARROLLO

La presente investigación tiene como propósito desarrollar un sistema de cuantificación automática de malezas en cultivos de papa, utilizando imágenes captadas por vehículos aéreos no tripulados (drones) y empleando la arquitectura de red neuronal convolucional Mask R-CNN. Este enfoque busca disminuir de forma considerable el tiempo requerido para la cuantificación manual de malezas, contribuyendo a una mayor eficiencia en los procesos de monitoreo y gestión agrícola.

2.1. Descripción del Proyecto

El presente proyecto tiene como objetivo el desarrollo de un sistema de cuantificación automática de malezas en cultivos de papa, a partir de imágenes capturadas mediante drones, utilizando la arquitectura de segmentación semantica. La solución propuesta será implementada como una aplicación web con arquitectura Cliente-Servidor, donde el Front-End será desarrollado en Angular y el Back-End será gestionado mediante Flask en Python, permitiendo así una comunicación eficiente a través de API RESTful.

La aplicación permitirá a los usuarios cargar imágenes aéreas adquiridas por drones, las cuales serán procesadas por el modelo de segmentación entrenado con datos previamente utilizados por el compañero Jorge Pazos, garantizando la continuidad del trabajo y la reutilización de recursos validados. El sistema realizará la segmentación e identificación de malezas, proporcionando métricas útiles para apoyar la toma de decisiones en el monitoreo y gestión de cultivos.

Para asegurar un desarrollo ordenado, riguroso y enfocado tanto en la calidad técnica del modelo como en la utilidad práctica del sistema, se emplearán dos metodologías complementarias: por un lado, la metodología CRISP-DM (Cross Industry Standard Process for Data Mining) orientará todo el proceso de análisis y modelado de datos; y por otro lado, la metodología ágil Scrum será utilizada para la planificación y gestión del desarrollo del software, promoviendo la colaboración, la iteración y la entrega continua de funcionalidades.

2.2.Comprensión del negocio.

La fase de la comprensión del negocio implica definir y analizar el problema central y los objetivos del proyecto. En este trabajo, dicha fase fue abordada previamente en la fase de Introducción, en donde se establecieron claramente la problemática relacionada con la cuantificación de malezas en cultivos de papa, los objetivos generales y específicos.

2.3.Comprensión de los datos

El dataset fue generado a partir de imágenes capturadas mediante un dron tipo DJI Mavic 2 Pro, este fue configurado en la aplicación DroneDeploy 5.7. Las imágenes fueron recolectadas en cultivos de papa ubicados en la provincia de Imbabura, Ecuador. El proceso de adquisición se realizó durante vuelos planificados con condiciones específicas de altitud y clima para asegurar la calidad de las capturas.

Las especificaciones de vuelo fueron las siguientes:

- Altura de vuelo: 9 metros
- Velocidad de vuelo: 1 m/s
- Modo de captura: Video
- Resolución de imagen: 5472px por 3648px

De estas imágenes se obtuvieron subimágenes con dimensiones de 250 x 250 píxeles. Se seleccionó este tamaño porque permite capturar una planta entera, lo cual resulta fundamental para realizar una segmentación exacta. Posteriormente las imágenes fueron redimencionadas a 128 x 128 píxeles y cada una de estas subimágenes fue etiquetada en el software Roboflow mediante máscaras binarias que indicaban la presencia o ausencia de las cinco clases objetivo: cow-tongue, dandelion, kikuyo, other, y potato.

El Dataset balanceado obtenido cuenta con un numero 6632 imágenes para el entrenamiento, 829 imágenes para la validación y 830 imágenes para las pruebas.

Tabla 3 Numeros de pixeles de cada Clase

Clase	Train	Validation	Test	Total
Background	89,043,401	11,156,549	11,209,413	111,409,363
Cow-tongue	3,075,798	376,656	439,269	3,891,723
Dandelion	1,427,813	161,499	179,105	1,768,417
Kikuyo	2,405,968	314,983	307,616	3,028,567
Other	691,744	84,028	87,433	863,205
Potato	12,013,964	1,488,621	1,375,884	14,878,469

Total	108,658,688	13,582,336	13,598,720	135,839,744

La Tabla 3 presenta la distribución de píxeles por clase en los conjuntos de entrenamiento, validación, prueba y el total general del dataset. Se observa que la clase Background concentra la mayor proporción de información, con un total de 111.409.363 píxeles, seguida por la clase Potato, con 14.878.469 píxeles. En contraste, las clases Dandelion y Other presentan la menor representación, alcanzando únicamente 1.768.417 y 863.205 píxeles respectivamente, lo que refleja un marcado desbalance en la cantidad de datos disponibles para dichas categorías.

Las clases Cow-tongue (3.891.723 píxeles) y Kikuyo (3.028.567 píxeles) mantienen una representación intermedia dentro del conjunto de datos. Esta disparidad en la distribución puede impactar en el proceso de entrenamiento del modelo de segmentación, generando una mayor sensibilidad hacia las clases dominantes y una menor capacidad de generalización frente a las categorías minoritarias. Por ello, resulta pertinente considerar el uso de técnicas de balanceo de clases, estrategias de ponderación en la función de pérdida o métodos de aumento de datos, con el propósito de mitigar posibles sesgos y optimizar el desempeño del modelo en todas las clases de interés.

2.4. Preparación de los datos

El objetivo de esta fase es realizar una revisión y adecuación del dataset previamente generado por el ex tesista Jorge Pazos, asegurando su integridad, limpieza y estructura correcta para su reutilización. Además, se busca adaptar el formato de anotaciones al estándar requerido por la implementación con la librería Matterport, dejando preparado el conjunto de datos para su procesamiento y uso en fases posteriores de entrenamiento del modelo Mask R-CNN.

Creación de nuevo Dataset

El proceso de balanceo y augmentación de datos se aplicó exclusivamente al conjunto de entrenamiento, con el objetivo de mejorar la representación de las clases minoritarias. Los conjuntos de validación y prueba se mantuvieron sin modificaciones, garantizando una evaluación imparcial y confiable del desempeño del modelo sobre datos no vistos.

Se implementó un proceso de aumento de datos balanceado por clase sobre el conjunto de entrenamiento. Para ello, se definieron reglas de duplicación específicas para las categorías con menor representación (Cow-tongue, Kikuyo, Other y Dandelion), de modo que cada vez que una imagen contenía alguna de estas clases, se generaban copias adicionales aplicando transformaciones aleatorias como volteo horizontal y vertical, rotación en un rango de ±30° y ajustes de brillo y contraste.

Tabla 4 Aumento de datos del conjunto de entrenamiento

Clase	**Duplicados generados por imagen**	**Técnicas de aumento aplicadas**	**Objetivo del aumento**
Cow-tongue	×2	Inversión horizontal y vertical, rotación aleatoria (±30°), ajuste aleatorio de brillo y contraste	Incrementar representatividad de la clase con menor presencia visual.
Kikuyo	×3	Inversión horizontal/vertical, rotación aleatoria, ajuste de brillo y contraste	Compensar el desbalance de esta clase en el conjunto de entrenamiento.
Other	×1	Rotación aleatoria, ajuste de brillo/contraste	Aumentar la diversidad visual de la categoría "otras malezas".
Dandelion	×1	Inversión horizontal, ajuste de brillo/contraste	Mejorar la generalización del modelo frente a variaciones de iluminación.
Potato	×0	—	No se aplicó augmentación; clase base con suficiente representatividad.

Background	×0	—	No se modificó; corresponde al fondo sin vegetación.

Estas imágenes aumentadas, junto con sus máscaras correspondientes, se almacenaron en un nuevo directorio, garantizando así una mayor diversidad y un mejor equilibrio entre clases. Este procedimiento tuvo como finalidad reducir el desbalance del dataset y fortalecer la capacidad del modelo de segmentación para aprender patrones de las categorías minoritarias, mitigando posibles sesgos hacia las clases dominantes.

Comparación del dataset original y el dataset creado

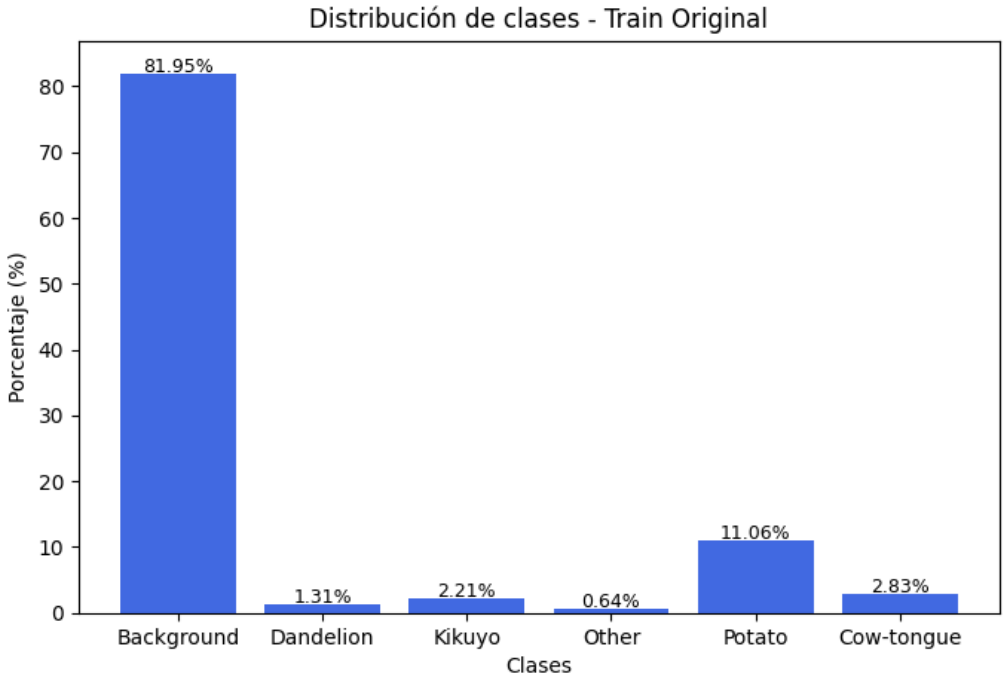

Fig. 5 Dataset original

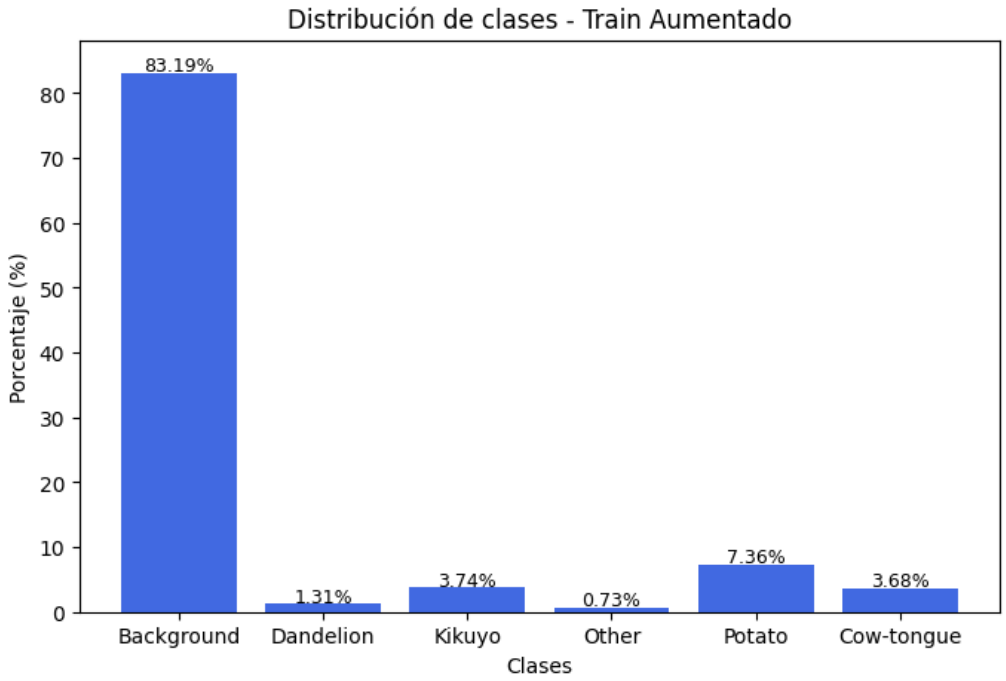

Fig. 6 Dataset modificado

Las figuras 5 y 6 muestran cómo el proceso de aumento de datos balanceado por clase impactó en la composición del conjunto de entrenamiento. En el dataset original predominaba ampliamente la clase Background (81.95%), seguida por Potato (11.06%), mientras que el resto de las clases apenas alcanzaban valores entre el 0.64% y el 2.83%. Tras aplicar la estrategia de oversampling dirigido, las clases minoritarias como Kikuyo y Cow-tongue incrementaron su participación relativa, reduciendo parcialmente el desbalance.

Aunque Background sigue siendo la categoría mayoritaria (83.19%), ahora existe una mayor representación proporcional de las clases minoritarias, lo que fortalece la capacidad del modelo para generalizar y evitar sesgos hacia las categorías dominantes.

Adecuación para usar Mask R-CNN

Para el entrenamiento con la arquitectura Mask R-CNN, es fundamental que las máscaras estén separadas por instancia y clase, ya que este modelo está diseñado específicamente para la segmentación de instancias, donde cada objeto debe estar claramente identificado de manera individual. Por esta razón, se desarrolló e implementó un script en Python que permite procesar las máscaras originales, las cuales contenían múltiples clases superpuestas en una sola imagen y generar máscaras binarias independientes para cada clase presente.

Además, durante este proceso se decidió ignorar la clase correspondiente al fondo (background), ya que no aporta información útil para el modelo y podría generar ruido en el aprendizaje. Esta separación facilita que el modelo identifique y aprenda a segmentar correctamente cada tipo de objeto o maleza de forma individual dentro del conjunto de datos.

```python
json_path = r"C:\Users\olive\Desktop\Tesis\Detectron2\dataset\via_region_data.json"
mask_input_dir = r"C:\Users\olive\Desktop\Tesis\Detectron2\dataset\masks\val"  # carpeta con tus PNG originales
output_mask_dir = r"C:\Users\olive\Desktop\Tesis\Detectron2\dataset\masks\val2"

os.makedirs(output_mask_dir, exist_ok=True)

# Cargar el diccionario de clases
with open(json_path, "r") as f:
    label_map = json.load(f)

# Convertir a: {"1": "Cow-tongue", ...}
label_dict = {k: v["class"] for k, v in label_map.items() if k != "0"}  # omitir fondo

# Iterar por cada máscara
for mask_filename in os.listdir(mask_input_dir):
    if not mask_filename.endswith(".png"):
        continue

    print(f"Procesando {mask_filename}")
    mask_path = os.path.join(mask_input_dir, mask_filename)
    mask_img = cv2.imread(mask_path, cv2.IMREAD_GRAYSCALE)

    base_name = os.path.splitext(mask_filename)[0]
    unique_labels = np.unique(mask_img)

    for label_value in unique_labels:
        label_str = str(label_value)
        if label_str == "0" or label_str not in label_dict:
            continue  # omitir fondo o valores no definidos

        class_name = label_dict[label_str]
        binary_mask = (mask_img == label_value).astype(np.uint8) * 255

        if np.sum(binary_mask) == 0:
            print(f" ⚠️  Máscara vacía para {class_name} en {mask_filename}")
            continue

        output_filename = f"{base_name}_{class_name}.png"
        output_path = os.path.join(output_mask_dir, output_filename)
        cv2.imwrite(output_path, binary_mask)
        print(f" ✅ Generado: {output_filename}")
```

Fig. 7 Código para la separación de las máscaras por clase

Fig. 8 Máscaras separadas por clases

2.5.Modelado

Para la fase 4 de la Metodología CRISP-DM se realizó el entrenamiento del Modelo con la arquitectura Mask-RCNN utilizando Backbone diferentes.

Configuración de entorno

Para llevar a cabo el entrenamiento del modelo con la arquitectura Mask R-CNN utilizando la implementación de Matterport, se configuró un entorno de desarrollo basado en Python 3.8 y TensorFlow 2.1, dado que esta versión es compatible con la biblioteca original. El entorno fue creado en un entorno virtual para mantener las dependencias aisladas y estables. Se instalaron las librerías requeridas como Keras, imgaug, OpenCV, scikit-image, entre otras.

Asimismo, se utilizó una GPU NVIDIA RTX 4050, aprovechando la aceleración por hardware mediante CUDA y cuDNN, lo cual permitió mejorar significativamente los tiempos de entrenamiento. El sistema operativo empleado fue Windows 11, y se empleó Jupyter Notebook como entorno de pruebas para el desarrollo interactivo de código y visualización de resultados. Este entorno permitió ejecutar entrenamientos, cargar datasets personalizados y visualizar predicciones de segmentación sobre las imágenes de validación.

Resumen de la arquitectura Mask RCNN

Tabla 5 Resumen de la aquitectura Mask R-CNN

Elemento	Valor
Nombre del modelo	mask_rcnn
Arquitectura base	ResNet-50
Componentes adicionales	FPN (Feature Pyramid Network), RPN (Region Proposal Network), ROIAlign, clasificación y segmentación
Total, de parámetros	44,684,470
Parámetros entrenables	44,625,206
Parámetros no entrenables	59,264
Capas convolucionales (aprox.)	Más de 150 capas
Entradas esperadas	Imagen RGB (None, None, 3)
Tamaño típico de entrada	128x128 (en entrenamiento por parches)
Salidas	Clases, cajas delimitadoras, máscaras segmentadas
N° de clases en salida	6 clases (Cow-tongue, Dandelion, Kikuyo, Other, Potato, fondo)
Función de pérdida	dice_ce_loss (combinación de Dice Loss y Categorical Crossentropy)
Tamaño de las máscaras	28x28 por instancia
Capa final de segmentación	TimeDistributed(Conv2D) con activación sigmoid

El proceso de entrenamiento del modelo Mask R-CNN se realizó en tres etapas progresivas para optimizar el aprendizaje y mejorar la precisión de la segmentación.

Primera Etapa.

En la primera etapa, se entrenaron únicamente las capas heads, que corresponden a las capas finales responsables de realizar las predicciones específicas del conjunto de datos personalizado, manteniendo congeladas las capas del backbone preentrenado.

Segunda Etapa

En la segunda etapa, se continuó el entrenamiento incluyendo las capas de las etapas superiores de la red (stage 4 y 5) del backbone, permitiendo una adaptación más profunda a las características visuales del nuevo dominio.

Tercera Etapa.

En la tercera etapa, se procedió a entrenar toda la red, lo que permitió un ajuste completo de todos los pesos del modelo. Esta estrategia gradual permitió estabilizar el aprendizaje, evitar el sobreajuste temprano y aprovechar el conocimiento transferido del entrenamiento previo en COCO.

Arquitectura Deeplabv3+

Con el objetivo de explorar una nueva arquitectura de segmentación que permita mejorar las métricas obtenidas previamente con el modelo Mask R-CNN.

Para ello, se configuró el entorno necesario para implementar la arquitectura DeepLabV3+, una red especializada en segmentación semántica que ha demostrado un excelente rendimiento en entornos de clasificación por píxel. Se optó por utilizar ResNet-50 como backbone, debido a su equilibrio entre profundidad, eficiencia computacional y precisión.

Resumen de la arquitectura Deeplabvplus

Tabla 6 Resumen del entrenamiento de Deeplabv3+

Elemento	**Descripción**
Nombre del modelo	DeepLabV3+
Backbone	ResNet-50 preentrenado en ImageNet
Tamaño de entrada	(128, 128, 3) — Imagen RGB
Número de clases	6 clases
Capas extraídas del backbone	conv4_block6_2_relu (deep features) y conv2_block3_2_relu (low-level)
Bloque ASPP	Convoluciones dilatadas con tasas 1, 6, 12, 18 + Pooling global
Profundidad ASPP	5 ramas paralelas, cada una con 256 filtros
Redimensionamiento ASPP	Redimensionamiento espacial para empatar dimensiones con el skip connection

Decoder	2 capas Conv2D de 256 filtros con activación ReLU
Skip connection	1 Conv2D de 48 filtros sobre feature bajo del backbone
Fusión decoder + skip	Concatenación + 2 convoluciones de 3x3
Salida del modelo	Conv2D con activación softmax (por clase)
Tamaño de salida	(128, 128, 6) — mapa de probabilidad por clase
Función de activación final	softmax

Con el objetivo de seguir optimizando el rendimiento del sistema de segmentación semántica centrada en la evaluación del modelo DeepLabV3+ utilizando una nueva arquitectura base: EfficientNetV2.

EfficientNetV2 ha sido ampliamente reconocido por su alta eficiencia y buen desempeño en tareas de visión por computadora, incluso con arquitecturas más compactas. Su incorporación como backbone en DeepLabV3+ representa una oportunidad para lograr un mejor balance entre rendimiento, velocidad y precisión, lo cual es especialmente valioso en escenarios de despliegue real o en dispositivos con recursos limitados.

Durante el entrenamiento del modelo, se utilizaron tres callbacks de Keras para mejorar el rendimiento y prevenir el sobreajuste. Estos mecanismos ofrecen un control dinámico sobre el proceso de aprendizaje y ayudan a mantener el mejor modelo que se ha logrado.

En primer lugar, se utilizó el callback ReduceLROnPlateau, que ajusta automáticamente la tasa de aprendizaje cuando una métrica específica deja de mejorar tras un cierto número de épocas. En este caso, se estuvo vigilando la métrica de validación mean IoU, configurando el modo en “max” porque queremos maximizar ese valor. El factor de reducción se estableció en 0.5, con una paciencia de 10 épocas y un límite mínimo de tasa de aprendizaje de 1e-6. Esta técnica ayuda a afinar el modelo a medida que se acerca a la convergencia.

Más adelante, se implementó el callback EarlyStopping, que tiene como objetivo detener el entrenamiento antes de tiempo si no se observa una mejora significativa en la pérdida de validación después de un número específico de iteraciones. Se definió una paciencia de 20 épocas y una mejora mínima de 0.001. Además, se activó la opción restore_best_weights=True,

lo que permite mantener los pesos de la mejor época alcanzada. Esta estrategia ayuda a prevenir el sobreentrenamiento y a mejorar la capacidad de generalización del modelo.

Finalmente, se implementó el callback ModelCheckpoint, que permite guardar automáticamente los pesos del modelo cada vez que se logra un mejor valor en la métrica que nos interesa. En este caso, se almacenó el modelo que alcanzó el mayor valor de IoU durante la validación.

El uso combinado de estos tres callbacks permitió un entrenamiento más eficiente, estable y controlado, garantizando la obtención del modelo con el mejor desempeño durante el proceso de validación.

2.6.Despliegue

Para la fase 6 del modelo Crisp-DM implica la integración del modelo de aprendizaje automático en un entorno de producción donde se pueda realizar predicciones en datos reales. El propósito es ofrecer un sistema que permita a los usuarios interactuar con el modelo de forma ágil.

Para llevar a cabo esta fase se aplicó la metodología ágil Scrum, la cual permitió organizar el desarrollo del sistema web de forma iterativa e incremental. Este enfoque facilitó la planificación de tareas en Sprints cortos, garantizando una mejor gestión del tiempo.

Fase de pre-juego
Asignación de roles.

En la tabla 7 se presenta a los miembros clave que intervienen en el desarrollo del presente trabajo, asignando sus respectivos nombres, roles dentro de la metodología Scrum y cargos institucionales en el contexto académico de la Universidad Técnica del Norte (UTN). Esta distribución permite establecer responsabilidades claras durante la planificación, ejecución y evaluación del proyecto, garantizando así una correcta gestión de este.

Tabla 7 Roles del proyecto

Nombre	Rol	Cargo
Iván García	Product Owner	Director del trabajo de Grado (Docente UTN)
Oliver Zamora	Scrum Máster, Equipo de desarrollo.	Tesista (Estudiante de la Carrera de software)

Marco Pusdá	Stakeholder	Asesor del trabajo de Grado (Docente UTN)

Requerimientos del sistema.

Para la definición de los requisitos del sistema se optó por el uso de historias de usuario, dado que esta técnica permite expresar las necesidades del usuario final de una forma clara, sencilla y orientada al valor. Las historias de usuario describen qué necesita hacer el usuario, por qué lo necesita y cuál es el beneficio que se espera obtener, lo cual se alinea adecuadamente con los principios de la metodología Scrum adoptada en el presente trabajo.

Tabla 8 Historia de usuario 1

Código: HU01	**Título: Cargar imagen desde el dispositivo**	**Prioridad: Alta**
Descripción:	Como usuario necesito cargar una imagen desde mi computadora para que el sistema pueda analizar las malezas presentes en el cultivo.	
Criterios de aceptación:	CA1: Se debe permitir seleccionar archivos JPG o PNG. CA3: Si el archivo no es válido, se debe mostrar un mensaje de advertencia.	

Tabla 9 Historia de usuario 2

Código: HU02	**Título: Visualizar resultados segmentados**	**Prioridad: Alta**
Descripción:	Como usuario necesito visualizar la imagen procesada con las malezas detectadas, para poder comparar con la imagen original.	

Criterios de aceptación:	CA1: Se debe mostrar la imagen original junto con la imagen segmentada. CA2: Las zonas con maleza deben estar delimitadas.

Tabla 10 Historia de usuario 3

Código: HU03	**Título: Obtener métricas del análisis**	**Prioridad: Media**
Descripción:	Como usuario necesito conocer las métricas del análisis realizado, para evaluar la precisión del modelo.	
Criterios de aceptación:	CA1: Se debe mostrar la métrica al finalizar el análisis. CA2: Las métricas deben estar expresadas en porcentaje y ser fácilmente interpretables.	

Tabla 11 Historia de usuario 4

Código: HU04	**Título: Descargar imagen segmentada**	**Prioridad: Media**
Descripción:	Como usuario necesito descargar la imagen segmentada con las malezas detectadas para tener una referencia sin conexión.	
Criterios de aceptación:	CA1: Debe haber un botón para descargar la imagen procesada. CA2: La imagen descargada debe conservar las delimitaciones generadas por el modelo.	

Tabla 12 Historia de usuario 5

Código: HU05	**Título: Procesamiento automático con el modelo.**	**Prioridad: Alta**
Descripción:	Como usuario necesito que el sistema analice automáticamente la imagen cargada utilizando un modelo de segmentación, para identificar las zonas con presencia de malezas.	

Criterios de aceptación:	CA1: El sistema debe ejecutar el modelo sobre la imagen cargada. CA2: El procesamiento debe iniciarse automáticamente una vez cargada la imagen válida. CA3: Si ocurre un error en la inferencia, debe mostrarse un mensaje de advertencia.

Product Backlog

A partir de las historias de usuario definidas previamente, se construyó el Product Backlog, que representa una lista priorizada de funcionalidades que deben ser desarrolladas para cumplir con los objetivos del sistema. Cada ítem del backlog ha sido estimado en puntos de historia, considerando la complejidad y la prioridad de cada funcionalidad. A continuación, se detalla el Product Backlog del sistema propuesto en la Tabla 13.

Tabla 13 Product Backlog del sistema

Código	Historia de Usuario	Prioridad	Estimación (pts)
HU01	Como usuario necesito cargar una imagen desde mi computadora para que el sistema pueda analizar las malezas presentes en el cultivo.	Alta	5
HU02	Como usuario necesito visualizar la imagen procesada con las malezas detectadas, para poder comparar con la imagen original.	Alta	5
HU03	Como usuario necesito conocer las métricas del análisis realizado de la imagen.	Media	3
HU04	Como usuario necesito descargar la imagen segmentada con las malezas detectadas para tener una referencia sin conexión.	Media	3

HU05	Como usuario necesito que el sistema analice automáticamente la imagen cargada utilizando un modelo de segmentación, para identificar las zonas con presencia de malezas.	Alta	10

Definición de la Arquitectura

La arquitectura de software propuesta en la figura 9 para este proyecto responde a un enfoque modular y escalable, basado en una arquitectura Cliente-Servidor. En esta se distingue claramente el frontend, desarrollado en Angular, encargado de la interacción con el usuario, y el backend, implementado con Flask en Python, responsable del procesamiento de imágenes y la ejecución del modelo de segmentación.

El sistema ha sido diseñado para que el usuario pueda cargar imágenes capturadas por drones a través de una interfaz web intuitiva, las cuales son enviadas al servidor mediante solicitudes API. Una vez recibida la imagen, el backend la procesa y realiza inferencias utilizando la red neuronal, devolviendo los resultados segmentados y las métricas correspondientes al usuario final.

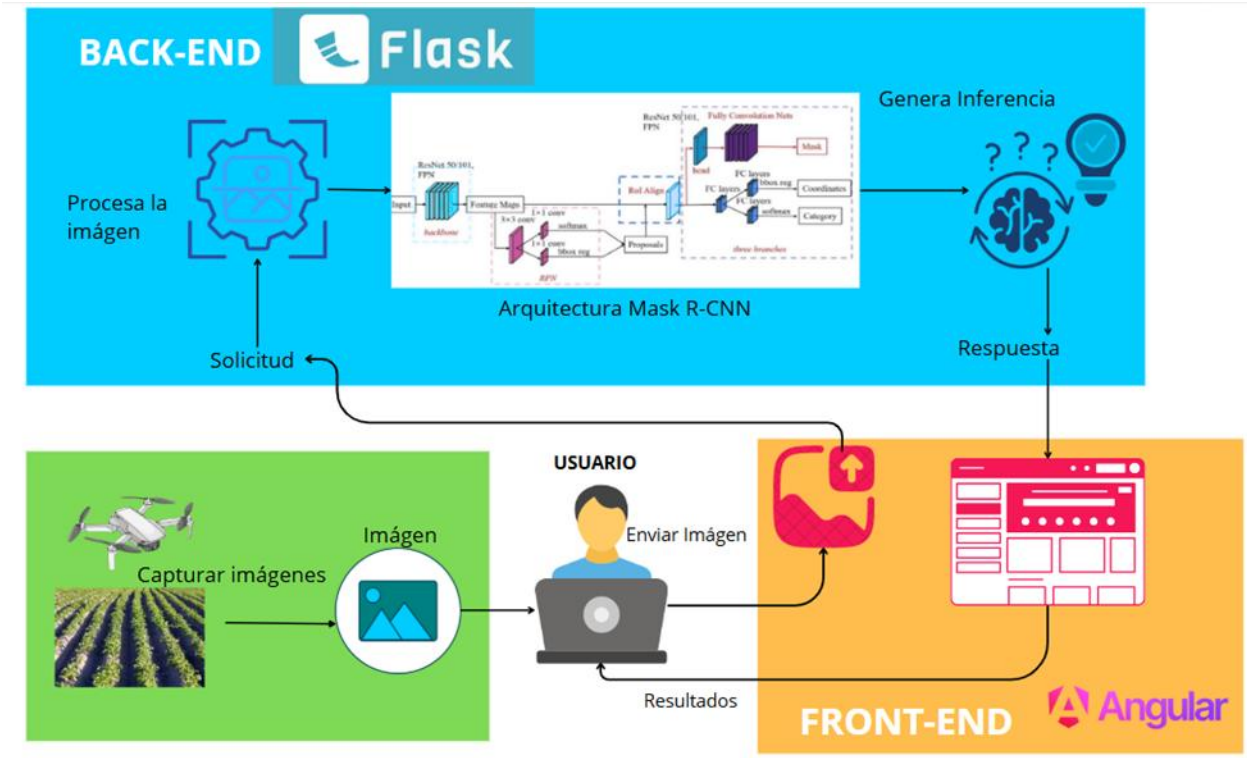

Fig. 9 Arquitectura del sistema

Fase de Juego

Sprint 1

Tabla 14 Planificación Sprint 1

Código	Nombre	Tarea
HU01	Cargar imagen desde el dispositivo	Crear endpoint de carga de imagen (/upload)
HU01	Cargar imagen desde el dispositivo	Validación del archivo de entrada
HU01	Cargar imagen desde el dispositivo	Dividir la imagen en subimágenes de 128×128 px
HU01	Cargar imagen desde el dispositivo	Agregar soporte para carga directa de imágenes 128×128 px
HU05	Procesamiento automático con DeeplabV3+	Cargar el modelo
HU05	Procesamiento automático con DeeplabV3+	Crear función de inferencia

HU05	Procesamiento automático con DeeplabV3+	Implementar endpoint /predict-deeplab (imagen completa con subdivisión)
HU05	Procesamiento automático con DeeplabV3+	Implementar endpoint /predict-deeplab-tile (imágenes 128×128)
HU02	Visualizar resultados segmentados	Generar imagen segmentada
HU02	Visualizar resultados segmentados	Crear endpoint /predict
HU03	Obtener métricas del análisis	Calcular métricas de segmentación
HU03	Obtener métricas del análisis	Crear endpoint /metrics
HU04	Descargar imagen segmentada	Guardar imagen procesada en carpeta temporal
HU04	Descargar imagen segmentada	Crear endpoint /download

- Explicación del Sprint:

El propósito de este Sprint es implementar la lógica de procesamiento en el servidor, construyendo un backend funcional que permita la carga de imágenes, su análisis mediante el modelo, el cálculo de métricas de evaluación y la entrega de resultados al usuario.

Este desarrollo constituye la base operativa del sistema, permitiendo que desde cualquier interfaz se pueda interactuar con el modelo de forma automatizada, precisa y accesible.

- Incremento

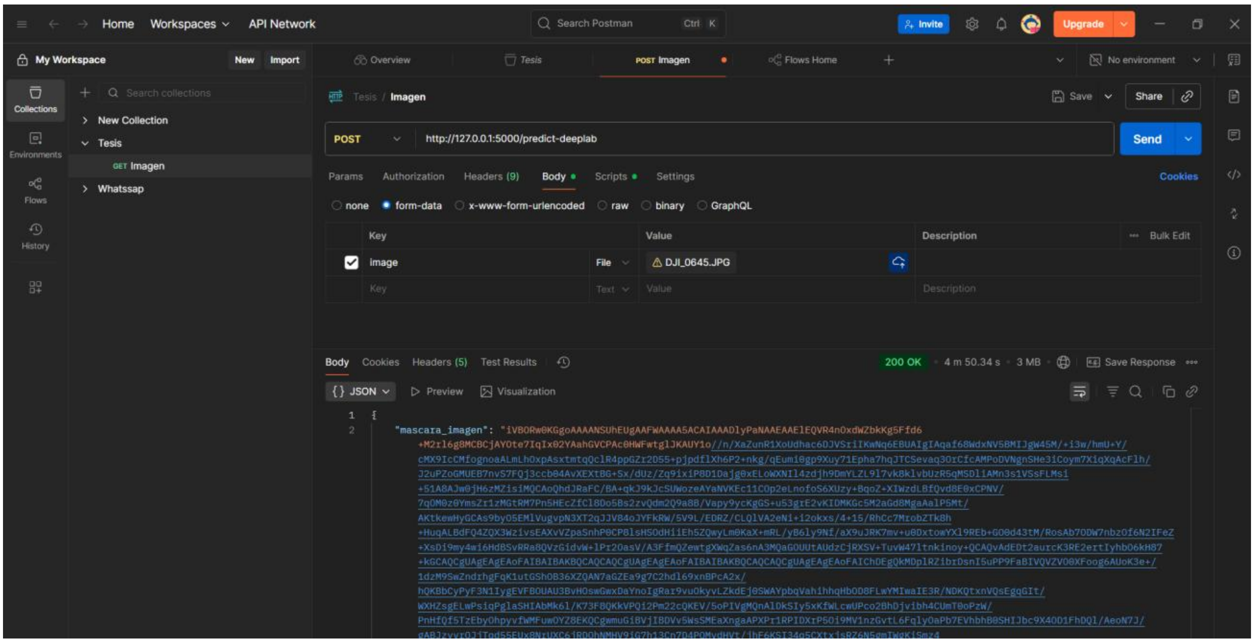

Fig. 10 Endpoint de la división y prediccion de las mascaras

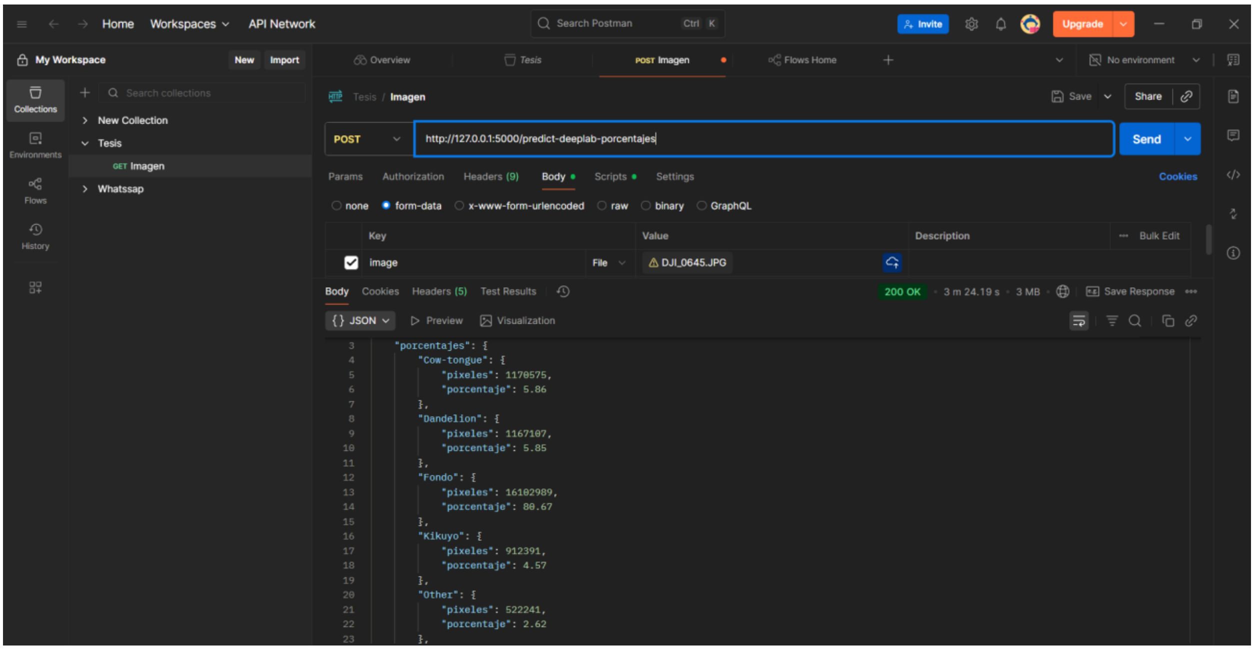

Fig. 11 Endpoint del cálculo de porcentaje de malezas de toda la imagen

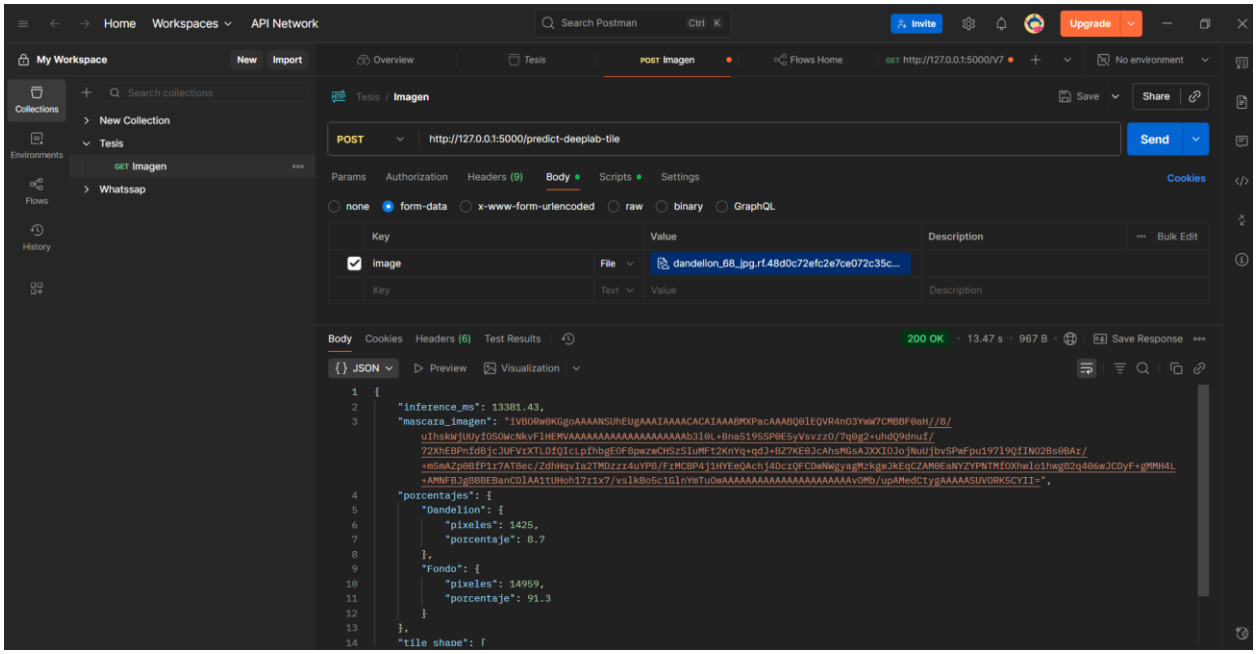

Fig. 12 Endpoint de la predicción de la imagen de 128 pixeles

Para la implementación de este Sprint se desarrollaron tres API especializadas: la primera destinada al procesamiento de imágenes de 128×128 píxeles, la segunda orientada al análisis de imágenes completas capturadas directamente desde el dron, y una tercera responsable de calcular el porcentaje correspondiente a cada clase dentro de la imagen procesada.

Sprint 2

Tabla 15 Planificación Sprint 2

Código	Nombre	Tarea
HU01	Cargar imagen desde el dispositivo	Crear componente de carga de imagen
HU01	Cargar imagen desde el dispositivo	Consumir endpoint /upload del backend
HU01	Cargar imagen desde el dispositivo	Mostrar imagen original y procesada
HU03	Obtener métricas del análisis	Llamar al endpoint -porcentajes por cada clase de maleza.
HU04	Descargar imagen segmentada	Habilitar botón de descarga de resultado

- Explicación del Sprint

Implementar una interfaz gráfica funcional que permita a los usuarios interactuar con el sistema de segmentación de malezas de forma intuitiva. Esta interfaz debe permitir cargar imágenes, visualizar los resultados procesados, consultar métricas del análisis, y descargar las imágenes segmentadas, todo mediante llamadas al backend previamente implementado.

- Incremento

Fig. 13 Interfaz para subir y procesar imagen

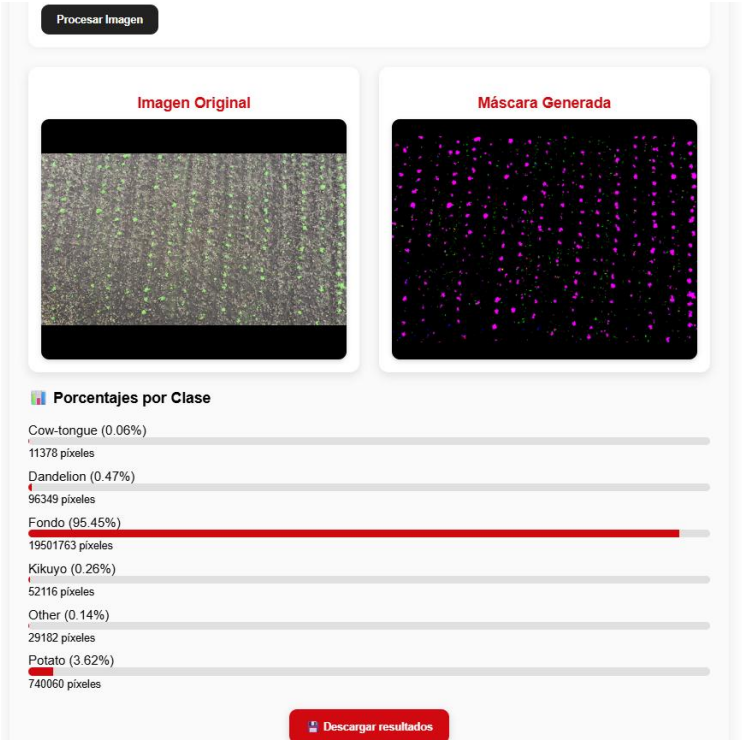

Fig. 14 Resultado de la imagen capturado por dron

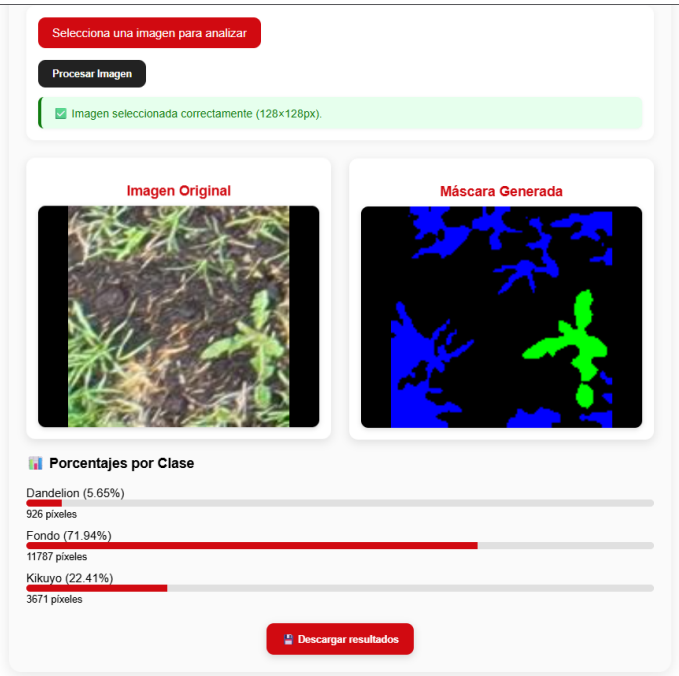

Fig. 15 Vista de la masca generada de las imágenes 128px

La interfaz gráfica desarrollada incorpora un mecanismo de selección que permite al usuario elegir el tipo de imagen a procesar, ya sea una imagen individual de 128×128 píxeles o una captura completa obtenida directamente desde el dron. En función de esta elección, el

sistema ejecuta automáticamente una de las dos API implementadas en el Sprint anterior, generando como resultado la máscara segmentada correspondiente junto con su leyenda de colores y los porcentajes asociados a cada clase identificada, así como un botón para descargar.

Pot-juego

Durante esta fase, se realizó el despliegue de la arquitectura del sistema, con el propósito de poner en funcionamiento el prototipo funcional de la aplicación web propuesta.

Para ello, se empleó Docker como herramienta de contenedorización, lo que permitió aislar cada componente del sistema y garantizar un entorno de ejecución homogéneo e independiente del sistema operativo o las configuraciones locales. Se utilizaron servicios en la nube proporcionados por Microsoft Azure, lo que permitió alojar tanto el backend desarrollado en Flask como el frontend construido en Angular.

Despliegue del Backend

Primero, se creó una imagen de Docker con el siguiente archivo Dockerfile, donde se configuraron las dependencias necesarias del modelo de segmentación y el servicio Flask

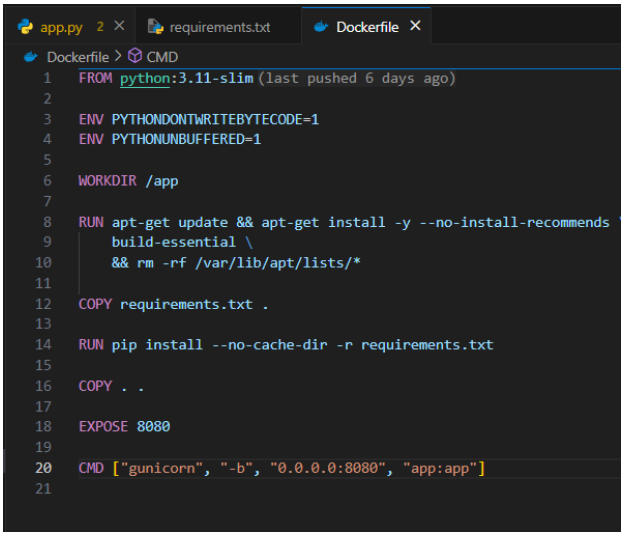

Fig. 16 Archivo Dockerfile

Una vez generada la imagen con el comando docker build -t flask-api:latest . Esta fue publicada en Docker Hub bajo el nombre opzamoraf/flask-api:latest como se muestran en la figura 17.

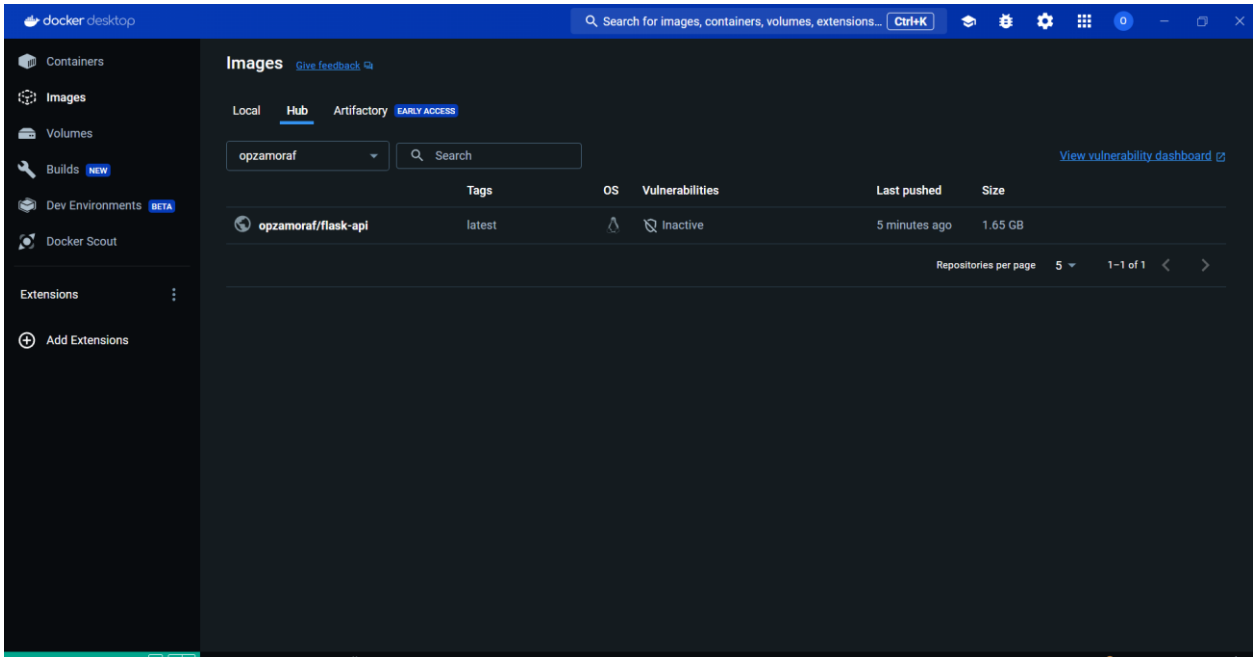

Fig. 17 Imagen subida a Docker Hub

Posteriormente se desplegó en el servicio Azure Container Apps con el siguiente comando:

az containerapp up --name flask-api --resource-group tesis-backend --image opzamoraf/flask-api:latest --target-port 8080 --ingress external.

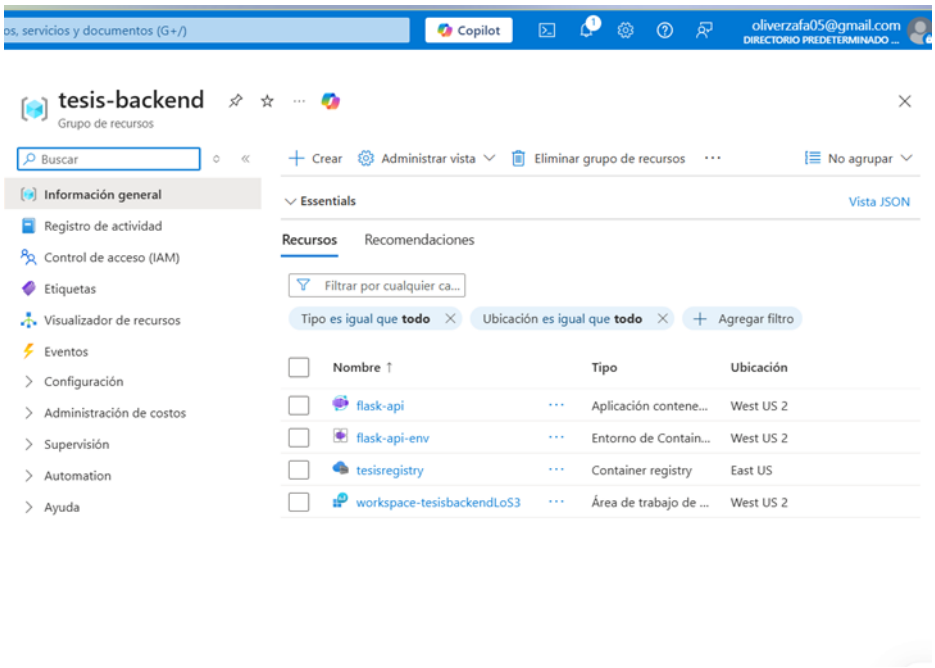

Fig. 18 Backen Desplegado

Despliegue del Front End

Para la capa de presentación, desarrollada en Angular, se realizó el proceso de compilación en modo producción mediante el comando que se muestra en la figura 19:

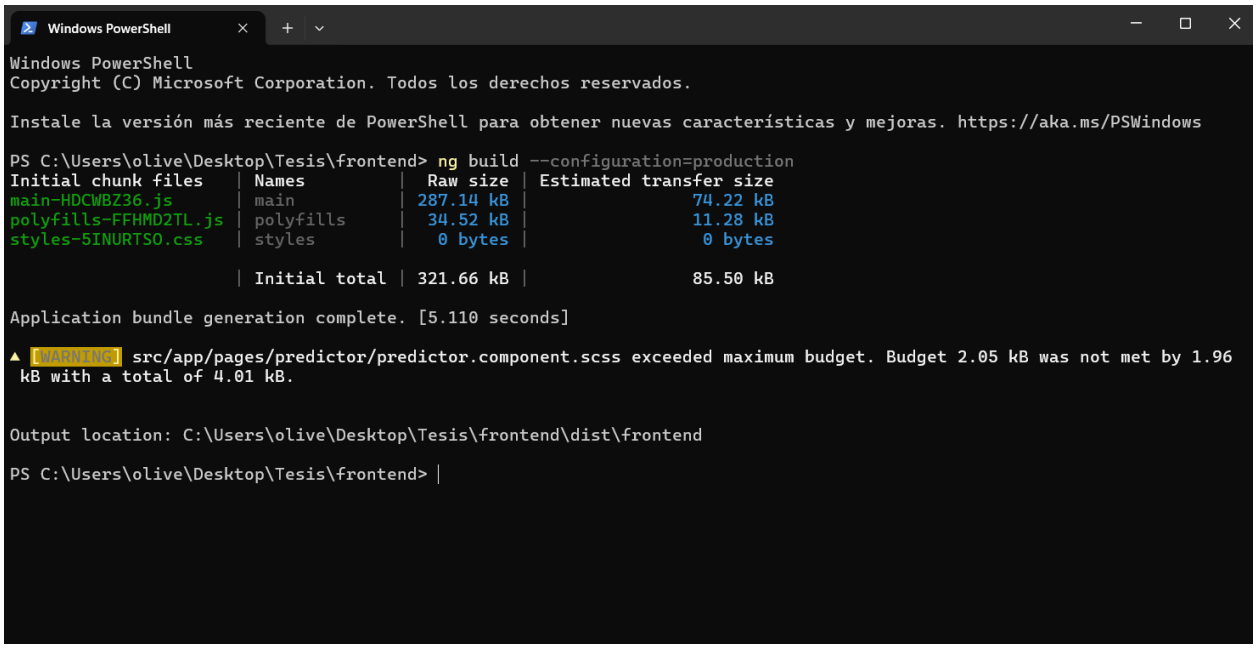

Fig. 19 Compilación en modo producción

Los archivos resultantes se almacenaron en la carpeta dist/frontend/browser, la cual fue publicada como sitio web estático utilizando Azure Storage Account.

Primero, se creó una cuenta de almacenamiento en Azure con el siguiente comando:

az storage account create --name tesisfrontendstorageopz --resource-group tesis-backend --location eastus2 --sku Standard_LRS --kind StorageV2

Posteriormente, se habilitó la funcionalidad de sitio web estático:

az storage blob service-properties update --account-name tesisfrontendstorageopz --static-website --index-document index.html --404-document index.html

Finalmente, los archivos compilados fueron cargados al contenedor $web con el siguiente comando:

az storage blob upload-batch –account-name tesisfrontendstorageopz –destination '$web' –source ./dist/frontend/browser

En la Figura 20 se muestra la interfaz web desplegada en la nube, accesible desde la URL:

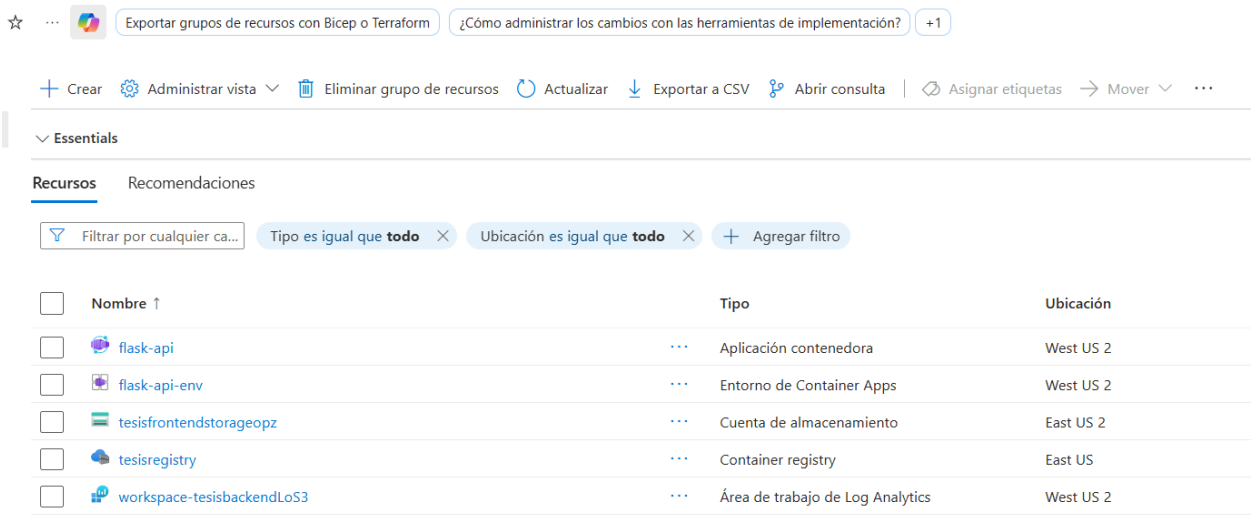

Fig. 20 Despliegue del front-end

III. VALIDACION

La validación de resultados en este trabajo se enfoca en evaluar la precisión y eficiencia del modelo de segmentación semántica desarrollado, así como del sistema automatizado propuesto. Esta evaluación se estructura en dos niveles: primero, se realiza una comparación entre los resultados obtenidos en los distintos entrenamientos del modelo, utilizando diversas arquitecturas y backbones como Mask R-CNN y DeepLabV3+, y contrastándolos además con los resultados obtenidos por compañeros en trabajos anteriores que abordaron problemáticas similares. En segundo lugar, se analiza el desempeño del sistema en cuanto al tiempo de procesamiento, midiendo el intervalo requerido para segmentar una imagen de entrada. A continuación, se describen en detalle cada uno de estos procesos de validación

3.1.Resultados Obtenidos.

Modelo Mask R-CNN

Como se mencionó en el capítulo anterior el modelo se entrenó en 3 etapas diferentes. A continuación, se exponen los resultados de las diferentes etapas.

Primera Etapa.

La tabla 16 presenta los resultados con el conjunto de entrenamiento y validación obtenidos en la primera etapa del entrenamiento. Adicionalmente en la tabla 17 se muestran los resultados obtenidos por clase de esta etapa.

Tabla 16 Resultados generales de la primera fase

Cojunto	Mean IOU	Mean Dice Coefficien	Mean Dice Loss	Mean Accuracy	Mean Precision	Mean Recall	Mean F1 Score
Validación	0.6572	0.7620	0.2380	0.9459	0.8544	0.7259	0.7620
Entrenamiento	0.6900	0.7870	0.2130	0.9511	0.8636	0.7546	0.7870

Tabla 17 Resultados por clase de la primera fase

Clase	Mean IOU	Mean Dice Coefficien	Mean Dice Loss	Mean Accuracy	Mean Precision	Mean Recall	Mean F1 Score
Background	0.9411	0.9680	0.0320	0.9594	0.9571	0.9805	0.9680
Dandelion	0.6947	0.8008	0.1992	0.9686	0.8482	0.7807	0.8008
Potato	0.7577	0.8385	0.1615	0.9550	0.9171	0.8042	0.8385
Kikuyo	0.5543	0.6945	0.3055	0.9266	0.7762	0.6513	0.6945
Other	0.5163	0.6166	0.3834	0.8675	0.7151	0.5897	0.6166
Cow-tongue	0.6371	0.7647	0.2353	0.9535	0.8456	0.7168	0.7647

Segunda etapa.

En la tabla 18 se presentan los resultados obtenidos de la segunda etapa. En la tabla 19 se presentan los resultados obtenidos por clase.

Tabla 18 Resultados de la segunda fase

Cojunto	Mean IOU	Mean Dice Coefficien	Mean Dice Loss	Mean Accuracy	Mean Precision	Mean Recall	Mean F1 Score
Validación	0.6887	0.7896	0.2104	0.9543	0.8715	0.7513	0.7896
Entrenamiento	0.7397	0.8304	0.1696	0.9648	0.8919	0.7953	0.8304

Tabla 19 Resultados por clase de la segunda fase

Clase	Mean IOU	Mean Dice Coefficien	Mean Dice Loss	Mean Accuracy	Mean Precision	Mean Recall	Mean F1 Score
Background	0.9404	0.9676	0.0324	0.9588	0.9543	0.9828	0.9676
Dandelion	0.7511	0.8483	0.1517	0.9868	0.8851	0.8274	0.8483

Potato	0.7932	0.8711	0.1289	0.9635	0.9376	0.8309	0.8711
Kikuyo	0.5720	0.7067	0.2933	0.9251	0.7866	0.6614	0.7067
Other	0.5496	0.6474	0.3526	0.8706	0.7561	0.6062	0.6474
Cow-tongue	0.6531	0.7768	0.2232	0.9556	0.8398	0.7384	0.7768

Tercera etapa

En la tabla 20 se presentan los resultados obtenidos en la última etapa del entrenamiento. En la tabla 21 se presentan los resultados obtenidos por clase.

Tabla 20 Resultados de la tercera fase

Cojunto	**Mean IOU**	**Mean Dice Coefficien**	**Mean Dice Loss**	**Mean Accuracy**	**Mean Precision**	**Mean Recall**	**Mean F1 Score**
Validación	0.7067	0.8050	0.1950	0.9591	0.8812	0.7657	0.8050
Entrenamiento	0.7659	0.8508	0.1492	0.9699	0.9089	0.8137	0.8508

Tabla 21 Resultados por clase de la tercera fase

Clase	**Mean IOU**	**Mean Dice Coefficien**	**Mean Dice Loss**	**Mean Accuracy**	**Mean Precision**	**Mean Recall**	**Mean F1 Score**
Background	0.9421	0.9685	0.0315	0.9600	0.9562	0.9827	0.9685
Dandelion	0.7632	0.8573	0.1427	0.9871	0.9048	0.8250	0.8573
Potato	0.8061	0.8794	0.1206	0.9658	0.9305	0.8495	0.8794
Kikuyo	0.5718	0.7082	0.2918	0.9311	0.7988	0.6579	0.7082
Other	0.5892	0.6881	0.3119	0.9018	0.7842	0.6442	0.6881
Cow-tongue	0.6613	0.7805	0.2195	0.9471	0.8505	0.7361	0.7805

En la figura 21 se visualiza el resultado de la predicción y la máscara real de una imagen de 128 x 128 pixeles.

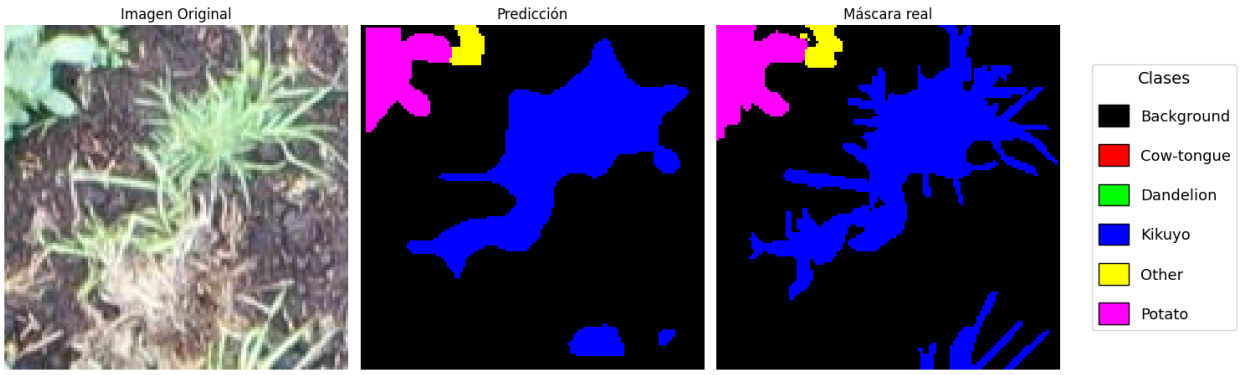

Fig. 21 Resultado de segmentación semántica con la arquitectura Mask R-CNN

En la Figura 22, se presentan los resultados de la predicción del modelo Mask R-CNN aplicado a una imagen aérea del cultivo de papa. En esta figura, se pueden ver las máscaras que el modelo ha generado, junto con los cuadros delimitadores y las etiquetas de clase que corresponden a cada objeto detectado. El modelo logró identificar correctamente las áreas que pertenecen a las clases de papa y maleza. Las máscaras de color destacan las zonas segmentadas para cada clase, mientras que los cuadros discontinuos marcan las áreas de detección.

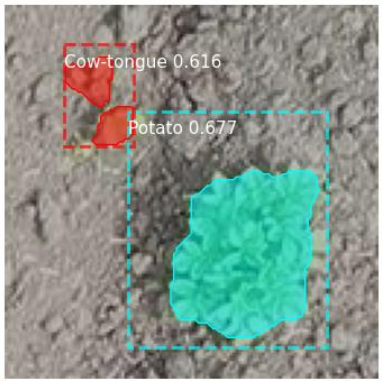

Fig. 22 Resultado de predicción del modelo Mask R-CNN.

Modelo Deeplabv3+

Con este modelo se probó con el backbone resnet 50 y el eficienetV2 con sus versiones B0, B1 Y B2. A continuación se presentan los resultados obtenidos con los diferentes backbones.

Resnet 50

En las figuras 23 y 24, podemos ver las curvas de precisión (accuracy) y pérdida (loss) que se generaron durante el entrenamiento del modelo con el backbone ResNet50. Se nota que la precisión se estabiliza rápidamente en las primeras épocas, alcanzando valores cercanos a

1.0, lo que indica que el modelo está aprendiendo bien las características distintivas de las clases en el conjunto de entrenamiento. La pérdida de entrenamiento va disminuyendo de manera constante hasta llegar a valores mínimos, mientras que la pérdida de validación muestra algunas oscilaciones al principio antes de estabilizarse. Esto es algo común en los modelos de aprendizaje profundo, ya que implica el ajuste de pesos y la regularización.

La coherencia entre las curvas de entrenamiento y validación sugiere que el modelo encuentra un buen equilibrio entre aprender y generalizar, sin señales claras de sobreajuste. En resumen, el comportamiento de estas curvas demuestra la efectividad del backbone ResNet50 para extraer características relevantes, lo que permite al modelo lograr un rendimiento satisfactorio en la segmentación de las clases de interés dentro del conjunto de imágenes aéreas de cultivos de papa.

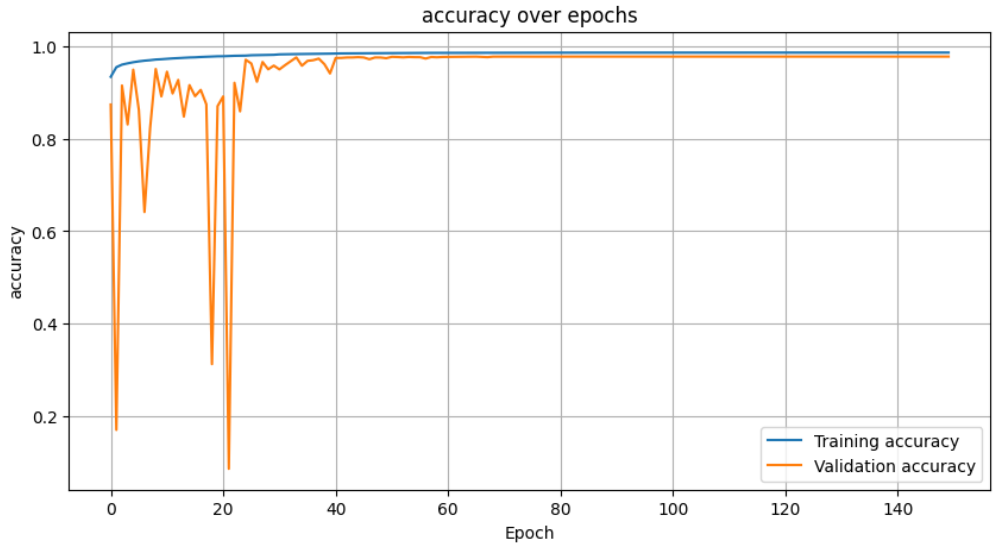

Fig. 23 Curva de accuracy durante el entrenamiento con resnet 50

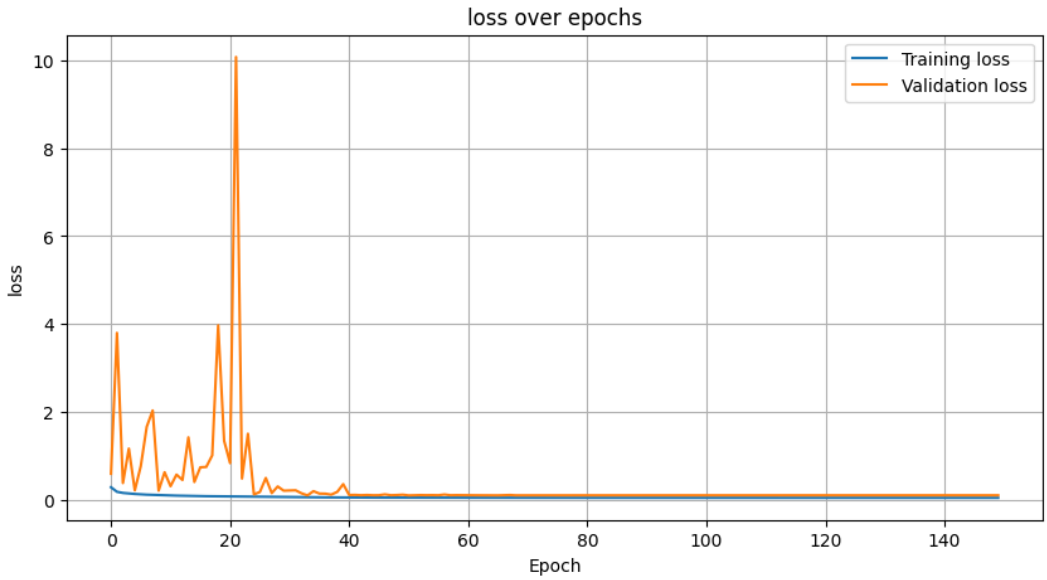

Fig. 24 Curva de perdida durante el entrenamiento con resnet 50

En la tabla 22 se muestran los resultados obtenidos con los conjuntos de entrenamiento, validación y prueba. En la tabla 23 se presentan los resultados por clase obtenidos con el conjunto de pruebas.

Tabla 22 Resultados obtenidos deeplavb3+ con backbone resnet 50

Cojunto	Mean IOU	Mean Dice Coefficien	Mean Dice Loss	Mean Accuracy	Mean Precision	Mean Recall	Mean F1 Score
Validación	0.8501	0.9160	0.084	0.9092	0.9239	0.9092	0.9160
Entrenamiento	0.8653	0.9259	0.074	0.9821	0.9294	0.9896	0.9259

Tabla 23 Resultados del conjunto de pruebas del modelo Deeplabv con Backcone Resnet 50

Clase	Mean IOU	Mean Dice Coefficien	Mean Dice Loss	Mean Accuracy	Mean Precision	Mean Recall	Mean F1 Score
Background	0.9789	0.9893	0.0107	0.9895	0.9891	0.9895	0.9893
Dandelion	0.8375	0.9115	0.0885	0.9153	0.9078	0.9153	0.9115

Potato	0.8061	0.9719	0.0281	0.9735	0.9703	0.9735	0.9719
Kikuyo	0.7997	0.8887	0.1113	0.9011	0.8766	0.9011	0.8887
Other	0.6871	0.8145	0.1855	0.7687	0.8662	0.7687	0.8145
Cow-tongue	0.8520	0.9200	0.08	0.9069	0.9336	0.9069	0.9200

En la figura 25 se visualiza el resultado de la predicción y la máscara real de una imagen de 128 x 128 pixeles.

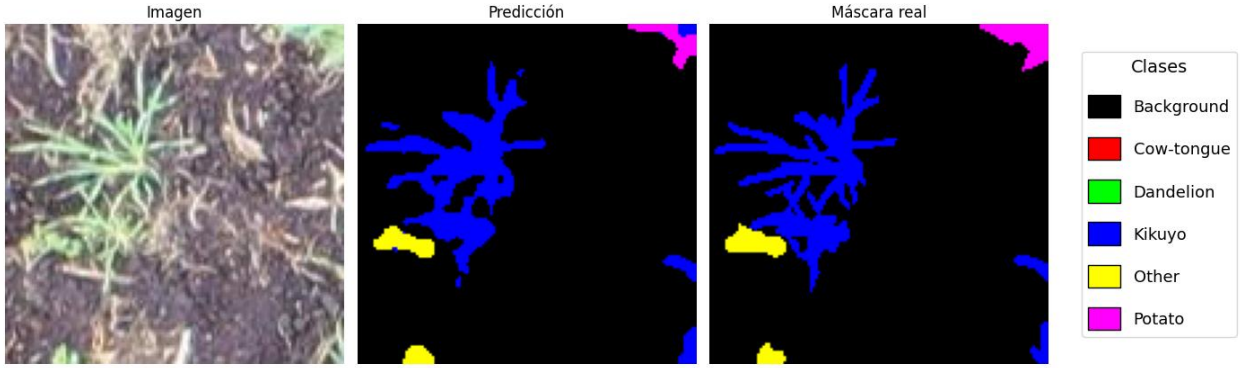

Fig. 25 Resultado de predicción con Backbone Resnet 50

EficientnetV2-B0

La figura 26 y 27 representan la curva de precisión y perdida durante el entrenamiento respectivamente. Se observa que el modelo alcanza una precisión elevada desde las primeras épocas, con una tendencia estable y convergente hacia valores cercanos al 0.98 tanto en entrenamiento como en validación.

Por su parte, la curva de pérdida muestra una disminución progresiva y sostenida, alcanzando valores mínimos y estables en las últimas épocas, lo cual refleja la correcta convergencia del modelo. Las ligeras fluctuaciones observadas en las primeras iteraciones son normales debido al ajuste inicial de los pesos

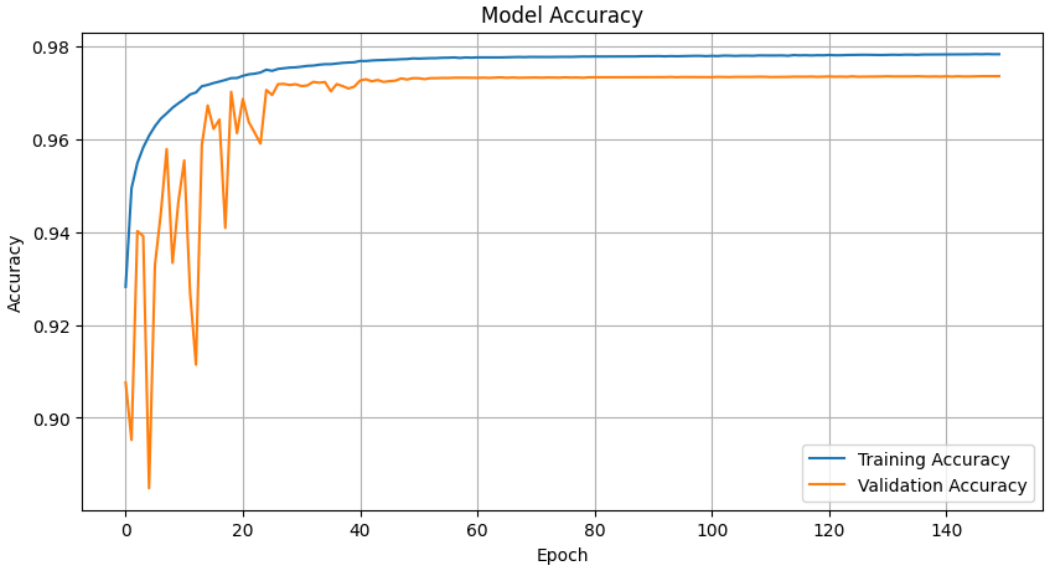

Fig. 26 Curva de accuracy durante el entrenamiento con Efincientnetv2-B0

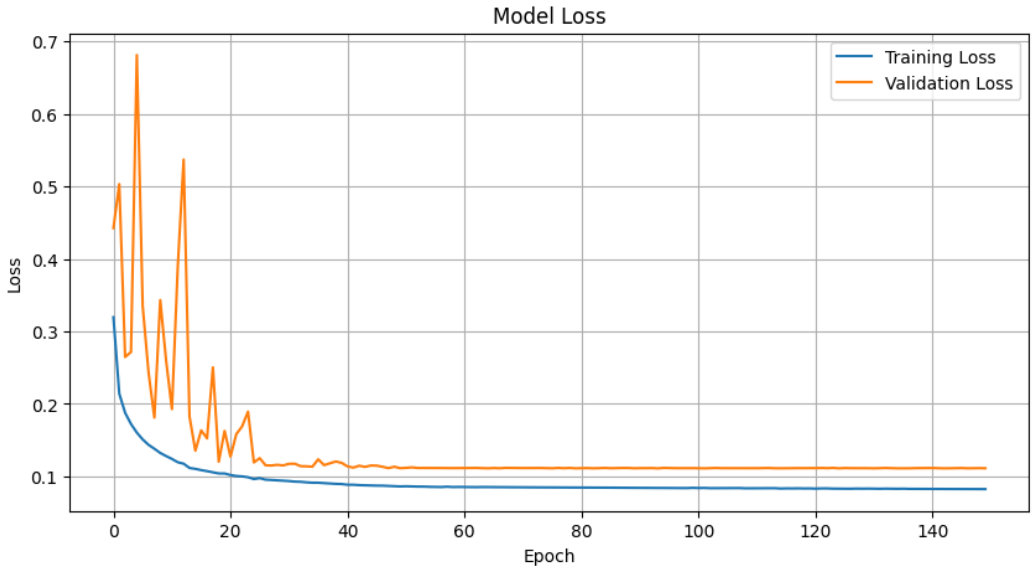

Fig. 27 Curva de perdida durante el entrenamiento con Efincientnetv2-B0

En la tabla 24 se expone los resultados obtenidos con el conjunto de entrenamiento, validación y pruebas, adicionalmente en la tabla 25 se presenta los resultados por clase obtenidos del conjunto de pruebas.

Tabla 24 Resultados obtenidos deeplavb3+ con backbone eficientnetV2-B0

Conjunto	**Mean IoU**	**Dice Coeff.**	**Dice Loss**	**Accuracy**	**Precision**	**Recall**	**F1 Score**
Entrenamiento	0.9201	0.9578	0.0422	0.9572	0.9584	0.9572	0.9578
Validación	0.9059	0.9494	0.0506	0.9452	0.9545	0.9452	0.9494

Pruebas	0.9169	0.9559	0.0441	0.9546	0.9573	0.9546	0.9559

Tabla 25 Resultados del conjunto de pruebas del modelo Deeplabv con Backcone Efincientnetv2-B0

Clase	Mean IOU	Mean Dice Coefficient	Mean Dice Loss	Mean Accuracy	Mean Precision	Mean Recall	Mean F1 Score
Background	0.9874	0.9936	0.0064	0.9934	0.9939	0.9934	0.9936
Dandelion	0.9077	0.9516	0.0484	0.9549	0.9484	0.9549	0.9516
Potato	0.9674	0.9834	0.0166	0.9855	0.9814	0.9855	0.9834
Kikuyo	0.8581	0.9236	0.0764	0.9249	0.9224	0.9249	0.9236
Other	0.8505	0.9192	0.0808	0.9036	0.9353	0.9036	0.9192
Cow-tongue	0.9303	0.9639	0.0361	0.9653	0.9625	0.9653	0.9639

En la figura 28 se visualiza el resultado de la predicción y la máscara real de una imagen de 128 x 128 pixeles.

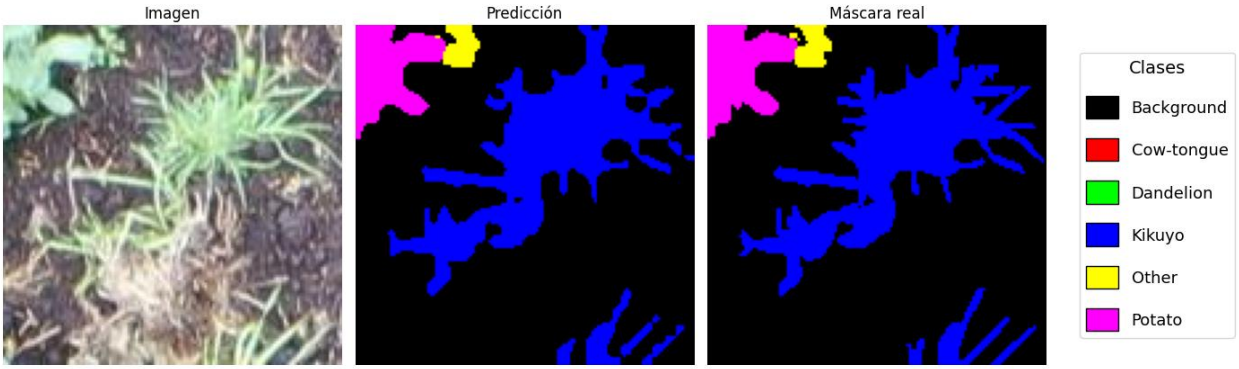

Fig. 28 Resultado de predicción con Backbone EficientnetV2-B0

EficientnetV2-B1

Las figuras 29 y 30 muestran la curva de precisión y pérdida que se obtuvo durante el entrenamiento con este backbone. Se puede notar que, después de una fase inicial de altibajos en las primeras épocas, la precisión, tanto en el entrenamiento como en la validación, se estabiliza rápidamente, alcanzando valores cercanos a 1.0. En lo que respecta a la curva de

pérdida, se observa una disminución constante hasta que se estabiliza en valores mínimos, lo que indica que los pesos de la red se han optimizado correctamente. Las fluctuaciones iniciales en la pérdida de validación reflejan el proceso de ajuste del modelo antes de que se logre la convergencia.

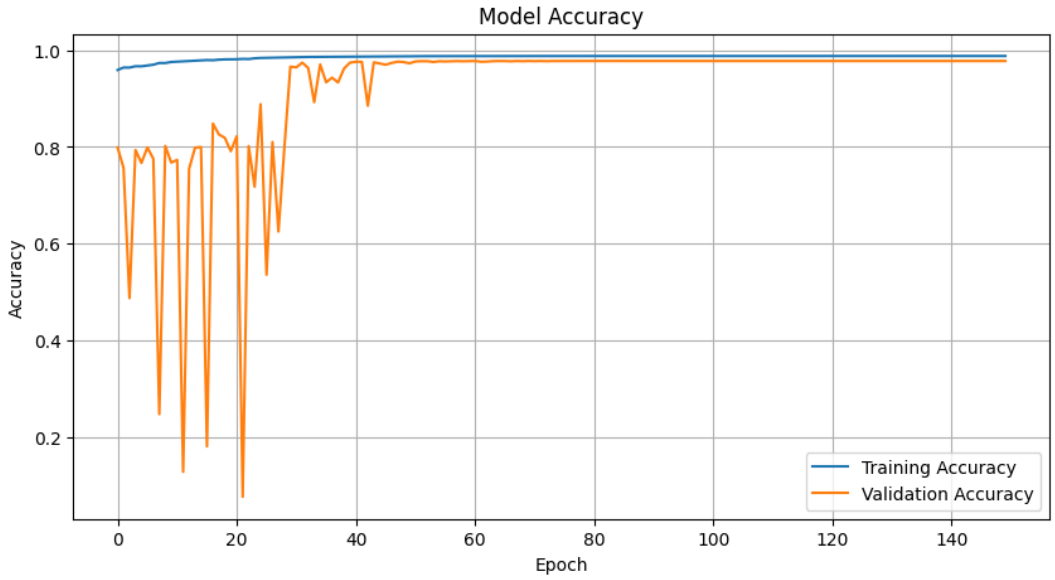

Fig. 29 Curva de accuracy durante el entrenamiento con Efincientnetv2-B1

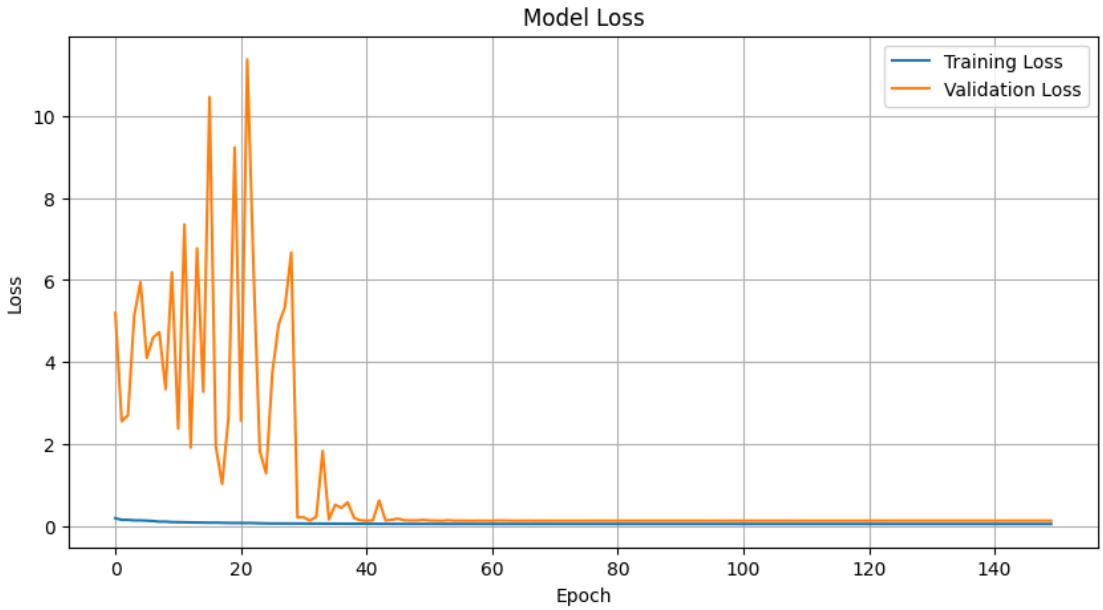

Fig. 30 Curva de perdida durante el entrenamiento con Efincientnetv2-B1

La tabla 26 expone los resultados obtenidos con los diferentes conjuntos de entrenamiento, validación y pruebas. La tabla 27 contiene los resultados por clases con el conjunto de pruebas.

Tabla 26 Resultados obtenidos deeplavb3+ con backbone eficientnetV2-B1

Conjunto	Mean IoU	Dice Coeff.	Dice Loss	Accuracy	Precision	Recall	F1 Score
Entrenamiento	0.9025	0.9479	0.0521	0.9469	0.9490	0.9469	0.9479
Validación	0.8956	0.9438	0.0562	0.9418	0.9461	0.9418	0.9438
Pruebas	0.8991	0.9458	0.0542	0.9446	0.9472	0.9446	0.9458

Tabla 27 Resultados del conjunto de pruebas del modelo Deeplabv con Backcone Efincientnetv2-B1

Clase	Mean IOU	Mean Dice Coefficient	Mean Dice Loss	Mean Accuracy	Mean Precision	Mean Recall	Mean F1 Score
Background	0.9840	0.9919	0.0081	0.9918	0.9921	0.9918	0.9919
Dandelion	0.8867	0.9400	0.0600	0.9450	0.9350	0.9450	0.9400
Potato	0.9581	0.9786	0.0214	0.9805	0.9768	0.9805	0.9786
Kikuyo	0.8350	0.9101	0.0899	0.9073	0.9128	0.9073	0.9101
Other	0.8210	0.9017	0.0983	0.8882	0.9156	0.8882	0.9017
Cow-tongue	0.9098	0.9528	0.0472	0.9546	0.9510	0.9546	0.9528

En la figura 31 se visualiza el resultado de la predicción y la máscara real de una imagen de 128 x 128 pixeles.

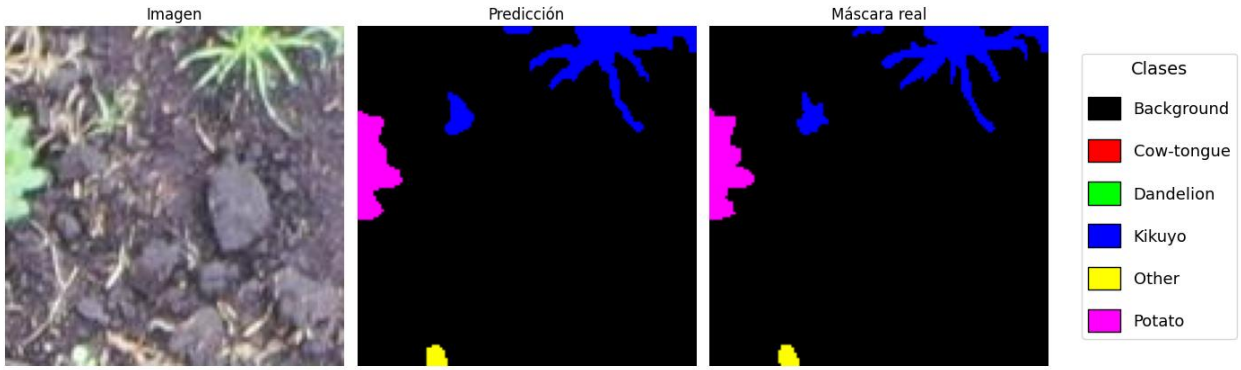

Fig. 31 Resultado de predicción con Backbone EficientnetV2-B1

EficientnetV2-B2

Las figuras 32 y 33 muestran las curvas de precisión y de pérdida obtenidas durante el proceso de entrenamiento con este backbone. En la primera gráfica, se puede ver que tanto la precisión de entrenamiento como la de validación aumentan rápidamente en las primeras épocas, alcanzando valores cercanos al 100%. Esto sugiere que el modelo tiene una buena capacidad de aprendizaje. En la segunda gráfica, la pérdida disminuye notablemente a medida que avanzan las épocas y luego se estabiliza, lo que indica que el modelo logró una buena convergencia y redujo los errores de predicción durante el proceso de entrenamiento.

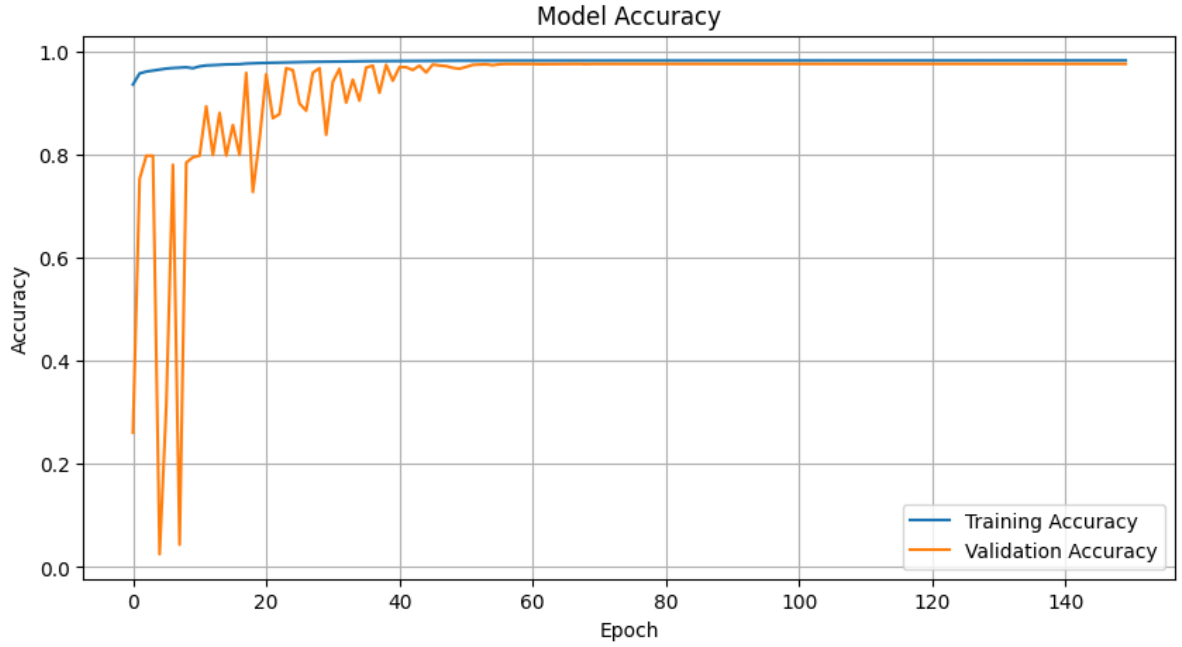

Fig. 32 Curva de accuracy durante el entrenamiento con Efincientnetv2-B2

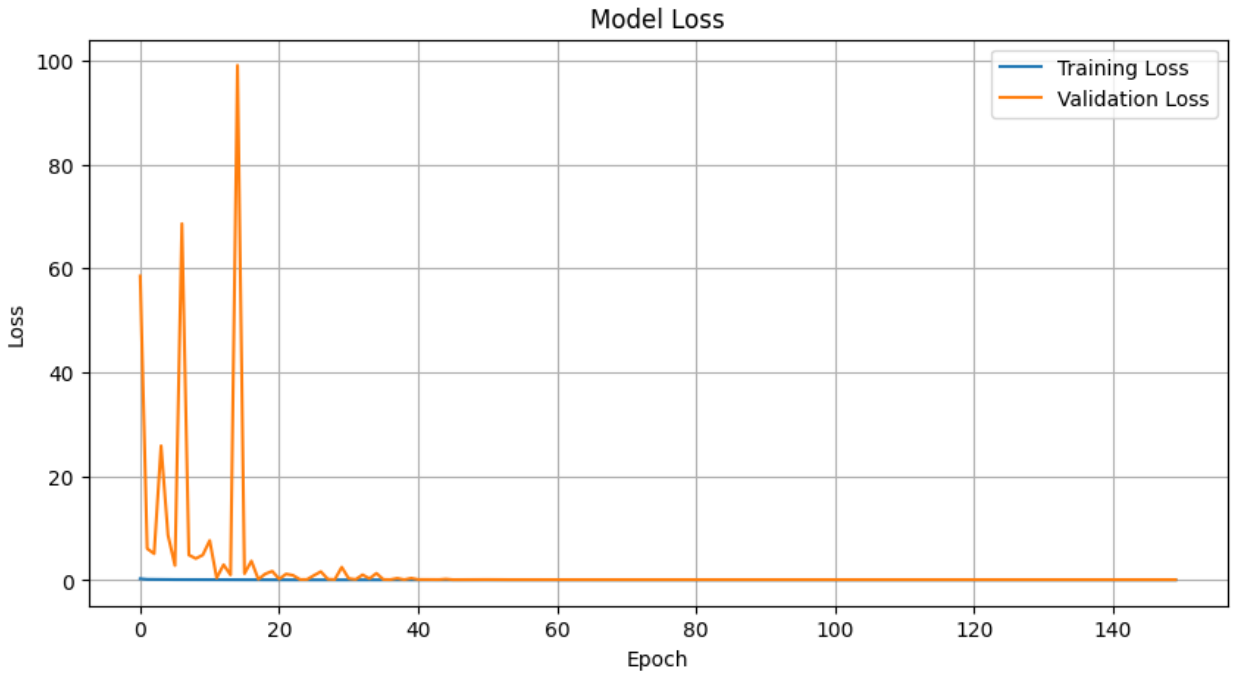

Fig. 33 Curva de perdida durante el entrenamiento con Efincientnetv2-B2

La tabla 28 presenta los resultados obtenidos a partir de los distintos conjuntos de entrenamiento, validación y prueba, mientras que la tabla 29 muestra los resultados por clase correspondientes al conjunto de pruebas.

Tabla 28 Resultados obtenidos deeplavb3+ con backbone eficientnetV2-B2

Conjunto	**Mean IoU**	**Dice Coeff.**	**Dice Loss**	**Accuracy**	**Precision**	**Recall**	**F1 Score**
Entrenamiento	0.8973	0.9449	0.0551	0.9425	0.9475	0.9425	0.9449
Validación	0.8874	0.9389	0.0611	0.9425	0.9475	0.9425	0.9449
Pruebas	0.8935	0.9426	0.0574	0.9394	0.9460	0.9394	0.9426

Tabla 29 Resultados del conjunto de pruebas del modelo Deeplabv con Backcone Efincientnetv2-B2

Clase	**Mean IOU**	**Mean Dice Coefficient**	**Mean Dice Loss**	**Mean Accuracy**	**Mean Precision**	**Mean Recall**	**Mean F1 Score**
Background	0.9833	0.9916	0.0084	0.9916	0.9915	0.9916	0.9916
Dandelion	0.8839	0.9384	0.0616	0.9407	0.9361	0.9407	0.9384

Potato	0.9579	0.9785	0.0215	0.9803	0.9766	0.9803	0.9785
Kikuyo	0.8310	0.9077	0.0923	0.9043	0.9111	0.9043	0.9077
Other	0.8075	0.8935	0.1065	0.8742	0.9137	0.8742	0.8935
Cow-tongue	0.8974	0.9459	0.0541	0.9452	0.9467	0.9452	0.9459

En la figura 34 se visualiza el resultado de la predicción y la máscara real de una imagen de 128 x 128 pixeles.

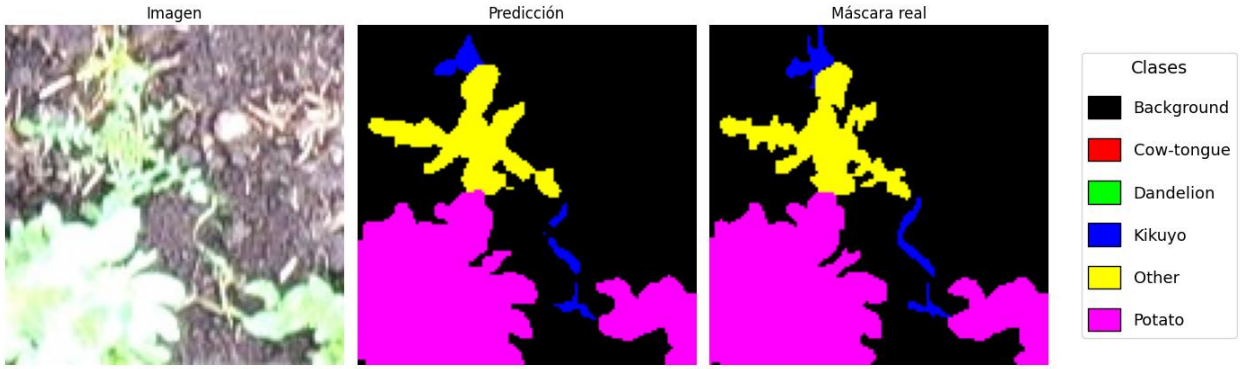

Fig. 34 Resultado de predicción con Backbone EficientnetV2-B2

3.2.Comparación de los resultados de los Modelos Obtenidos

La primera validación se enfoca en comparar los resultados obtenidos durante los diferentes entrenamientos de los modelos de segmentación semántica utilizados en este trabajo, específicamente Mask R-CNN y DeepLabV3+, cada uno con backbones distintos como ResNet-50 y EfficientNetV2. Para esta evaluación se aplican métricas ampliamente utilizadas en el ámbito de la inteligencia artificial, tales como la pérdida Dice (Dice Loss), el coeficiente medio de Dice (Mean Dice Coefficient) y el índice de intersección sobre unión (Intersection over Union, IoU).

El propósito de esta validación es analizar y contrastar el rendimiento de cada modelo entrenado bajo diferentes configuraciones, con el fin de determinar cuál de ellos presenta una mayor capacidad para segmentar con precisión las malezas en las imágenes capturadas. Esto permitirá seleccionar el modelo más adecuado para el despliegue final del sistema. A continuación, se detallan los resultados obtenidos de cada modelo.

Comparación con conjunto de entrenamiento.

Tabla 30 Comparación General de Modelos con conjunto de entrenamiento

Modelo	Backbone	Mean IoU	Dice Coeff.	Dice Loss	Accuracy	Precision	Recall	F1 Score
Mask R-CNN - Fase 1	ResNet-50	0.6900	0.7870	0.2130	0.9511	0.8636	0.7546	0.7870
Mask R-CNN - Fase 2	ResNet-50	0.7397	0.8304	0.1696	0.9648	0.8919	0.7953	0.8304
Mask R-CNN - Fase 3	ResNet-50	0.7659	0.8508	0.1492	0.9699	0.9089	0.8137	0.8508
DeepLabV3+	ResNet-50	0.8653	0.9259	0.0740	0.9821	0.9294	0.9896	0.9259
DeepLabV3+	EfficientNetV2-B0	0.9201	0.9578	0.0422	0.9572	0.9584	0.9572	0.9578
DeepLabV3+	EfficientNetV2-B1	0.9025	0.9479	0.0521	0.9469	0.9490	0.9469	0.9479
DeepLabV3+	EfficientNetV2-B2	0.8973	0.9449	0.0551	0.9425	0.9475	0.9425	0.9449

Como se muestra en la Tabla 30, durante el entrenamiento de los modelos se evidenció un comportamiento progresivo y consistente en la arquitectura Mask R-CNN con backbone ResNet-50. En la primera fase, el modelo alcanzó un IoU de 0.6900 y un Dice Coefficient de 0.7870, con una pérdida (Dice Loss) de 0.2130, lo que demuestra que logró aprender patrones relevantes, aunque con un margen de error considerable. En la segunda fase, al liberar más capas del backbone, el rendimiento mejoró alcanzando un IoU de 0.7397 y un Dice Coefficient de 0.8304, con incrementos en precisión y F1 Score. Finalmente, en la tercera fase, se optimizó aún más el desempeño, con un IoU de 0.7659 y un Dice Coefficient de 0.8508, reduciendo la pérdida a 0.1492. Esta evolución progresiva confirma que la estrategia de entrenamiento escalonado permitió refinar los pesos del modelo y mejorar su capacidad de segmentación en el conjunto de entrenamiento.

En contraste, la arquitectura DeepLabV3+ mostró un rendimiento marcadamente superior desde su primera implementación. Con ResNet-50 como backbone, alcanzó un IoU de 0.8653 y un Dice Coefficient de 0.9259, con una pérdida de apenas 0.0740. Estos resultados reflejan una mayor fidelidad entre las segmentaciones predichas y las máscaras reales,

confirmando la eficiencia de DeepLabV3+ para aprender de los datos incluso con un backbone menos complejo.

Los mejores resultados del conjunto de entrenamiento se obtuvieron con DeepLabV3+ utilizando EfficientNetV2-B0 como backbone, alcanzando un IoU de 0.9201 y un Dice Coefficient de 0.9578. Este modelo registró la menor pérdida de todos los experimentos (0.0422) y mantuvo un equilibrio sobresaliente entre precisión (0.9584), recall (0.9572) y F1 Score (0.9578). Las configuraciones con EfficientNetV2-B1 (IoU = 0.9025, Dice = 0.9479) y EfficientNetV2-B2 (IoU = 0.8973, Dice = 0.9449) también superaron al modelo con ResNet-50, aunque sin alcanzar los valores de B0.

Comparación con conjunto de validación

Tabla 31 Comparación General de Modelos con conjunto de Validación

Modelo	Backbone	Mean IoU	Dice Coeff.	Dice Loss	Accuracy	Precision	Recall	F1 Score
Mask R-CNN - Fase 1	ResNet-50	0.6572	0.7620	0.2380	0.9459	0.8544	0.7259	0.7620
Mask R-CNN - Fase 2	ResNet-50	0.6887	0.7896	0.2104	0.9543	0.8715	0.7513	0.7896
Mask R-CNN - Fase 3	ResNet-50	0.7067	0.8050	0.1950	0.9591	0.8812	0.7657	0.8050
DeepLabV3+	ResNet-50	0.8501	0.9160	0.0840	0.9092	0.9239	0.9092	0.9160
DeepLabV3+	EfficientNetV2-B0	0.9059	0.9494	0.0506	0.9452	0.9545	0.9452	0.9494
DeepLabV3+	EfficientNetV2-B1	0.8956	0.9438	0.0562	0.9418	0.9461	0.9418	0.9438
DeepLabV3+	EfficientNetV2-B2	0.8874	0.9389	0.0611	0.9425	0.9475	0.9425	0.9449

El análisis comparativo de los modelos de segmentación implementados en este trabajo evidencia una evolución progresiva en el rendimiento del Mask R-CNN con backbone ResNet-50 a lo largo de sus tres fases de entrenamiento. En la primera fase, el modelo alcanzó un IoU de 0.6572, un Dice Coefficient de 0.7620 y un F1 Score de 0.7620, lo que refleja un desempeño inicial limitado. Conforme se incorporaron más capas del backbone en las fases siguientes, se

observaron mejoras constantes, alcanzando en la tercera fase un IoU de 0.7067 y un Dice Coefficient de 0.8050. Estos resultados confirman que la estrategia de entrenamiento progresivo permitió una mayor adaptación del modelo a las características del conjunto de validación, aunque sin superar las limitaciones propias de la arquitectura.

En comparación, la arquitectura DeepLabV3+ con backbone ResNet-50 mostró un rendimiento superior, alcanzando un IoU de 0.8501 y un Dice Coefficient de 0.9160, con una reducción significativa del Dice Loss a 0.0840. Este resultado confirma que DeepLabV3+ ofrece una mejor capacidad de segmentación multiclase que Mask R-CNN, con un balance favorable entre precisión (0.9239) y recall (0.9092).

La mejor configuración se obtuvo con DeepLabV3+ utilizando EfficientNetV2-B0 como backbone, que alcanzó un IoU promedio de 0.9059 y un Dice Coefficient de 0.9494 en el conjunto de validación, consolidándose como la opción más precisa. Estos valores superan a los alcanzados con EfficientNetV2-B1 (IoU = 0.8956, Dice = 0.9438) y EfficientNetV2-B2 (IoU = 0.8874, Dice = 0.9389). El bajo valor de Dice Loss (0.0506) y el equilibrio entre precisión (0.9545), recall (0.9452) y F1 Score (0.9494) confirman la robustez de esta configuración.

En síntesis, la experimentación demuestra que DeepLabV3+ con EfficientNetV2-B0 es la arquitectura más adecuada en términos de exactitud y generalización para el conjunto de validación, superando tanto al modelo Mask R-CNN como a las variantes con ResNet-50, EfficientNetV2-B1 y EfficientNetV2-B2.

Validación comparativa por clase de los modelos de segmentación

Tabla 32 Comparación por clases de los modelos

Clase	Mask R-CNN (Fase 3) IoU	DeepLabV3+ ResNet-50 IoU	DeepLabV3+ B0 IoU	DeepLabV3+ B1 IoU	DeepLabV3+ B2 IoU
Background	0.94	0.97	0.99	0.98	0.98
Dandelion	0.69	0.83	0.90	0.88	0.88
Potato	0.75	0.80	0.96	0.95	0.95

Kikuyo	0.57	0.79	0.87	0.85	0.84
Other	0.58	0.68	0.79	0.79	0.76
Cow-tongue	0.66	0.85	0.92	0.90	0.89

El análisis comparativo por clase evidencia que el desempeño de los modelos varía significativamente en función de la especie segmentada. En la clase Background, todos los modelos alcanzaron métricas elevadas, con valores de IoU superiores a 0.94. El mejor resultado correspondió a DeepLabV3+ con EfficientNetV2-B0, con un IoU de 0.99, lo que confirma la facilidad del modelo para identificar las áreas de fondo debido a su alta homogeneidad y representación en el dataset.

En la clase Dandelion, se observa una mejora considerable al pasar de Mask R-CNN (IoU = 0.69) a DeepLabV3+ con ResNet-50 (IoU = 0.83), alcanzando un máximo de 0.90 con EfficientNetV2-B0. Esto refleja que DeepLabV3+ ofrece una mayor sensibilidad para detectar estructuras de malezas con formas más definidas y patrones recurrentes.

Para la clase Potato, los resultados fueron sobresalientes con los backbones de EfficientNet, alcanzando un IoU de 0.96 en B0 y 0.95 en B1 y B2, frente al 0.75 de Mask R-CNN y 0.80 de ResNet-50. Este comportamiento indica que las características morfológicas de la papa fueron mejor capturadas por los modelos más avanzados, lo cual es de gran relevancia en aplicaciones agrícolas donde este cultivo es prioritario.

La clase Kikuyo se mantuvo como uno de los mayores desafíos. El IoU pasó de 0.57 en Mask R-CNN a 0.79 en ResNet-50, alcanzando un máximo de 0.87 en EfficientNetV2-B0. Aunque la mejora es significativa, los valores se mantienen por debajo de los obtenidos en Potato o Background, lo que se explica por la alta variabilidad morfológica de esta especie y su similitud visual con otras malezas.

En la clase Other, que agrupa especies diversas, los resultados muestran las mayores dificultades, con IoU de apenas 0.58 en Mask R-CNN y un máximo de 0.79 en DeepLabV3+ B0 y B1. La heterogeneidad de este grupo dificulta la correcta segmentación, lo que se traduce en un desempeño inferior respecto a las demás clases.

Finalmente, en la clase Cow-tongue, se observa un avance sustancial de 0.66 en Mask R-CNN a 0.85 en DeepLabV3+ con ResNet-50, alcanzando un máximo de 0.92 en

EfficientNetV2-B0. Esto demuestra que la combinación de DeepLabV3+ con backbones modernos permite identificar con mayor precisión especies con hojas más largas y características menos homogéneas.

En síntesis, los resultados por clase confirman que DeepLabV3+ con EfficientNetV2-B0 es el modelo más robusto y preciso, logrando superar de manera consistente tanto a Mask R-CNN como a DeepLabV3+ con ResNet-50 y otras variantes de EfficientNet. Este comportamiento valida la elección de B0 como backbone final, al ofrecer un equilibrio superior entre exactitud global y rendimiento específico en cada clase de maleza.

Complejidad computacional de Modelos.

La Tabla 33 presenta la comparación global de complejidad y eficiencia computacional entre los modelos implementados. Se incluyen el número de parámetros, tamaño del modelo, tiempo de entrenamiento, tiempo de inferencia por imagen y la plataforma de entrenamiento utilizada.

Tabla 33 Complejidad computacional de modelos

Modelo / Backbone	**# Parámetros (M)**	**Tamaño (MB)**	**Tiempo de entrenamiento (h)**	**Tiempo de inferencia (s/img)**	**Plataforma de entrenamiento**
Mask R-CNN ResNet-50	44,684,470	175.244 KB	12h 20min	303.38 ms	GPU RTX 4050, 16GB RAM, TensorFlow 2.1
DeepLabV3+ ResNet-50	11,847,094	139.337 KB	6h 32min	67.26 ms	GPU RTX 4050, 16GB RAM, TensorFlow 2.1
DeepLabV3+ EfficientNet-B0	7,557,565	89.137 KB	4h 21min	33.61 ms	GPU RTX 4050, 16GB RAM, TensorFlow 2.1
DeepLabV3+ EfficientNet-B1	7,918,369	93.573 KB	4h 50min	45.22 ms	GPU RTX 4050, 16GB RAM, TensorFlow 2.1

DeepLabV3+ EfficientNet-B2	8,500,117	100.374 KB	5h 15min	53.15 ms	GPU RTX 4050, 16GB RAM, TensorFlow 2.1

El modelo Mask R-CNN con backbone ResNet-50 resultó ser el más pesado, con 44,6 millones de parámetros y un tamaño aproximado de 175 MB, requiriendo 12 horas 20 minutos de entrenamiento y alcanzando una latencia promedio de 303,38 ms por imagen. Si bien esta arquitectura permitió obtener predicciones correctas, su elevado costo computacional limita su aplicabilidad en entornos donde se requiere velocidad y eficiencia.

La versión DeepLabV3+ con ResNet-50 redujo significativamente la complejidad, con 11,8 millones de parámetros y un tamaño de 139 MB, logrando entrenarse en 6 horas 32 minutos y alcanzando un tiempo de inferencia de 67,26 ms por imagen. Este resultado refleja un avance en eficiencia respecto a Mask R-CNN, aunque aún conserva una carga computacional considerable.

Las variantes de DeepLabV3+ con EfficientNet demostraron un mejor balance entre precisión y eficiencia. El modelo con EfficientNet-B0 destacó como la opción más ligera, con 7,5 millones de parámetros, un tamaño de 89 MB, un tiempo de entrenamiento de 4 horas 21 minutos y la menor latencia de 33,61 ms por imagen. Estos valores confirman que EfficientNet-B0 ofrece un rendimiento competitivo con el menor costo computacional.

Por su parte, EfficientNet-B1 y B2 incrementaron levemente el número de parámetros y el tamaño del modelo (hasta 100 MB en B2), lo que se tradujo en mayores tiempos de entrenamiento (4h 50min y 5h 15min) y tiempos de inferencia más altos (45,22 ms y 53,15 ms, respectivamente), sin aportar mejoras significativas frente a EfficientNet-B0.

La comparación evidencia que DeepLabV3+ con EfficientNet-B0 constituye la mejor alternativa para la segmentación semántica multiclase, al combinar un excelente desempeño con un bajo costo computacional, lo que lo convierte en el modelo más adecuado para su implementación en entornos reales.

Resultados del modelo elegido usando en conjunto de pruebas

Tabla 34 Resultados del modelo elegido usando el dataset de pruebas

Modelo	Backbone	Mean IoU	Dice Coeff.	Dice Loss	Accuracy	Precision	Recall	F1 Score
DeepLabV3+	EfficientNetV2-B0	0.9169	0.9559	0.0441	0.9546	0.9573	0.9546	0.9559

El modelo DeepLabV3+ con EfficientNetV2-B0 alcanzó un Mean IoU de 0.9169 y un Dice Coefficient de 0.9559 en el conjunto de test, lo que confirma su elevada capacidad de segmentación y su excelente generalización más allá de los datos de entrenamiento y validación. La pérdida de Dice (0.0441) se mantuvo en valores bajos, indicando una mínima discrepancia entre las predicciones generadas por el modelo y las máscaras reales.

La exactitud global (Accuracy = 0.9546) evidencia que el modelo clasifica correctamente la gran mayoría de los píxeles, mientras que los valores de precisión (0.9573) y recall (0.9546) muestran un equilibrio adecuado entre la detección de verdaderos positivos y la reducción de falsos positivos. En conjunto, el F1 Score de 0.9559 refleja que el modelo logra un rendimiento armónico y consistente en todas las clases.

Los resultados obtenidos en el conjunto de test confirman la solidez del modelo final DeepLabV3+ con EfficientNetV2-B0 frente a datos no vistos durante el entrenamiento. Al contrastarlos con los valores obtenidos en el entrenamiento (Mean IoU = 0.9201, Dice = 0.9578) y en la validación (Mean IoU = 0.9059, Dice = 0.9494), se observa una consistencia notable entre los tres conjuntos, sin indicios de sobreajuste severo.

La ligera disminución de rendimiento en validación respecto al entrenamiento es un comportamiento esperado y evidencia que el modelo no memoriza los datos, sino que logra generalizar de manera efectiva. Asimismo, el desempeño en test incluso supera marginalmente a la validación, lo que refuerza la confianza en que el modelo es capaz de mantener un equilibrio adecuado entre precisión y recall en condiciones reales.

Estos resultados confirman que la arquitectura seleccionada no solo superó a las alternativas evaluadas en fases anteriores, sino que también se consolida como una solución robusta y confiable para la segmentación semántica multiclase en entornos agrícolas, garantizando un equilibrio óptimo entre precisión, sensibilidad y eficiencia computacional.

Tabla 35 Resultados del modelo elegido por clase

Clase	Mean IOU	Mean Dice Coefficient	Mean Dice Loss	Mean Accuracy	Mean Precision	Mean Recall	Mean F1 Score
Background	0.9874	0.9936	0.0064	0.9934	0.9939	0.9934	0.9936
Dandelion	0.9077	0.9516	0.0484	0.9549	0.9484	0.9549	0.9516
Potato	0.9674	0.9834	0.0166	0.9855	0.9814	0.9855	0.9834
Kikuyo	0.8581	0.9236	0.0764	0.9249	0.9224	0.9249	0.9236
Other	0.8505	0.9192	0.0808	0.9036	0.9353	0.9036	0.9192
Cow-tongue	0.9303	0.9639	0.0361	0.9653	0.9625	0.9653	0.9639

Los resultados por clase obtenidos en el conjunto de prueba evidencian un desempeño sobresaliente y balanceado del modelo DeepLabV3+ con EfficientNetV2-B0, aunque con diferencias según la complejidad de cada especie.

Background alcanzó el mejor rendimiento global, con un IoU de 0.9874 y un Dice Coefficient de 0.9936, lo que indica que el modelo identifica de manera prácticamente perfecta las regiones de fondo, sin introducir errores significativos.

En la clase Potato, también se observó un desempeño sobresaliente (IoU = 0.9674, Dice = 0.9834), confirmando que el modelo reconoce con gran precisión los cultivos de papa, que presentan características visuales consistentes en las imágenes.

La clase Cow-tongue obtuvo métricas elevadas (IoU = 0.9303, Dice = 0.9639), evidenciando una buena capacidad de detección pese a las similitudes morfológicas con otras especies de hojas anchas.

En Dandelion, los resultados fueron igualmente sólidos (IoU = 0.9077, Dice = 0.9516), mostrando que el modelo logra discriminar esta especie, aunque con una ligera tendencia a confundir algunos bordes de las plantas con el fondo.

Las clases Kikuyo (IoU = 0.8581, Dice = 0.9236) y Other (IoU = 0.8505, Dice = 0.9192) presentaron los valores más bajos en comparación con las demás especies. Este

comportamiento se explica por la alta variabilidad morfológica de estas categorías, ya que Kikuyo tiene un patrón de crecimiento más irregular y Other agrupa múltiples especies heterogéneas. Esto se traduce en una mayor dificultad para el modelo al momento de diferenciar límites precisos y evitar confusiones con otras clases.

En términos generales, todas las clases superaron un IoU del 0.85, lo que confirma que el modelo final mantiene un rendimiento robusto y confiable en la segmentación multiclase. No obstante, los resultados más bajos en Kikuyo y Other ponen de manifiesto la necesidad de aumentar la representación de estas clases en el dataset o considerar una redefinición de categorías para reducir la heterogeneidad y mejorar la precisión futura.

Ejemplos de mejores y peores predicciones del modelo final

La Fig. 35 presenta ejemplos de las mejores predicciones realizadas por el modelo DeepLabV3+ con EfficientNetV2-B0, en los que se alcanzó un desempeño sobresaliente con valores de IoU de 0.97. En estos casos, la segmentación obtenida es prácticamente idéntica a la verdad de terreno, evidenciando la alta precisión del modelo en clases específicas.

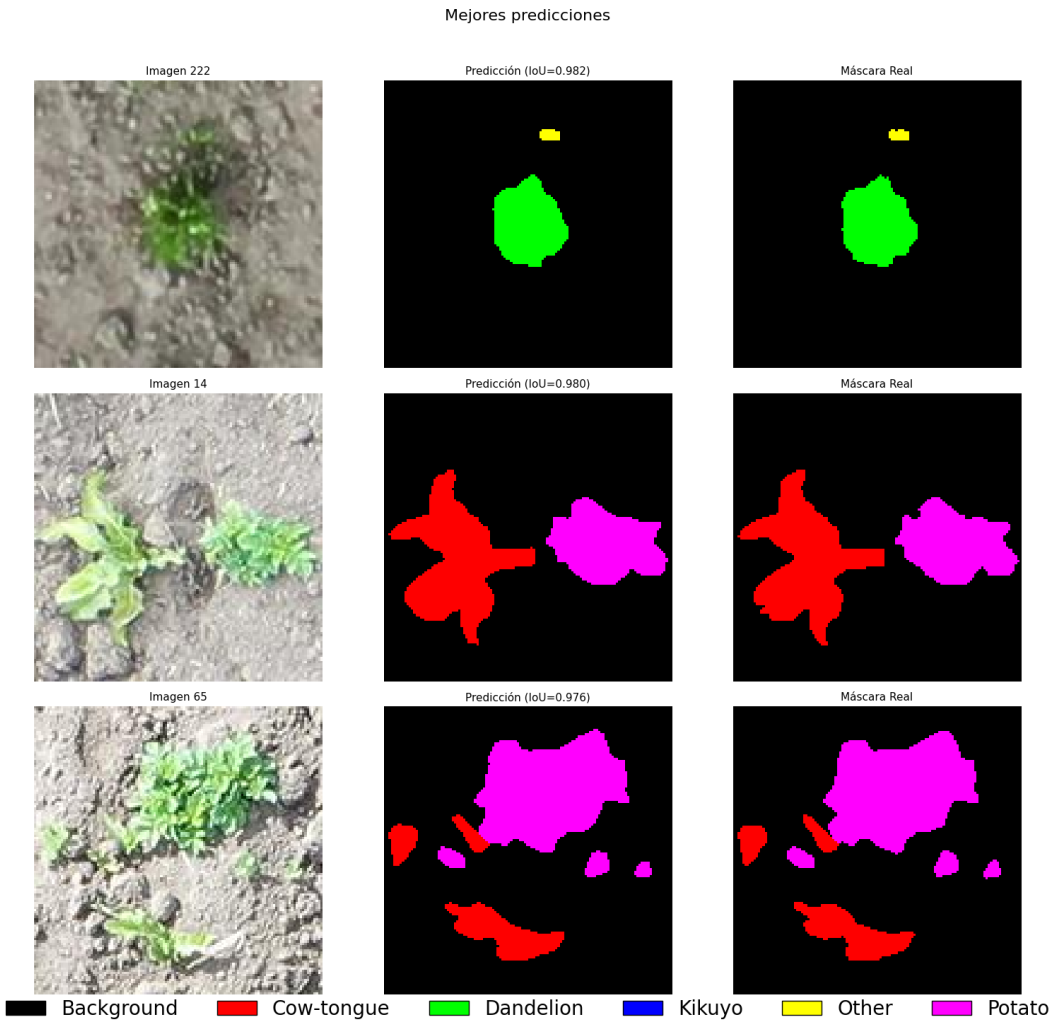

Fig. 35 Mejores predicciones del modelo elegido

Se observan las predicciones más acertadas del modelo DeepLabV3+ con backbone EfficientNetV2-B0, encontrando las clases Dandelion, Cow-tongue, Potato y Background. Estas clases son las que más aparecen en el conjunto de datos presentado y son lo suficientemente representativas en cuanto a sus patrones morfológicos, lo que favorece su segmentación por parte del modelo.

La clase Dandelion destaca por su compacta morfología y su color homogéneo, permitiendo una delimitación precisa de sus bordes. En el caso de Cow-tongue y Potato, ambas poseen hojas ampleas y contrastan respecto del suelo, lo que favorece su correcta identificación. El Background se mantiene como la clase dominante en todas las imágenes al representar las zonas de suelo desnudo y sin vegetación.

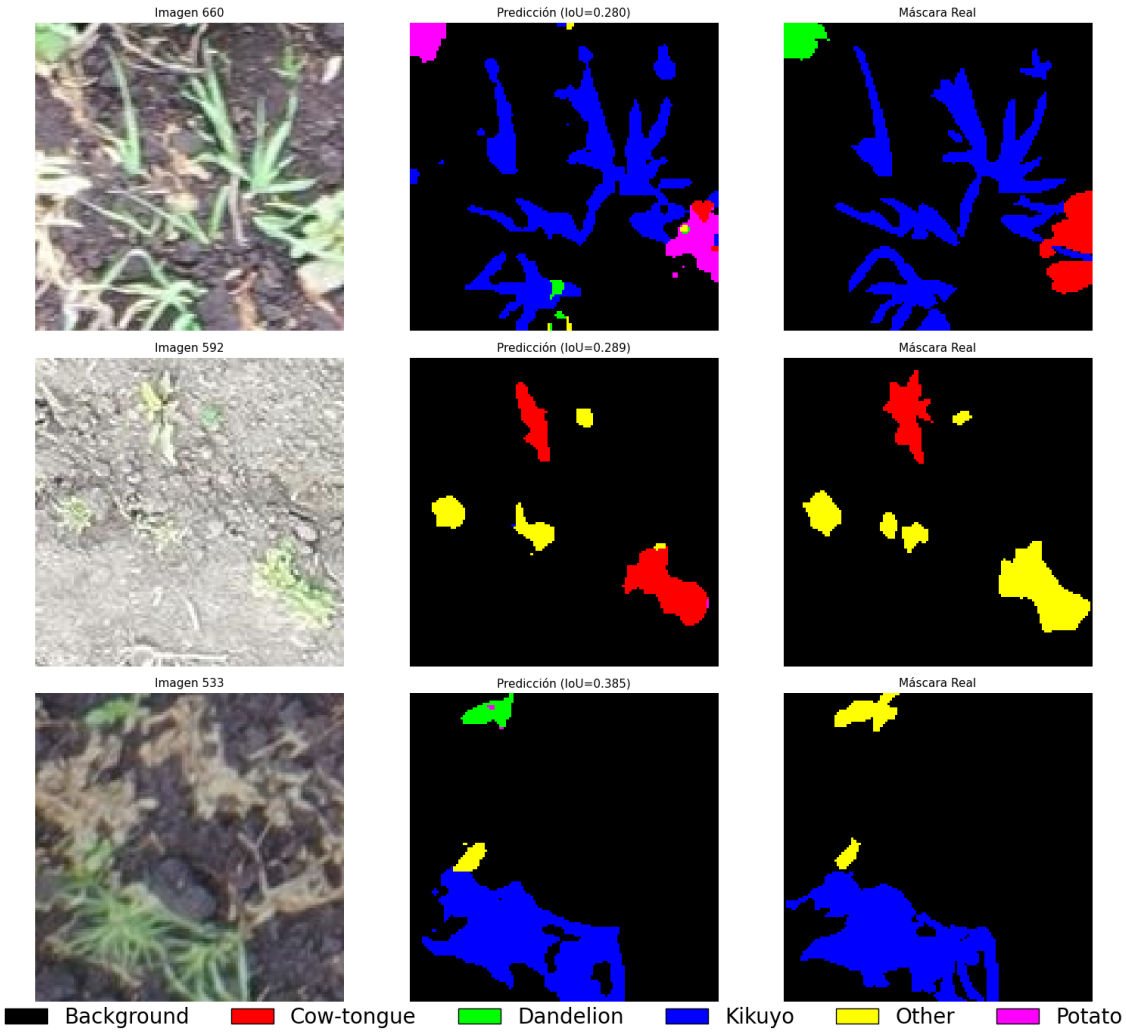

Fig. 36 Peores predicciones del modelo elegido

La Fig. 36 muestra ejemplos representativos de las peores predicciones realizadas por el modelo DeepLabV3+ con EfficientNetV2-B0, en los cuales el desempeño fue bajo (IoU entre 0.28 y 0.39). En estas imágenes se observa una marcada discrepancia entre las máscaras predichas y las anotaciones reales, con tendencia a la sobresegmentación y a la inclusión de clases falsas positivas.

En el primer ejemplo (Imagen 660), el modelo identificó simultáneamente hasta seis clases, incluyendo Potato y Other, que no estaban presentes en la verdad de terreno. Esto refleja una confusión del modelo en escenarios con múltiples estructuras vegetales superpuestas.

En la Imagen 592, el bajo contraste del suelo y la dispersión de pequeñas malezas ocasionaron que el modelo predijera Potato y Kikuyo, clases ausentes en la anotación real. En este caso, la red mostró alta sensibilidad, pero a costa de reducir la precisión.

En la Imagen 533, se evidenció nuevamente la confusión entre especies morfológicamente similares, al etiquetar regiones como Potato y Dandelion cuando en la verdad de terreno solo se encontraban Kikuyo y Other.

Estos resultados confirman que las clases Kikuyo y Other constituyen los mayores desafíos para la segmentación semántica en este proyecto, debido a su alta variabilidad y similitudes visuales con otras especies. En consecuencia, las fallas observadas en estas predicciones coinciden con los valores más bajos de IoU reportados en la evaluación cuantitativa, lo que pone en evidencia la necesidad de disponer de un mayor número de muestras o aplicar técnicas de balance de clases para mejorar la capacidad discriminativa del modelo.

Un aspecto relevante es que la clase Other aparece tanto en los ejemplos de mejores predicciones como en los de peores. Este comportamiento se debe a que dicha categoría agrupa varias especies con morfologías disímiles, lo que genera escenarios opuestos: en casos donde la especie dentro de Other presenta rasgos consistentes y diferenciables, el modelo alcanza IoU cercanos al 0.99. Sin embargo, en imágenes con individuos más heterogéneos o visualmente similares a otras clases (Potato, Kikuyo), el desempeño se degrada notablemente, con IoU inferiores a 0.40.

Esta dualidad refleja que la clase Other constituye un desafío particular para la segmentación semántica, ya que no representa una categoría homogénea, sino un conjunto de especies diversas.

DISCUSIÓN

En el presente trabajo de investigación se analizan arquitecturas de redes profundas para la tarea de la segmentación semántica de la maleza en cultivos de papa. A partir del modelo Mask R-CNN, se pasa a realizar una implementación del modelo DeepLabV3+ utilizando diferentes backbones, con la finalidad de determinar cuál de estas arquitecturas logra un mejor compromiso entre la precisión, el tiempo de entrenamiento y la capacidad de generalización de imágenes obtenidas mediante vehículos aéreos no tripulados (UAV).

A lo largo del experimento, para adecuar el modelo Mask R-CNN a las necesidades de detección y segmentación, se decidió entrenar el modelo en tres fases progresivas. Aun así, a pesar de que se obtuvieron progresos entre las diferentes fases, el rendimiento obtenido no es competitivo, obteniendo valores medios de Mean IoU de 0.7067 y Dice Coefficient de 0.8050 en la mejor fase. Esto hace evidente que, a pesar de que de la arquitectura es efectiva en la segmentación de instancias, eficiente en escenarios donde se prioriza la segmentación semántica.

Por el contrario, la arquitectura DeepLabV3+ tuvo un rendimiento notablemente mejor a partir de los primeros experimentos, alcanzando un Mean IoU de 0.8501 y un Dice Coefficient de 0.9160 con la utilización de un backbone ResNet-50, obteniendo ya una notable mejora en la calidad de segmentación; de hecho, al ir integrando backbones cada vez más eficientes, como EfficientNetV2-B0, el rendimiento siguió mejorando, de hecho, alcanzó un Mean IoU de 0.9201 en el entrenamiento, 0.9059 en la validación, y 0.9169 en las pruebas, lo cual muestra que el modelo era capaz de obtener un rendimiento sostenido independientemente del subconjunto de datos a evaluar.

A continuación, en la tabla 36 se muestra la comparación con resultados de trabajos realizados anteriormente con el mismo número de clases usando otras arquitecturas

Tabla 36 Comparación con trabajos anteriores conjunto de entrenamiento

Modelo Final	**Mean IoU**	**Dice Coeff.**	**Dice Loss**
U-Net Residual Balanceada	0.9096	0.9521	0.0478
ResNeXt50 modificada	0.735	0.797.	0.202
DeepLabV3+ EfficientNetV2-B0	**0.9201**	**0.9578**	**0.0422**

En el conjunto de entrenamiento, el modelo DeepLabV3+ con EfficientNetV2-B0 alcanzó los mejores resultados entre todas las arquitecturas comparadas, con un Mean IoU de 0.9201, un Dice Coefficient de 0.9578 y una Dice Loss de apenas 0.0422. Estos valores reflejan un ajuste preciso a los datos, con mínima discrepancia entre las predicciones generadas y las máscaras de referencia.

El modelo U-Net Residual Balanceada obtuvo métricas también destacables (IoU = 0.9096; Dice = 0.9521), pero ligeramente inferiores al desempeño de DeepLabV3+, lo que indica una capacidad de aprendizaje sólida, aunque con menor eficiencia en la representación de los patrones del dataset.

Por su parte, el Modelo Rexnet50, presentó valores considerablemente más bajos (IoU = 0.735; Dice = 0.797; Loss = 0.202), lo que sugiere que, incluso en entrenamiento, no logró alcanzar el mismo nivel de ajuste que los otros modelos, evidenciando limitaciones en su capacidad de aprendizaje.

Tabla 37 Resultados de los modelos con el conjunto de validación

Modelo Final	**Mean IoU**	**Dice Coeff.**	**Dice Loss**
U-Net Residual Balanceada	0.8021	0.8763	0.1236
ResNeXt50 modificada	0.626	0.679	0.320
DeepLabV3+ EfficientNetV2-B0	0. 9169	0.9494	0.0506

En el conjunto de validación, los resultados reafirman la superioridad del modelo DeepLabV3+ con EfficientNetV2-B0, que alcanzó un Mean IoU de 0. 9169, un Dice Coefficient de 0.9494 y una Dice Loss de 0.0506. Estos valores demuestran una excelente capacidad de generalización, con una mínima pérdida de precisión respecto a los resultados obtenidos en entrenamiento, lo que evidencia que el modelo no incurrió en sobreajuste significativo.

El modelo U-Net Residual Balanceada presentó un rendimiento aceptable (IoU = 0.8021; Dice = 0.8763; Loss = 0.1236), aunque sensiblemente menor al de DeepLabV3+. Esto indica que, si bien fue capaz de generalizar sobre el conjunto de validación, su desempeño se vio limitado en la segmentación de las clases con mayor variabilidad.

Por otro lado, el Modelo Rexnet50 mostró un rendimiento considerablemente más bajo en este conjunto (IoU = 0.626; Dice = 0.679; Loss = 0.320), lo que refleja dificultades para mantener un buen ajuste frente a datos no vistos durante el entrenamiento.

Tabla 38 Resultados de los modelos por clase

Clase	U-Net Residual Balanceada	ResNeXt50 modificada	DeepLabV3+ EfficientNetV2-B0
Background	0.951	0.956	0.987
Cow-tongue	0.776	0.826	0.930
Dandelion	0.817	0.583	0.907
Kikuyo	0.724	0.703	0.858
Other	0.665	0.434	0.850
Potato	0.866	0.908	0.967
Mean IoU	0.802	0.735	0.9059

El análisis por clase evidencia con mayor claridad las diferencias de desempeño entre los modelos evaluados. El modelo DeepLabV3+ con EfficientNetV2-B0 se posiciona consistentemente como la mejor alternativa, alcanzando los valores más altos de IoU en todas las clases. En Background y Potato logra métricas cercanas a la perfección (0.987 y 0.967 respectivamente), confirmando su capacidad para segmentar con alta precisión categorías mayoritarias y estructuralmente definidas. En contraste, aunque el Modelo RexNet50 presenta un rendimiento competitivo en la clase Papa (0.908) y aceptable en Background (0.956), su desempeño decae notablemente en categorías más complejas como Other (0.434) y Dandelion (0.583), evidenciando limitaciones para generalizar en clases con alta variabilidad.

Por su parte, la U-Net Residual Balanceada mantiene un rendimiento intermedio (Mean IoU = 0.802), destacando en Potato (0.866) y Background (0.951), aunque con menores resultados en Kikuyo (0.724) y Other (0.665). Finalmente, la comparación global muestra que DeepLabV3+ EfficientNetV2-B0 no solo supera a los modelos previos en promedio (Mean

IoU = 0.9059), sino que también ofrece un desempeño equilibrado en todas las clases, incluyendo aquellas consideradas más desafiantes, como Kikuyo y Other, donde logra un IoU de 0.858 y 0.850 respectivamente.

Comparación con trabajos relacionados.

En tabla 39 se presentan trabajos relacionados con la identificación de malezas utilizando segmentación semántica. Se recalca que la mayoría de los estudios se enfocan principalmente en dos clases, suelo, cultivos y malezas.

Tabla 39 Resultados de trabajos relacionados

Autores	Modelo	Numero de clases	Cultivos	Numero de malezas	Mean IOU
Cheong S, Lee S.	KDOSS-Net	3(Suelo, Cultivo, Maleza)	Arroz, Zanahoria, Remolacha azucarera	Sagittaria trifolia, Grass weeds, Dicot weeds, Malezas que no se especifica.	0.7524
Gomroki M, Benaragama D.	CWRepViT-Net	4(Suelo, Cultivo, Maleza dividido en 3 tipos)	Cultivo de soya	Canola voluntaria, Otras malezas de hoja ancha, Malezas gramíneas.	0.9735
Sivakumar S, Payyappilly A	Federated Ensemble DeepLabV3+ (con U-Net, SegNet y PSPNet)	3(Suelo, Cultivo, Maleza)	Maíz, Soya, Trigo, Arroz	(Malezas mixtas, incluye gramíneas y dicotiledóneas)	0.892

Los modelos comparados muestran un rango de desempeño con valores de Mean IoU entre 0.75 y 0.97, dependiendo de la arquitectura empleada, el tipo de cultivo y la diversidad de clases utilizadas.

El modelo propuesto en este trabajo, DeepLabV3+ con backbone EfficientNetV2-B0, alcanzó un Mean IoU de 0.92 en el conjunto de prueba, posicionándose como un resultado competitivo frente a estudios recientes. Si bien el modelo CWRepViT-Net obtuvo un valor superior de 0.97, este se entrenó con un conjunto de datos más homogéneo y con una menor variabilidad ambiental. En cambio, el presente estudio enfrentó el desafío de segmentar imágenes UAV con condiciones de iluminación, textura y color muy variables, propias de cultivos de papa del norte del Ecuador.

En comparación con KDOSS-Net, que alcanzó un Mean IoU de 0.75, y con el Federated Ensemble DeepLabV3+, que reportó 0.89, el modelo desarrollado en esta investigación demuestra una mejor segmentación. Esto confirma la eficacia de la combinación de DeepLabV3+ y EfficientNetV2-B0.

CONCLUSIONES

- La integración de las metodologías CRISP-DM y Scrum resultó ser una estrategia efectiva para el desarrollo del sistema de segmentación y cuantificación de malezas. CRISP-DM proporcionó una estructura analítica sólida para abordar las fases del ciclo de ciencia de datos desde la comprensión del problema agrícola hasta la evaluación de los modelos de segmentación, mientras que Scrum permitió gestionar el desarrollo de la aplicación web de manera iterativa e incremental, facilitando la planificación de tareas, seguimiento del avance y adaptación continua.
- Para realizar la evaluación de los modelos de segmentación semántica no solo es necesario las métricas comunes de Inteligencia artificial, también son necesarias métricas enfocadas a la segmentación semántica tales como Mean IOU, Mean Dice que permiten evaluar con mayor precisión la calidad de la segmentación semántica, al comparar píxel a píxel la superposición entre las máscaras predichas y las reales.
- A pesar de que Mask R-CNN ofrece resultados aceptables en la segmentación semántica de malezas, los experimentos realizados demostraron que DeepLabV3+ obtuvo un mejor desempeño global, reflejado en métricas más altas de IoU y DICE. Esta diferencia se atribuye a que DeepLabV3+ está específicamente optimizado para tareas de segmentación semántica pixel a pixel. En contraste, Mask R-CNN, al estar originalmente diseñado para segmentación instancia, puede presentar limitaciones al delimitar correctamente bordes precisos de objetos densamente agrupados como las malezas.

- El modelo presentó un desempeño significativamente menor al segmentar la clase "otros", lo cual se debe principalmente a la alta heterogeneidad y ambigüedad visual que caracteriza a esta categoría. A diferencia de las clases específicas de malezas, que presentan patrones morfológicos definidos, la clase "otros" incluye elementos diversos como suelo desnudo, sombras, residuos vegetales o bordes de cultivo, los cuales no comparten una textura ni forma coherente.

- El despliegue del modelo con mejor desempeño, seleccionado tras la evaluación comparativa entre Mask R-CNN y DeepLabV3+, permitió integrar de forma exitosa una solución de segmentación semántica dentro de la aplicación web desarrollada. Esta implementación demostró ser eficiente, al ofrecer predicciones precisas en tiempos razonables y con una interfaz accesible para el usuario final. Además, el uso de

contenedores Docker garantizó la portabilidad y reproducibilidad del sistema en distintos entornos, facilitando su mantenimiento y posible escalado.

RECOMENDACIONES

- Se recomienda priorizar el uso de DeepLabV3+ o arquitecturas similares en futuras aplicaciones orientadas exclusivamente a segmentación semántica, especialmente en escenarios agrícolas con objetos densos y solapados.
- Se recomienda redefinir y dividir la clase “otros” en subcategorías más coherentes (por ejemplo: suelo, sombra, restos vegetales), así como aumentar la cantidad y calidad de datos anotados para estas subclases, aplicando técnicas de aumento de datos específicas para mejorar la representación del modelo.
- Se recomienda trabajar en la integración del modelo con dispositivos edge o drones con capacidad de procesamiento embarcado para una segmentación en tiempo real durante vuelos.
- Aumentar la diversidad del dataset incluyendo imágenes de distintas fechas, tipos de cultivo, condiciones climáticas y estados de crecimiento, para lograr un modelo más robusto.
- Proponer líneas de investigación futuras, como la detección plagas, enfermedades o estrés hídrico mediante técnicas similares

BIBLIOGRAFÍA

[1] S. P. Siddique Ibrahim, U. Nithin, S. M. A. Kareem, and G. V. Kailash, "Weed Net: Deep Learning Informed Convolutional Neural Network Based Weed Detection in Soybean Crops," in *3rd IEEE International Conference on Mobile Networks and Wireless Communications, ICMNWC 2023*, Institute of Electrical and Electronics Engineers Inc., 2023. doi: 10.1109/ICMNWC60182.2023.10435726.

[2] S. N, M. Sundaram, R. Ranjan, and Abhishek, "Weedspedia: Deep Learning-Based Approach for Weed Detection using R-CNN, YoloV3 and Centernet," in *2023 International Conference on Quantum Technologies, Communications, Computing, Hardware and Embedded Systems Security (iQ-CCHESS)*, 2023, pp. 1–5. doi: 10.1109/iQ-CCHESS56596.2023.10391389.

[3] A. Giradkar, R. Adpawar, R. Agrawal, C. Dhule, and N. C. Morris, "Design of Autonomous Weed Elimination using Maching Learning Techniques," in *International Conference on Sustainable Computing and Smart Systems, ICSCSS 2023 - Proceedings*, Institute of Electrical and Electronics Engineers Inc., 2023, pp. 102–106. doi: 10.1109/ICSCSS57650.2023.10169371.

[4] P. M. P. Correia, J. Najafi, and M. Palmgren, "De novo domestication: what about the weeds?," 2024, *Elsevier Ltd*. doi: 10.1016/j.tplants.2024.03.001.

[5] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," *IEEE Trans Pattern Anal Mach Intell*, vol. 42, no. 2, pp. 386–397, 2020, doi: 10.1109/TPAMI.2018.2844175.

[6] P. Chapman *et al.*, "CRISP-DM 1.0 Step-by-step data mining guide," DaimlerChrysler, 2000.

[7] RoboFlow, "Roboflow: Computer vision tools for developers and enterprises," 2024. [Online]. Available: https://roboflow.com/

[8] "Scrum Guide | Scrum Guides." Accessed: Jan. 11, 2025. [Online]. Available: https://scrumguides.org/scrum-guide.html

[9] U. Nations, "Objetivos de Desarrollo Sostenible | Naciones Unidas," 2015, *United Nations*. [Online]. Available: https://www.un.org/es/impacto-acad%C3%A9mico/page/objetivos-de-desarrollo-sostenible

[10] "Plan de Creación de Oportunidades 2021-2025 – Secretaría Nacional de Planificación," 2021. [Online]. Available: https://www.planificacion.gob.ec/plan-de-creacion-de-oportunidades-2021-2025/

[11] K. Jabran, T. Ahmad, A. O. Siddiqui, İ. Üremiş, and M. N. Doğan, "Chapter 7 - Weed management in potato," in *Potato Production Worldwide*, M. E. Çalişkan, A. Bakhsh, and K. Jabran, Eds., Academic Press, 2023, pp. 121–131. doi: https://doi.org/10.1016/B978-0-12-822925-5.00013-X.

[12] R. Goyal, A. Nath, and U. Niranjan, "Weed detection using deep learning in complex and highly occluded potato field environment," *Crop Protection*, vol. 187, p. 106948, 2025, doi: https://doi.org/10.1016/j.cropro.2024.106948.

[13] K. M. B. Roncal Valderrama, "Efecto de tres tipos de coberturas para el control de malezas en Mangifera indica L. cv. Kent en Chimbote, Ancash," 2024.

[14] J. Aliloo, E. Abbasi, E. Karamidehkordi, E. Ghanbari Parmehr, and M. Canavari, “Dos and Don’ts of using drone technology in the crop fields,” *Technol Soc*, vol. 76, p. 102456, 2024, doi: https://doi.org/10.1016/j.techsoc.2024.102456.

[15] M. Canicattì and M. Vallone, “Drones in vegetable crops: A systematic literature review,” *Smart Agricultural Technology*, vol. 7, p. 100396, 2024, doi: https://doi.org/10.1016/j.atech.2024.100396.

[16] P. S. Minz and C. S. Saini, “RGB camera-based image technique for color measurement of flavored milk,” *Measurement: Food*, vol. 4, p. 100012, 2021, doi: https://doi.org/10.1016/j.meafoo.2021.100012.

[17] W. Liu *et al.*, “Identification of varieties of wheat seeds based on multispectral imaging combined with improved YOLOv5,” *Food Physics*, vol. 2, p. 100042, 2025, doi: https://doi.org/10.1016/j.foodp.2024.100042.

[18] B. Liu, S. Men, Q. Yu, D. Li, Z. Ding, and Z. Liu, “Internal scanning hyperspectral imaging system for deep sea target detection,” *Opt Lasers Eng*, vol. 185, p. 108722, 2025, doi: https://doi.org/10.1016/j.optlaseng.2024.108722.

[19] G. Batchuluun, S. H. Nam, and K. R. Park, “Deep learning-based plant classification and crop disease classification by thermal camera,” *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 10, Part B, pp. 10474–10486, 2022, doi: https://doi.org/10.1016/j.jksuci.2022.11.003.

[20] G. Rivera, R. Porras, R. Florencia, and J. P. Sánchez-Solís, “LiDAR applications in precision agriculture for cultivating crops: A review of recent advances,” *Comput Electron Agric*, vol. 207, p. 107737, 2023, doi: https://doi.org/10.1016/j.compag.2023.107737.

[21] Carrillo Velarde and Cristhyan Javier, “Segmentación semántica de imágenes para la obtención de rutas libres de obstáculos,” 2021, Accessed: Jan. 06, 2025. [Online]. Available: https://reunir.unir.net/handle/123456789/11278

[22] L. P. Altamirano, P. F. Guía Rodrigo Salas, and P. M. Co-Guía Daira Velandia Dra, “Deep Learning aplicado a la Segmentación Semántica de Imágenes aéreas,” 2021. Accessed: Jan. 06, 2025. [Online]. Available: http://repositoriobibliotecas.uv.cl/handle/uvscl/3958

[23] J. Cheng, C. Deng, Y. Su, Z. An, and Q. Wang, “Methods and datasets on semantic segmentation for Unmanned Aerial Vehicle remote sensing images: A review,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 211, pp. 1–34, 2024, doi: https://doi.org/10.1016/j.isprsjprs.2024.03.012.

[24] X. Yuan, J. Shi, and L. Gu, “A review of deep learning methods for semantic segmentation of remote sensing imagery,” *Expert Syst Appl*, vol. 169, p. 114417, 2021, doi: https://doi.org/10.1016/j.eswa.2020.114417.

[25] T. Antamis, A. Drosou, T. Vafeiadis, A. Nizamis, D. Ioannidis, and D. Tzovaras, “Interpretability of deep neural networks: A review of methods, classification and hardware,” *Neurocomputing*, vol. 601, p. 128204, 2024, doi: https://doi.org/10.1016/j.neucom.2024.128204.

[26] B. Neupane, T. Horanont, and J. Aryal, “Deep Learning-Based Semantic Segmentation of Urban Features in Satellite Images: A Review and Meta-Analysis,” *Remote Sens (Basel)*, vol. 13, no. 4, 2021, doi: 10.3390/rs13040808.

[27] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” *IEEE Trans Pattern Anal Mach Intell*, vol. 39, no. 6, pp. 1137–1149, 2017, doi: 10.1109/TPAMI.2016.2577031.

[28] A. M. Hafiz and G. M. Bhat, “A survey on instance segmentation: state of the art,” *Int J Multimed Inf Retr*, vol. 9, no. 3, pp. 171–189, 2020, doi: 10.1007/s13735-020-00195-x.

[29] R. Sapkota, D. Ahmed, and M. Karkee, “Comparing YOLOv8 and Mask R-CNN for instance segmentation in complex orchard environments,” *Artificial Intelligence in Agriculture*, vol. 13, pp. 84–99, 2024, doi: https://doi.org/10.1016/j.aiia.2024.07.001.

[30] Y. Zhao *et al.*, “YOMASK: An instance segmentation method for high-throughput phenotypic platform lettuce images,” *Comput Electron Agric*, vol. 230, p. 109868, 2025, doi: https://doi.org/10.1016/j.compag.2024.109868.

[31] A. Picon *et al.*, “Crop-conditional semantic segmentation for efficient agricultural disease assessment,” *Artificial Intelligence in Agriculture*, 2025, doi: https://doi.org/10.1016/j.aiia.2025.01.002.

[32] A. Yadav and E. Kumar, “Object Detection on Real-Time Video with FPN and Modified Mask RCNN Based on Inception-ResNetV2,” *Wirel Pers Commun*, vol. 138, no. 4, pp. 2065–2090, 2024, doi: 10.1007/s11277-024-11539-9.

[33] J. K. Appati and I. A. Yirenkyi, “A cascading approach using se-resnext, resnet and feature pyramid network for kidney tumor segmentation,” *Heliyon*, vol. 10, no. 19, p. e38612, 2024, doi: https://doi.org/10.1016/j.heliyon.2024.e38612.

[34] Y. Zhong, J. Wang, J. Peng, and L. Zhang, “Anchor box optimization for object detection,” in *Proceedings - 2020 IEEE Winter Conference on Applications of Computer Vision, WACV 2020*, Institute of Electrical and Electronics Engineers Inc., 2020, pp. 1275–1283. doi: 10.1109/WACV45572.2020.9093498.

[35] S. Wang, G. Sun, B. Zheng, and Y. Du, “A crop image segmentation and extraction algorithm based on mask RCNN,” *Entropy*, vol. 23, no. 9, 2021, doi: 10.3390/e23091160.

[36] B. Ahmed, T. A. Gulliver, and S. alZahir, “Image splicing detection using mask-RCNN,” *Signal Image Video Process*, vol. 14, no. 5, pp. 1035–1042, 2020, doi: 10.1007/s11760-020-01636-0.

[37] F. F. Liu and M. Fang, “Semantic Segmentation of Underwater Images Based on Improved Deeplab,” *J Mar Sci Eng*, vol. 8, no. 3, 2020, doi: 10.3390/jmse8030188.

[38] Q. Zhao, J. Cao, J. Ge, Q. Zhu, X. Chen, and W. Liu, “Multi-UNet: An effective Multi-U convolutional networks for semantic segmentation,” *Knowl Based Syst*, vol. 309, p. 112854, 2025, doi: https://doi.org/10.1016/j.knosys.2024.112854.

[39] S. Turk, O. Bingol, A. Coskuncay, and T. Aydin, “The impact of implementing backbone architectures on fracture segmentation in X-ray images,” *Engineering Science and Technology, an International Journal*, vol. 59, p. 101883, 2024, doi: https://doi.org/10.1016/j.jestch.2024.101883.

[40] O. Rainio, J. Teuho, and R. Klén, “Evaluation metrics and statistical tests for machine learning,” *Sci Rep*, vol. 14, no. 1, 2024, doi: 10.1038/s41598-024-56706-x.

[41] R. Zhao *et al.*, “Rethinking Dice Loss for Medical Image Segmentation,” in *2020 IEEE International Conference on Data Mining (ICDM)*, 2020, pp. 851–860. doi: 10.1109/ICDM50108.2020.00094.

[42] “Home | Scrum Guides.” Accessed: Jan. 11, 2025. [Online]. Available: https://scrumguides.org/

[43] A. Azevedo and M. F. Santos, “KDD, SEMMA and CRISP-DM: a parallel overview,” *IADS-DM*, 2008.

[44] J. S. Saltz, “CRISP-DM for Data Science: Strengths, Weaknesses and Potential Next Steps,” in *2021 IEEE International Conference on Big Data (Big Data)*, 2021, pp. 2337–2344. doi: 10.1109/BigData52589.2021.9671634.

[45] J. H. Moyano, K. M. Cenci, and J. R. Ardenghi, “Arquitectura Cliente-Servidor de Alto Rendimiento para servicio RTK,” in *XXVI Congreso Argentino de Ciencias de la Computación (CACIC)(Modalidad virtual, 5 al 9 de octubre de 2020)*, 2020.

[46] “What’s New In Python 3.13 — Python 3.13.1 documentation.” Accessed: Dec. 08, 2024. [Online]. Available: https://docs.python.org/3/whatsnew/3.13.html

[47] “Welcome to Flask — Flask Documentation (3.1.x).” Accessed: Dec. 08, 2024. [Online]. Available: https://flask.palletsprojects.com/en/stable/

[48] “TypeScript: Documentation - TypeScript for JavaScript Programmers.” Accessed: Dec. 08, 2024. [Online]. Available: https://www.typescriptlang.org/docs/handbook/typescript-in-5-minutes.html

[49] “What is Angular? • Angular.” Accessed: Dec. 08, 2024. [Online]. Available: https://angular.dev/overview

[50] Z. Guo, D. Cai, Z. Jin, T. Xu, and F. Yu, “Research on unmanned aerial vehicle (UAV) rice field weed sensing image segmentation method based on CNN-transformer,” *Comput Electron Agric*, vol. 229, p. 109719, 2025, doi: https://doi.org/10.1016/j.compag.2024.109719.

[51] H. Jiang, Q. Chen, R. Wang, J. Du, and T. Chen, “SWFormer: A scale-wise hybrid CNN-Transformer network for multi-classes weed segmentation,” *Journal of King Saud University - Computer and Information Sciences*, vol. 36, no. 7, p. 102144, 2024, doi: https://doi.org/10.1016/j.jksuci.2024.102144.

[52] S. P. Samuel, K. Malarvizhi, and S. Karthik, “Weed detection in agricultural fields via automatic graph cut segmentation with Mobile Net classification model,” *Signal Image Video Process*, vol. 18, no. 2, pp. 1549–1560, 2024, doi: 10.1007/s11760-023-02863-x.

[53] S. Thiagarajan, A. Vijayalakshmi, and G. H. Grace, “Weed detection in precision agriculture: leveraging encoder-decoder models for semantic segmentation,” *J Ambient Intell Humaniz Comput*, vol. 15, no. 9, pp. 3547–3561, Sep. 2024, doi: 10.1007/s12652-024-04832-9.

[54] B. Xu, J. Fan, J. Chao, N. Arsenijevic, R. Werle, and Z. Zhang, “Instance segmentation method for weed detection using UAV imagery in soybean fields,” *Comput Electron Agric*, vol. 211, p. 107994, 2023, doi: https://doi.org/10.1016/j.compag.2023.107994.

[55] S. D. Khan, S. Basalamah, and A. Lbath, “Weed–Crop Segmentation in Drone Images with a Novel Encoder–Decoder Framework Enhanced via Attention Modules,” *Remote Sens (Basel)*, vol. 15, no. 23, 2023, doi: 10.3390/rs15235615.

[56] M. Gomroki, D. Benaragama, C. J. Henry, N. Badreldin, and R. Gulden, “CWRepViT-Net: An encoder-decoder deep learning framework with RepViT blocks for crop weed semantic segmentation in soybean fields through their life journey,” *Smart Agricultural Technology*, vol. 12, p. 101472, 2025, doi: https://doi.org/10.1016/j.atech.2025.101472.

[57] S. H. Cheong, S. J. Lee, S. J. Im, J. Seo, and K. R. Park, “KDOSS-net: Knowledge distillation-based outpainting and semantic segmentation network for crop and weed images,” *Plant Phenomics*, vol. 7, no. 3, p. 100098, 2025, doi: https://doi.org/10.1016/j.plaphe.2025.100098.

[58] S. Sivakumar, A. J. Payyappilly, A. Rajan, R. Arumuga Arun, and R. S. Rampriya, “A computer vision-based crop and weed segmentation using federated ensemble learning in diverse multi-crop fields,” *Eng Appl Artif Intell*, vol. 159, p. 111773, 2025, doi: https://doi.org/10.1016/j.engappai.2025.111773.

ANEXOS

Anexo A: Repositorio del Backend Flask

GitHub: https://github.com/Oliverzam/Backend-tesis-Oliver-Zamora.git

Anexo B: Repositorio del Frontend Angular

GitHub: https://github.com/Oliverzam/Frontend-tesis-Oliver-Zamora.git

Anexo C: Repositorio del entrenamiento del modelo

GitHub: https://github.com/Oliverzam/Entrenamient-tesis-Oliver-Zamora.git