UNIVERSIDAD DE LA REPÚBLICA URUGUAY

FACULTAD DE INGENIERÍA

Sistema de generación de instancias y reducción de carga del etiquetado de imágenes de malezas para detección de objetos

Informe de Proyecto de Grado presentado por

Daniel Padrón Simón, Edison Steven Estramil Moreira,
Nicolás Alejandro Núñez Hosco

en cumplimiento parcial de los requerimientos para la graduación de la carrera de Ingeniería en Computación de Facultad de Ingeniería de la Universidad de la República

Supervisores

Gonzalo Tejera
Mercedes Marzoa

Montevideo, 23 de julio de 2025

Agradecimientos

Primero, un especial agradecimiento a Gonzalo Tejera y Mercedes Marzoa, tutores a cargo de esta tesis, por habernos guiado y apoyado durante el transcurso del proyecto, con el que cerramos esta etapa.

De igual modo, queremos agradecer a los familiares respectivos de cada integrante del grupo por habernos respaldado incondicionalmente a lo largo de este proyecto, y durante toda nuestra carrera.

Extendemos nuestro agradecimiento a todos los involucrados, tanto los autores de los antecedentes que motivaron este trabajo, así como a los equipos de desarrollo de las diferentes herramientas utilizadas. Además, un especial agradecimiento a Nicolás Blumetto por habernos ayudado en la integración con CVAT, y también a Juana Villabla y Agustina Azcoitia, ingenieras agrónomas, por habernos arrojado luz durante la evaluación de las imágenes generadas.

Un afectuoso agradecimiento a todos los docentes y funcionarios de la Facultad de Ingeniería de la UdelaR, por habernos acompañado durante el desarrollo de nuestras carreras.

Por último, pero no por eso menos importante, queremos agradecer al equipo completo de Winterpixel Games por haber creado y disponibilizado Rocket Bot Royale, un juego capaz de mantener nuestra cordura durante estos últimos 8 meses.

Resumen

Técnicas asociadas al área de visión artificial, como la detección de objetos, son ampliamente utilizadas en la búsqueda de la automatización en diversos dominios; en particular, en este trabajo se aplica en lo referido al control de malezas en cultivos.

Actualmente, los métodos de detección de objetos más populares, que hacen uso de redes neuronales, requieren de un gran número de imágenes con sus correspondientes etiquetas, conformadas por la clase y localización de cada objeto que se busca detectar. Con el objetivo de mitigar esta limitación, en este trabajo se relevan diversas técnicas orientadas a mejorar el desempeño de modelos de detección sin aumentar el esfuerzo humano, a costa de un mayor uso de poder computacional.

Posteriormente, se proponen, implementan y evalúan distintos flujos que a partir de un conjunto de imágenes y un número máximo de imágenes a etiquetar manualmente, devuelven un subconjunto completamente etiquetado, buscando que este sea el más significativo para un modelo de detección, junto con imágenes artificiales autoetiquetadas de los objetos a detectar. Esto permite su libre utilización en futuros entrenamientos de modelos de detección de objetos, sin depender de la arquitectura base utilizada.

En particular, los flujos son variaciones en el uso de los diferentes componentes implementados, como modelos difusos para la generación de imágenes, Plug and Play Active Learning (PPAL) para la selección de imágenes, así como YOLOv11m, Faster R-CNN y RetinaNet para la evaluación, todos ellos implementados en Python.

El primero de los componentes es una implementación propia de PPAL, el cual, al igual que el resto de las técnicas de aprendizaje activo, realiza la selección de imágenes a etiquetar de forma progresiva, a medida que el modelo es entrenado, con el fin de priorizar el etiquetado de aquellas imágenes que sean más significativas para el entrenamiento del modelo.

El segundo de dichos componentes consiste en el entrenamiento de modelos generativos que sean capaces de capturar los detalles de los objetos a detectar, a partir de Stable Diffusion v1.5, utilizando como método de ajuste Dreambooth, a fin de expandir el conjunto de datos de entrenamiento.

Por último, las imágenes generadas por el modelo generativo no serían aprovechables para el entrenamiento sin sus etiquetas correspondientes. Por esta razón, y con el fin de no requerir intervención humana para el etiquetado de

estas, se propone un último componente, el cual se encargará de etiquetar de forma autónoma las imágenes generadas, utilizando los canales de colores de las imágenes, además de otros métodos que los combinan con el uso de SAM (Segment Anything Model), o mediante modelos de detección en conjunto con filtros que utilizan las imágenes etiquetadas por el usuario.

Se evalúa el aumento de rendimiento provisto tanto por cada componente individualmente, como por los flujos que los combinan. Utilizando el conjunto de datos CottonWeedDet12, se compara el rendimiento de diferentes modelos entrenados, primero con las imágenes resultantes de haber aplicado las tres componentes, y luego sobre las imágenes resultantes de no haber aplicado ninguna de estas. Al calcular la mAP@50, el método propuesto obtiene, al utilizar un 5 % del conjunto de datos, una mejora de 0,119 en Faster R-CNN, 0,052 en YOLOv11m y 0,191 en RetinaNet con respecto a entrenar el detector sin utilizar las imágenes sintéticas, y mediante selección aleatoria. Al usar un 10 % se obtienen mejoras de 0,049 en Faster R-CNN, 0,058 en YOLOv11m y 0,087 en RetinaNet. Por último, al usar un 20 %, se obtienen mejoras de 0,053 en Faster R-CNN, 0,035 en YOLOv11m y 0,034 en RetinaNet. Estos resultados muestran que la metodología presenta una clara mejora en todos los detectores, especialmente en los casos en lo que el número de imágenes es particularmente bajo, lográndolo sin incrementar el esfuerzo humano requerido.

Palabras clave: Modelos generativos de imágenes, reducción de carga de etiquetado, aprendizaje automático, aprendizaje activo, detección de objetos.

Índice general

Capítulo 1

Introducción

Según la División de Producción y Protección Vegetal (NSP por sus siglas en inglés) de la Organización de las Naciones Unidas para la Agricultura (FAO por sus siglas en inglés), se definen las malezas como plantas no deseadas que se desarrollan fuera de lugar e interfieren con las actividades o el bienestar humano. El control de malezas, en consecuencia, es un área de gran interés en lo que respecta al desarrollo del sector agrícola, en especial si se tiene en cuenta que las malezas representan uno de los mayores riesgos en la reducción de la producción de cultivos, junto con los patógenos, hongos, bacterias y plagas (Cordeau, Triolet, Wayman, Steinberg, y Guillemin, 2016). A modo de ejemplo, se estima que los costos anuales provocados por malezas en la producción de grano en Australia ronda los 3.300 millones de dólares australianos (Bajwa, Walsh, y Chauhan, 2017). Por lo tanto, es habitual la utilización de herbicidas para el control de las mismas. Sin embargo, su uso prolongado y recurrente puede derivar en la aparición de especies de malezas resistentes a estos productos, además de generar impactos ambientales negativos, especialmente en lo referido a la contaminación de los recursos hídricos y al deterioro de la calidad de los suelos agrícolas. Este problema no escapa a nuestro país, en el informe CEUTA (2006) se expone el significativo aumento en el uso agrotóxicos en Uruguay, implicando un potencial riesgo en la calidad de los suelos y el agua en el territorio nacional.

A raíz de esto, en los últimos años se han desarrollado distintas soluciones robóticas enfocadas en el control de malezas, en particular, el grupo de Gestión de Redes e Inteligencia Artificial (MINA) de la Facultad de Ingeniería de la Universidad de la República, financiado por el Fondo de Promoción de Tecnología Agropecuaria (FPTA) del Instituto Nacional de Investigación Agropecuaria (INIA), se encuentra desarrollando un robot cuya función será localizar y eliminar una serie de malezas en determinados cultivos. El proyecto de grado aquí presentado tiene como objetivo el apoyo al desarrollo antes mencionado. Puntualmente, en lo que respecta al proceso de creación del conjunto de datos que será posteriormente utilizado para elaborar el detector de malezas del robot.

Los detectores se caracterizan por localizar varios objetos determinados den-

tro de imágenes, clasificando cada objeto detectado e indicando su ubicación dentro de la imagen. Actualmente, son utilizadas técnicas de aprendizaje automático profundo para la elaboración de los detectores que conforman el estado del arte, implicando la necesidad de disponer de un conjunto de datos de entrenamiento para poder entrenar dichos modelos de detección. Estos conjuntos de datos consisten en un considerable número de imágenes, donde cada una de ellas se encuentra etiquetada. En otras palabras, cada imagen tiene especificada la localización y el tipo de cada objeto que se busca aprender a detectar.

Cabe destacar que este proceso, requiere de un considerable trabajo manual, tanto la captura de las imágenes, como el proceso de etiquetado. Sin mencionar que en algunos dominios, como el de las malezas, se requiere de expertos para realizar dicho proceso, ya que no es trivial la localización de las malezas en los cultivos, ni mucho menos distinguir su especie. Esto implica que el proceso sea usualmente costoso, y naturalmente sea de interés buscar mecanismos para optimizarlo.

El presente proyecto tiene como principal objetivo la investigación y la experimentación de técnicas que permitan reducir dicho esfuerzo (medido en el número de imágenes a etiquetar), obteniendo modelos de detección de objetos con el mayor rendimiento posible. En este sentido, existen varios mecanismos que se adecuarían al anterior objetivo. Por ejemplo, métodos que realizan sugerencias de posibles localizaciones de objetos en cada imagen (Papadopoulos, Uijlings, Keller, y Ferrari, 2017), métodos que ordenan las imágenes según su relevancia para el modelo a entrenar (Brust, Käding, y Denzler, 2018), métodos que combinan imágenes no etiquetadas con imágenes etiquetadas (Zhou, Yu, Wang, Qian, y Li, 2021), métodos que generan nuevas imágenes (Y. Zhang, Song, y Li, 2021), entre otros.

Dado los más recientes avances en la generación de imágenes sintéticas, y los prometedores resultados presentados por las técnicas de aprendizaje activo, se acordó con los tutores del proyecto acotar el alcance del trabajo a los siguientes dos mecanismos:

1. Partiendo de un conjunto de imágenes no etiquetadas, seleccionar las imágenes más informativas para el modelo de detección, mediante aprendizaje activo.

2. Generar imágenes artificiales autoetiquetadas de malezas mediante modelos generativos.

La tesis tiene como objetivo relevar y explorar diversas técnicas de ambos mecanismos, analizando el beneficio obtenido en cuanto a rendimiento al entrenar diferentes modelos de detección. Aquellas técnicas que presenten mejores resultados serán combinadas e integradas en una única solución, de forma que simplifique la experimentación y su posterior uso en el proyecto principal, independientemente de cuál modelo de detección se utilice.

Dado el hecho de no encontrase finalizada la recolección de imágenes para el conjunto de datos que será utilizado en el entrenamiento del detector del robot, y considerando que no existen conjuntos de datos públicos con imágenes

de las especies de malezas objetivo encontradas en los cultivos de nuestro país, se optó por utilizar el conjunto de datos CottonWeedDet12 (Dang, Chen, Lu, y Li, 2023a), el cual consiste de 5.648 imágenes de alta resolución de 12 especies de malezas encontradas en campos de algodón del sur de los Estados Unidos.

Se espera desarrollar un marco teórico adecuado en los mecanismos de selección y aumento de datos con generación de imágenes artificiales, acompañado con un relevamiento de diferentes técnicas para cada mecanismo. Asimismo, se busca diseñar propuestas de flujos que combinen tanto las mejores técnicas de selección, como las de aumento de datos.

Por último, se espera que las mejoras al utilizar estos métodos se encuentren dentro de las limitaciones inherentes de las propias técnicas. En otras palabras, el aumento de rendimiento esperado está alineado con el relevado. En particular, cuando el esfuerzo humano disponible es bajo, se espera una mejora considerable al usar técnicas de selección de imágenes. Por otro lado, en lo que respecta a la generación de imágenes, se esperan mejoras relativas a la dificultad asociada a la tarea, es decir, considerando el hecho de que la generación de imagen debe ser lo suficientemente realista como para que un modelo de detección pueda extraer información de utilidad para su aprendizaje. De todas formas, en caso de obtener imágenes con alta fidelidad y diversidad, se esperan resultados no menores en especial cuando el número de imágenes manualmente etiquetadas es bajo o desbalanceado en alguna de sus especies.

1.1. Objetivos

Específicamente, el actual proyecto cuenta con los siguientes objetivos:

1. Relevar, explorar y proponer diferentes procesos que, utilizando la selección de imágenes con aprendizaje activo, y la generación de imágenes con modelos generativos, permitan mejorar el rendimiento de distintos modelos de detección de objetos, variando el esfuerzo humano, medido en el número de imágenes manualmente etiquetadas de las que se dispone durante dicho proceso.

2. Proveer de una implementación que, tomando como entrada las imágenes sin etiquetar, permita el uso de las diferentes técnicas. De forma tal que su salida sean las imágenes de entrada, junto con sus correspondientes etiquetas, y adicionalmente las imágenes generadas, también junto a sus correspondientes etiquetas. La implementación debe ser configurable, permitiendo el uso de diferentes técnicas y parámetros establecidos por el usuario. Se debe integrar con el software de asistencia para etiquetado CVAT (Computer Vision Annotation Tool) para mayor usabilidad.

3. Evaluar el rendimiento de diferentes modelos de detección utilizando las imágenes salientes de la anterior implementación, probando las técnicas individualmente y en conjunto, variando el esfuerzo humano disponible.

4. Evaluar empíricamente las imágenes generadas mediante expertos en malezas, así como utilizando métricas que midan la calidad y fidelidad de las mismas.

1.2. Organización del documento

El presente documento se organiza en seis capítulos, siendo la introducción el primero de ellos. El capítulo 2 consiste en el marco teórico del proyecto, con una breve pero suficiente presentación de los conocimientos necesarios para el pleno entendimiento del resto del documento, como es lo referido al problema de detección de objetos, el aprendizaje activo, y a la generación de imágenes.

En el capítulo 3 serán presentados los principales antecedentes al actual proyecto, incluyendo precedentes sobre aprendizaje activo en detección de objetos, la utilización del aprendizaje activo junto con generación de imágenes, generación de imágenes sintéticas de malezas, y mecanismos de auto-etiquetado de imágenes generadas artificialmente.

En el capítulo 4 se presenta el desarrollo principal del proyecto. Aquí se describe en detalle el diseño, análisis e implementación de las diferentes partes que conforman la solución propuesta. Se expone el flujo de datos y la interacción entre los diferentes componentes, además de criterios y métodos empleados en cada etapa; sus ventajas, limitaciones, y su impacto en el desempeño final del sistema.

En el capítulo 5 se describe el proceso de experimentación aplicado a las técnicas implementadas. Primero, se evalúa cada componente por separado, analizando sus resultados y las mejoras obtenidas. Posteriormente, se compara el rendimiento de los modelos de detección Faster R-CNN, YOLOv11m y RetinaNet, entrenados sin aplicar la selección mediante aprendizaje activo ni la técnica de aumento de datos propuesta, frente a los resultados obtenidos al emplear las distintas estrategias implementadas.

Finalmente, en el capítulo 6 se presentan las conclusiones del trabajo y se proponen posibles líneas de trabajo futuro.

Capítulo 2

Marco teórico

En este capítulo son presentadas las bases teóricas necesarias para el entendimiento de los componentes principales y técnicas utilizadas a lo largo del presente proyecto. Si se desea profundizar en las técnicas abordadas, dirigirse al documento del estado del arte (Padrón, Estramil, y Núñez, 2025). La estructura del capítulo se organiza en cuatro secciones, cada una enfocada en un aspecto importante del trabajo.

En la sección 2.1 se aborda de manera resumida el aprendizaje automático y las redes neuronales, el problema de la detección de objetos, con una descripción general del enfoque, y sus variantes más comunes. También se detallan los principales métodos de evaluación utilizados para medir su rendimiento.

En la sección 2.2 está dedicada a las técnicas de aprendizaje activo, destacando su funcionamiento básico, componentes, problemas y desafíos, para luego centrarse en su utilización dentro de la tarea de detección de objetos.

Por último, en la sección 2.3 se aborda la generación de imágenes sintéticas, enfocándose en los dos métodos principales, redes generativas antagónicas (GANs, Generative Adversarial Networks) y modelos difusos (DMs, Diffusion Models). Se describen sus principios de funcionamiento, ventajas y limitaciones.

2.1. Detección de objetos

La detección de objetos es una tarea fundamental en el procesamiento de imágenes y visión por computadora, cuyo objetivo es identificar y localizar automáticamente instancias de objetos dentro de una imagen o video. A diferencia de la clasificación de imágenes, donde se asigna una única etiqueta representativa a toda la imagen, la detección de objetos permite reconocer múltiples elementos de distintas clases y determinar su ubicación mediante el uso de cajas delimitadoras (bounding boxes). Esta tarea resulta esencial para aplicaciones que requieren no solo saber qué objetos están presentes, sino también dónde se encuentran.

2.1.1. Aprendizaje automático y redes neuronales

El aprendizaje automático (machine learning) es una rama de la inteligencia artificial que se enfoca en desarrollar algoritmos capaces de aprender patrones a partir de datos de entrada, sin ser explícitamente programados para realizar tareas específicas. En lugar de seguir instrucciones fijas, un modelo de aprendizaje automático busca mejorar su desempeño a medida que analiza más ejemplos, lo que se conoce como entrenamiento.

Aprendizaje supervisado

En el caso del aprendizaje supervisado, el modelo aprende a partir de ejemplos etiquetados. Estos ejemplos consisten en pares formados por una entrada y su correspondiente salida (o etiqueta). Por ejemplo, si quisiéramos crear un modelo de clasificación que determine si una imagen contiene un gato o un perro, los datos de entrada podrían consistir en los valores de los canales RGB (rojo, verde y azul, que representan la intensidad de color de cada píxel) en la imagen, mientras que las etiquetas asociadas a cada imagen serían las categorías correspondientes, en este caso, "gato" o "perro".

A medida que el modelo procesa más datos de entrenamiento, este ajusta su estructura interna, buscando una configuración que minimice el error en las predicciones y los valores reales. El entrenamiento consiste en iterar sobre los datos, permitiendo al modelo mejorar gradualmente su capacidad para realizar mejores predicciones. Un ejemplo de este tipo de modelo es k-NN (k vecinos más cercanos, k nearest neighbors), que clasifica un nuevo dato considerando los k puntos más cercanos en el espacio de características del conjunto de entrenamiento.

Aprendizaje no supervisado

A su vez, existen otras formas de aprendizaje automático, como el aprendizaje no supervisado, donde el modelo trabaja únicamente con datos de entrada sin sus correspondientes etiquetas, debiendo encontrar estructuras en los datos en si mismos. Por ejemplo, K-Means, el cual divide los datos en k grupos (clusters) basados en su similitud. Para esto, asigna a cada dato en el espacio de características al grupo cuyo centroide se encuentra más cercano, y luego actualiza los centroides en función de este nuevo estado, repitiendo el proceso hasta alcanzar una condición de parada (por ejemplo, que ningún dato cambie de grupo de una iteración a la siguiente).

Redes neuronales

Las redes neuronales son un tipo de arquitectura, especialmente relevante hoy en día, la cual es utilizada principalmente en el aprendizaje automático supervisado. Su funcionamiento se inspira en el cerebro humano y están compuestas por "neuronas" (o perceptrones) organizadas en capas. En la figura 2.1 se muestra una representación abstracta de una red neuronal. Cada neurona

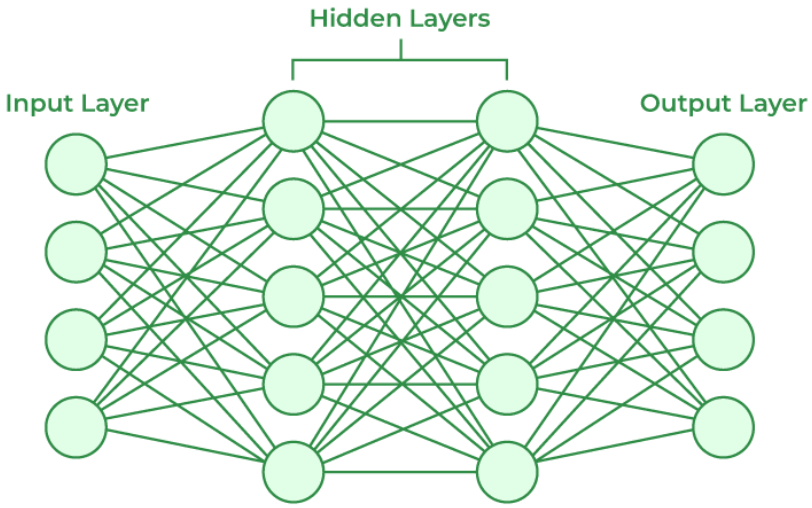


Figura 2.1: Visualización de la arquitectura conceptual de una red neuronal. Se muestran la capa de entrada (Input Layer), las capas ocultas (Hidden Layers), y la capa de salida (Output Layer), junto con sus conexiones. (GeeksforGeeks, 2025)

recibe un conjunto de entradas, las procesa mediante operaciones matemáticas, y produce una salida. Estas entradas pueden ser los datos de entrada originales del modelo, o las salidas generadas por neuronas de la capa anterior. Cada neurona toma sus entradas, las multiplica por un conjunto de pesos y le suma un sesgo. A esto se le aplica una función de activación, y se utiliza este resultado como salida de la neurona. Por último, para evaluar el rendimiento de la red, se utiliza una función de pérdida, que mide la discrepancia entre las predicciones generadas por el modelo y los valores esperados.

Son entonces estos pesos y sesgos los que el modelo ajusta a medida que es entrenado para reducir el error entre la salida predicha y la esperada, utilizando un algoritmo como descenso por gradiente junto con el resultado de la función de perdida. Esto permite que las redes neuronales sean capaces de aprender patrones más complejos en datos como imágenes, texto, o audio. Se puede profundizar sobre redes neuronales en el trabajo de Nielsen (2015).

2.1.2. Métodos clásicos para detección de objetos

Antes de entrar en detalle sobre la detección de objetos en conjunto con redes neuronales, se hace un breve resumen sobre métodos clásicos que no hacen uso de estas para generar una idea del problema, además de que, en caso de ser de interés, se pueda profundizar en el tema.

Previo a que las redes neuronales se volviesen populares, la detección de objetos se realizaba con técnicas basadas en reglas definidas manualmente. Estos métodos no aprenden directamente de los datos de entrada, sino que se especifican reglas diseñadas por una persona experta en los objetos a detectar.

Una de las estrategias más comúnmente usada consiste en la búsqueda de partes importantes de la imagen, las cuales permitan distinguir un objeto de otro, como por ejemplo, bordes, esquinas, texturas o formas en una imagen. Estos elementos son llamados "descriptores". Algunas de las técnicas para obtener estos descriptores son Transformada Invariante a la Escala (SIFT, Scale-Invariant Feature Transform) (Lowe, 2004), Transformada de Características Invariantes a la Escala (SURF, Speeded-Up Robust Features) (Bay, Tuytelaars, y Van Gool, 2006), Histograma de Gradientes Orientados (HOG, Histogram of Oriented Gradients) (Dalal y Triggs, 2005) o Patrones Binarios Locales (LBP, Local Binary Patterns) (Ojala, Pietikainen, y Maenpaa, 2002).

Una vez que se obtiene esta información, se procesa haciendo uso de un clasificador, como una máquina de vectores de soporte (SVM, Support Vector Machine), o un conjunto de árboles de decisión. Estos son encargados de clasificar el elemento como válido (perteneciente a la clase a detectar) o no. Para buscar los descriptores, una de las técnicas más usadas era la de ventana deslizante, lo cual significa recorrer toda la imagen, probando diferentes tamaños y posiciones, para comprobar si alguno coincide con el objeto que se busca.

Uno de los métodos más conocidos y utilizados en su momento fue el de Viola-Jones (Viola y Jones, 2001), diseñado especialmente para detectar rostros. Este método combina descriptores visuales sencillos, descriptores Haar, presentadas en el mismo artículo, con un algoritmo que descarta rápidamente zonas en las cuales seguramente no haya un rostro. Gracias a esto, funciona excepcionalmente rápido, incluso en sistemas de pocos recursos.

Otro enfoque importante fue el de los Modelos de Partes Deformables (DPM, Deformable Part Models) (Felzenszwalb, Girshick, McAllester, y Ramanan, 2009), el cual representa los objetos como un conjunto de partes conectadas entre sí, como si fuesen piezas móviles. Este método ayuda a reconocer objetos que puedan aparecer en diferentes localizaciones o posturas.

Por último, se comenzaron a utilizar otros métodos que, en lugar de recorrer toda la imagen, primero tratan de encontrar zonas que probablemente tengan un objeto, reduciendo así el esfuerzo necesario por la ventana deslizante. Esto se hacía con técnicas como Regiones Extremales Máximamente Estables (MSER, Maximally Stable Extremal Regions) (Donoser y Bischof, 2006) o Búsqueda Selectiva (Uijlings, Van De Sande, Gevers, y Smeulders, 2013), los cuales dividían la imagen en regiones prometedoras que luego eran analizadas en detalle.

Si bien estos métodos son efectivos en ciertos contextos, poseen limitaciones al enfrentarse a escenarios más complejos o con muchas variaciones. Con el aumento en la capacidad de cómputo, las redes neuronales comenzaron a ser utilizadas para esta tarea. A diferencia de los métodos explicados, estas permiten aprender directamente de los datos de entrada, sin la necesidad de diseñar de forma manual las reglas o los descriptores. En la siguiente sección se busca explorar técnicas modernas de detección de objetos, basadas en redes neuronales.

2.1.3. Redes neuronales en detección de objetos

Como se mencionó en la sección 2.1.1 el aprendizaje supervisado se basa en la utilización de un conjunto de datos etiquetados donde cada entrada está asociada a su salida esperada. Este enfoque, donde se aprende a partir de imágenes etiquetadas de ejemplo, resulta adecuado para tareas de visión por computadora. Gracias a este proceso, el modelo logra generalizar su conocimiento a datos no vistos y resolver problemas como la clasificación de imágenes o detección de objetos. La detección de objetos plantea un desafío complejo, ya que involucra interpretar información visual altamente variable, por ejemplo, los objetos pueden aparecer bajo diferentes escalas, posiciones, iluminaciones y perspectivas dentro de una imagen. Resolver simultáneamente las tareas de clasificación y localización requiere un modelo capaz de capturar patrones de alta complejidad y adaptarse a variaciones espaciales. Los modelos que utilizan las redes neuronales profundas conforman el estado del arte actual en lo que respecta a la detección de objetos, en particular aquellos basados en Transformers (Vaswani y cols., 2017).

Los modelos neuronales de detección de objetos se basan en el aprendizaje supervisado, donde el modelo aprende a partir de ejemplos etiquetados, esto quiere decir que se necesita un conjunto de datos (dataset) previamente etiquetado, es decir que cada ejemplo de entrenamiento consta de:

- **Una imagen**
- **Conjunto de etiquetas (labels):** La posición de cada objeto mediante la caja delimitadora y la clase a la que pertenece el objeto detectado.

Formalmente cada imagen x_i está asociada a un conjunto de anotaciones y_i, donde cada anotación tiene un conjunto de tuplas de la forma (c, b) siendo:

- c la clase del objeto
- $b = (bx, by, bh, bw)$ las coordenadas de la caja delimitadora, representado el centro (bx, by), la altura bh y el ancho bw.

El objetivo del entrenamiento es que el modelo aprenda una función que, dado una imagen x_i, prediga correctamente el conjunto de tuplas (c, b) correspondiente a los objetos presentes en la imagen. En otras palabras, el modelo debe ser capaz de identificar la clase y la localización de cada instancia de objeto en x_i, reproduciendo lo más fielmente posible el conjunto de anotaciones y_i.

2.1.4. Arquitecturas para detección de objetos

A continuación, se describen las principales arquitecturas utilizadas en la detección de objetos. Estas arquitecturas se clasifican en tres grandes categorías, según la forma en que abordan el problema de localizar y clasificar los objetos:

- Detectores en dos etapas (two-stage detectors): Primero generan un conjunto reducido de regiones de interés y, posteriormente, clasifican y refinan las predicciones sobre esas regiones. Ejemplos de este tipo de detectores incluyen a Faster R-CNN (S. Ren, He, Girshick, y Sun, 2015).

- Detectores en una etapa (one-stage detectors): Predicen directamente, en un único paso, tanto las cajas delimitadoras como las clases correspondientes para los objetos en la imagen. Entre estos detectores de una etapa destaca YOLO (Redmon, Divvala, Girshick, y Farhadi, 2016).

- Con la introducción de la arquitectura Transformer, y ante la imposibilidad de encuadrarlo como detector de una o dos etapas, se creó recientemente una nueva categoría de detectores, los cuales modelan relaciones globales en la imagen, eliminando la necesidad de generar regiones propuestas. Actualmente algunas variantes del mismo conforman el estado del arte en la detección de objetos. Un ejemplo de un detector con esta arquitectura es DETR (Carion y cols., 2020).

En el contexto del desarrollo de este proyecto, no se ha definido la arquitectura específica a utilizar, dado que esta decisión depende de la definición general del proyecto mayor en el que se integra este trabajo.

Cabe destacar que la elección entre una arquitectura está estrechamente relacionada con los requerimientos particulares del sistema:

- Los detectores en una etapa (como YOLO) presentan un mayor velocidad de inferencia, resultando más adecuados para aplicaciones en tiempo real.

- Los detectores en dos etapas (como Faster R-CNN) suelen ofrecer una mayor precisión a costa de un mayor tiempo de procesamiento, no siendo lo mejor para trabajar en tiempo real.

- Los detectores basados en Transfomers (como DETR) eliminan la necesidad de generar regiones propuestas, ofrecen un balance entre precisión y velocidad, aunque suelen requerir un mayor tiempo de entrenamiento.

2.1.5. Evaluación de sistemas de detección de objetos

La evaluación de los sistemas de detección de objetos es fundamental para comparar modelos e identificar problemas o posibles mejoras. A continuación se presenta de manera resumida algunas de las métricas más utilizadas en el área de aprendizaje automático en lo que a detección de objetos refiere.

Intersección sobre unión (IoU, Intersection over Union)

Es la métrica estándar para evaluar la superposición entre dos cajas delimitadoras. Se define como el cociente entre el área de intersección y el área de unión de dos cajas. Sea A el área de la primera caja y B el de la segunda. Entonces, el IoU se calcula como:

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

Podemos definir un umbral para considerar que una detección es correcta. Por ejemplo, un umbral común es de 0.5. Si al calcular el IoU entre una caja predicha y una real, este supera el umbral definido, y la clase predicha se corresponde a la real, se considera como una detección correcta.

En base a esto, en el contexto de detección de objetos podemos definir los siguientes valores:

- **Verdadero positivo (TP, True Positive):** Cantidad de predicciones las cuales su caja se superpone con una caja real, con la que además coincide en su clase predicha.
- **Falso positivo (FP, False Positive):** Cantidad de predicciones las cuales cumplen que, o se superponen con una caja real, pero difieren en la clase predicha, o no se superpone con ninguna caja real, o es una detección duplicada.
- **Falso negativo (FN, False Negative):** Es la cantidad de cajas reales que no se superponen con ninguna predicción.

Métricas generales

La precisión mide la proporción de detecciones positivas correctas. En este contexto, una proporción alta indica que cuando el modelo detecta un objeto, es más probable que sea una detección correcta. La precisión se calcula como:

$$Precision = \frac{TP}{TP + FP}$$

La recuperación (recall) mide la proporción de cajas reales que son predichas correctamente. Una proporción más alta indica que el modelo es capaz de detectar una mayor cantidad de los objetos presentes en la imagen. La recuperación se calcula como:

$$Recuperacion = \frac{TP}{TP + FN}$$

Por último, el puntaje F1 se utiliza como forma de calcular una única métrica que balancee tanto la precisión como la recuperación, representando de forma resumida el rendimiento del modelo. El puntaje F1 se calcula como:

$$F1 = 2 \cdot \frac{Precision \cdot Recuperacion}{Precision + Recuperacion}$$

Métricas específicas

La curva de precisión-recuperación muestra la relación entre la precisión y la recuperación del modelo, evaluadas en diferentes umbrales de confianza para las predicciones.

- Cada predicción tiene asociada una puntuación de confianza dada por el propio modelo.
- Al variar ese umbral entre 0 y 1, se modifican las predicciones que se aceptan como válidas.
- Para cada valor del umbral, se calculan la precisión y la recuperación, y se grafíca un punto cuyas coordenadas son (precisión, recuperación).
- En general, a medida que la recuperación aumenta, la precisión disminuye, puesto que se aceptan más predicciones con menor confianza, lo que puede incluir falsos positivos.

En base a esta curva se pueden calcular otras métricas que sinteticen el rendimiento del modelo.

- **Precisión Promedio (AP):** Es el área bajo la curva precisión-recuperación para una clase especifica.
- **Media de Precisión Promedio (mAP)**:Se calcula como el promedio del AP sobre todas las clases detectadas por el modelo. Resume el rendimiento general. Si C es la cantidad de clases en el modelo, el mAP se calcula como:

$$mAP = \frac{1}{C} \cdot \sum_{1}^{C} AP_i$$

- **mAP5@50:** Se calcula como el mAP del modelo al fijar el umbral de IoU en 0.5.
- **mAP@50:95:** Similar al mAP@50, se calcula como el mAP del modelo al utilizar 10 umbrales de IoU entre 0.5 y 0.95, con incrementos de 0.05, promediando los resultados.

2.2. Aprendizaje activo

A continuación se presenta el marco teórico para el segundo de los componentes utilizados. Veremos que es aprendizaje activo (AL, active learning), cuales son sus objetivos y aspectos que hay que tener en cuenta al hablar de aprendizaje activo.

Cuando se entrena un modelo de aprendizaje automático hay que tener en cuenta la información con la que se entrena el modelo, ya que a partir de esa

información el modelo va a aprender aspectos de la realidad que le van a servir para realizar la tarea específica en la que se esté entrenando. Las preguntas básicas que surgen cuando se enfrenta el problema de determinar que ejemplos se deberían etiquetar para entrenar el modelo son: ¿los ejemplos que se están eligiendo son buenos? ¿Son representativos? ¿Qué cantidad de imágenes se necesita para tener un buen desempeño? Si hay una cantidad limitada de recursos para etiquetar los ejemplos de entrenamiento, ¿qué ejemplos conviene etiquetar? La respuesta a estas preguntas se responde bajo el nombre de AL. AL es el concepto por el cual nos referimos a todas estas preguntas, es importante destacar que es un concepto en si y no una técnica específica, siguiendo las definiciones que se verán a continuación se desencadenan numerosas formas de utilización que dependen fuertemente del problema en el que se esté trabajando y el método de aprendizaje automático que se utilice.

Definamos formalmente que es AL. Sabiendo que los algoritmos supervisados aprenden de un conjunto de entrenamiento inicial conformado por instancias etiquetadas, denotamos este conjunto como:

$L = \{\langle \mathbf{x_1}, y_1\rangle, \langle \mathbf{x_2}, y_2\rangle, \ldots, \langle \mathbf{x_n}, y_n\rangle\}$.

donde n es el número de instancias etiquetadas, y x_i, y_i son la instancia y su etiqueta respectivamente. El objetivo es aprender una hipótesis $h \in H$ que haga que cada instancia se corresponda con su etiqueta, donde H es el espacio de hipótesis.

Por otro lado se tiene el conjunto $U = \{\mathbf{u_1}, \mathbf{u_2}, \ldots, \mathbf{u_m}\}$, donde m es el número de instancias sin etiquetar, y u_i son estas instancias. La idea principal consiste en entrenar un aprendiz con el conjunto inicial L, y luego usar ese aprendiz sobre el conjunto U para elegir b instancias que sean más informativas o tengan mayor incertidumbre. Estas instancias son etiquetadas por un oráculo humano; luego, el modelo se vuelve a entrenar con las nuevas instancias etiquetadas y se repite el proceso nuevamente hasta alcanzar una condición de parada. En la figura 2.2 se puede observar este ciclo.

2.2.1. Componentes del aprendizaje activo

Cualquier proceso de AL consiste en cuatro componentes principales.

- **Datos:** Etiquetados y no etiquetados.
- **Algoritmo de aprendizaje:** Es un modelo usado para evaluar el proceso de anotación. El modelo se va actualizando iterativamente a medida que se agregan nuevas instancias etiquetadas.
- **Presupuesto:** El presupuesto es la cantidad de instancias que voy a etiquetar del total.
- **Estrategia de consulta (QS, Query Strategy):** También conocido como función de adquisición (Acquisition Function), el cual es una función que se usa para evaluar las instancias en U y seleccionar las más informativas y representativas.

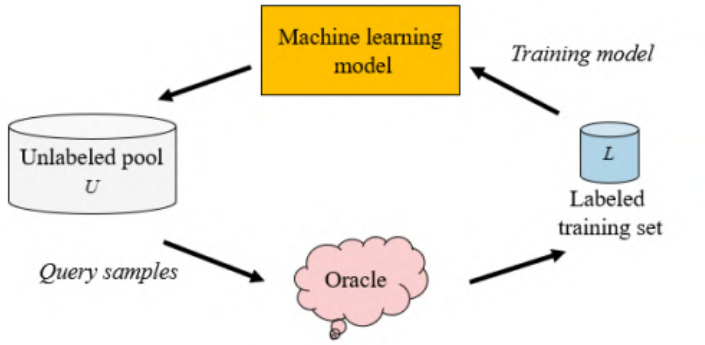


Figura 2.2: Ciclo de funcionamiento de aprendizaje activo, el cual se compone de: Conjunto de entrenamiento etiquetado y no etiquetado, el oráculo y el modelo de aprendizaje automático. (Tharwat y Schenck, 2023)

- **Experto:** Persona experta, usualmente llamada oráculo, quien anota las imágenes elegidas por la QS.

Estas definiciones representan la esencia de AL. Nótese que se define a un nivel lo suficientemente abstracto para que abarque cualquier modelo de aprendizaje automático. Cuando se determina exactamente el modelo y la forma de etiquetar los datos es cuando se comienza a utilizar una técnica o conjunto de técnicas específicas, implementando una solución particular.

Las instancias a elegir se toman a partir de la QS. Lo cierto es que existen muchas maneras para determinar qué instancias tomar, en la anexo D se muestran las formas más conocidas de selección y en particular dos de las que se usarán más adelante en este trabajo, que son incertidumbre de menor confianza y basado en diversidad. La forma más simple de selección claramente es realizarla de forma aleatoria, que establece una línea base para todas las otras formas posibles. Cabe destacar que de por sí la selección de forma aleatoria empíricamente no es una línea base fácil de superar, sobre todo si se utiliza un conjunto bien balanceado, o la proporción de imágenes seleccionadas es alta.

Aunque AL puede utilizarse con cualquier modelo, la eficacia e incluso la viabilidad de las distintas estrategias de selección que mencionamos se ven afectadas por el tipo de modelo que se este usando.

Originalmente, cuando se hablaba de AL, todo giraba en torno a problemas de baja dimensionalidad, es decir, aquellos donde los modelos trabajan con unas pocas decenas o cientos de parámetros, como en tareas de modelado de decisiones, clusterización o clasificación tradicional.

La introducción de las redes neuronales profundas (DL), en cambio, llevó a trabajar con modelos que manejan miles o millones de parámetros, lo que cambia radicalmente el panorama, ya que no todos los métodos de selección son aplicables en estos contextos.

A modo de ejemplo, cuando trabajamos con pocos parámetros, sería viable

medir el cambio esperado del modelo, ya que podemos seguir fácilmente cómo varían esos parámetros al actualizar el modelo. Sin embargo, cuando se trabaja con redes neuronales profundas —donde existen millones de parámetros—, realizar un seguimiento individual del cambio se vuelve impracticable.

En vista de la necesidad de distinguir entre baja y alta dimensionalidad, comienzan a surgir numerosos trabajos bajo el nombre de aprendizaje activo profundo (DAL, Deep Active Learning), cuyo relevamiento se hace en en el documento P. Ren y cols. (2021). El objetivo detrás es aplicar Active Learning (AL) específicamente en escenarios de alta dimensionalidad.

A grandes rasgos, el ciclo de funcionamiento de DAL es similar al del AL tradicional, con la diferencia principal de que el modelo que se entrena es una red neuronal. En cuanto a la operativa, uno de los aspectos más destacados es que DAL presenta una limitación importante, ya que la estrategia de etiquetar una sola muestra por vez no resulta efectiva en el contexto de DL. Por este motivo, la mayoría de los estudios de DAL, según P. Ren y cols. (2021), emplean estrategias de consulta por lotes, donde se seleccionan múltiples muestras de forma simultánea.

En este enfoque la decisión no se basa en evaluar individualmente cada instancia para luego elegir las mejores, sino que se analiza la calidad del conjunto de instancias seleccionado, buscando maximizar la representatividad global del lote.

2.2.2. Desafíos al usar aprendizaje activo

Algunos de los principales desafíos que se encuentran al trabajar con AL son:

- **Ruido en los datos:** Uno de los principales problemas que afecta a AL son las instancias de entrenamiento mal etiquetadas, estas instancias tienen un impacto negativo que puede ser más prejudicial que tener un conjunto de entrenamiento chico.

- **Desbalance de clases:** El desbalance en la cantidad de instancias de cada clase puede ocasionar que el aprendiz activo escoja las instancias de la clase con menor ratio de instancias, ya que son las que tiene mayor nivel de incertidumbre. Muchos aprendices activos tratan de usar algoritmos de muestreo para balancear la información, por ejemplo, SMOTE (Chawla, Bowyer, Hall, y Kegelmeyer, 2002).

- **Elección de instancias iniciales:** Otro desafío de AL consiste en determinar que instancias iniciales tomar. En el proceso del AL es necesario tener un conjunto etiquetado para entrenar por primera vez al aprendiz, pero la pregunta aquí es cómo determinamos cuál será el mejor conjunto para comenzar a realizar el proceso, además del tamaño de dicho conjunto.

- **Criterio de parada:** En AL el proceso continúa hasta que se alcanza un criterio de parada. El desafío aquí es determinar dicho criterio que mejor

se adapte al problema en el que se esté trabajando, ya que la elección es un balance entre el costo de etiquetado y la eficiencia del algoritmo. El algoritmo puede ser terminado luego de una cierta cantidad de instancias etiquetadas o por alcanzar un cierto grado de aciertos en el modelo. El problema es que si no se presta atención al criterio de parada se podría continuar etiquetando más allá de lo que era previsto, además etiquetar una cantidad fija tampoco puede incurrir en grandes niveles de eficiencia del modelo.

- **Instancias anómalas (Outliers):** Las instancias anómalas son instancias que tienen una desviación significativa del resto de las instancias. Los conjuntos de datos que contienen anomalías afectan considerablemente a AL si estas instancias anómalas son seleccionadas y etiquetadas. Según Tharwat y Schenck (2023) etiquetar estas instancias no solo es una pérdida en costo y tiempo, ya que son instancias que se encuentran alejadas de los datos normales, sino que también afecta considerablemente el desempeño en el resto de las instancias.

  Una solución a las anomalías podría ser removerlas o al menos evitar etiquetarlas, esto se puede hacer detectándolas de forma geométrica (Klidbary, Shouraki, Ghaffari, y Kourabbaslou, 2017). Otro de los desafíos es que si se tiene un conjunto desbalanceado, la clase minoritaria puede verse como anomalías.

2.2.3. Aprendizaje activo profundo y detección de objetos

En el contexto de este proyecto, el conjunto de instancias U está compuesto por imágenes no etiquetadas. El objetivo es seleccionar un subconjunto de U con b imágenes para ser etiquetadas, siguiendo el procedimiento de AL. La información disponible en un problema de detección de objetos es más rica y compleja que en clasificación como se menciona en la sección 2.1.3. A medida que se cuenta con mayor información de entrada, la aplicación de DAL comienza a volverse más compleja. Para entenderlo mejor, es útil analizar primero cómo funcionaría DAL en el caso más sencillo: la clasificación de imágenes.

En este escenario, se dispone de un conjunto de imágenes, cada una asociada a una única clase. El modelo toma una imagen del conjunto de entrenamiento, realiza una pasada hacia adelante (forward pass) y, en base a sus creencias, genera una distribución de probabilidad sobre las clases posibles. Esta distribución refleja qué tan probable considera el modelo que la imagen pertenezca a cada clase.

En este contexto, una forma común de cuantificar la incertidumbre del modelo es mediante la entropía de la distribución de probabilidad que genera sobre las clases. Cuando la entropía es baja, significa que el modelo asigna una probabilidad alta a una única clase, lo que indica alta confianza en su predicción. Por el contrario, una entropía alta refleja una distribución más uniforme entre las clases, lo que implica mayor incertidumbre, ya que el modelo no está claramente inclinado hacia una única respuesta.

Una vez realizada la asignación, evaluar la incertidumbre de cada caja implica analizar no solo la confianza de clasificación sino también la precisión espacial (qué tanto se ajusta la predicción al objeto real). Una caja podría tener alta confianza en su clase, pero mala localización, o viceversa. Se deben combinar ambas fuentes de error para decidir si una instancia es incierta o segura.

Estos desafíos —asignación entre predicciones y anotaciones, definición de incertidumbre, y toma de decisiones de selección— complican significativamente la aplicación de DAL en tareas de detección de objetos respecto al caso de clasificación simple.

Todas estas cuestiones serán abordadas en la sección 4.2.1, donde se detallará la implementación del método Plug and Play Active Learning (PPAL), aplicado específicamente a conjuntos de datos de malezas, considerando las particularidades propias de los problemas de detección de objetos en imágenes agrícolas.

2.3. Modelos generativos de imágenes

Hoy en día existe una gran cantidad de modelos de aprendizaje automático, con diferentes objetivos, tamaños y arquitecturas, que pueden ser clasificados en dos grandes categorías: los modelos discriminativos y los modelos generativos, donde la principal diferencia entre ellos subyace en la distribución de probabilidad aprendida.

Formalmente, para el caso de aprendizaje supervisado, dada la distribución de probabilidad de los datos de entrada D y de su clase objetivo L, los modelos discriminativos aprenden la distribución condicional $P(L|D)$, mientras que los generativos aprenden la distribución conjunta $P(L, D)$. Intuitivamente, los modelos discriminativos, tiene como principales objetivos tareas de clasificación y/o regresión; mientras que los modelos generativos pueden ser utilizados tanto para realizar predicciones (como por ejemplo el método de Naive Bayes), como también para generar nuevos datos, similares a los utilizados en el entrenamiento del modelo.

En particular, existe una gran variedad de modelos generativos capaces de crear nuevos datos. Desde la generación de texto (por ejemplo, ChatGPT, Gemini, Llama), generación de video (Sora, Veo 2), generación de audio (MusicLM, MusicGen, Suno AI) y generación de imágenes; en este último se enfoca esta sección. A lo largo del tiempo, lo referido a la generación de imágenes artificiales es, sin lugar a duda, un área de gran valor e interés, no solo por sus posibles usos en el sector audiovisual, sino también para complementar técnicas de aprendizaje automático, como por ejemplo el aumento de datos. Las técnicas de aumento de datos son ampliamente utilizadas en el entrenamiento de diferentes modelos (como los modelos de detección de objetos); estas consisten en expandir el conjunto de datos de entrenamiento, permitiendo así aumentar sus capacidades de generalización. Diferentes técnicas son posibles: desde modificaciones de los colores de la imagen (por ejemplo, contraste, brillo, saturación), modificar la geometría de la imagen (rotaciones, recortes, transformación), combinar múltiples imágenes, entre otras técnicas.

En lo que respecta a este proyecto, se busca poder realizar un aumento de datos generando artificialmente imágenes de las diferentes malezas que se busca aprender a reconocer. Esta sección tiene el propósito de introducir al lector a los principales conocimientos relacionados a los modelos generativos de imágenes, los cuales son necesarios para comprender la primera componente del proyecto. Primero se presentan las principales arquitecturas que conforman hoy en día el estado del arte en la generación de imágenes, luego se profundiza en el caso de las redes generativas antagónicas y los modelos difusos, y finalmente métodos de evaluación de los modelos generativos.

2.3.1. Principales arquitecturas de modelos generativos de imágenes

Actualmente, existen diferentes arquitecturas de modelos generativos, contando cada una de ellas con un gran número de variantes. En 2013 fueron presentados los Autocodificadores Bayasianos Variacionales (VAE, Auto-Encoding Variational Bayes) (Kingma, Welling, y cols., 2013), los cuales fueron un avance significativo en lo que respecta a los autocodificadores, y en particular en la obtención de imágenes con una composición de elementos coherente en colores y forma. Posteriormente, en 2014, se introdujeron las Redes Generativas Antagónicas (GANs, Generative Adversarial Networks) (Goodfellow y cols., 2020) las cuales fueron uno de los mayores avances en la generación de imágenes, permitiendo una notable mejora en nitidez y fidelidad con respecto a las VAEs.

En 2020 fueron presentados los modelos difusos (Ho, Jain, y Abbeel, 2020), los cuales significaron una gran mejora frente a las GANs, sobre todo en lo que respecta a estabilidad de entrenamiento y calidad de las imágenes generadas. Hoy en día este tipo de modelos, en particular ciertas implementaciones como Stable Diffusion (Rombach, Blattmann, Lorenz, Esser, y Ommer, 2022), conforma el estado del arte en la generación de imágenes.

Como será profundizado en la sección 4.3, tanto las GANs, como los modelos difusos, eran opciones posibles para ser utilizadas en el actual proyecto, aunque finalmente se optó por la última. De todas formas, con el fin de brindar al lector el marco teórico necesario para desarrollar dicha elección, se presenta brevemente en la siguiente sección la arquitectura GAN, y posteriormente la difusa.

2.3.2. Redes Generativas Antagónicas

En 2014 fueron presentadas las Redes Generativas Antagónicas (GANs, Generative Adversarial Networks) (Goodfellow y cols., 2020), mostrando una significativa mejora en la capacidad de generación de imágenes artificiales. Tal fue su impacto, que a lo largo de los últimos años, en especial previo a la presentación de los modelos difusos, fueron propuestos un considerable número de variantes, algunas mostrando resultados sorprendentes.

Las GANs se componen de dos modelos, un generador (G) y un discriminador (D), los cuales tendrán objetivos contrarios, entrenándose mutuamente

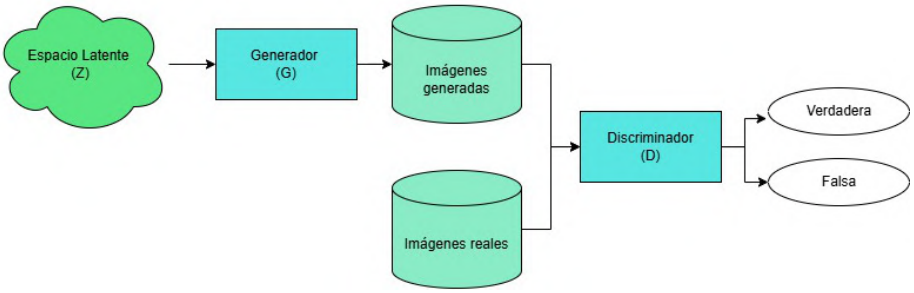


Figura 2.3: Arquitectura general de un modelo GAN. Los componentes que se entrenan son el generador (G) y el discriminador (D)

producto de dicha interacción. En particular, el generador tiene como objetivo crear imágenes sintéticas capaces de engañar al discriminador, mientras este último debe ser capaz de distinguir las imágenes reales de las generadas.

En la figura 2.3 se presenta la arquitectura en alto nivel de un modelo GAN. A partir de un vector z, perteneciente a lo que se conoce como espacio latente (Z), el generador produce una imagen ($G(z)$). Varias de las imágenes generadas y reales (x) son ingresadas al discriminador, el cual deberá clasificarlas como reales o falsas (generadas). Por un lado, el objetivo del discriminador D, es discernir entre imágenes reales y falsas, eso es $D(G(z)) = 0$ y $D(x) = 1$. Por otro lado, el objetivo del generador es que $D(G(z)) = 1$, es decir, que el discriminador etiquete como reales las imágenes generadas.

Aunque esta arquitectura haya logrado avances significativos en la generación de imágenes artificiales, también es cierto que tiene varias desventajas. La mayoría de ellas se debe a la propia naturaleza antagónica del entrenamiento del modelo, donde, ante desbalances entre el generador y el discriminador, producen efectos negativos que pueden comprometer el aprendizaje del modelo. Un caso concreto, y además uno de los más desafiantes, es el realismo en la interacción entre los elementos que componen la imagen generada, por ejemplo, que un objeto no esté apoyado sobre una superficie, o que un objeto se superponga con otro.

En la sección 4.3 se describe en mayor detalle varios de los principales problemas que presentan las GANs, cómo estos afectan al presente proyecto, y la posterior mejora que presentaron los modelos difusos, los cuales serán presentados a continuación. Para mayor detalle en lo que refiere a variantes de GANs, dificultades y trabajos asociados, dirigirse al segundo capítulo del documento del estado del arte (Padrón y cols., 2025).

2.3.3. Modelos difusos

En 2020 fueron presentados los modelos difusos (DM, diffusion models) (Nichol y Dhariwal, 2021), los cuales recientemente han logrado establecerse como el estado del arte en diferentes tareas asociadas a la generación de imáge-

nes. A diferencia de las GANs que generaban la imagen en una sola iteración del generador, los modelos difusos tienen una naturaleza iterativa, donde cada imagen es procesada múltiples veces, eliminando ruido en cada iteración. Otra diferencia fundamental con respecto a las GANs, es que no tiene un entrenamiento de tipo antagónico, sino que son modelos basados en probabilidad, los cuales tienen como ventaja un entrenamiento significativamente más estable y simple.

En particular, el proceso se divide en dos partes iterativas, ambas con T iteraciones: el "proceso hacia adelante" $q(x_{t+1}|x_t)$ (FP, forward process) y el "proceso hacia atras" $p(x_t|x_{t+1})$ (BP, backward process). El primero consiste, en cada iteración t, en la agregación gradual de ruido gaussiano al estado x_t, formando así el estado x_{t+1}; en contraposición, el segundo proceso tiene como entrada el estado x_{t+1}, y mediante el modelo difuso, recuperar gradualmente la imagen mediante la eliminación de ruido, formando así el estado x_t. Entonces el estado x_0 consiste en la imagen sin alteraciones, mientras que el estado x_T está formado exclusivamente de ruido gaussiano. Notar que ambos procesos afectan al estado x_t de forma gradual, se conoce como planificador (scheduler) al componente cuya función es determinar en qué grado se modifica cada estado en ambos procesos, o sea, cuánto ruido se agrega, o cuánta información de la imagen se recupera.

Se define el parámetro β_t que determina el grado de afectación en cada etapa dado por el planificador.

Sin lugar a duda, lo más relevante en lo que refiere a la generación de imágenes es el "proceso hacia atrás", ya que es este el que permitirá la generación de imágenes a partir de ruido. En lo que respecta a la inferencia del modelo, el punto de partida será el estado x_T, el cual tiene las mismas dimensiones que la imagen, pero se compone exclusivamente de ruido gaussiano. En cada iteración el estado x_i será ingresado al modelo difuso, el cual puede ser visto como un auto-codificador, cuya función es eliminar el ruido de la entrada (Rombach y cols., 2022). El modelo difuso hará una predicción sobre la imagen real (x_0), pero en vez de terminar el proceso con dicha predicción, se combina (según el valor de β_t del planificador) el estado x_t con la predicción antes hecha, dando como resultado x_{t-1}. Este proceso es repetido hasta alcanzar el estado x_0, donde se obtiene la imagen generada.

Existen diferentes variantes de modelos difusos, comúnmente todos tienen al menos como entrada el estado x_t y el número de la iteración actual (t). Una de las implementaciones más populares es Stable Diffusion (se presenta en la sección 2.3.3), el cual, como una de sus varias características, permite la entrada de información adicional que permita condicionar la imagen que se busca generar. También existen variantes según en qué espacio están los estados x_t, la arquitectura del modelo difuso, los planificadores, entre otros. Para mayor información sobre los modelos difusos, se recomienda leer el trabajo (Erdem, 2023), y en lo que respecta a las variantes y aplicaciones, consultar el capítulo 3 del documento del estado del arte (Padrón, Estramil, y Núñez, 2025).

Modelos de difusión latentes

Aunque la propuesta original de los modelos difusos haya sido un gran avance en la generación de imágenes sintéticas en comparación con las GANs, varias mejoras han sido necesarias. En particular, en lo que respecta a los mecanismos de control a la hora de generar imágenes, como también en la elevada demanda computacional necesaria para el entrenamiento e inferencia de los DM.

En esta sección se presentarán brevemente los modelos de difusión latentes (LDM, latent diffusion models) (Rombach y cols., 2022), los cuales buscan reducir el poder de cómputo necesario, como también agregar mecanismos de control. Una de las implementaciones más populares, y utilizadas en este proyecto, es Stable Diffusion, desarrollada por Stability AI en colaboración con CompVis y RunwayML.

Originalmente, el espacio de los estados x_t, era el mismo espacio que el de las imágenes, eso quiere decir que si se está generando una imagen RGB de resolución 1024×1024, entonces, a menos de orden, cada estado x_t tiene dimensión $1024 \times 1024 \times 3$. Una consecuencia evidente es el elevado consumo de memoria y tiempo de cómputo. En cambio, los LDM, como su nombre lo indica, proponen que los estados x_t se encuentren en un espacio de menor dimensión (bidimensional) y tamaño, llamado espacio latente. El proceso de entrenamiento consiste en dos etapas:

1. Primero se entrena un autocodificador, el cual usando un codificador $\mathcal{E}$ reduce la dimensión de la imagen de entrada por un factor (usualmente entre 4 u 8), generando una representación de la misma ($\mathcal{E}(x)$) en un espacio latente (Z), y luego, pudiendo reconstruirla a partir del decodificador $\mathcal{D}$.

2. Las imágenes del conjunto de entrenamiento son llevadas al espacio latente, y posteriormente es entrenado el modelo difuso de forma similar al explicado en la anterior sección. La arquitectura del modelo difuso es de tipo U-Net (Ronneberger, Fischer, y Brox, 2015), y fueron agregados bloques de atención cruzada con el fin de agregar entradas adicionales para condicionar la generación del modelo.

Es importante resaltar que el uso del espacio latente disminuye drásticamente las demandas computacionales de los modelos difusos. Además que al tratarse de modelos basados en probabilidad, es común que el mismo tienda a consumir mucho tiempo del entrenamiento en reflejar detalles imperceptibles de alta frecuencia, mientras que el espacio latente obtenido al entrenar el autocodificador con regularizadores específicos, tiende a ignorar estos detalles, y por ende el modelo difuso se enfoca en los detalles visualmente más relevantes (Rombach y cols., 2022). Notar que, en lo que respecta a la inferencia, una vez que se obtiene el estado x_0, el resultado final será la imagen $\mathcal{D}(x_0)$, resultante de aplicar el decodificador sobre x_0.

En lo que respecta al condicionamiento, se utiliza el mecanismo de atención, en particular la atención cruzada, con el fin de poder condicionar al modelo en la

generación de imágenes mediante entradas externas. Se pueden usar diferentes tipos de entrada, texto, imágenes, mapas semánticos, entre otros. En cualquier caso, se cuenta con un codificador específico para el tipo de entrada, el cual debe ser entrenable, convirtiendo a la entrada condicional, en un vector compatible con las dimensiones esperadas por mecanismo de atención cruzada.

Ahora que se cuenta con modelos difusos computacionalmente eficientes, y que además permiten condicionar la generación de imágenes, se continuará en la próxima sección mecanismos para ajustar estos modelos; de forma tal que utilizando modelos preentrenados, se pueda aprender a generar imágenes de nuevos concepto, en nuestro caso, las malezas que buscamos aprender a detectar.

2.3.4. Ajuste de modelos difusos

Como se explicó en la introducción a este proyecto, las imágenes artificiales que se buscan generar tienen como propósito ser utilizadas en un aumento de datos del conjunto de entrenamiento, en consecuencia, es necesario contar con la capacidad de generar imágenes de los conceptos que se buscan aprender; en este caso, cada una de las especies de las malezas objetivo. Sin embargo, los modelos difusos son incapaces de imitar la apariencia de un objeto poco frecuente en el conjunto de entrenamiento. Además, debido a su falta de expresividad, intentar generar un concepto mediante una descripción detallada del mismo, resultará en una imagen cuya apariencia seguirá difiriendo significativamente de la buscada (Ruiz y cols., 2023). Tal como se preveía, la situación antes descrita ocurre con las malezas que buscamos generar, indicando la necesidad de ajustar el modelo difuso con imágenes de las malezas.

Los LDM requieren de un gran número de imágenes para poder ser entrenados, es por ello que comúnmente se ajustan modelos preentrenados con el fin de adoptar los nuevos conceptos. Existen varios métodos para poder realizar la adopción de nuevos conceptos, cada uno con particularidades. Un método posible, y comúnmente utilizado, es simplemente continuar con el entrenamiento del modelo, pudiendo el proceso ser optimizado opcionalmente, como LoRA (Hu y cols., 2022).

En lo que respecta a métodos con salidas de poco tamaño, Textual Inversion (Gal y cols., 2022a) se destaca por realizar su entrenamiento sobre la representación vectorial de un token especificado. También existen métodos como Custom Difussion (Kumari, Zhang, Zhang, Shechtman, y Zhu, 2023), especializados en el ajuste de un modelo en múltiples conceptos, y la generación de imágenes compuestas por varios de estos. Otro método muy popular es Dreambooth (Ruiz y cols., 2023), que se beneficia del conocimiento de conceptos más generales para inducir en un determinado token, el concepto más específico que se busca aprender a generar.

Dadas las características del proyecto, Dreambooth resulta el método de ajuste más adecuado; la razón es discutida en detalle en la sección 4.3. Por lo tanto, se presenta a continuación en qué consiste Dreambooth.

Dreambooth

Dreambooth (Ruiz y cols., 2023) es un popular método de ajuste de modelos difusos, el cual se distingue por su particular forma de establecer las instrucciones (prompt) para cada imagen, y la función de regularización utilizada, "Preservación previa específica de la clase" (Class-specific Prior Preservation).

Este método utiliza un mecanismo de entrenamiento análogo al utilizado cuando el modelo difuso fue entrenado en primera instancia, utilizando la misma función de pérdida (a menos de la regularización), y ajustando los pesos de todas las capas del modelo, incluso las que dependen de las representaciones vectoriales del texto (text embeddings).

Una de las principales diferencias es en la forma de las instrucciones, tienen la forma en inglés de "A [identificador] [clase]". El [identificador] es un token el cual debe tener poca información previa en el modelo de lenguaje y el modelo de difusión, en otras palabras, debe ser un token poco usado, con poca influencia semántica; usualmente en Stable Diffusion es usado el token "sks". Asimismo, [clase] es un descriptor aproximado del sujeto. A modo de ejemplo, si se busca aprender a generar un modelo de automóvil determinado, la [clase] podría ser "car". Un ejemplo de instrucción seria entonces: "A sks car".

Observar que esta técnica tiene cualidades. Primero, al usar un identificador con poco valor semántico, el proceso de inducir el nuevo concepto en dicho token es óptimo, ya de lo contrario, primero se debería desasociar el conocimiento previo para adquirir el nuevo. Segundo, al usar un descriptor de clase, el proceso de aprendizaje se apoya en el conocimiento actual del modelo para generar la clase del objeto, y por ende permite un entrenamiento más veloz y con mejores resultados.

Se observa que al entrenar meticulosamente los modelos de difusión de esta forma, los mismos tienen una gran capacidad en adquirir nuevos conceptos, pero se presentan varios desafíos. El primero de ellos se conoce como Deriva lingüística (Language Drift), que consiste en que un modelo de lenguaje al ser ajustado pierde progresivamente conocimiento sintáctico y semántico del lenguaje. Otro problema es que la diversidad en las imágenes generadas por el modelo difuso tiende a verse reducida; y, por último, el modelo tiende a ser muy propenso a sobre ajustarse.

Para mitigar esto, se propuso un regularizador llamado Preservación previa específica de la clase (Class-Specific Prior Preservation), el cual consiste en entrenar al modelo también utilizando las propias imágenes generadas por el modelo previo al ajuste. En particular, para dichas imágenes se utiliza como instrucción únicamente la descripción de la clase ("A [clase]", sin el identificador). El uso del este regularizador tiene como ventajas el aumento de la diversidad en las imágenes generadas por el modelo, y también reduce el riesgo de sobreajuste. En consecuencia, se observa una leve disminución en la fidelidad del concepto en la imagen.

La figura 2.4 ilustra el proceso de entrenamiento de un modelo difuso mediante Dreambooth, en el cual, a partir de unas pocas imágenes del nuevo concepto (en este caso, un perro en específico) el modelo se entrena usando instrucciones

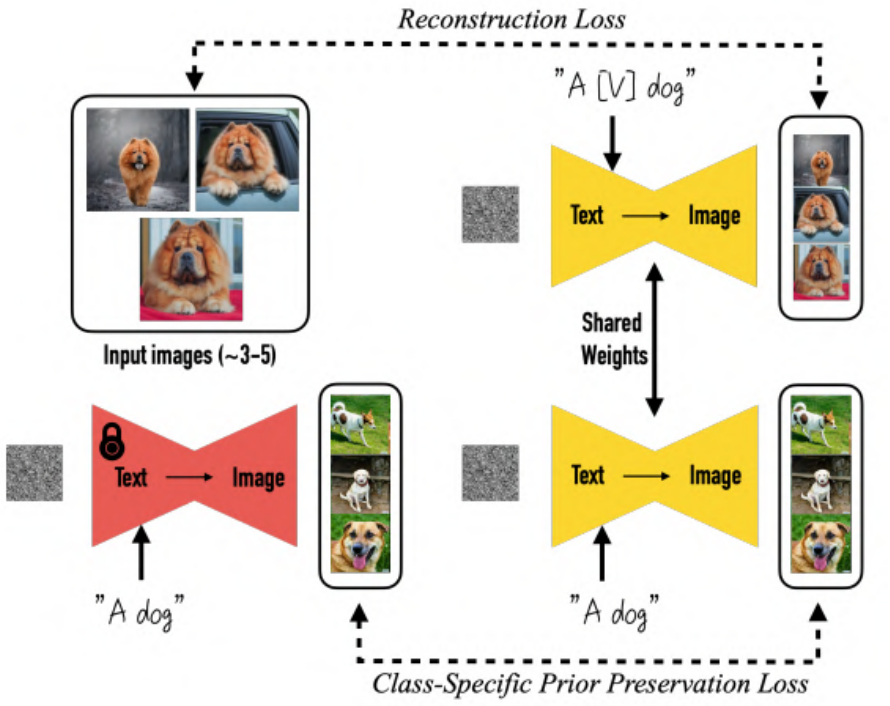


Figura 2.4: Proceso de entrenamiento mediante Dreambooth. En primer lugar, se cuentan con un conjunto de imágenes de entrada (Input images, esquina superior-izquierda), y se generan un conjunto de ejemplos para la instrucción "A [clase]", en este caso "A dog", con el modelo difuso sin modificar (modelo en rojo, esquina inferior-izquierda). Luego se procede con el entrenamiento, modificando los parámetros del modelo (modelo en amarillo), donde la función de costo es la suma de dos términos: Pérdida por Reconstrucción (Reconstruction Loss) y Pérdida de Preservación Previa Específica de la Clase (Class-Specific Prior Preservation Loss), donde la primera es la misma función de perdida utilizada en el pre entrenamiento, y la segunda es la propuesta por los autores de Dreambooth. La pérdida por reconstrucción es la diferencia con las imágenes de ejemplo, mientras que la Pérdida de Preservación Previa Específica de la Clase es la diferencia con las imágenes generadas con el propio modelo al principio. (Ruiz y cols., 2023)

personalizadas, como "A [V] dog", donde [V] es el token especial, como sks, con poco peso semántico. Al mismo tiempo, se utiliza el regularizador "Preservación previa específica de la clase", donde las imágenes de la clase general generadas con el modelo sin ninguna modificación, en este caso, obtenidas mediante la instrucción "A dog" (sin el token especial). Mediante el uso este enfoque, Dreambooth permite incorporar el concepto de un perro en específico, sin olvidar el conocimiento previo que se tenia sobre lo que es un perro.

2.3.5. Evaluación de modelos generativos de imágenes

En diferentes campos del aprendizaje automático es de especial relevancia el uso de métricas, no únicamente para comparar diferentes modelos, sino para determinar objetivos durante el entrenamiento o ajuste del modelo.

En el área de la generación de imágenes, sin embargo, no existe una métrica universalmente aceptada que mida de manera objetiva y definitiva la calidad o realismo de las imágenes generadas. Por ejemplo, nos puede interesar más que la morfología de los objetos generados se adhiera más a la realidad, y no así sus colores. En su lugar, se han propuesto y utilizado diversas métricas que varían según la tarea específica y el dominio en el que se aplica.

A continuación se detallan dos de las métricas más empleadas.

Puntaje Inception (IS, Inception Score)

El puntaje Inception (Salimans y cols., 2016) evalúa la calidad y diversidad de las imágenes generadas sin usar imágenes reales como referencia. Como su nombre indica, utiliza la red Inception (Szegedy y cols., 2015) preentrenada con el conjunto de datos ImageNet, compuesta por alrededor de 1000 clases diferentes. La idea es que si una imagen es clasificada con alta probabilidad en una sola clase por la red Inception, esto indica que la imagen tiene alta calidad. Por otro lado, si las imágenes son clasificadas de manera uniforme a través de las diferentes clases, esto sugiere que las imágenes son diversas.

Entonces, IS se calcula primero promediando la distancia de divergencia Kullback-Leiber (Divergencia KL) entre la distribución de probabilidad condicional $p(y|x)$ (la salida del clasificador Inception para una imagen generada) y la distribución marginal $p(y)$, obtenida al promediar $p(y|x)$ sobre todas las imágenes generadas. La divergencia Kullback-Leiber mide cuan diferente es una distribución de probabilidad respecto a otra.

Por último, se aplica la función exponencial al promedio de las divergencias KL para asegurar un valor positivo. La formula final es entonces:

$$\mathrm{IS} = \exp\left(E_x\left[D_{\mathrm{KL}}\left(p(y \mid x) \,\|\, p(y)\right)\right]\right)$$

Un IS alto indica que las imágenes generadas son claras y de alta calidad, con una distribución de probabilidad concentrada en una clase, y también diversas, con una distribución marginal amplia. En cambio, un IS bajo sugiere que las imágenes generadas no son de buena calidad ni muy diversas.

Cabe destacar que, al depender del clasificador previamente entrenado sobre ImageNet, el IS está condicionado por las clases presentes en este conjunto de datos. Por lo tanto, su capacidad para evaluar imágenes generadas de categorías que no se encuentren en ImageNet puede verse afectada, limitando la interpretación hecha para dominios específicos.

Distancia de Inception Fréchet (FID, Fréchet Inception Distance)

Similar a IS, la distancia de Inception Fréchet (FID) (Heusel, Ramsauer, Unterthiner, Nessler, y Hochreiter, 2017) busca evaluar la calidad y diversidad de un conjunto de imágenes generadas, con la diferencia de que FID utiliza un conjunto de imágenes reales de referencia para la comparación, permitiendo la evaluación en imágenes cuya clase no sea parte de las incluidas en ImageNet. En particular, FID busca comparar que tan parecidas son las imágenes del conjunto de referencia con las generadas.

A diferencia de IS, que compara las distribuciones de probabilidad de las imágenes a través de Inception, FID se basa en los mapas de características extraídos de una capa intermedia de la red Inception. Estos mapas contienen información sobre las características de las imágenes, permitiendo comparar imágenes que, aunque no sean similares en términos de píxeles, si lo pueden ser en lo que respecta a sus características.

Luego de obtener dichos mapas para todas las imágenes, se calcula la distancia entre ambos conjuntos utilizando la distancia de Fréchet, la cual utiliza la media y la matriz de covarianza de las características extraídas.

Formalmente:

$$FID = ||\mu_r - \mu_g||^2 + Tr(O_r + O_g - 2(O_r O_g)^{\frac{1}{2}})$$

Donde:

- μ_r, μ_g son las medias entre los mapas de características de las imágenes de referencia y las generadas respectivamente.
- O_r, O_g son las matrices de covarianza entre los mapas de características de las imágenes de referencia y las generadas respectivamente.

Otra diferencia con IS, en el cual donde mientras más elevado el valor, mejor desempeño, en FID mientras más cercano a cero sea el resultado, mejor desempeño este indica. La diferencia se debe a que IS busca que la distribución de probabilidades de la imagen y la marginal sean lo más diferentes posibles; pero FID busca que las características de las imágenes generadas sean lo más similares posible a las de las imágenes de referencia.

Capítulo 3

Revisión de antecedentes

Debido a la naturaleza de investigación de este trabajo, resultó fundamental realizar un relevamiento de antecedentes, abarcando proyectos y artículos académicos, además de diversas técnicas desarrolladas en contextos similares. En este capítulo se presenta un resumen de los diferentes elementos relevados, con el objetivo de dar un mejor contexto sobre el problema y destacar los aportes más relevantes de investigaciones anteriores. En particular, como se mencionó en la introducción, uno de nuestros objetivos consiste en la selección de las imágenes más significativas para el modelo de detección, por lo tanto, las secciones 3.1 y 3.2 presentan dos trabajos relacionados que hacen uso de aprendizaje activo. Además, en el trabajo presentado en la sección 3.2 se integra esta selección con aumentación de datos vía generación de imágenes. Por otro lado, en la sección 3.3 se resumen dos artículos cuya metodología incluye la utilización de modelos de difusión latentes para la aumentación de datos en conjuntos de imágenes de malezas. Por último, en la sección 3.4 se explican diferentes trabajos relacionados con el autoetiquetado de imágenes. En particular, se detallan tanto técnicas que hacen uso del estado interno del modelo generativo al momento de generar la imagen, como formas de obtener una etiqueta unicamente usando la información contenida en la imagen.

3.1. Aprendizaje activo para detección de objetos

El trabajo Plug and Play Active Learning for Object Detection (Yang, Huang, y Crowley, 2024) propone una aplicación directa de aprendizaje activo en la detección de objetos, mediante un método independiente de la arquitectura del detector utilizada. El algoritmo consta de dos etapas, la primera de estas, llamada “Muestreo de Incertidumbre Calibrada por Dificultad” (DCUS), tiene como objetivo calcular un coeficiente de dificultad por clase que considera tanto la dificultad de clasificación como la de localización. Este coeficiente se utiliza para ponderar las incertidumbres de las instancias, permitiendo seleccio-

nar un conjunto de imágenes candidatas con alta incertidumbre. En la segunda etapa, "Similitud de Emparejamiento Condicionado por Categoría" (CCMS), se mide la similitud entre imágenes con múltiples instancias mediante el emparejamiento de objetos similares entre las mismas. Esta medida se emplea en un algoritmo k-center-greedy para seleccionar un subconjunto diverso y representativo de imágenes para su anotación. Se destaca entre los antecedentes por ser una de las propuestas más recientes, publicada inicialmente en 2022, y actualizada en 2024, lo que la posiciona como un enfoque actual y relevante. Además, los propios autores resaltan su simplicidad de integración con distintos detectores, y su estatus como estado del arte. Se omite la explicación en detalle del trabajo ya que se explicará en la sección 4.2.

3.2. Aprendizaje activo y aumento de datos con imágenes sintéticas

En el trabajo Reducing Annotating Load: Active Learning with Synthetic Images in Surgical Instrument Segmentation (Peng y cols., 2024) se propone una metodología para la segmentación de instrumentos quirúrgicos que busca reducir la carga de anotación manual. El enfoque combina aprendizaje activo con generación de imágenes sintéticas, con el objetivo de entrenar modelos de segmentación capaces de detectar estos instrumentos en imágenes, utilizando una menor cantidades de muestras reales etiquetadas.

La principal motivación de este trabajo es similar a la de nuestro proyecto; al ser el entorno quirúrgico extremadamente complicado para etiquetar, es necesario contar con expertos en el área para esta tarea, sin embargo, para entrenar un modelo de segmentación, especialmente en un área tan delicada como la medicina, es esencial contar un conjunto extenso de datos, generando así un conflicto entre la necesidad de reducir la carga de anotación, y la necesidad de un gran volumen de datos.

Para afrontar este problema, los autores integran dos técnicas diferentes, el aprendizaje activo, cuyas bases fueron explicadas en la sección 2.2, junto con la generación de datos sintéticos mediante recorte y pegado (cut-and-paste).

Su propuesta funciona de manera iterativa. En cada ciclo, se seleccionan las imágenes más informativas del conjunto de datos no etiquetado, mediante el criterio de Aprendizaje Activo Bayesiano Por Desacuerdo (BALD, Bayesian Active Learning By Disagreement), sobre el cual se puede profundizar en el documento del estado del arte (Padrón, Estramil, y Núñez, 2025).

Estas imágenes son luego anotadas manualmente por un experto, y empleadas para generar imágenes sintéticas. Dichas imágenes se crean recortando instrumentos quirúrgicos (de las imágenes anotadas) y pegándolos sobre fondos seleccionados, aplicando técnicas de fusión de forma de suavizar la transición entre el instrumento y el fondo, mejorando así el realismo y la eficacia del método. En la figura 3.1 se ilustra este proceso.

El método fue evaluado en tres conjuntos de datos diferentes. Los expe-

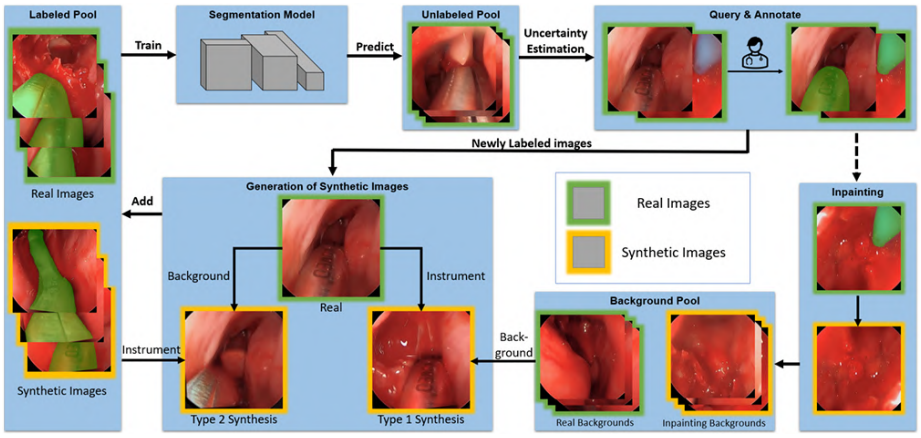


Figura 3.1: Flujo de trabajo de aprendizaje activo asistido por generación de imágenes sintéticas. El modelo de segmentación se entrena con un conjunto de imágenes reales etiquetadas. Se realiza inferencia sobre un conjunto no etiquetado, donde se estima la incertidumbre para seleccionar muestras a ser anotadas manualmente. Paralelamente, se generan imágenes sintéticas combinando fondos reales o generados mediante recorte y pegado con instrumentos. Las imágenes sintéticas se incorporan al conjunto etiquetado para reforzar el entrenamiento del modelo. (Peng y cols., 2024)

rimentos demostraron que el uso de imágenes sintéticas generadas mejora el rendimiento de los modelos de segmentación, especialmente en escenarios en los cuales se cuenta con cantidades pequeñas de imágenes anotadas (por ejemplo, el 10 %). Incluso al utilizar el 100 % del conjunto de datos, la utilización de imágenes sintéticas presentó mejoras, aunque siendo estás menos significativas respecto a los demás experimentos.

Este trabajo es particularmente relevante para nuestro proyecto, ya que plantea una estrategia similar a la nuestra en lo que a reducción de carga de anotación refiere, mediante aprendizaje activo y generación de imágenes sintéticas. Sin embargo, en nuestro caso, proponemos el uso de un criterio de selección diferente, introducido en la sección 3.1, además de implementar la generación de imágenes mediante modelos difusos.

3.3. Generación de imágenes sintéticas con modelos difusos latentes

En el primero de los trabajos presentados en esta sección, Weed image augmentation by ControlNet-added stable diffusion for multi-class weed detection (Deng y Lu, 2025) se propone un enfoque para el entrenamiento de modelos de detección de malezas basado en la generación de imágenes sintéticas mediante

ControlNet en Stable Diffusion. El objetivo principal es mitigar el problema de la escasez de datos etiquetados para entrenar dichos modelos.

La metodología planteada consiste en utilizar ControlNet, un modelo de control para redes generativas, que permite guiar la generación de imágenes en base a condiciones predefinidas. A diferencia de otros enfoques, ControlNet permite especificar diferentes características visuales de la imagen a generar mediante otras entradas más allá de texto. Por ejemplo, los autores plantean el uso de las cajas delimitadoras presentes en la imagen con la intención de guiar al modelo en la ubicación de las malezas a aprender.

La propuesta funciona entonces etiquetando un subconjunto de datos, utilizando estas imágenes etiquetadas junto con sus cajas delimitadoras para entrenar, mediante ControlNet, el modelo difuso, junto con el texto de entrada "muchos detalles, plantas en el campo".

Para evitar que el modelo mezcle los conceptos y morfologías de diferentes malezas, en cada imagen se tapan, mediante un recuadro gris, malezas que no sean pertenecientes a la misma clase.

Durante el proceso de generación, el modelo crea nuevas instancias de malezas, manteniendo las características morfológicas que pudo aprender mediante el conjunto de entrenamiento.

En cuanto al proceso de etiquetado de las imágenes generadas, la intención de los autores al utilizar ControlNet para guiar el entrenamiento con las cajas delimitadoras era poder generar imágenes sintéticas sabiendo de antemano la ubicación en la cual se encontraría cada instancia. Sin embargo, esto no dio resultado, por lo tanto se utilizó un enfoque semiautomático. Inicialmente, se aplica el modelo de detección entrenado previamente en imágenes reales para predecir las ubicaciones de las malezas en las imágenes sintéticas. Estas predicciones son luego revisadas y corregidas manualmente por una persona experta.

Por último, se evalúa el rendimiento del modelo utilizando YOLOv8l como detector, comparando el desempeño del modelo entrenado con los siguientes conjuntos de datos:

- Únicamente datos reales.
- Únicamente datos sintéticos.
- Un conjunto de datos compuesto por datos reales y datos sintéticos.

En la figura 3.2 se puede observar los resultados obtenidos en el trabajo. Los autores señalan como el agregado de imágenes generadas al conjunto de datos reales tiene un impacto limitado, aunque positivo, al evaluar el rendimiento del modelo de detección. Aun así, se puede observar como clases con la menor cantidad de cajas delimitadoras en el conjunto de datos reales utilizado por los autores, como pueden ser *eclipta* y *goosegrass*, vieron el mayor aumento en las métricas de la evaluación.

Otro trabajo que hace uso de modelos difusos para el aumento de datos es Analysis of Stable Diffusion-derived fake weeds performance for training Convolutional Neural Networks (Moreno, Gómez, Altares-López, Ribeiro, y Andújar,

Weed species	R only	1.5S only	R+0.5S	R+S	R+1.5S
Carpetweed	79.2% (89.2%)	28.4% (35.0%)	80.1% (89.5%)	79.9% (88.9%)	80.2% (88.8%)
Eclipta	**91.1% (95.3%)**	**30.9% (34.6%)**	**93.0% (96.0%)**	**93.0% (97.0%)**	**94.1% (97.7%)**
Goosegrass	**91.8% (95.3%)**	**46.6% (62.7%)**	**93.2% (96.7%)**	**93.3% (96.9%)**	**94.0% (97.9%)**
Lambsquarters	79.9% (93.3%)	42.5% (52.9%)	80.9% (93.9%)	80.3% (93.4%)	80.7% (93.6%)
MorningGlory	93.1% (97.5%)	50.0% (65.6%)	94.7% (98.0%)	93.7% (97.4%)	93.5% (97.1%)
PalmerAmaranth	89.8% (96.8%)	27.1% (30.1%)	91.6% (97.0%)	91.1% (96.7%)	90.7% (97.1%)
Purslane	76.0% (89.2%)	23.9% (28.7%)	77.4% (89.6%)	77.6% (90.3%)	77.0% (89.1%)
Ragweed	88.1% (94.0%)	62.1% (77.0%)	89.3% (94.2%)	88.6% (94.0%)	88.5% (94.7%)
SpottedSpurge	86.7% (94.5%)	55.2% (68.5%)	88.4% (94.9%)	89.1% (95.9%)	88.5% (95.2%)
Waterhemp	93.5% (96.6%)	58.3% (66.0%)	94.6% (97.2%)	94.7% (97.2%)	94.2% (97.1%)
Total	**86.9% (94.2%)**	**42.5% (52.1%)**	**88.3% (94.7%)**	**88.1% (94.8%)**	**88.1% (94.8%)**

Figura 3.2: Resultados en la detección de objetos. Los valores 0.5, 1 y 1.5 son las proporciones en relación al tamaño del conjunto de datos real. R es el tamaño del conjunto de datos reales y S el del conjunto de datos sintéticos. El valor antes de los paréntesis es map@50:95. El valor dentro del paréntesis es map@50 (explicados en 2.1.5). (Deng y Lu, 2025)

2023). En este trabajo, similar al anterior, se aborda el problema de la escasez de datos etiquetados para el entrenamiento de redes neuronales aplicadas a la detección de malezas. Para esto, los autores plantean el ajuste de un modelo difuso (Stable Difussion) para la generación de imágenes sintéticas de las especies a detectar, *solanum nigrum L*, *portulaca oleracea L* y *setaria verticillata L*. A partir de 30 imágenes reales de cada especie, se generan imágenes sintéticas con el fin de usarlas para el entrenamiento de los modelos YOLOv8l y RetinaNet.

La generación de imágenes sintéticas se realiza en dos etapas. En la primera, se obtienen las imágenes a utilizar. Debido a que los autores eligieron usar resoluciones de tamaño 512×512 y 768×768 para entrenar los modelos difusos, y que el conjunto de datos original contiene imágenes que superan el tamaño 5000×3000, se sigue un algoritmo para construir el conjunto de datos de entrenamiento que cumplan las condiciones requeridas. Cada caja delimitadora se comienza a agrandar hasta que alcanza alguna de las resoluciones necesarias, en caso de hacerlo sin que se solape con la caja delimitadora de otra instancia, se utiliza para el entrenamiento del modelo generativo, en caso contrario, es descartada. Mediante este algoritmo, además, se asegura que la maleza ocupe una proporción considerable de la imagen, ya que de lo contrario, el modelo podría aprender principalmente las características del fondo en lugar de la maleza. En la segunda etapa, se generan y autoetiquetan imágenes sintéticas para el entrenamiento del modelo de detección. Este autoetiquetado funciona utilizando ecuaciones basadas en los canales de colores de las imágenes. En particular, estas ecuaciones hacen uso del Índice Exceso de Verde menos Exceso de Rojo (ExGR) (para más detalles, consultar el trabajo original).

En la figura 3.3 se muestran los resultados obtenidos en el trabajo. Se puede

Model	Experiment	Real Bboxes	Artificial Bboxes	AP@0.5			mAP@0.5
				SOLNI	POROL	SETVE	
Yolov8l	1-Y	0	228	0.928	0.849	0.909	0.895
	2-Y	0	450	0.958	0.912	0.928	0.933
	3-Y	0	900	0.933	0.829	0.885	0.882
	4-Y	0	1,800	0.964	0.87	0.893	0.909
	5-Y	228	0	0.976	0.924	0.987	0.962
	6-Y	228	228	0.991	0.991	0.994	0.992
	7-Y	228	450	0.985	0.983	0.992	0.987
	8-Y	228	900	0.995	0.984	0.993	0.99
	9-Y	228	1,800	0,99	0,972	0,986	0,983
RetinaNet	1-R	0	228	0.874	0.816	0.877	0.856
	2-R	0	450	0.915	0.849	0.916	0.893
	3-R	0	900	0.916	0.877	0.926	0.907
	4-R	0	1,800	0.923	0.872	0.887	0.894
	5-R	228	0	0.975	0.945	0.961	0.96
	6-R	228	228	0.998	0.977	0.985	0.987
	7-R	228	450	0.996	0.982	0.974	0.984
	8-R	228	900	0.999	0.984	0.972	0.985
	9-R	228	1,800	0.998	0.983	0.987	0.989

Figura 3.3: Resultados al calcular mAP@50 al entrenar los modelos YOLOv8l y RetinaNet. Se muestran resultados al utilizar solo imágenes reales, solo imágenes sintéticas, y al combinarlas en diferentes proporciones. (Moreno y cols., 2023)

observar como, tanto para el modelo YOLOv8l como RetinaNet, los mejores resultados son obtenidos mediante una combinación de datos reales en conjunto con datos sintéticos, alcanzando una map@50 de 0.992, mientras que los peores son obtenidos al utilizar únicamente datos sintéticos, teniendo su mínimo en 0.882.

Figura 3.4: Ejemplos de imágenes generadas mediante Stable Difussion para las tres clases de malezas. (Moreno y cols., 2023)

Se destaca que, si bien los resultados parecen alentadores, como se puede observar en las figuras 3.3 y 3.4, al contrario del presente proyecto, el conjunto de datos utilizado en este trabajo contiene unicamente tres clases diferentes de malezas (al contrario de doce), con fondos conteniendo casi en su totalidad tierra, el cual provoca un contraste notorio que beneficia el uso de algoritmos

sencillos para el autoetiquetado. En la sección 3.4.2 se introducirá un trabajo que aplica un algoritmo más adecuado para el conjunto de datos utilizado en este proyecto para la experimentación, y en 4.4.1 se presentarán ligeros cambios con la idea de mejorarlo.

3.4. Autoetiquetado de imágenes

Con el fin de utilizar las imágenes generadas en el entrenamiento de los detectores de objeto, es tanto necesaria la imagen, como su etiqueta adjunta, la cual debe contener tanto la ubicación y la especie de cada maleza que se encuentre contenida en la imagen. Existen varias formas de poder abordar el anterior problema, pero las mismas pueden ser categorizadas en dos grupos: en aquellos que solo utilizan la imagen para poder generar las detecciones; y aquellos que, además de utilizar la imagen, también utilizan el estado interno del propio modelo generativo.

3.4.1. Autoetiquetado basado en modelos generativos

Generalmente, la tarea de detección de objetos es realizada disponiendo únicamente de la imagen, y como fue desarrollada en la sección 2.1, se cuenta con un amplio repertorio de opciones para dicha tarea.

En el caso particular de las imágenes generadas, se tiene un elemento adicional de suma importancia, en el propio modelo generativo. Por lo tanto, se presentan a continuación un par de trabajos que buscan aprovechar el conocimiento del modelo generativo, de forma que se pueda extraer la posición donde las malezas se encuentran dentro de las imágenes generadas.

El primero de estos trabajos es HandsOff: Labeled Dataset Generation With No Additional Human Annotations (Xu, Vasileva, Dave, y Seshadri, 2023), el cual comparte una motivación muy similar a la del actual proyecto. Se buscan con pocas imágenes etiquetadas ~ 50, ser capaces de poder generar un número indefinido de instancias artificiales autoetiquetadas, para el entrenamiento de futuros detectores. Para ello, primero es realizado un ajuste sobre un modelo generativo preentrenado con arquitectura StyleGAN2 (Karras y cols., 2020).

Una vez que se es capaz de generar imágenes, se entrena una serie de redes neuronales de tipo perceptron multicapa (MLP), la cual tiene como objetivo etiquetar la imagen generada. El proceso es ilustrado en la figura 3.5. Observar que la entrada es la concatenación de diferentes capas intermedias del modelo generativo, y la salida la etiqueta de la imagen. Esta última es ajustable según el tipo de etiquetado que se busca obtener; variando, por ejemplo, si se trata de cajas delimitadoras, puntos claves, o una máscara semántica. El resultado es obtenido mediante la votación de los diferentes MLP previamente entrenados.

Observar que, debido al carácter supervisado en el entrenamiento del etiquetador, es necesario contar tanto con imágenes etiquetadas, como con la capacidad de reproducir imágenes del conjunto de entrenamiento por el generador. Esto último es necesario, ya que el etiquetador que se busca entrenar tiene como

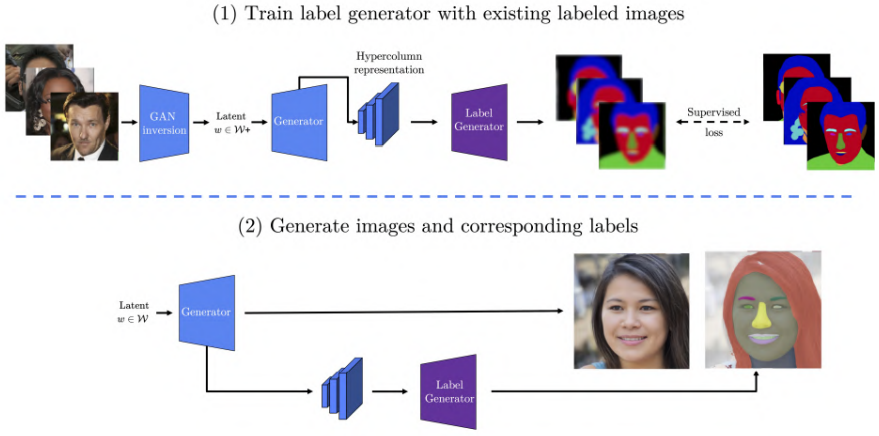


Figura 3.5: Entrenamiento y generación de imágenes artificiales autoetiquetadas en el framework Hands-off. Primero es descrito el entrenamiento, donde se busca, para cada imagen del conjunto de entrenamiento, los vectores latentes más próximos utilizando Inversión de GAN, luego se calculan los vectores de hipercolumnas utilizando los estados internos del modelo generativo. Este último es la entrada del etiquetador, generando así la correspondiente etiqueta, lo que permite su comparación con la etiqueta esperada, y el posterior entrenamiento del etiquetador. En lo que respecta a la generación de imágenes consiste en primero generar la imagen, y en simultaneo el vector de hipercolumnas, obteniendo de esa forma la etiqueta para la imagen generada. (Xu y cols., 2023)

entrada una concatenación de capas intermedias del generador. Para lograr que el generador cree una imagen similar a la del conjunto de entrenamiento, se busca un vector latente en el espacio latente, el cual, al ser usado como entrada en el modelo generativo, este último genere una imagen lo más similar posible a la buscada. Esta familia de técnicas son conocidas bajo el nombre de Inversión de GAN.

El trabajo muestra que utilizando grandes conjuntos de imágenes no etiquetadas para ajustar el modelo generativo (en todos los casos más de 5000 imágenes), y entrenando el etiquetador con solamente 50 imágenes etiquetadas, se logran generar imágenes sintéticas autoetiquetadas las cuales logran superar ampliamente en rendimiento a los modelos de detección si solamente hubieran sido entrenados con dichas 50 imágenes.

Notar que este trabajo es significativamente relevante, ya que muestra que es posible reducir sustancialmente el número de instancias etiquetadas, aprovechando lo aprendido por el modelo generativo utilizando un gran número de instancias no etiquetadas. Como ya fue mencionado anteriormente, y profundizado en la sección 4.3, en este trabajo se utilizan modelos de difusión, no siendo directamente aplicable lo propuesto en HandsOff, es más, actualmente no se tiene conocimiento de ningún método equivalente que utilice modelos de difusión

con el mismo propósito.

Por otro lado, y al contrario de HandsOff, en el artículo DiffuGen: Adaptable Approach for Generating Labeled Image Datasets using Stable Diffusion Models (Shenoda y Kim, 2023) se presenta una metodología que hace uso de modelos difusos, en lugar de GANs, utilizándolos para generar conjuntos de datos etiquetados de manera automática. El objetivo principal es abordar la falta de datos etiquetados, permitiendo la generación de imágenes sintéticas en conjunto con sus correspondientes etiquetas mediante técnicas de etiquetado no supervisado. La propuesta para la generación consiste en el uso de plantillas de texto para definir atributos específicos en las imágenes generadas, como nombres de clases, condiciones climáticas o puntos de vista, permitiendo así crear variaciones de las mismas clases, incrementando la diversidad del conjunto de datos.

En cuanto al etiquetado, en el artículo se propone el etiquetado no supervisado, que utiliza mapas de atención cruzada, generados durante el proceso de difusión para identificar las regiones de interés de cada token en el texto de entrada, y de esta forma crear máscaras semánticas y cajas delimitadoras que indiquen la posición correspondiente del token en la imagen.

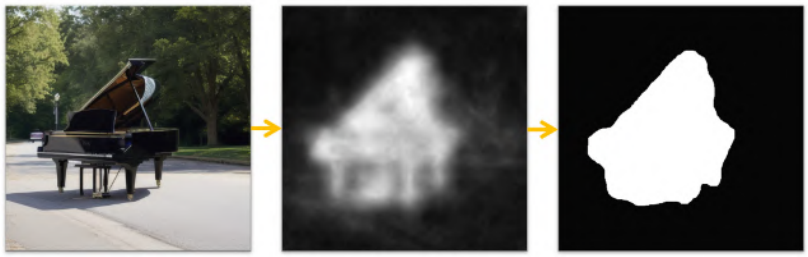


Figura 3.6: Visualización de la generación de una máscara semántica mediante el uso de los mapas de atención cruzada. (Shenoda y Kim, 2023)

En la figura 3.6 se muestra una representación de las diferentes etapas del autoetiquetado propuesto. En el artículo se menciona que, si bien en la mayoría de casos las imágenes fueron etiquetadas correctamente, especialmente para imágenes con pocos objetos, en aquellas que presenten una cantidad elevada de objetos de cada clase, se hallan inconsistencias en las detecciones, debido a las limitaciones inherentes al uso de máscaras de atención cruzada, ya que no son capaces de separar entre diferentes instancias de una misma clase.

Observar que, si bien ambos métodos (DiffuGen y HandsOff) parecen similares, los mismos no son comparables, ya que DiffuGen no es una adaptación de HandsOff para modelos difusos. Esto es debido a que DiffuGen es un método del tipo zero-shot, eso es, que no requiere de un entrenamiento específico para generar el etiquetado, a diferencia de HandsOff que es de tipo few-shot, el cual requiere de pocas instancias para ser adaptado. HandsOff por lo tanto presenta una mayor flexibilidad y robustez al utilizar una red externa para el proceso de autoetiquetado.

3.4.2. Autoetiquetado basado en canales de colores

El proyecto Detección y clasificación de cultivos y malezas a partir de imágenes RGB (Pérez, Rodríguez, y Rován, 2023), hecho por estudiantes de la Facultad de Ingeniería UdelaR, tiene como objetivo detectar y clasificar instancias de plantas en imágenes a color (RGB), diferenciando entre las clases maleza o cultivo. Para ello, se aprovecha la información de los canales de color para localizar las regiones relevantes (que podrían tener una instancia) y clasificarlas.

La metodología emplea una combinación de técnicas deterministas, como las presentadas en la sección 2.1.2, en conjunto con métodos más avanzados como K-Means y k-NN. El proceso se divide en dos etapas principales:

Segmentación

En esta primera etapa se busca identificar regiones de interés en la imagen, es decir, zonas donde probablemente haya una planta. Para ello, se aplica una técnica de agrupamiento por color (Color Clustering), que consiste en agrupar los píxeles en dos grupos (clusters) utilizando la información de los canales RGB. Cada píxel se asigna entonces a un grupo según su proximidad a los centroides, y se clasifica como área de interés o fondo.

Este enfoque puede relacionarse con el descrito en la sección 2.1.2, donde primero se identifican posibles zonas con objetos para luego generar recortes, evitando así evaluar todas las posiciones posibles.

Clasificación

Una vez extraídos los recortes de las regiones de interés, se analizan utilizando técnicas de extracción de descriptores como SIFT o FFT, obteniendo descriptores como Momentos Hu o Blobs. Posteriormente, se entrena un modelo de clasificación k-NN que utiliza estos descriptores para clasificar las regiones de interés entre malezas y cultivos.

De este proyecto, nos interesa particularmente la etapa de segmentación, ya que permite autoetiquetar las imágenes generadas. Si bien en el trabajo descrito en la sección 3.3 también se utiliza segmentación por color para generar las etiquetas de las imágenes sintéticas, se considera que este enfoque es más completo y funcional. Esta metodología será abordada en detalle en la sección 4.4.1.

Capítulo 4

Parte Central

En este capítulo se aborda en detalle el análisis e implementación de los diferentes componentes construidos, explicando su utilidad y como son integrados en un único sistema, usando como base las técnicas relevadas anteriormente.

Según lo explicado en la introducción, el problema a resolver consiste en, dado un conjunto de imágenes, devolver un conjunto compuesto por imágenes reales y falsas con sus respectivas etiquetas, requiriendo que una persona solo etiquete una cantidad determinada de las primeras. Esto implica un esfuerzo en lograr obtener el mayor rendimiento posible dada esta limitación, por lo que es necesario hacer un mejor uso posible de los datos de entrada.

Para dar un contexto de los distintos módulos que componen la solución, en la sección 4.1 se describe en alto nivel el flujo general de procesamiento que seguirán las imágenes desde su entrada, hasta su respectivo resultado. Los diferentes componentes que conforman este flujo de datos buscan activamente mejorar el resultado final, respetando el limite de esfuerzo humano definido anteriormente.

El primero de estos componentes, explicado en 4.2, se encarga de la selección de imágenes, la cual comúnmente, en los casos que no se etiquete todo el conjunto de datos, se hace de manera aleatoria. En el caso de este proyecto, se propone la implementación de PPAL, explicado en 3.1, para lograr que el esfuerzo humano se centre principalmente en el etiquetado de aquellas imágenes más significativas para el modelo de detección.

Una vez seleccionadas las imágenes, estas son utilizadas (además de como resultado final) para entrenar un modelo generativo, a fin de aumentar el conjunto de datos disponible para su uso. Este proceso se desarrolla en la sección 3.3, donde se detallan los métodos utilizados, el preprocesamiento necesario y las desventajas del uso de otos modelos.

Con el mismo propósito, el siguiente componente, explicado en la sección 4.4, hace uso de las imágenes sintéticas para generar sus etiquetas correspondientes de manera automática, utilizando los canales de colores de las imágenes, modelos de detección o segmentación.

Con el objetivo de dar una solución que sea utilizable no solo desde el punto

de vista técnico, sino también práctico, se integró el proyecto con la herramienta CVAT, herramienta la cual es utilizada para el etiquetado de imágenes para visión por computadora. Esta integración es explicada en 4.5.

Finalmente, en la sección 4.6 se presenta la arquitectura del sistema desarrollado, describiendo como se integran y comunican los diferentes componentes, para lograr una solución escalable y eficiente.

4.1. Flujo de datos

Como primer paso, en esta sección se proponen y analizan diferentes flujos de datos posibles para el problema antes especificado. Se busca definir diferentes flujos que describan como, a partir de las imágenes de entrada, se pueda obtener otro conjunto de imágenes etiquetadas de salida. En particular, los flujos propuestos deben cumplir con las tres siguientes restricciones:

- La entrada consiste en un conjunto de imágenes, las cuales pueden contener una o varias malezas de diferentes especies. No se cuenta con ningún tipo de información sobre el contenido de la imagen.
- Se cuenta con un esfuerzo humano disponible, el cual será medido en el número de imágenes que se puede solicitar (a un persona) que se etiquete. En otras palabras, se encuentra limitado y especificado el número máximo de imágenes que pueden ser etiquetadas por humanos.
- La salida deberá ser un conjunto de imágenes etiquetadas, las cuales puedan ser posteriormente utilizadas en el entrenamiento de algoritmos de detección de objetos.

Los flujos de datos están compuestos por diferentes fases, donde en cada una de ellas se realiza un cierto proceso basándose en la información de su fase previa o en colaboración con un agente externo (como personas etiquetando imágenes), existiendo diferentes métodos para cada una de las fases. Los flujos de datos aquí presentados son basados en los antecedentes relevados en el capítulo 3.

El flujo de datos más básico es ilustrado en la figura 4.1, el cual consiste exclusivamente en la selección de un subconjunto de las imágenes de entrada, y su posterior etiquetado. Notar que la forma más simple de poder realizar este proceso, sería mediante la selección aleatoria de las imágenes. Dicho en otras palabras, dado un conjunto de imágenes, escoger aleatoriamente un subconjunto cuyo tamaño corresponda al máximo esfuerzo humano disponible.

Existen mecanismos más sofisticados que permiten realizar dicha selección, obteniendo conjuntos más representativos, especialmente si el esfuerzo humano disponible es escaso. En el trabajo se exploran tanto la selección de imágenes de forma aleatoria, como también mediante técnicas de aprendizaje activo, ambas tratadas en la sección 4.2.

Cuando se busca integrar el aumento de datos vía generación de imágenes artificiales, existen diferentes flujos de datos posibles, aunque principalmente

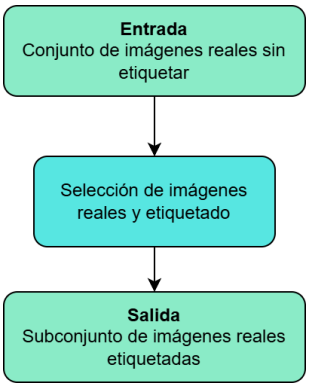


Figura 4.1: Diagrama de flujo de datos utilizando únicamente la selección de imágenes.

pueden ser categorizados en dos grupos: los flujos donde el autoetiquetado de las imágenes generadas es realizado mediante el propio modelo generativo; y los flujos donde el etiquetado de las imágenes generativas es hecho únicamente con la imagen generada.

Un ejemplo de un flujo de datos que corresponde a la primera de estas categorías es el trabajo HandsOff, presentado en la sección 3.4, e ilustrado en la figura 4.2. En este flujo, el modelo generativo es ajustado con todas las imágenes disponibles, aunque las mismas no hayan sido etiquetadas. Eso implica que si el ajuste es correctamente realizado, las imágenes generadas serán parecidas a las del conjunto de entrenamiento, pero para poder ser utilizadas en algoritmos de detección es necesaria su etiqueta, entonces, ¿cómo localizar en la imagen generada las malezas buscadas y distinguir su clase? Se utiliza un detector cuya entrada es una concatenación de ciertas capas intermedias del modelo generativo, de forma tal que el propio modelo generativo no solamente crea la imagen, sino que también distingue su localización y especie de cada maleza que contiene.

El detector es entrenado utilizando un bajo número de imágenes etiquetadas (~ 50), mostrando elevadas capacidades de detección en las imágenes generadas, pero también en las imágenes reales. Observar que, sí el modelo generativo es ajustado con imágenes reales modificadas, es posible que el detector ya no sea útil en las imágenes reales.

Lo anterior permite que todas las imágenes generadas y también las imágenes de entrada al flujo sean etiquetadas y devueltas como salida del mismo. La utilización de todo el conjunto de imágenes de entrada, junto con el bajo número de imágenes etiquetadas necesarias, y al hecho de que el detector podría ser utilizado tanto con imágenes generadas como con reales, hicieron que este flujo sea de gran interés para este proyecto, de forma tal que durante el primer transcurso del mismo se aspiraba su utilización.

De todas formas, como fue explicado en la sección 3.4, HandsOff fue desarrollado para modelos generativos basados en GANs, y como se justifica en

la sección 4.3, este trabajo es realizado utilizando modelos de difusión. Dadas las diferencias entre ambas arquitecturas, en especial la naturaleza iterativa de los modelos difusos, la aplicación del anterior método no es directa, y por ende estudiar su viabilidad y posible implementación supera las limitaciones de este proyecto de grado.

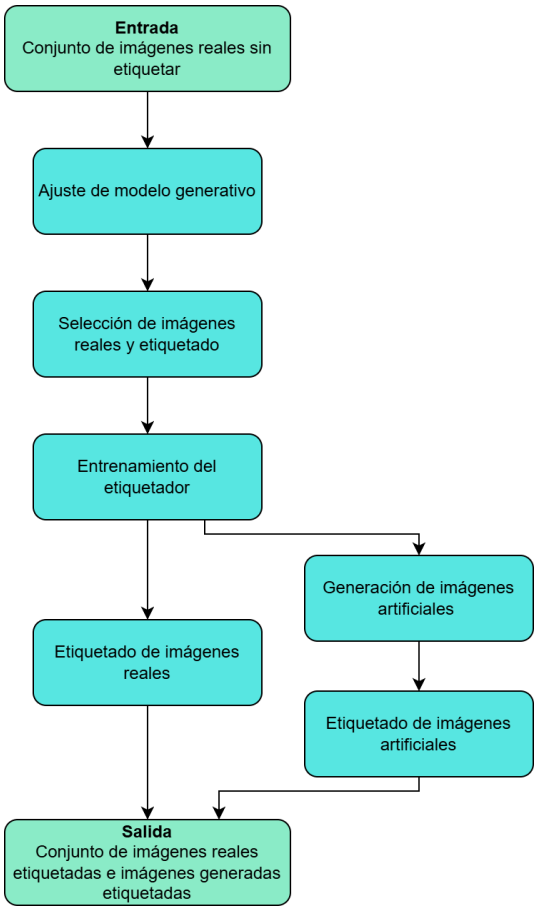


Figura 4.2: Diagrama de flujo de datos derivado del trabajo HandsOff (Xu y cols., 2023), el cual utiliza un etiquetador cuya entrada son estados parciales del modelo generativo de imágenes.

Dado que durante el proceso de relevamiento de antecedentes no se encontró ningún trabajo el cual, a partir de una imagen generada por modelos de difusión, logre detectar los objetos junto con su respectiva clase, se buscan flujos de datos en los cuales se tenga control sobre cuál especie de maleza se está generando, y la imagen sea lo suficientemente simple para inferir la localización de la maleza en la imagen.

En línea con los trabajos descritos en la sección 3.3, el flujo de datos realizado

en este proyecto es ilustrado en la figura 4.3 junto con el nombre de cada una de las componentes implementadas para cada fase del flujo. En particular, en este flujo, se propone primero seleccionar un conjunto de imágenes y etiquetarlas manualmente. Con el fin de poder controlar el tipo de maleza que está siendo generada en un momento dado, durante el proceso de ajuste solo se utilizan imágenes con una única maleza, especificando para dicha imagen cuál es la especie de maleza en cuestión. En efecto, como se explicará en la sección 4.3, se obtuvieron mejores resultados al ajustar diferentes modelos generativos por maleza.

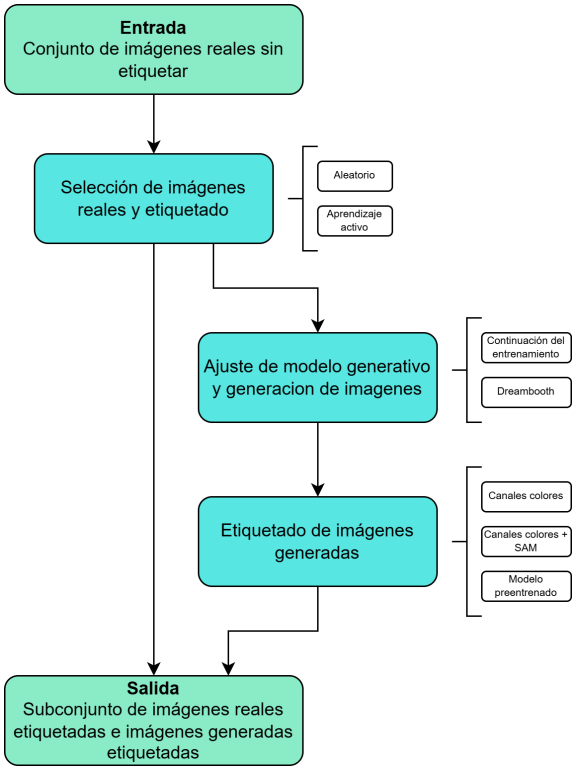


Figura 4.3: Diagrama de flujo de datos utilizando selección y generación de imágenes artificiales autoetiquetadas, creadas utilizando modelos generativos ajustados con las imágenes seleccionadas. En cada etapa son especificados los diferentes componentes propuestos.

Observar que, debido a que durante el proceso de ajuste son utilizadas imágenes de una sola maleza con su correspondiente especie, es necesario que previamente las imágenes hayan sido etiquetadas. Entonces, una vez que las imágenes de entrada son etiquetadas, las malezas dentro de la misma son recortadas y clasificadas, y luego utilizadas en el ajuste de los modelos generativos.

Una vez que los modelos son ajustados, ya es posible generar imágenes arti-

ficiales, las cuales deben ser posteriormente etiquetadas. Notar que la especie de la maleza es conocida, ya que es especificada a la hora de generar la imagen, solo resta la localización de la maleza en la imagen, dicho proceso es desarrollado en la sección 4.4. La salida del flujo serán las imágenes seleccionadas y etiquetadas, junto con las imágenes generadas y autoetiquetadas.

Observar que el anterior flujo de datos propuesto tiene un claro inconveniente, si el número de imágenes que se pueden etiquetar manualmente es inferior al número de imágenes de la entrada, entonces existirán imágenes de entrada que no contaran con etiquetado, y por ende, no serán parte de la salida del flujo. Existen mecanismos para poder afrontar la anterior situación, creando pseudo-etiquetas para dichas imágenes utilizando métodos de detección de objetos semisupervisados. Los anteriores métodos fueron relevados, y propuestos en el anexo B, aunque por cuestiones estrictamente temporales, los mismos no son parte de la experimentación realizada.

4.2. Selección de imágenes

Esta sección se centrará en el componente de selección. Como se muestra en la figura 4.3 tiene como entrada las imágenes reales sin etiquetar, por lo que su objetivo será seleccionar un subconjunto de imágenes que serán las que luego pasarán a la siguiente fase de etiquetado.

Como se comenta en la sección 2.2, existen diversas formas de seleccionar estas imágenes, donde la más básica de todas es realizar una selección de manera aleatoria. La selección aleatoria define la línea base en este proyecto, ya que es la forma más simple de realizar la selección.

También en la sección 2.2 se menciona que existen formas más inteligentes de realizar la selección, trabajando en esto bajo el nombre de aprendizaje activo, pero lo cierto es que el aprendizaje activo es un concepto muy amplio que solamente refiere al proceso y no a la forma, existiendo trabajos en diversas áreas como el procesamiento del lenguaje natural, extracción de información, problema de preguntas/respuestas, y por supuesto el área de visión por computadora con trabajos relacionados con la detección de objetos y la segmentación semántica.

En línea con el objetivo de este proyecto, en particular en lo que refiere a la selección de imágenes y aprendizaje activo, se busca implementar una técnica que muestre una ganancia de desempeño al elegir B imágenes de forma inteligente respecto a elegir B imágenes de forma aleatoria. Cabe destacar que según los objetivos del proyecto se limita a la implementación de un método existente.

Una de las cosas que se tuvieron en cuenta a la hora de relevar el método a implementar era que primero debía ser un trabajo de detección de objetos, por otro lado, idealmente el método no debía ser dependiente de la arquitectura de detección de objetos (que deja como ventaja poder elegir cualquier arquitectura), ya que el proyecto más grande sobre el cual se encuentra este proyecto no tiene aún una arquitectura definida, por lo que la implementación particular podría llevar a la incompatibilidad en el proyecto más grande.

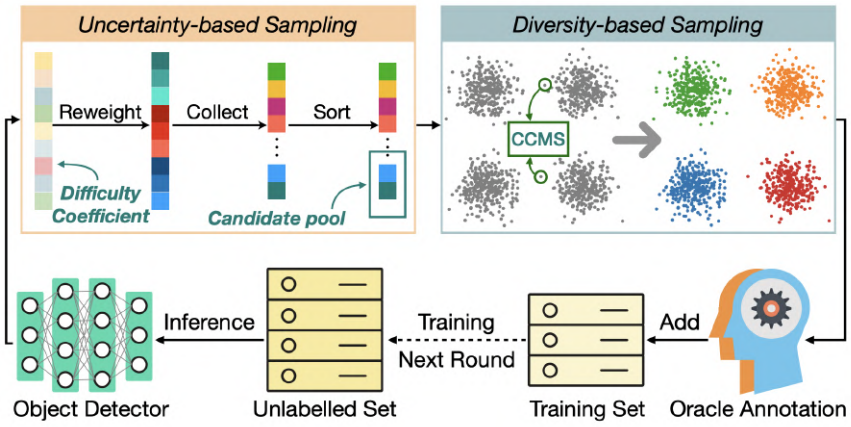


Figura 4.4: Se detalla el proceso realizado por PPAL. En primer lugar se realiza la etapa DCUS (Uncertainty-based Sampling), luego se pasa a la etapa de diversidad (Diversity-based Sampling) que retorna las imágenes que el humano debe etiquetar, esas imágenes son usadas luego para entrenar el modelo en la siguiente iteración, agregándolas al conjunto de entrenamiento y quitándolas del conjunto sin etiquetar, luego el proceso se repite. (Yang y cols., 2024)

Bajo este panorama se relevó el trabajo Plug And Play Active Learning (PPAL) (Yang y cols., 2024) como se menciona en 2.2. El trabajo PPAL tiene como objetivo precisamente romper la barrera de que los métodos existentes hacen uso de la arquitectura interna de los detectores, haciendo inutilizable el método para otros, además de la complejidad existente en integrar dichos cambios de arquitectura en proyectos existentes, según se comenta en el artículo.

Otra de las razones que inclinó la elección sobre este trabajo fue que se afirma ser el estado del arte, mostrando un gran desempeño por encima de la selección aleatoria para el conjunto de datos de COCO (Lin y cols., 2014), que es un conjunto de datos de gran escala ampliamente utilizado en visión por computadora, teniendo más de 200.000 imágenes y 80 clases diferentes, por lo que PPAL se integra bien con la filosofía de este proyecto.

4.2.1. PPAL

Esta sección será dedicada a explicar como funciona el algoritmo propuesto en el método PPAL (Yang y cols., 2024), se realizará un extracto de las partes esenciales con el objetivo de comprender lo implementado en este proyecto de grado. Este algoritmo consta de dos etapas que combinan el muestreo basado en la incertidumbre y en la diversidad.

Para este algoritmo se encuentra disponible una implementación oficial, la misma está realizada haciendo uso de la librería mmdetection (Chen y cols., 2019), esta implementación está orientada a experimentación por lo que se en-

cuentra sobrecargada de dependencias y componentes que no son competentes en este proyecto y por ende contraproducentes para la integración con la arquitectura desarrollada, considerando que la librería de detección utilizada en este proyecto es principalmente Torchvision. Dadas estas razones, se optó por una implementación propia completa, tomando la implementación oficial como insumo.

El algoritmo consta de dos etapas, la primera etapa "Muestreo de incertidumbre calibrada por dificultad" (DCUS) tiene como objetivo calcular un coeficiente de dificultad por clase que considera tanto la dificultad de clasificación como de localización. Este coeficiente se utiliza para ponderar las incertidumbres de las instancias, permitiendo seleccionar un conjunto de imágenes candidatas con alta incertidumbre.

La segunda etapa, "Similitud de emparejamiento condicionado por categoría" (CCMS), mide la similitud entre imágenes con múltiples instancias mediante el emparejamiento de objetos similares entre imágenes. Esta medida se emplea en un algoritmo "k-center-greedy" para seleccionar un subconjunto diverso y representativo de imágenes para su anotación.

El procedimiento cumple las condiciones de aprendizaje activo, ya que tiene una naturaleza iterativa, y se cuenta con un presupuesto B de imágenes a etiquetar. De esas B imágenes se toma una cantidad de imágenes n para realizar un primer entrenamiento, siendo esta cantidad un hiperparámetro del modelo.

Si se realizan un total de r iteraciones de aprendizaje activo y se eligen b imágenes del conjunto sin etiquetar en cada iteración, entonces $B = br + n$. En cada iteración i el modelo se vuelve a entrenar desde el comienzo con las $b(i-1)+n$ imágenes etiquetadas hasta el momento.

El proceso se describe en la figura 4.4, se puede apreciar que para obtener las b imágenes primero se pasa por la primera etapa DCUS donde a partir de la incertidumbre se obtiene un conjunto reducido llamado conjunto candidato (candidate pool) que es la entrada para la segunda etapa CCMS de diversidad que hará uso del algoritmo "k-center-greedy" para obtener las b imágenes. A continuación se explica individualmente cada etapa.

4.2.2. Muestreo de incertidumbre calibrada por dificultad (DCUS)

DCUS provee una forma de medir la incertidumbre tanto en la clasificación como en la localización, intuitivamente el método intenta aumentar la importancia de las clases en las que el modelo no tiene buen desempeño.

Específicamente, se suponen C clases en el conjunto de datos. Al principio del algoritmo se define las dificultades por clase $D^r = \{d_i\}_{i\in[C]}$ y se inicializan todas en 1. Posteriormente, se calcula la dificultad de cada caja delimitadora predicha durante el entrenamiento como:

$$q(b|\hat{b}) = 1 - P(b|\hat{b})^{\xi} \cdot \mathrm{IoU}(b,\hat{b})^{1-\xi} \tag{4.1}$$

donde b es la caja predicha, $\hat{b}$ es la caja real (ground-truth) a la que se

asigna b. Notar que en el entrenamiento se cuenta con las cajas delimitadoras verdaderas, a estas cajas se le llama caja real. $P(b|\hat{b})$ es la probabilidad de clasificación respecto a la clase de $\hat{b}$, $\text{IoU}(b, \hat{b})$ es la Intersección sobre Unión entre b y $\hat{b}$, y $0 \leq \xi \leq 1$ es un hiperparámetro. La dificultad de detección definida en la ecuación 4.1 tiene en cuenta tanto la clasificación como la localización; por lo tanto, ambas contribuyen al muestreo de incertidumbre. Luego, las dificultades registradas por clase se actualizan mediante el promedio de la dificultad de cada objeto utilizando un promedio móvil exponencial (EMA) durante el entrenamiento:

$$d_i^k \leftarrow m_i^{k-1} d_i^{k-1} + (1 - m_i^{k-1}) \cdot \frac{1}{N_i^k} \sum_{j=1}^{N_i^k} q_j \tag{4.2}$$

$$m_i^k \leftarrow \begin{cases} m^0 & \text{if } N_i^k > 0 \\ m^0 \cdot m_i^{k-1} & \text{if } N_i^k = 0 \end{cases} \tag{4.3}$$

donde k es la iteración de entrenamiento, d_i^k es la dificultad actualizada para la categoría i, N_i^k es el número de objetos predichos de la clase i en el lote de entrenamiento, q_j es la dificultad de entrenamiento del objeto j-ésimo, m_i^* es el momentum EMA, y m^0 es el momentum inicial para todas las categorías. Como se muestra en la ecuación (4.3), si un lote de entrenamiento no contiene objetos de la categoría i, disminuye su momentum por clase multiplicándolo por m^0, lo que asegura un ritmo de actualización similar de las dificultades para las diferentes clases.

Después de entrenar el detector sobre el conjunto de datos etiquetado, se calcula el coeficiente de dificultad por categoría $W^r = \{w_i\}_{i \in [C]}$ para la ronda r de aprendizaje activo como:

$$w_i = 1 + \alpha\beta \cdot \log\left(1 + \gamma \cdot d_i\right) \quad \text{donde} \quad \gamma = e^{1/\alpha} - 1 \tag{4.4}$$

donde α controla qué tan rápido cambia el coeficiente de dificultad con respecto a la dificultad por clase, y β controla el límite superior del coeficiente de dificultad. Finalmente, se calcula $U(I)$ la incertidumbre de cada imagen no etiquetada sumando la entropía de cada objeto detectado, ponderada por el coeficiente de dificultad correspondiente:

$$U(I) = \sum_{i=1}^{M_I} w_{c(i)} \cdot \sum_{j=1}^{C'} -p_{ij} \cdot \log(p_{ij}) \tag{4.5}$$

donde M_I es el número de objetos detectados en la imagen I; $w_{c(i)}$ es el peso de la categoría predicha del objeto i; C' es el número de formas de clasificación, p_{ij} es la probabilidad predicha para la categoría j. Finalmente, se utiliza una estrategia simple para seleccionar el conjunto candidato: para un presupuesto de aprendizaje activo b, se ordenan las imágenes por su incertidumbre y se seleccionan las δb más inciertas del conjunto no etiquetado. Se llama al hiperparámetro $\delta > 1$ la "razón de expansión del presupuesto".

4.2.3. Similitud por Correspondencia Condicionada a la Categoría (CCMS)

La intuición detrás de CCMS es que la similitud entre dos imágenes de instancias múltiples puede calcularse midiendo cuán similares son sus objetos contenidos. Formalmente, para dos imágenes de instancias múltiples I_a e I_b, el detector identifica varios objetos en cada una, de los cuales $O_a = \{o_{a,i}\}_{i\in[M_a]}$ y $O_b = \{o_{b,i}\}_{i\in[M_b]}$, donde cada objeto $o_{*,i}$ es una tripleta $o_{*,i} = (f_{*,i}, t_{*,i}, c_{*,i})$ donde $f_{*,i}$ son las características visuales extraídas de los mapas de características, $t_{*,i}$ es el puntaje de detección, $c_{*,i}$ es la etiqueta de clase predicha, y M_a y M_b son la cantidad de objetos en I_a e I_b respectivamente. Se usa la similitud de O_a y O_b como un intermediario para calcular la similitud de I_a e I_b, la cual se calcula emparejando cada objeto con su contraparte más similar en la misma categoría en la otra imagen. Específicamente, para un objeto $o_{a,i}$ en O_a, su similitud con O_b se calcula como:

$$s(o_{a,i}, O_b) = \begin{cases} \max\limits_{c_{b,j}=c_{a,i}} \frac{f_{a,i}\cdot f_{b,j}}{\|f_{a,i}\|\cdot\|f_{b,j}\|} + 1 & \text{si existe } c_{b,j} = c_{a,i} \\ 0 & \text{si no existe } c_{b,j} = c_{a,i} \end{cases} \tag{4.6}$$

donde se define $S(o_{a,i}, O_b)$ como 0 si ningún objeto en O_b está en la misma categoría que $o_{a,i}$. Luego, la similitud entre O_a y O_b se calcula promediando las similitudes de los objetos ponderadas por sus puntajes de detección:

$$S'(O_a, O_b) = \frac{1}{\sum_i t_{a,i}} \sum_{i=1}^{M_a} t_{a,i} \cdot s(o_{a,i}, O_b) \tag{4.7}$$

$$S(O_a, O_b) = \frac{1}{2} \cdot (S'(O_a, O_b) + S'(O_b, O_a)) \tag{4.8}$$

donde la similitud final es el promedio de $S'(O_a, O_b)$ y $S'(O_b, O_a)$, lo que asegura la simetría de la similitud.

CCMS se usa para seleccionar un conjunto representativo de imágenes del conjunto candidato como la consulta Q de aprendizaje activo. El objetivo del muestreo basado en diversidad se expresa formalmente como:

$$Q = \min_{Q'\subseteq H:|Q'|=b} \left\{ \max_{I_i, I_j \in Q'} S(O_i, O_j) \right\} \tag{4.9}$$

donde H es el conjunto candidato obtenido en la primera etapa, y b es el presupuesto de AL. Sin embargo, la ecuación (4.9) corresponde al problema k-center, el cual es de tipo NP-hard; por lo que se propone el uso del algoritmo k-center-greedy, el cual es una solución eficiente que retorna un resultado que se corresponde a un óptimo local, en particular garantiza que reemplazar cualquiera de los b elementos por otro resulta en aumentar la distancia entre ellos, siendo esta combinación más distante de la óptima. Otro problema del objetivo en la ecuación (4.9) es que solo maximiza la diversidad de las muestras seleccionadas, ignorando cuán representativas son, lo que puede llevar a seleccionar

valores atípicos (outliers) de los datos. Para lidiar con esto, además se corre un algoritmo k-means++ usando los resultados del algoritmo k-center-greedy como centroides iniciales. Sin embargo, aquí solo se puede calcular las similitudes de cada par de imágenes y no calcular la media real de cada clúster para actualizar sus centroides. Se resuelve este problema asignando como nuevos centroides aquellas imágenes cuyas similitudes sumadas con las demás imágenes de su clúster son máximas.

4.2.4. Implementación de PPAL

Esta sección tiene como objetivo delinear las principales características de la implementación de PPAL realizada en este trabajo, además de detallar las consideraciones necesarias que se llevaron a cabo para poder realizar la implementación de las fórmulas descritas en la sección anterior. En la figura 4.5 se puede apreciar el flujo de datos correspondiente a una iteración de aprendizaje activo, en línea con este diagrama se continuará la explicación de las partes involucradas.

Antes de comenzar con las iteraciones, el primer paso consiste en realizar el entrenamiento del modelo a partir de n imágenes etiquetadas iniciales. Ya con el modelo entrenado se procede a realizar las iteraciones, donde cada iteración comienza con la etapa DCUS que toma las imágenes etiquetadas hasta la iteración actual y le aplica la función *obtencion_cajas_delimitadoras y_probabilidad_por_cada_clase* que genera, por cada imagen, la distribución de probabilidad por clase, el puntaje de la imagen (que representa la seguridad del modelo sobre la imagen), y las cajas delimitadoras que el modelo cree hay en esa imagen. Si bien estas imágenes ya están etiquetadas, son utilizadas por PPAL para ejecutar el operador *computar_coeficientes_de_dificultad*, el cual como se nombre lo indica, actualiza los coeficientes de dificultad para cada clase. Para lograr este cómputo se utiliza la fórmula 4.9 que para poder implementar es necesario conocer la correspondencia entre las cajas reales y las cajas generadas por el modelo. Por cada imagen se tienen n cajas reales y m cajas generadas, esto se da porque el modelo no necesariamente devuelve la cantidad exacta de objetos, menos aún en etapas tempranas de entrenamiento.

Para cada caja delimitadora de las m es necesario generar una correspondencia con cada imagen n, para esto se implementa una matriz $M_{m\mathbf{x}n}$ donde la entrada M_{ij} corresponde al cálculo IoU del objeto generado i con el objeto real j, luego la correspondencia para cada objeto se realiza mediante el máximo por filas, intuitivamente esto es asignar cada caja generada a la que más se parece de las reales.

Otro de los aspectos que la fórmula 4.9 necesita es $P(b|\hat{b})$, esto es la probabilidad de la clase de la caja delimitadora, respecto a la caja real a la que fue asignada, es importante remarcar que es a la que fue asignada y no a la que el modelo predijo, ya que pueden diferir. Las arquitecturas devuelven, como salida, la clase asignada, el puntaje correspondiente, y la probabilidad asociada a cada clase. Sin embargo, implementaciones como mmdetection y torchvision generalmente no retornan directamente las probabilidades para cada clase du-

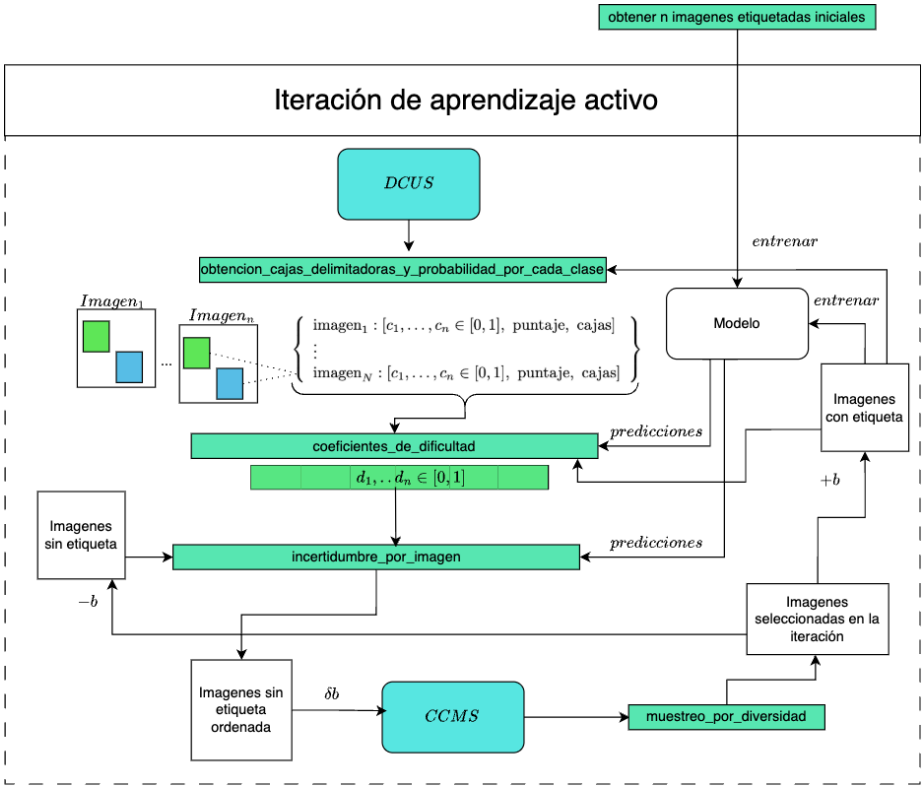


Figura 4.5: El primer paso consiste en entrenar el modelo con las n imágenes iniciales, luego se procede a realizar las iteraciones de aprendizaje activo. Cada iteración comienza en DCUS, se obtienen las cajas delimitadoras junto con su probabilidad por clase de cada imagen, se calcula los coeficientes de dificultad que son la clave para calcular la incertidumbre por imagen que determinan las δb imágenes que entran a la etapa CCMS, que a través del muestreo por diversidad selecciona las imágenes a etiquetar para esta iteración. Se repite este proceso hasta la última iteración.

rante la inferencia, por lo que se tuvo que modificar los cabezales de clasificación del modelo, permitiendo así que el sistema retorne la distribución completa de probabilidades por clase para cada caja delimitadora. Esto no es contrario a lo que se dice en artículo de que es realizable en cualquier arquitectura, pero es necesario para esa arquitectura determinar cuál es la forma de poder retornar en la salida la distribución de probabilidad. Una vez computados los coeficientes de dificultad se comienza a utilizar las imágenes sin etiqueta, precisamente en la etapa *incertidumbre_por_imagen*, donde se hace una pasada hacia adelante de dichas imágenes para obtener su entropía y utilizarla junto con los coeficientes de dificultad en la ecuación 4.5, para este punto se tienen la incertidumbre asociada a cada imagen sin etiqueta. Posteriormente, se ordenan por incertidumbre y se toman las primeras δb (en la experimentación se usa principalmente $\delta = 4$) que pasarán a la etapa CCMS aplicando la operación *muestreo_por_diversidad* para obtener las b imágenes de la iteración. Una consideración que surge en este cómputo es que en la ecuación 4.6 f_a corresponde al mapa de características correspondiente al objeto a, para implementar esto se logra extrayendo primero el mapa de características de toda la imagen, luego recorta la región de interés correspondiente a cada predicción (RoI pooling), y finalmente pasando cada una de esas regiones por la cabeza de la red (box head) para obtener el vector final de características. De esta forma, cada objeto detectado se representa por un vector numérico que captura su información visual permitiendo realizar la comparación de similitud. Una vez seleccionadas las b imágenes, las mismas se agregan al conjunto de imágenes con etiqueta, y se quitan del conjunto de imágenes sin etiqueta, completando la iteración de aprendizaje activo.

4.3. Generación de imágenes sintéticas

La siguiente fase del flujo de datos propuesto a tratar consiste en la generación de imágenes artificiales para el aumento de datos; es decir, entrenar modelos generativos con el objetivo de ser capaces de crear nuevas imágenes de las malezas en cuestión, de forma tal que al agregar dichas imágenes al conjunto de entrenamiento, se obtenga un mayor rendimiento al entrenar detectores con dicho conjunto de entrenamiento aumentado.

Es prudente primero destacar que este objetivo es ambicioso, principalmente debido a que se está buscando que, mediante el aprendizaje utilizando imágenes artificiales de un concepto, un detector sea capaz de poder distinguir dicho concepto en imágenes reales; por lo que en consecuencia tanto la calidad de las imágenes, como la diversidad, resultan factores claves para el anterior cometido.

En particular, resulta critico que las imágenes generadas preserven aspectos claves de la especie de la maleza, como por ejemplo la morfología foliar de las hojas (base, ápice, borde de la hoja, entre otros), sus colores, la separación entre las hojas, el número de hojas, entre otras propiedades. Pero al mismo tiempo, que el modelo a partir de lo aprendido durante su pre-entrenamiento, sea capaz ofrecer diversidad, por ejemplo alterando aspectos como la ubicación de la planta, la posición, tamaño y la proporción de hojas, su orientación, y la

superficie donde la maleza yace, entre otras. Por último, pero no por eso menos importante, como fue explicado en la sección 4.1, se busca tener control de la imagen generada, en particular en lo que respecta a la especie, e idealmente el número de las malezas por imagen; de esta forma, solamente restaría la ubicación para obtener su etiqueta, como sera explicado en la sección 4.4.

Por lo tanto, en esta sección se describe la elección del modelo generativo utilizado, como también el proceso de ajuste y la generación de instancias.

4.3.1. Elección del modelo generativo

Una de las decisiones más importantes tomadas durante el transcurso de este proyecto es, sin lugar a duda, cual arquitectura de modelo generativo utilizar. Como se presenta en la sección 2.3, existen diferentes arquitecturas de modelos generativos de imágenes, como las VAEs, GANs, y modelos de difusión; donde hoy en día la arquitectura que conforma el estado del arte en lo que respecta a la generación de imágenes artificiales son los modelos difusos, como por ejemplo los modelos de difusión latentes (LDM).

Hoy en día los modelos de difusión son ampliamente utilizados, tanto en el mundo audiovisual como en el académico, en particular, implementaciones de LDM como Stable Diffusion se han vuelto especialmente populares, dada su gran capacidad de generación de imágenes de propósito amplio (no acotado a un solo dominio, como por ejemplo imágenes de personas, animales, entre otros). Es más, diversos estudios han mostrado que los modelos difusos más recientes han logrado obtener una significativa mejora en la calidad y diversidad en las imágenes generadas en comparación con las GANs (Dhariwal y Nichol, 2021) (H. Wang, 2024), a costa de un mayor costo computacional.

Adicionalmente, debido al aprendizaje de tipo adversario entre el generador y el discriminador, las GANs presentan problemas adicionales. Han sido propuestas diferentes variantes de GANs con el fin de poder abordar dichos problemas, aunque lo cierto es que aun siguen siendo relevantes. Algunos de estos problemas son:

- **Colapso de moda:** Consiste en que el generador crea (en su mayoría o en su totalidad) imágenes con las mismas características, teniendo solamente muy sutiles variaciones entre sí. El origen del problema se remonta al discriminador, el cual resulta tener un peor desempeño al discernir la veracidad de imágenes con determinadas características, por lo que el generador tiende a generar imágenes con dichas características con mayor frecuencia debido a la naturaleza del entrenamiento antagónico. Variantes como las Wasserstein GANs (WGAN) (Arjovsky, Chintala, y Bottou, 2017) y Wasserstein GANs con penalización de gradiente (WGAN-GP, Wasserstein GAN with gradient penalization) (Gulrajani, Ahmed, Arjovsky, Dumoulin, y Courville, 2017) atenúan parcialmente el problema. Notar que este problema afecta directamente a la diversidad, y por lo tanto es notoriamente perjudicial.

- **Inestabilidad y desvanecimiento del gradiente:** Debido a la naturaleza competitiva de la función de costo, ocurren momentos en que pequeños cambios en una de las dos redes, pueden resultar en grandes cambios en la otra, dando como resultado que si se pierde la sincronización en sus estados, se puede perder lo aprendido. Estos problemas limitan el aprendizaje del modelo, e implican cómputo adicional. Además, como los dos modelos son entrenados en simultáneo, puede ocurrir que uno de ellos aprenda con mayor velocidad que el otro. En el caso de las GANs, es común que esto ocurra con el discriminador, lo que resulta en que el mismo logre en una etapa temprana del entrenamiento, distinguir con gran precisión qué imágenes son reales o generadas. En estos casos, el discriminador no produce información en su gradiente de utilidad para el generador, siendo esto extremadamente negativo para el aprendizaje de este último. Variantes como GANs de crecimiento progresivo (ProGAN, Progressive Growing of GANs) (Karras, Aila, Laine, y Lehtinen, 2017), WGAN, WGAN-GP, o GAN desenrolladas (Unrolled GAN) (Metz, Poole, Pfau, y Sohl-Dickstein, 2016) atenúan dicho problema. Notar que en el contexto de este proyecto, el flujo no debería incurrir en esfuerzo humano adicional al realizado en el etiquetado de imágenes seleccionadas por el selector, por lo tanto, los casos de inestabilidad son especialmente problemáticos, ya que podrían requerir una intervención humana adicional.

- **Falta de modelos preentrenados de uso general:** Por lo general, los modelos generativos requieren de un elevado número de imágenes a la hora de ser entrenados, es por ello que en la mayoría de los casos, resulta de mayor conveniencia realizar ajustes sobre modelos preentrenados. Usualmente las GANs son entrenadas sobre un domino particular, donde son utilizados conjuntos de entrenamiento en el orden de las decenas de miles de imágenes. Por ejemplo en StyleGAN3 se utilizó el conjunto Faces-HQ (70.000 imágenes) y Cityscapes (12.700 imágenes). Por el contrario, los modelos de difusión actuales son entrenados para la generación de imágenes de amplio propósito, donde no se está limitado a un dominio en particular. En consecuencia, se utilizan conjuntos de entrenamiento con un número considerablemente mayor de imágenes, por ejemplo, uno de los conjuntos de datos utilizados en el entrenamiento de Stable Diffusion v1 es LaionAesthetics v2 5+ (*LAION-Aesthetics — LAION*, 2022) con aproximadamente 600 millones de imágenes. Por lo tanto, puede ser un desafió encontrar un modelo preentrenado en un dominio específico a la hora de ajustar una GAN, mientras que en los modelos difusos se puede ajustar en cualquier dominio partiendo de un modelo preentrenados de propósito general. Siendo esto último de especial utilidad considerando la baja disponibilidad de modelos generativos sobre malezas.

Al tener en cuenta los anteriores argumentos, la decisión de optar por utilizar los LDM como modelo generativo parece clara; aunque debido al contexto de este proyecto, las anteriores desventajas podrían ser aceptadas ante la posibilidad

de utilizar un flujo de datos como el de HandsOff (sección 4.1 y 3.3) el cual, como fue previamente detallado, ofrece sustanciales ventajas en el proceso de autoetiquetado de las imágenes sintéticas. En tal escenario, se vuelve necesario analizar la viabilidad de utilizar la variante StyleGAN2 (Karras y cols., 2020) como modelo generativo, y compararla en aspectos claves con los LDM. Como fue detallado en la sección 3.3 el flujo de datos propuesto en HandsOff está específicamente diseñado para la arquitectura StyleGAN2.

Una de las características de la variante StyleGAN2, es que busca generar la imagen mediante una serie de "Bloques de estilo", los cuales tiene el objetivo de poder variar el estilo de un sujeto sin perder propiedades claves que lo definen. En lo que respecta a este proyecto, la anterior característica no es deseable, ya que si un modelo generativo fuera entrenado con todas las imágenes de todas las malezas (como lo indica el flujo de datos de HandsOff), entonces el generador potencialmente podría producir imágenes de malezas que combinen características de diferentes especies, produciendo instancias que no corresponderían por ende a ninguna especie y no sean útiles para el entrenamiento de un detector. El anterior efecto es ilustrado en la figura 4.6, donde en el trabajo Phil Wang (2020) se entrena un generador StyleGAN2 con imágenes de flores de diferentes especies, obteniendo algunas imágenes de flores que no existen.

En consecuencia, para poder aplicar HandsOff al actual proyecto, se debería adaptar el método a un generador con una arquitectura diferente. Notar que por lo general, en modelos generativos donde se busca poder separar claramente diferentes conceptos (como especies en este caso), se utiliza algún mecanismo de condicionamiento en el modelo generativo, de forma que se pueda agregar información extra a la hora de generar, como una clase o descripción.

En caso de que efectivamente sea utilizado un generador condicional, o directamente se entrenen diferentes modelos generativos para cada especie, ya no se podría aplicar el flujo de datos propuestos en HandsOff, ya que primero sería necesario separar las imágenes por especie de maleza antes de poder entrenar el o los modelos generativos. Como se detalla en la sección 5.2, el conjunto de datos de entrenamiento consta de 4.700 imágenes de alta resolución, con 12 especies de malezas, además se busca no etiquetar más de un 20 % de dichas imágenes, resultando en menos de 1.000 imágenes, y en consecuencia (en el caso uniforme), menos de 90 imágenes por especie. Existen estudios que han explorado el tamaño del conjunto de datos necesario para realizar exitosamente un ajuste en una GAN (Y. Wang y cols., 2018), se observa que el número de imágenes necesaria está estrechamente vinculado a la resolución de las imágenes generadas, destacando ademas que cuando la resolución es de 64×64, se requieren entre cientos y miles de imágenes, o incluso más. Es más, se destaca que un modelo preentrenado con el conjunto ImageNet, posteriormente ajustado con un subconjunto de las imágenes de LUSUN Bedrooms, y evaluado sobre todo el conjunto LUSUN Bedrooms utilizando como métrica FID (sección 2.3.5), continúa mejorando de forma no despreciable, incluso cuando se utilizan un millón de imágenes para el proceso de ajuste. En el caso de los ajustes de modelos generativos realizados en HandsOff, se utilizaron conjuntos de datos como CelebAMask-HQ (30.000 imágenes), Car-Parts (1.800 imágenes), DeepFashion

Figura 4.6: Imágenes de flores generadas por modelo StyleGAN2 al ser entrenados con de diversas especies de flores. Múltiples de las flores generadas no corresponden a ninguna especie existente (Phil Wang, 2020).

MultiModal (44.100 imágenes), y Cityscapes (25.000 imágenes).

Entonces, hasta primero no explorar nuevas arquitecturas de generadores GANs no condicionales, compatibles con lo propuesto en HandsOff, se descarta la posibilidad del uso de generadores de imágenes sintéticas con GANs. A los objetivos y plazos del actual proyecto, no se abordará dicha investigación.

A diferencia del ajuste de las GANs, existen varios métodos de ajuste de modelos difusos condicionales, como por ejemplo Dreambooth (sección 2.3.4) los cuales logran con un pequeño número de instancias ($\sim$ 5) incorporar un nuevo concepto (en este caso cada especie) en el modelo generativo.

Finalmente, se opta por la arquitectura de modelos generativos de tipo LDM; en particular por la implementación de Stable Diffusion v1.5 por razones de

compatibilidad con las implementaciones de los métodos de ajustes utilizados y explicados en la sección 4.3.3.

4.3.2. Conjunto de entrenamiento de los modelos generativos

Con el fin de ajustar los modelos generativos, son necesarios conjuntos de imágenes de malezas de las especies que se buscan aprender a generar. Dado el flujo de datos utilizado (sección 4.1) los modelos serán ajustados distinguiendo las malezas de cada especie; específicamente, en caso de utilizar un solo modelo, se especificará, en la descripción de la imagen, la especie. Mientras que en caso de utilizar varios modelos, cada uno de ellos será entrenado con las imágenes de una sola especie.

Los modelos difusos condicionales, como Stable Diffusion, son ajustados utilizando únicamente un conjunto de imágenes junto con su descripción asociada, lo cual significa que no se indica de ninguna forma que objeto en la imagen es el que se busca aprender, y por consiguiente, que partes de la imagen se deben ignorar. A modo de ejemplo, si se presentan múltiples imágenes de determinada especie, lo buscado sería aprender a generar nuevas instancias de dicha especie y no el suelo, es más, seria ideal si aspectos como el tipo de suelo, posición de la maleza, o iluminación pudieran ser cambiados a la hora de generar las nuevas imágenes. En consecuencia, las imágenes utilizadas para ajustar el modelo generativo deberían componerse mayoritariamente de la maleza.

Inspirados por el método propuesto en el trabajo Analysis of Stable Diffusion-derived fake weeds performance for training Convolutional Neural Networks (Moreno y cols., 2023) (sección 3.3) las imágenes utilizadas para ajustar los modelos generativos son creadas a partir de las cajas delimitadoras de las imágenes previamente etiquetadas, y luego ampliadas hasta cierto limite configurable; un ejemplo resultante del anterior proceso se presenta en la figura 4.7c. En el caso de que las imágenes generadas (junto con sus respectivas etiquetas) sean directamente utilizadas en el entrenamiento de modelos de detección (como es realizado en este proyecto), la ampliación de las cajas delimitadoras antes mencionada debe ser necesariamente realizada, ya que en caso contrario, los modelos generativos serían ajustados con imágenes donde la maleza abarcaría la totalidad de la fotografía (figura 4.7a), entonces sería esperable (como se ejemplifica en la figura 4.7b) que la maleza en la imagen generada también abarque la totalidad imagen. En tal caso, la caja delimitadora asociada a la imagen generada sería trivialmente toda la imagen, y en consecuencia, la anterior instancia no sería de gran utilidad al entrenar posteriormente detectores, especialmente en lo que respecta a la tarea de localización.

Los procesos de ajustes de modelos difusos, en particular en aquellos donde se busca incorporar un nuevo concepto utilizando un bajo número de instancias de entrenamiento, tienen la desventaja de ser muy susceptibles a ejemplos de entrenamiento con características no deseadas, como un desbalance en el ajuste de blancos (Figura 4.8a) o en el contraste, desenfoque en la maleza (Figura 4.8b), oclusiones, entre otras.

(a) (b) (c)

Figura 4.7: (a) Recorte de maleza *eleusine indica* según caja delimitadora. (b) Imagen generada por modelo difuso ajustado con imágenes recortadas según su caja delimitadora. (c) Recorte de maleza *eleusine indica* utilizado su caja delimitadora y posteriormente ampliada.

(a) (b)

Figura 4.8: Imágenes creadas por modelos difusos ajustados con fotografías de la especie *ambrosia artemisiifolia* (a) y *sida rhombifolia* (b) respectivamente. En (a) las imágenes de entrenamiento no tenían un balance de ajuste de blancos, mientras que en (b) no fueron filtrados los casos donde las malezas estaban desenfocadas.

Notar que los antecedentes presentados en la sección 3.3, los cuales también se enfocan en la generación de imágenes artificiales para el aumento de datos en malezas, no tiene como limitante el esfuerzo humano, y por consiguiente, se realiza manualmente el proceso de selección y tratamiento de las imágenes utilizadas para ajustar los modelos generativos.

En el caso de este proyecto, este proceso es realizado de forma automática mediante una serie de filtros y ajustes. Los mismos (junto con los detalles del proceso de ampliación de las cajas delimitadoras), son tratados en detalle el ane-

xo A y presentados a continuación de forma general. Dada una caja delimitadora de una maleza en una imagen:

1. Si la caja delimitadora intercepta con cualquier otra caja, la instancia es descartada.

2. Se estima el nivel de desenfoque utilizando la varianza del Laplaciano. Se descartan las instancias cuya estimación de desenfoque no supere el límite configurado.

3. Se expande la caja delimitadora en todas las direcciones posibles, de forma tal que no se intercepte ninguna caja delimitadora correspondiente a otra maleza en la imagen. La máxima expansión permitida es proporcional al largo de la caja delimitadora, donde su factor está definido por configuración.

4. Si las dimensiones del recorte obtenidas luego de la expansión son menores a las establecidas como mínimas, se descarta la instancia.

5. Se normaliza el brillo y el contraste de la imagen. Luego, la misma es rescalda a la resolución con la que será ajustado el modelo generativo.

6. En el caso de que el método de ajuste requiera una descripción asociada a la imagen, para cada recorte se genera su respectiva descripción, que es configurable (se discute en detalle en la sección 5.4) e incluyendo un identificador de la especie y, opcionalmente, la posición de la maleza en la imagen.

7. Como será profundizado en la sección 4.3.3, se define un número máximo de instancias por maleza. Se conservan las instancias de mayor resolución por especie.

El resultado de este proceso es un conjunto acotado (por el máximo número de instancias por especie) de recortes clasificados por especie, cuyo brillo y contraste se encuentran balanceados. Estos recortes (de igual resolución) contienen una única maleza (idealmente no borrosa), ubicada de forma aleatoria dentro de la imagen, donde la misma ocupa la mayoría del recorte. Estas imágenes son utilizadas a continuación para el proceso de ajuste de los modelos generativos.

4.3.3. Ajuste de los modelos generativos

Dado el conjunto de imágenes confeccionado en la anterior sección, se procede al ajuste de los modelos generativos. Como se describe en la sección 2.3.4, existe una amplia variedad de métodos de ajuste de modelos difusos, cada uno con características particulares dependiendo del contexto del ajuste dado.

En particular en este trabajo se busca que cada imagen contenga una única especie de maleza, entonces, métodos especializados en la generación multi-concepto, como Custom Diffusion (Kumari y cols., 2023) no serían adecuados.

Además, aspectos como la preservación de la morfología de la especie resultan determinantes, lo cual es obtenido de mejor manera si son ajustados los pesos internos del modelo generativo (e incluso mejores si también se ajusta el codificador de texto), por lo tanto no seria adecuado el uso de métodos que no optimicen dichos parámetros, como Textual Inversion (Gal y cols., 2022b). En la figura 4.9 se observa como el anterior método presenta dificultades en la preservación de la morfología en comparación con métodos como Dreambooth.

Figura 4.9: Imágenes generadas por los modelos generativos Stable Diffusion e Imagen, luego de haber sido ajustados por los métodos Dreambooth y Textual Inversion. Se muestra que Dreambooth ofrece mayor fidelidad con respecto al florero rojo mostrado en la primera fila de la imagen (Ruiz y cols., 2023).

Por lo tanto, en este trabajo se exploran dos mecanismos de ajustes que modifican todos los pesos del generador. El primero de ellos se trata de un ajuste mediante continuación de entrenamiento; en otras palabras, se continua entrenando al modelo de igual forma a como se preentrenó, pero utilizando como conjunto de entrenamiento las imágenes y descripciones obtenidas del proceso explicado en la sección 4.3.2. Este método fue el realizado en el trabajo Analysis of Stable Diffusion-derived fake weeds performance for training Convolutional Neural Networks (Moreno y cols., 2023) (sección 3.3), donde fueron entrenados

dos modelos diferentes por especie de maleza, uno cuando la maleza se encuentra en su etapa de crecimiento, siendo apenas unos brotes, y otro modelo cuando la misma ya se encuentra desarrollada. En el caso de este trabajo no se cuenta con información sobre su etapa de crecimiento, si no que solo se conoce su especie. Se experimenta tanto entrenando en un solo modelo como entrenando en un modelo diferente por especie.

Ha sido observado que los modelos de difusión tienen excelentes capacidades para integrar nuevos conceptos al realizar un cuidadoso ajuste del modelo mediante la continuación de su entrenamiento, incluso utilizando pequeños conjuntos de entrenamientos (Ruiz y cols., 2023). De todas formas, es usual que el entrenamiento tienda a sobreajustarse con facilidad, y que el modelo pierda diversidad en las imágenes generadas; ademas se ha observado que en caso de que los parámetros del codificador de texto también sean ajustados, el modelo progresivamente pierda capacidades del entendimiento del lenguaje (Ruiz y cols., 2023).

En consecuencia, en este trabajo se experimenta principalmente con el método Dreambooth, como segundo método de ajuste de modelos difusos. Como fue presentado en la sección 2.3.4, el método de ajuste Dreambooth es muy parecido a simplemente continuar con el entrenamiento, pero con dos propuestas fundamentales. La primera es su técnica para formar las descripciones de las imágenes de entrenamiento, donde se utiliza un token de bajo valor semántico para embeber el nuevo concepto, y un descriptor de clase el cual representa un concepto general al que se busca aprender, permitiendo lo anterior un entrenamiento más veloz y estable. La segunda propuesta es el regularizador llamado "Perdida de Preservación Previa Especifica de Clase" (Class-specific Prior Preservation Loss), el cual tiene como objetivo que el modelo no pierda capacidad de generación de conceptos previos al entrenamiento; dicho regularizador permite preservar la diversidad del modelo generativo, ademas de reducir el riesgo de sobreajuste, aunque el mismo, como se evidencia en la sección 5.4 sigue siendo un riesgo latente.

Dreambooth en su versión original no propone ningún mecanismo para ajustar múltiples conceptos en un mismo modelo, por lo tanto en este trabajo se ajusta para cada especie un modelo generativo distinto.

Es importante aclarar que en el caso de ambos métodos, los mismos fueron ejecutados mediante la librería Diffusers de Hugging Face, específicamente utilizando sus implementaciones para ambos métodos de ajuste.

En el siguiente capítulo, en particular, en la sección 5.4 se desarrollan las formas en que los anteriores métodos fueron realizados, experimentando con los diferentes hiperparametros y mostrando los resultados obtenidos, tanto en la calidad de las imágenes obtenidas, como en su impacto al entrenar detectores.

4.3.4. Generación de instancias sintéticas

Una vez ajustados los modelos de difusión, se procede a generar las imágenes sintéticas.

La generación es realizada utilizando el mismo planificador y número de iteraciones que cuando fue ajustado el modelo generativo, los cuales no fueron modificados de sus opciones por defecto, en particular, el planificador es DDIMScheduler (Song, Meng, y Ermon, 2020) y se utilizan 50 iteraciones.

Notar que la imagen puede ser generada utilizando cualquier instrucción (prompt), el cual debe contener el token que identifique a la maleza. Seria de gran interés poder especificar la localización de la maleza en la imagen, por ejemplo, haciendo uso de métodos como ControlNet (L. Zhang, Rao, y Agrawala, 2023). Desafortunadamente esta alterativa es desestimada dado el precedente del trabajo Weed image augmentation by ControlNet-added stable diffusion for multi-class weed detection (Deng y Lu, 2025) (sección 3.3), donde utilizando el anterior método, intentan generar imágenes de malezas especificando su localización vía cajas delimitadoras; no lograron que el generador cree la maleza en la posición indicada. En la sección 5.4, se experimenta en mayor detalle sobre el contenido de la instrucción utilizada para generar las imágenes.

Finalmente, una vez que la imagen es generada, con el fin de prevenir situaciones donde lo producido por el generador difiere demasiado de lo esperado, se procede mediante un detector (entrenado con las imágenes seleccionadas y etiquetadas del flujo de datos) a buscar la maleza dentro de la imagen generada. Si no se obtiene alguna detección de la especie esperada, con un nivel de confianza mayor al mínimo configurado, entonces la imagen generada es descartada.

4.4. Autoetiquetado de imágenes sintéticas

En esta sección se describe el componente encargado de autoetiquetar las imágenes sintéticas. Este componente recibe como entrada un conjunto de imágenes, de las cuales se sabe que existe al menos una instancia de maleza, así como la clase a la que pertenecen. Esto es posible, ya que, al momento de generar conjuntos de imágenes sintéticas, el generador es condicionado para generar muestras de una clase específica. Debido a esto, el problema al que se enfrenta el componente no es de detección de objetos per se -lo cual incluye tanto la localización como la clasificación-, sino unicamente de localización de la clase hiperónima que abarca a las demás.

Como se explicó en 3.4.1, es posible categorizar los métodos de autoetiquetado de imágenes en dos grupos principales.

El primero incluye aquellos los cuales hacen uso del estado interno del modelo al momento de generar la imagen, de forma de aprovechar lo aprendido por el modelo generativo durante el entrenamiento, entre los que se incluyen HandsOff y DiffuGen.

El método HandsOff fue descartado por consecuencia de haber prescindido de las GANs a favor de modelos difusos, por las razones ya explicadas en 4.3.

Esta categoría de métodos requiere que el modelo generador sea capaz de separar y comprender explícitamente cada concepto que ha aprendido, y pueda aplicarlos en condiciones diferentes a las cuales ha visto durante el entrenamiento. Por ejemplo, se espera que, tras ser entrenado para generar imágenes

de malezas, el modelo pueda generar nuevas muestras con diferentes cantidades, o en localizaciones específicas dentro de la imagen. De no cumplirse, esto implicaría que el modelo interpreta que la clase está definida por el objeto en conjunto con el fondo en el que se encuentra.

Durante la experimentación, se observó como este no era el caso para los modelos obtenidos. Si bien eran capaces de generar imágenes con gran fidelidad a las reales, estos no eran lograban separar los conceptos. Se profundiza más sobre esto en el anexo C.

Además, al probar la implementación oficial de DiffuGen, se encontró que poseía dependencias incompatibles (algunas de ellas deprecadas) con las usadas en nuestro proyecto. Adaptarse a estas requería prescindir de versiones más actuales y funcionales de librerías para el entrenamiento de modelos generativos.

Debido a estas razones, se optó por descartar Diffugen, y priorizar la segunda categoría de métodos de autoetiquetado, compuesta por aquellos que se basan únicamente en la información contenida en la imagen.

En particular, uno de estos es el utilizado en Analysis of Stable Diffusion-derived fake weeds performance for training Convolutional Neural Networks (Moreno y cols., 2023), explicado en 3.3, donde se implementa un método sencillo en el que cada pixel es categorizado como vegetación o fondo, dependiendo de sus valores en los canales RGB, favoreciendo aquellos con más valor en el canal G (verde, green) y penalizando aquellos con más valor en R (rojo, red). Esta implementación es suficiente para el dominio en el cual fue aplicada, en el cual las imágenes generadas poseían un fondo marrón con una maleza muy contrastada.

Sin embargo, este enfoque presenta limitaciones cuando se aplica en dominios ligeramente más complejos, donde los artefactos verdes no siempre se corresponden necesariamente a un objeto a etiquetar, y pueden llegar a introducir ruido en las ubicaciones. En el caso que se busquen métodos más complejos, que separen los diferentes artefactos verdes, y luego se busque agruparlos por maleza, surge el problema que los mismos se encuentran conectados mediante secciones de otro color, como un tallo marrón, lo cual podría provocar que una misma instancia sea detectada como múltiples ubicaciones distintas.

En la figura 4.10 se presenta un ejemplo en el cual el fondo de la imagen tiene musgo de color verde oscuro, el cual es confundido como zona de interés.

Por otro lado, en la figura 4.11 se presenta un ejemplo en el cual, debido a la iluminación, las partes que conectan algunas de las hojas de las diferentes instancias toman un color más cercano al amarillo, por lo que se considera como fondo, separando las cajas delimitadoras generadas.

Ambos casos se pueden ver como un problema de trazar una linea clara la cual separe los colores considerados de interés y los que no. Por ejemplo, en el primer caso, si definimos que es necesario un verde más claro, el musgo dejaría de ser considerado, pero también lo harían hojas de plantas con tonalidades más oscuras. En el segundo caso, si se aceptan colores más cercanos al amarillo, podrían también incluirse erróneamente artefactos que adopten esa tonalidad por la iluminación directa del sol. A continuación se explican tres enfoques diferentes que buscan afrontar estas limitaciones.

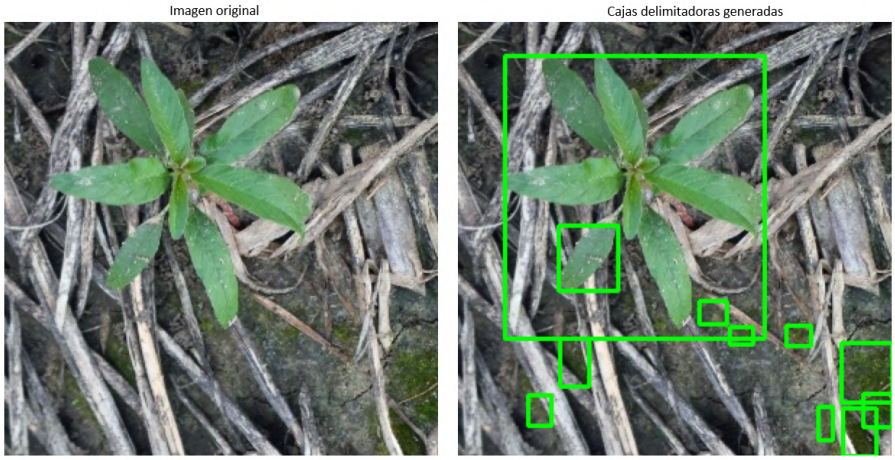


Figura 4.10: Resultado obtenido al aplicar un método sencillo de autoetiquetado mediante canales colores a una imagen sintética donde no todos los artefactos verdes son parte de una planta.

Figura 4.11: Resultado obtenido al aplicar un método sencillo de autoetiquetado mediante canales colores a una imagen sintética donde algunas partes de la planta están ligeramente separadas.

4.4.1. Canales de colores

El primero de estos enfoques, el cual ya fue explicado superficialmente en la sección 3.4.2, hace un uso mejorado de la información obtenida de los canales de colores.

En particular, para la segmentación de las regiones de interés dentro de

cierta imagen RGB, esta es convertida a formato Lab. Este formato separa la luminancia de la información de color, lo cual resulta especialmente útil en entornos donde la iluminación puede variar, como es el caso de imágenes de malezas en un campo.

Una vez convertida la imagen, se extraen únicamente los canales a y b, los cuales contienen la información del color en dimensiones ortogonales. El canal a representa la variación de color entre verde y magenta, y el canal b lo hace entre azul y amarillo. Cada pixel de la imagen queda entonces representado como un punto en un espacio de dos dimensiones, definido por sus valores en a y b.

Sobre este espacio (conocido como espacio de características), se aplica el método de aprendizaje no-supervisado K-Means explicado en 2.1.1, con el objetivo de agrupar los píxeles en dos grupos. Por lo tanto, el valor de k se fija en 2, lo cual proporciona una segmentación binaria, que diferencia entre aquellos píxeles de interés, que posiblemente pertenezcan a una maleza, y aquellos que posiblemente pertenezcan al fondo.

Una vez se termina la segmentación de la imagen, cada pixel pertenece a uno de dos grupos, sin embargo, aún no sabemos cuál de los dos grupos indica la zona de interés, dado que K-Means no etiqueta, sino que agrupa. Para obtener la etiqueta correspondiente, se calcula la media de los valores del canal b dentro de cada grupo, y se selecciona el que obtenga la menor media como zona de interés, lo cual indica mayor presencia de colores verdes, asumiendo que las malezas tienden a presentar valores más chicos en el canal b dado su color verde.

Luego de esto, se aplica un filtro de mediana para reducir el ruido provocado por píxeles aislados que no deberían pertenecer a la máscara. Se aplica además dilatación, de forma de aumentar el tamaño ocupado por las máscaras, y luego erosión, lo cual disminuye el tamaño de las mismas. Esto elimina pequeños huecos y separaciones entre partes de una misma maleza, para evitar generar más de una caja delimitadora por instancia.

Por último, se filtran regiones cuya área no supere un umbral mínimo, calculado como un porcentaje del área total de la imagen (por ejemplo, 5 %), con el objetivo de eliminar zonas pequeñas que posiblemente no sean de interés, y a cada región conexa se le asigna una caja delimitadora.

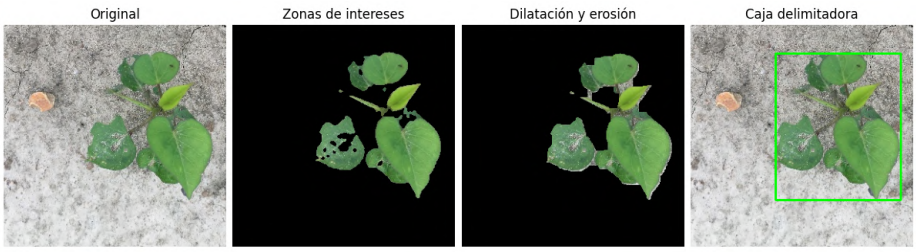


Figura 4.12: Etapas del autoetiquetado por canales de colores mediante agrupamiento por color

En la figura 4.12 se observa los resultados de las diferentes etapas. Notar

que, previo a la etapa de dilatación y erosión, se hubiesen generado tres cajas delimitadoras, una por cada zona conexa. Sin embargo, gracias a esta etapa, una sola caja es devuelta finalmente.

Si bien este método es capaz de encontrar las zonas de interés casi en la totalidad de los casos, surge un problema al momento de convertir las diferentes máscaras en cajas delimitadoras. Por ejemplo, si dos instancias de malezas diferentes se encuentran muy cercanas, o superpuestas, sus mascaras generarán una sola componente conexa, lo cual se corresponderá a una única caja delimitadora. Por el contrario, si una instancia de maleza posee hojas ligeramente separadas, que no son unidas por ninguna mascara luego de la segmentación, generará varias cajas delimitadoras para una única instancia.

Dada la dificultad para combinar los diferentes segmentos, finalmente se determinó que, al finalizar el proceso, se combinen todas las mascaras generadas como una única caja delimitadora, puesto que durante la experimentación se observó que los generadores eran capaces de crear muestras sintéticas mayoritariamente con una única instancia. De todas formas, se experimenta en la siguiente sección con el uso del modelo SAM para explorar otros métodos que permitan una mejor agrupación.

4.4.2. Canales de colores y SAM

Con el objetivo de solucionar los problemas inherentes al enfoque basado exclusivamente en la segmentación por canales de colores, se propone el agregado del modelo SAM (Segment Anything Model) como complemento dentro del proceso de autoetiquetado. Este modelo permite generar mascarás semánticas correspondientes a las distintas instancias, en base a un conjunto de puntos bidimensionales sobre la imagen, los cuales actúan como guías de regiones de interés.

El comienzo del proceso es similar al presentado anteriormente. Sin embargo, una vez obtenidas las máscaras, en lugar de calcular las cajas delimitadoras directamente de las zonas encontradas, se introduce una etapa adicional. En esta etapa, se seleccionan una serie de puntos equiespaciados dentro de cada región segmentada. Estos puntos son luego utilizados como entrada para el modelo SAM, el cual genera nueva máscaras que se ajusten con mayor precisión a las instancias en la imagen.

En la figura 4.13 se presenta un caso en el cual la planta posee dos hojas claramente separadas de las demás, y para las cuales el proceso de dilatación y erosión no fue suficiente para juntar en un solo segmento. En este caso, al utilizar SAM con la misma imágen, usando varios puntos que indiquen zonas en las cuales se está seguro que hay una planta, este es capaz de segmentar correctamente la imagen, utilizando el tallo que une las diferentes partes.

Por otro lado, en la figura 4.14 se observa un caso en el cual SAM no termina de unir correctamente una de las hojas mediante el tallo, provocando así un error al generar dos cajas delimitadoras en lugar de una.

Este agregado permite mejorar la segmentación en muchos casos donde las malezas presentan estructuras más complejas, como hojas o partes separadas

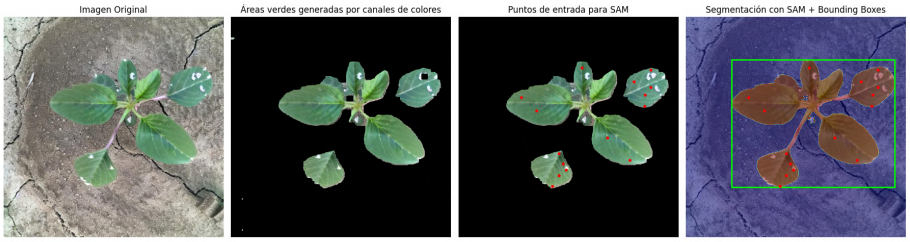


Figura 4.13: Etapas del autoetiquetado por canales de colores y SAM. La instancia contiene hojas separadas de las demás, pero SAM es capaz de unirlas correctamente mediante una máscara.

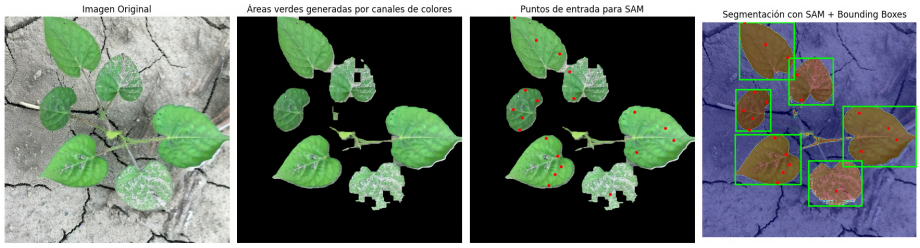


Figura 4.14: Etapas del autoetiquetado por canales de colores y SAM. Se obtiene un resultado erróneo debido a que SAM no es capaz de unir correctamente las hojas.

especialmente. Sin embargo, esta solución no resuelve completamente todos los casos problemáticos. En particular, cuando existen múltiples malezas muy próximas entre sí, este enfoque continua generando una única mascara conexa que posteriormente se corresponderá con una única caja delimitadora.

4.4.3. Modelo de detección

Otro enfoque posible para el autoetiquetado de imágenes es aquel presentado en Weed image augmentation by ControlNet-added stable diffusion for multi-class weed detection, explicado en 3.4.1. En este, una vez generada la imagen sintética, la misma es etiquetada automáticamente por un modelo de detección previamente entrenado, y corregida manualmente por una persona.

En nuestro caso, se propone una variante de este enfoque que prescinde de la intervención manual, con el objetivo de que este componente continúe siendo completamente automático. Para esto, una vez seleccionadas las imágenes reales que se utilizarán para entrenar el modelo generativo, estas mismas se aprovechan para entrenar un modelo de detección. Este modelo, entrenado sobre datos reales, es posteriormente utilizado para etiquetar las imágenes generadas

por el modelo generador.

Las etiquetas dadas por el modelo son luego filtradas bajo los siguientes tres criterios:

- Todas las etiquetas generadas por el modelo de detección deben superar cierto umbral de confianza preestablecido. Esto asegura que solo se utilicen predicciones en las que el modelo tiene mucha certeza.
- Al haber utilizado un modelo de generación, se sabe que debe existir al menos una instancia de una maleza. Por lo tanto, si el modelo de detección no encuentra ninguna instancia, la imagen y sus etiquetas son descartadas.
- Nuevamente, al haber generado la imagen, se sabe exactamente a que clase debe pertenecer cada instancia, por lo que, en caso de que alguna caja delimitadora no se corresponda con la clase esperada, la etiqueta es descartada.

Si bien utilizar un modelo de detección para entrenar otro modelo de detección puede parecer contradictorio, este procedimiento puede resultar útil siempre que se apliquen los filtros mencionados, los cuales introducen sesgos que favorecen la calidad de las etiquetas.

4.5. Integración con CVAT

El sistema desarrollado fue integrado con CVAT (Computer Vision Annotation Tool), una herramienta utilizada para la anotación manual de imágenes en proyectos de visión por computadora. Esta integración tiene como objetivo permitir al usuario realizar el etiquetado de las imágenes directamente al utilizar el programa construido, sin necesidad de coordinar el proceso de anotación de manera externa al proyecto.

Es necesario contar previamente con un servidor que esté ejecutando una instancia de CVAT para poder utilizar esta función.

Con esta integración, el etiquetado manual se convierte en una parte más del flujo de datos. Cuando el componente de selección de imágenes determina, en una de sus iteraciones, que conjunto de imágenes es necesario etiquetar, se crea una nueva tarea (task) en CVAT que incluye dichas imágenes. A continuación, se le permite al usuario etiquetar este conjunto de imágenes directamente desde la interfaz de CVAT.

Una vez creada la tarea, el sistema entra en estado de espera activa (busy waiting), durante el cual consulta de forma continua el estado actual de la tarea en el servidor de CVAT. Durante esta espera, el sistema se encuentra en pausa, hasta que el usuario marca la tarea como completada (completed), momento en el cual el sistema detecta automáticamente este cambio, y reanuda su ejecución, usando las etiquetas proporcionadas por el usuario.

Al utilizar la API expuesta por el servidor, el sistema es capaz de utilizar incluso instancias modificadas de CVAT, independientemente de su configuración.

4.6. Arquitectura

Para concluir este capítulo, se describe la arquitectura implementada en este proyecto, fundamentando las decisiones tomadas y detallando los componentes que la conforman.

Dado que el sistema está compuesto por funcionalidades claramente diferenciadas, siendo estas la selección de imágenes, la generación de datos sintéticos, y el autoetiquetado, se optó por una arquitectura modular. Este enfoque facilita el desarrollo, mantenimiento y escalabilidad del sistema, permitiendo que cada módulo se encargue de una responsabilidad específica.

Además, como cada funcionalidad puede contar con diferentes estrategias o técnicas para su implementación, las mismas son desarrolladas mediante interfaces abstractas para cada módulo. Estas definen funciones que encapsulan el comportamiento esperado del componente, pero sin anclarse a detalles concretos de su implementación. Esto permite extender o modificar el sistema sin alterar la estructura global, ni afectar otros componentes.

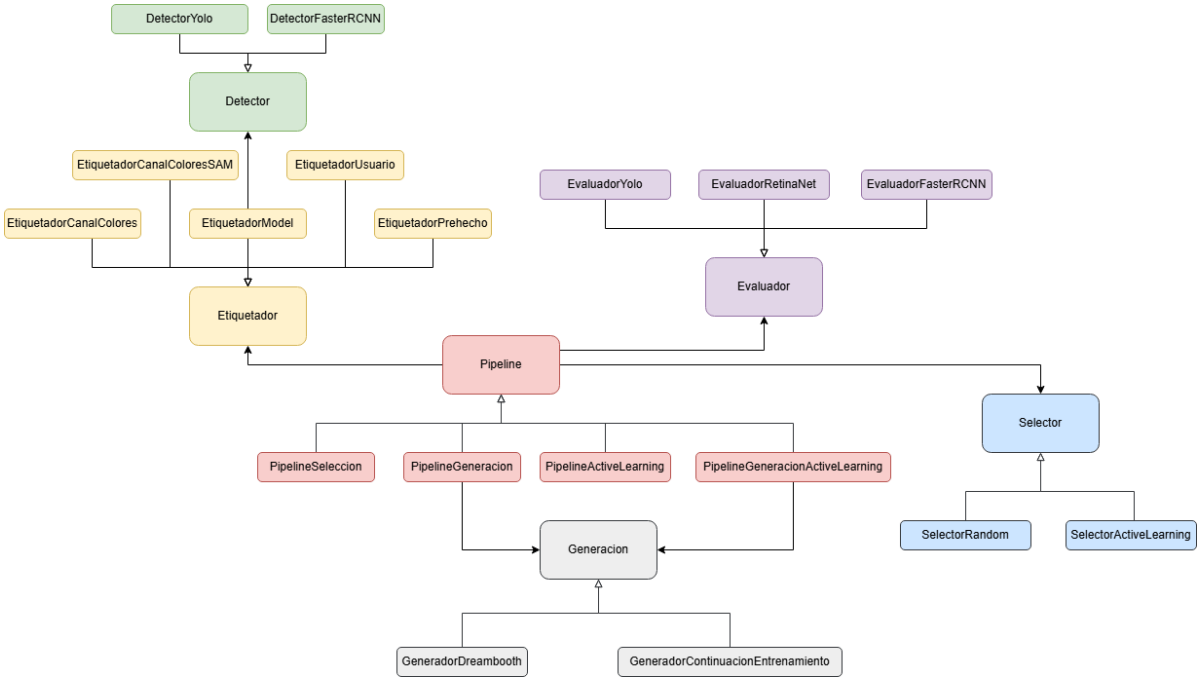


Figura 4.15: Arquitectura propuesta. Las flechas con relleno negro indican dependencia, mientras que las flechas con relleno blanco indican herencia.

En la figura 4.15 se ilustra en alto nivel la arquitectura del sistema. A continuación, se describen los diferentes módulos que la componen, junto con sus respectivas implementaciones.

El módulo *Selector* es responsable de seleccionar un subconjunto de imágenes

a ser etiquetadas, dada una cantidad específica de muestras a etiquetar. Para implementar este módulo, se cuenta con sus variantes:

- **SelectorRandom**: Realiza una selección de forma aleatoria.
- **SelectorActiveLearning**: Emplea aprendizaje activo, mediante PPAL, para elegir las imágenes más informativas.

Por otro lado, el módulo *Generador* tiene como objetivo crear imágenes sintéticas a partir de un conjunto de imágenes reales ya etiquetadas. En este caso, se cuenta con dos implementaciones, ambas explicados en la sección 3.3.

- **GeneradorContinuacionEntrenamiento**: Continua el entrenamiento de un modelo difuso.
- **GeneradorDreambooth**: Emplea Dreambooth para el entrenamiento de un modelo de difuso.

Encargado de asignar etiquetas a las imágenes, dado un conjunto de imágenes, el módulo *Etiquetador* devuelve sus etiquetas correspondientes. Dicho módulo cuenta con las siguientes implementaciones:

- **EtiquetadorPrehecho**: Devuelve etiquetas previamente hechas. Aunque su principal uso es el de la experimentación, es posible utilizarlo si ya se cuenta con un conjunto etiquetado y se quiere hacer uso de la aumentación de datos.
- **EtiquetadorUsuario**: Permite el etiquetado manual mediante la integración con CVAT, detallada en la sección 4.5.
- **EtiquetadorModel**: Emplea un modelo de detección preentrenado junto con filtros de cajas delimitadoras para etiquetar las imágenes.
- **EtiquetadorCanalColores**: Genera las etiquetas mediante el uso de la información contenida en los canales de colores de las imágenes.
- **EtiquetadorCanalColoresSAM**: Un enfoque que toma como base el anterior, utiliza los canales de colores junto con puntos guía para un modelo SAM que genera las etiquetas.

Siendo los últimos tres explicados en 4.4.

Para evaluar la calidad de las imágenes sintéticas y la efectividad del sistema al completo, el módulo *Evaluador* calcula diferentes métricas basándose en los resultados obtenidos. Sus implementaciones están compuestas por:

- **EvaluadorFasterRCNN**, **EvaluadorRetinaNet** y **EvaluadorYolo**: Entrenan modelos de detección y evalúan su desempeño usando las imágenes y sus etiquetas obtenidas.

Por último, el módulo *Pipeline* se encarga orquestar los flujos de trabajo, coordinando y comunicando los módulos según los diferentes procesos explicados en 4.1. Aunque gestiona la interacción entre componentes, no depende de sus implementaciones concretas, lo cual permite reutilizar el mismo pipeline con diferentes variantes de módulos. Se implementan diversas variantes, como:

- **PipelineSeleccion**: Unicamente realiza la selección de imágenes una vez, utilizado principalmente para establecer líneas base.
- **PipelineActiveLearning**: Realiza la selección de imágenes de forma iterativa, principalmente para combinarlo con selectores que hagan uso de aprendizaje activo.
- **PipelineGeneración**: También selecciona imágenes una única vez, pero las utiliza para generar y autoetiquetar imágenes sintéticas a fin de ampliar el resultado final.
- **PipelineGeneracionActiveLearning**: Incorpora tanto la selección iterativa junto con la generación de imágenes autoetiquetadas.

Gracias a las diferentes decisiones tomadas, la arquitectura construida posee las siguientes características:

- **Extensibilidad:** Las clases abstractas exponen una interfaz la cual encapsula cierta funcionalidad, sin imponer restricciones sobre sus implementaciones. Esto permite incorporar nuevas funcionalidades mediante subclases especializadas, sin modificar la arquitectura existente. Por ejemplo, la clase Generador se encarga de crear imágenes sintéticas, y si bien sus implementaciones utilizan modelos difusos, se podría implementar una subclase que utilice métodos cut-and-paste (como en 3.2) sin alterar la estructura del sistema.
- **Mantenibilidad:** La separación clara de responsabilidades entre los módulos facilita la modificación y corrección de componentes individuales, sin afectar el sistema al completo, como por ejemplo, cambios en como se seleccionan o eligen las imágenes.
- **Modularidad:** La arquitectura modular propuesta permite añadir, modificar o reemplazar componentes sin alterar la estructura general, lo cual facilita la experimentación con los diferentes componentes.

Capítulo 5

Experimentación

En este capítulo se presenta el proceso de experimentación y evaluación de los distintos componentes desarrollados, así como del sistema completo. Para esto, y dados los objetivos del trabajo, la experimentación se enfocó en analizar el impacto de cada una de las técnicas utilizadas (tanto de forma individual como al combinarlas) sobre el rendimiento obtenido en los distintos modelos de detección, siendo estos Faster R-CNN (S. Ren y cols., 2015), RetinaNet (Lin, Goyal, Girshick, He, y Dollár, 2017) y YOLOv11m (Jocher y Qiu, 2024), como es desarrollado en la sección 5.3.

En particular, para evaluar la contribución del selector de imágenes implementado mediante aprendizaje activo, se compara el mismo con una implementación basada en selección aleatoria, con el objetivo de cuantificar el beneficio real obtenido al incorporar aprendizaje activo en contextos de menor esfuerzo humano. Por otro lado, en la sección 5.4, la generación de imágenes y su impacto en el resultado final se evalúa mediante un estudio centrado en el proceso de ajuste de los modelos generativos, el preprocesamiento de imágenes, y la proporción entre imágenes reales y generadas que maximicen el aprendizaje del modelo de detección. Además, las imágenes generadas son evaluadas mediante las métricas IS y FID, introducidas anteriormente, así como a partir las opiniones de dos ingenieras agrónomas, las cuales se encuentran trabajando activamente en el proyecto del robot eliminador de malezas, en el cual se engloba este proyecto.

Asimismo, en la sección 5.5 se analiza el desempeño de los diferentes mecanismos de autoetiquetado propuestos, evaluando la métrica mAP@50 y mAP@50:95 (sección 2.1.5) obtenida por cada uno sobre el conjunto compuesto exclusivamente por imágenes generadas, con el fin de identificar qué técnica(s) proveen las mejores etiquetas sin requerir de intervención humana.

Por último, en la sección 5.6 se sintetizan los resultados obtenidos al aplicar las distintas técnicas, cuantificando y comparando las mismas según sus aportes en el rendimiento del sistema utilizando el conjunto de datos CottonWeedDet12 (sección 5.2). La infraestructura utilizada en los anteriores procesos es descrita en la sección 5.1, mientras que las configuraciones de los métodos empleados durante el proceso de experimentación son agrupadas en el anexo E.

5.1. Hardware e infraestructura

Esta sección describe la infraestructura utilizada durante el proyecto, tanto para el desarrollo como para la experimentación de los modelos finales. Se detallan los recursos de hardware empleados y las herramientas que facilitaron la gestión y el análisis de los experimentos.

El proyecto requirió la realización de numerosas pruebas y experimentos, fundamentales para la toma de decisiones a lo largo del desarrollo. Para mantener un entorno colaborativo y ordenado entre los integrantes del equipo, fue crucial documentar las instancias de cada prueba, el código empleado, los parámetros utilizados y los resultados obtenidos. Además, centralizar esta información resultó esencial para posibilitar estudios comparativos entre diferentes experimentos.

Con estos objetivos en mente, se adoptó la herramienta ClearML (ClearML, 2024). ClearML es una plataforma centralizada en la nube para la gestión de experimentos de aprendizaje automático. De la misma se utilizó:

- Versionado y gestión de conjuntos de datos.
- Organización en proyectos y tareas experimentales.
- Registro automático de parámetros y resultados.
- Visualización de gráficas y métricas en tiempo real.

ClearML cuenta con una librería para Python que permite crear tareas y enviar información al servidor directamente desde el código, posibilitando el monitoreo y registro en tiempo real. Esta funcionalidad resultó especialmente útil al trabajar con ClusterUY, uno de los recursos de cómputo utilizados en este proyecto.

Dado que el entrenamiento de redes neuronales profundas demanda una gran capacidad de procesamiento por GPU, fue necesario acceder a equipos con este hardware específico. Para ello, se utilizó ClusterUY que dispone de una amplia variedad de GPUs. Sin embargo, trabajar en ClusterUY presentó desafíos adicionales, como la necesidad de comprender su funcionamiento para lanzar tareas y las restricciones de instalación, ya que los usuarios no cuentan con permisos de instalación directa. Para resolver esto, se empleó Singularity (Sylabs, 2025), una solución de virtualización que permite definir imágenes con todas las dependencias necesarias, esto se utilizó en etapas tempranas del proyecto para generar imágenes con Automatic111 (Automatic, s.f.), que consiste en una interfaz gráfica para generar imágenes a través de Stable Difussion. Una de las limitaciones que tiene Singularity es que modificar una imagen requiere reconstruirla completamente, lo que dificulta el desarrollo.

Estas limitaciones motivaron el uso complementario de una GPU propia (NVIDIA 5070 Ti, acompañada de un CPU Intel i5 12400f y 32 GB de Ram DDR5) durante la etapa de pruebas y desarrollo. Esto permitió una mayor flexibilidad y agilidad, reservando el uso del cluster para experimentos finales y controlados.

En resumen, la combinación de hardware adecuado y herramientas de gestión como ClearML fue clave para asegurar la reproducibilidad y el análisis efectivo de los resultados experimentales en este proyecto.

5.2. Conjunto de datos

Como ya ha sido mencionado, el conjunto de datos utilizado durante la construcción y evaluación de este proyecto es CottonWeedDet12 (Dang, Chen, Lu, y Li, 2023b), debido al elevado número de imágenes que posee, su alta resolución, la variedad de clases con las que cuenta, y que los datos que lo componen son suficientemente claros para ajustar los modelos generativos, pero no tanto como para realizar el proceso de autoetiquetado de forma trivial. Este conjunto está compuesto por 5648 imágenes y 9388 cajas delimitadoras, repartidas entre 12 clases de malezas, que son: *eclipta*, *ipomoea indica*, *eleusine indica*, *sida rhombifolia*, *physalis angulata*, *senna obtusifolia*, *amaranthus palmeri*, *euphorbia maculata*, *portulaca oleracea*, *mollugo verticillata*, *amaranthus tuberculatus*, *ambrosia artemisiifolia*.

Las imágenes fueron tomadas en campos de algodón situados en el sur de los Estados Unidos, durante el periodo de junio a septiembre de 2021, bajo condiciones de luz natural, a un ángulo de 90 grados respecto al suelo, y a una altura aproximada de 40cm. El proceso de etiquetado fue realizado por personal experto en el área.

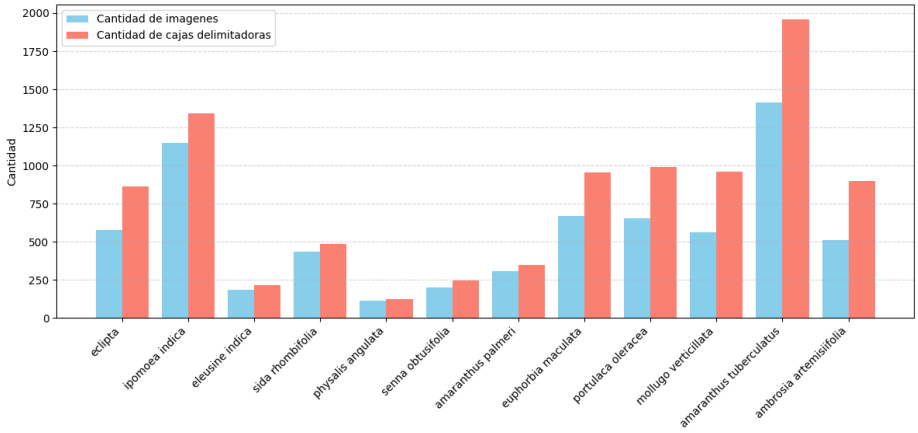


Figura 5.1: Distribución de imágenes y cajas delimitadoras por clase en CottonWeedDet12

En la figura 5.1 se presenta como se distribuyen las imágenes y cajas delimitadoras por cada especie de maleza en el conjunto. Puede observarse que clases como *amaranthus tuberculatus* e *ipomoea indica* presentan una mayor cantidad tanto de imágenes como cajas delimitadoras, mientras que otras, como *physalis angulata* y *senna obtusfolia* se encuentran menos representadas. Este desbalance

debe ser considerado al momento de entrenar y evaluar los diferentes modelos, así como cuando se lean los resultados finales.

Para realizar la experimentación, el conjunto de datos fue dividido en tres particiones:

- **Entrenamiento (80 % del conjunto):** Utilizado para entrenar los modelos de detección y generación empleados.
- **Validación (10 % del conjunto):** Usado para la selección de hiperparámetros y componentes que maximicen el rendimiento final.
- **Evaluación (10 % del conjunto):** Reservado exclusivamente para medir el desempeño final del sistema, sin haber sido utilizado previamente durante el entrenamiento o validación.

5.3. Metodología de evaluación

Con el fin de evaluar los métodos propuestos, a lo largo de este capítulo se experimenta con cada uno de los componentes descritos en el anterior capítulo. En particular, el objetivo final es estudiar el rendimiento obtenido al entrenar modelos de detección de objetos, utilizando como conjunto de entrenamiento las imágenes resultantes de las técnicas desarrolladas.

En consecuencia, en las secciones 5.4.1 y 5.5 se discuten los métodos propuestos sobre el proceso de ajuste de los modelos generativos, y el de autoetiquetado de imágenes respectivamente; donde se experimenta con las componentes desarrolladas, y comparando sus diferentes configuraciones utilizando el conjunto de validación.

De todas formas, se omite en este capítulo una experimentación sobre la implementación de PPAL, ya que dicho análisis es realizado en profundidad por los autores de la técnica; y por ende, en el resto de este capítulo se utilizan las configuraciones propuestas.

Finalmente, en la sección 5.6 las técnicas son comparadas entre sí; eso es, se contrastan los resultados obtenidos al emplear el mecanismo de selección PPAL frente la selección aleatoria, se analiza el impacto en la utilización de imágenes sintéticas auto-etiquetadas, y se valora el desempeño al combinar ambas técnicas. Dado que el módulo de detección de malezas aún no fue desarrollado en el robot eliminador de malezas (donde se encuadra este proyecto), se busca que los beneficios obtenidos sean independientes del modelo de detección de objetos utilizado. Es por ello que se propone evaluar el desempeño utilizando el conjunto evaluación sobre los siguientes tres detectores: Faster R-CNN (S. Ren y cols., 2015), RetinaNet (Lin y cols., 2017) y YOLOv11m (Jocher y Qiu, 2024).

La elección de dichos modelos de detección busca experimentar con modelos cuyas arquitecturas y propósitos sean distintos. En particular, el modelo Faster R-CNN es un modelo de detección de dos etapas (ver sección 2.1.4) el cual se destaca por ser una de las opciones que ofrece mayor precisión. Por el contrario,

los modelos YOLO fueron diseñados para ser veloces, producto de su arquitectura en una etapa y su bajo número de capas; lo que lo hace ideales para aplicaciones de tiempo real, a un costo de una menor precisión, especialmente en objetos de bajo tamaño o poco frecuentes. Un modelo intermedio entre los dos anteriores es RetinaNet, el cual es más veloz que Faster R-CNN debido a su arquitectura en una etapa, pero menos veloz que YOLO debido a capas adicionales, como puede ser la capa Red Piramidal de Funciones (FPN), la cual a costa de un mayor tiempo de cómputo, incrementa las capacidades de RetinaNet en la detección de objetos pequeños, producto de generar mapas de características de diferentes resoluciones. Por lo tanto, la elección de los anteriores modelos abarca tanto modelos enfocados en la precisión, como en la velocidad.

La configuración utilizada en los detectores es desarrollada en la sección 5.6, donde la mayoría no fueron modificadas de sus versiones por defecto, dado que el objetivo es observar el rendimiento de las técnicas propuestas, sin que las mismas sean perturbadas por otros mecanismos.

Dado el elevado tiempo necesario para la realización de cada prueba y los recursos limitados, solamente fue posible realizar una única vez los experimentos, presentando un claro problema en la rigurosidad estadística a la hora de afirmar la veracidad de los resultados.

5.4. Evaluación del proceso de generación de imágenes

Partiendo del proceso de generación de imágenes artificiales (desarrollado en la sección 4.3.3), esta sección se enfoca en la experimentación realizada sobre dicho proceso. Como fue previamente descrito, el proceso de ajuste de los modelos generativos, y la posterior generación de instancias, se trata de un proceso particularmente sensible a la configuración utilizada y a las instancias de entrenamiento. Es por ello que primero se exploran distintos aspectos claves sobre el proceso de ajuste, posteriormente se evalúan las imágenes generadas, y por último se estudian formas de integrarlas con las imágenes reales en el entrenamiento de modelos de detección.

5.4.1. Estudio del proceso de ajuste de los modelos generativos

Dado el proceso de ajuste de los modelos generativos descrito en la sección 4.3.3, se procede a comparar los diferentes métodos descritos, discutiendo sus configuraciones y resultados obtenidos.

Según las características del flujo de datos utilizado, se proponen dos métodos de ajuste, donde el primero se trata directamente de continuar con el entrenamiento del modelo, mientras que el segundo es Dreambooth.

Dado que se está utilizando Stable Diffusion, cada instancia de entrenamiento cuenta tanto con la imagen como de su respectiva descripción. Entonces se

comienza el proceso de experimentación discutiendo sobre la descripción utilizada. En el caso de Dreambooth, el propio método propone el formato de las descripciones, siendo esta *"A [identificador] [clase]"* (ver sección 2.3.4), donde el [identificador] debe ser un token de bajo valor semántico, y [clase] una descripción del tipo de concepto que se busca aprender a generar.

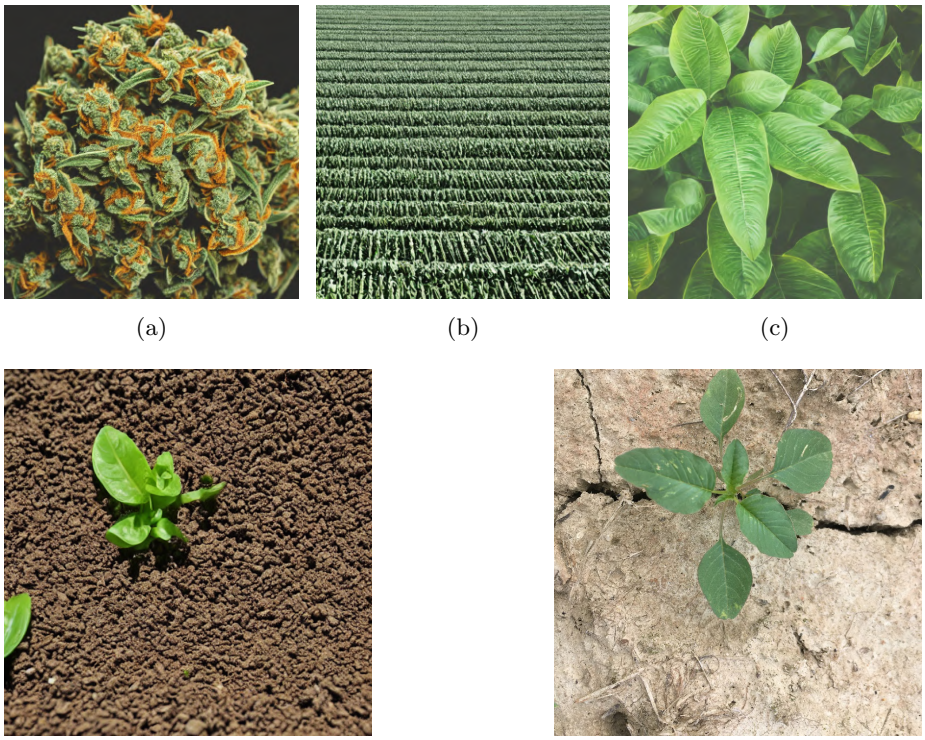
En el caso del otro método, no está especificado cuál debe ser el formato de la descripción. Por ejemplo, en el trabajo Moreno y cols. (2023) utilizan únicamente el nombre de la especie de la maleza, mientras que en Deng y Lu (2025) utilizan "high details, plants in the field".

Con el fin de utilizar el conocimiento codificado en [clase], explorar técnicas de control, y la posibilidad de ajustar un único modelo para todas las especies de malezas, se propone el siguiente formato de descripción: *"photo of [especie] [clase] in the [posición_x] [posición_y]"*. Donde [especie] es el nombre científico de la especie de la maleza (según los descritos en la sección 5.2), [clase] es el mismo término utilizado en Dreambooth, y [posición_x] [posición_y] describe la posición de la maleza en la imagen; la información de posición intenta que el modelo distinga de mejor forma que parte de la imagen contiene la maleza.

Además, para escoger el concepto a utilizar en [clase], se debe buscar alguno que genere una imagen visualmente similar a la que se busca generar, teniendo en cuenta que se busca generar imágenes de: una planta pequeña, única en la imagen, donde alrededor solo se encuentre el suelo donde yace, y la perspectiva sea perpendicular al suelo (imagen de referencia en figura 5.2e). En consecuencia se exploraron los diferentes conceptos buscando cuál podría ser el más próximo, los mismos son: *"weed"* (Figura 5.2a), *"crop"* (Figura 5.2b), *"plant"* (Figura 5.2c), y *"seedling"* (Figura 5.2d). Observar que el concepto *"seedling"* es el que presenta mayor similitud con la imagen de referencia, y por lo tanto se determinó hacer uso del mismo en las descripciones.

Una vez definidos los formatos de las descripciones asociadas a las imágenes de entrenamiento, se puede proceder con el ajuste de los modelos, y en particular, a su comparación. Es importante primero observar que los métodos de continuación del entrenamiento, y Dreambooth son similares, es más, de hecho (como fue explicado en la sección 2.3.4), Dreambooth extiende a la continuación del entrenamiento mediante dos propuestas: el formato de las descripción y un regularizador. Por lo tanto, se intuye que es más conveniente el uso de Dreambooth como método de ajuste. Con el fin de validar la anterior suposición se presentan resultados de haber realizado ajustes con dichos métodos. Se experimenta nuevamente sobre la especie *"amaranthus palmer"* (Imagen de referencia en figura 5.3a), utilizando en todos los casos el mismo tamaño de lote (batch), taza de aprendizaje (learning rate), conjunto de entrenamiento (32 imágenes), entre otros. Si los modelos son ajustados vía continuación del entrenamiento, se obtiene resultados como los ilustrados en la figura 5.3b, al ajustarlo por 11.200 pasos (350 por imagen de entrenamiento), mientras que si se extiende a 25.600 pasos (800 por imagen) se obtienen resultados como los de la figura 5.3c.

Se observa en el primero de los casos, las nervaduras de las hojas, como el ángulo de la cámara comienzan a tener cierto parecido a la imagen de referencia, pero aun distando mucho en la morfología esperada en la especie de la maleza.



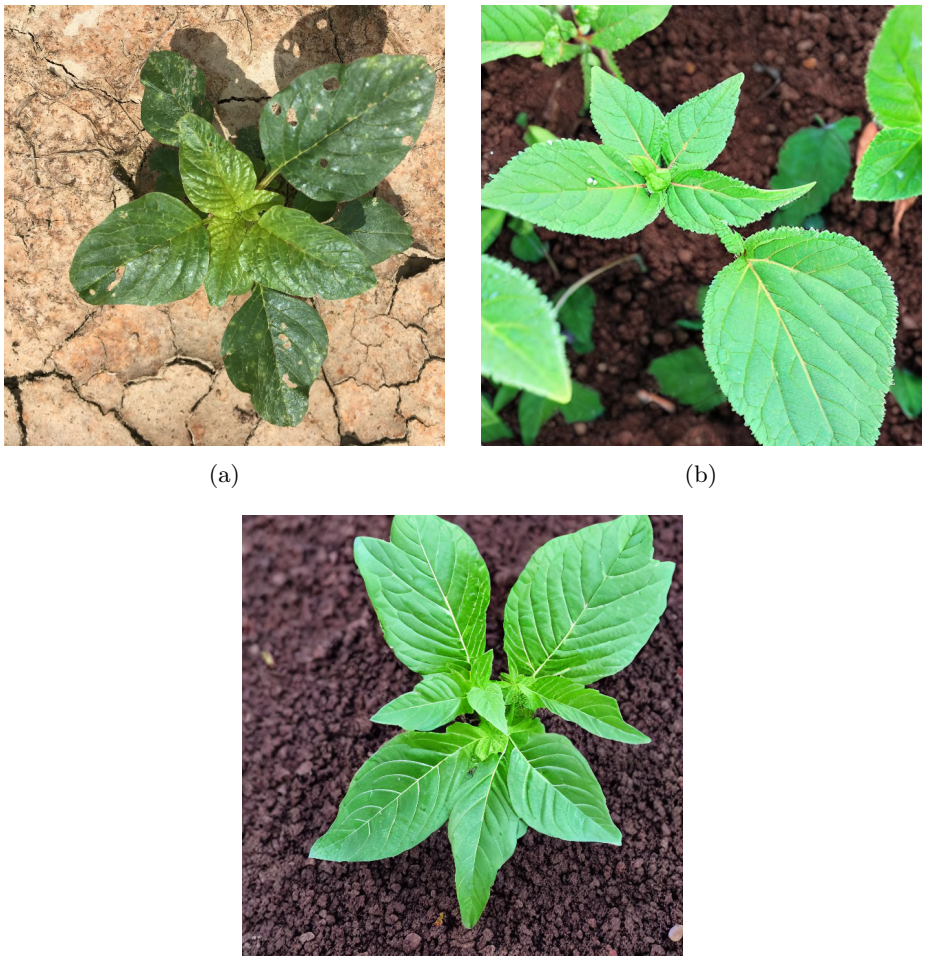
(a) (b) (c)

(d) (e)

Figura 5.2: (a), (b), (c) y (d) son imágenes generadas con Stable Diffusion v1.5 bajo la instrucción *"a photo of [class]"*, donde [class] es *"weed"*, *"crop"*, *"plant"* y *"seedling"* respectivamente. (e) es una imagen de referencia de la especie *"amaranthus palmer"*. Se observa que la mayor similitud con (e) es obtenida por (d).

En el caso del entrenamiento más prolongado, se destacan que la morfología tiene un mayor parecido, sobre todo en lo que refiere a la forma, tamaño y densidad de las hojas en las malezas, de todas formas siguen difiriendo significativamente en aspectos como el color y las imperfecciones de las hojas (manchas, agujeros, rupturas). Los cuales, pueden provocar variaciones en el entrenamiento de un modelo de detección.

En el caso del ajuste con Dreambooth, el cambio en el formato de la descripciones acelera significativamente la adopción del nuevo concepto, lo cual, según sus autores, es principalmente debido al uso del token identificador de bajo valor semántico; por lo cual sustituir el concepto que representa resulta más veloz en comparación con tokens con elevado valor semántico. Siguiendo las instrucciones adjuntas a la implementación de Dreambooth utilizada, el token "sks" es utilizado como identificador. Se observa el anterior comportamiento en la figura 5.4a, producto de haber ajustado el modelo durante 9.600 pasos (300



(a) (b)

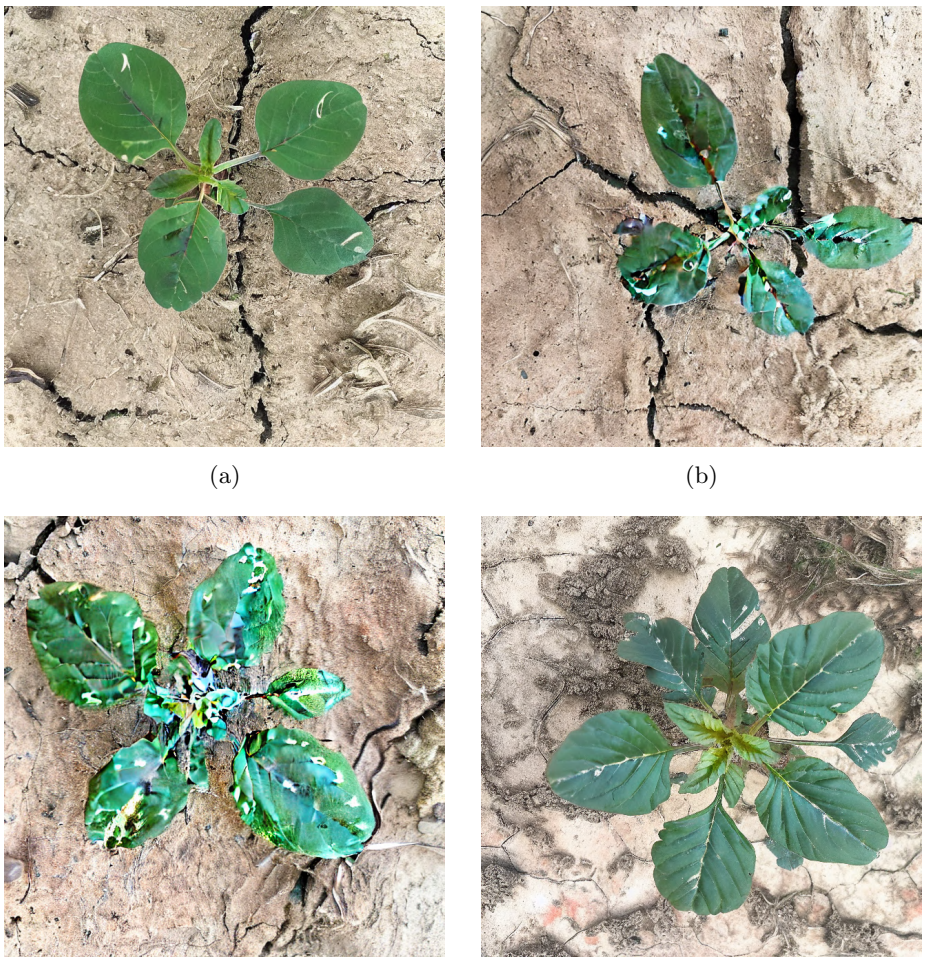
(c)

Figura 5.3: (a) es una imagen de referencia de la especie *"amaranthus palmer"*; la cual presenta notorias imperfecciones, como manchas, agujeros o rupturas de las hojas. (b) y (c) son imágenes generadas con modelos Stable Diffusion v1.5 ajustado con continuación de entrenamiento durante 11.200 y 25.600 pasos respectivamente

pasos por imagen de entrenamiento), unicamente ajustando los pesos internos del modelo de difusión y sin utilizar el regularizador. Notar también que con un menor número de pasos (en comparación con continuación del entrenamiento), se obtiene una mayor similitud con la imagen de referencia (figura 5.3a).

Según los autores de Dreambooth, se suelen obtener incluso mejores resultados si se ajustan todas las capas, incluso las del codificador de texto (text encoder). Esto último debe ser realizado con precaución, ya que en tal caso, el

modelo tiende a ser particularmente sensible al sobreajuste. Además se suelen presentar problemas como deriva lingüística (language drift), donde el modelo pierde progresivamente conocimiento sintáctico y semántico del lenguaje. Por ejemplo, si se repite el proceso de ajuste antes realizado, pero ahora también es ajustando el codificador de texto, se obtiene resultados como en la figura 5.4b.



(a) (b)

(c) (d)

Figura 5.4: (a) es producto de ajustar un modelo Stable Diffusion v1.5 con Dreambooth durante 9.600 pasos, sin entrenar el codificador de texto, y utilizando imágenes de la especie *"amaranthus palmer"* como datos de entrenamiento. (b) es análoga a (a), pero entrenando el codificador de texto. (c) es análoga a (b), pero utilizando el regularizador de Preservación Previa Específica de la Clase con 100 ejemplos por imagen de entrenamiento. La figura (d) es análoga a la (c) pero utilizando 5.760 pasos.

Para estos casos, los autores de Dreambooth proponen la Preservación Previa Específica de la Clase, la cual tiene como principal objetivo buscar que el modelo no pierda su conocimiento previo sobre el concepto identificado por "[clase]", con la intención de preservar diversidad en la generación del modelo. Este regularizador también ayuda a prevenir el sobreajuste, aunque no es su principal objetivo. En la figura 5.4c se muestra el resultado del uso de dicho regularizador, donde por cada imagen de entrenamiento se utilizan 100 de ejemplo. Se observa que, en este caso, el uso del regularizador no disminuye el sobreajuste. Notar que si se ajusta el modelo durante un menor número de épocas (evitando el sobreajuste), efectivamente ajustar el codificador de texto resulta en que las imágenes generadas presenten una mayor fidelidad con respecto a las principales características de la especie que se busca ajustar, como se observa en la figura 5.4d, junto con la presencia de detalles adicionales como rupturas y manchas en las hojas de las malezas generadas.

Dado que las imágenes serán utilizadas para entrenar detectores, la fidelidad de los conceptos y la diversidad resultan primordiales; en consecuencia se determina que durante el ajuste se entrene el codificador de texto y se emplee el regularizador de Preservación Previa Específica de la Clase, pero resulta necesario estudiar el número de épocas.

Partiendo de lo sugerido en los trabajos Packer (2023) y Suraj Patil, Pedro Cuenca y Valentine Kozin (2022), se recomienda el uso de una tasa de aprendizaje de e^{-6}, 200 épocas, ajustar el codificador de texto, y aproximadamente de 200 ejemplos por imagen de entrenamiento para el regularizador de Preservación Previa Específica de la Clase. Luego de experimentar con dichas configuraciones, se determinó adoptarlas en el ajuste de modelos generativos, menos el número de épocas. Es usual, luego de finalizado el ajuste de un modelo generativo, que el mismo se encuentre sobreajustado, y por ende, vía revisión manual, se escoja un estado intermedio el cual no presente signos de sobreajuste. Debido a las restricciones en cuanto al esfuerzo humano de este proyecto, el proceso de ajuste no dispone de la revisión manual de cada especie por parte del usuario. Además, se desconoce, en el dominio de la generación de imágenes, la existencia de métricas y/o métodos que detecten el sobreajuste, y permitan detener el ajuste de forma automática.

Entonces, con el fin de abordar la anterior problemática, se busca definir un único número de épocas para todas las especies. En primer lugar, buscando atenuar la diferencias al ajustar los modelos generativos, se limitó el número de imágenes de entrenamiento entre 7 y 32 instancias por especie; ya que mientras mayor sea el número de instancias de entrenamiento, menor el número de épocas que se deberían utilizar antes de que se manifieste el sobreajuste.

Seguidamente, vía revisión manual se escogió entre modelos ajustados durante 150 a 240 épocas (en la figura 5.5 se visualiza sobre la especie *"amaranthus palmer"*); se observa el sobreajuste es claro, con 240 épocas e incipiente con 210 épocas. Con 180 épocas se observa un mayor nivel de detalle en las hojas que con 150. Por lo tanto, se escogió entrenar el modelo generativo de cada especie con 180 épocas, y el resto de las configuraciones ya mencionadas.

Producto de la fase de experimentación, se identificó que cuando algunas

Figura 5.5: Imágenes generadas al ajustar todos los parámetros y codificador de texto de un modelo Stable Diffusion v1.5 vía Dreambooth, utilizando el regularizador de Preservación Previa Específica de la Clase. Realizando dicho ajuste por 150, 180, 210 y 240 épocas respectivamente

pocas imágenes con la que es ajustado el modelo generativo presentan defectos como desbalance de blancos, de contraste, o de desenfoque, el modelo generativo tiende a generar con frecuencia imágenes con dichas características; el anterior caso es mostrado en la figura 4.8 del anterior capítulo. Es por ello que para evitar tales casos con los modelos generativos ajustados, en la sección 4.3.2 se proponen diferentes filtros y ajustes de color sobre las instancias del conjunto de entrenamiento de los modelos generativos.

De todas formas, estos filtros no evitan el caso de las oclusiones con objetos que no sean parte de las malezas etiquetadas, como por ejemplo otras plantas cercas, rocas, o troncos que obstaculicen la visual de la maleza, y en consecuencia afecte la capacidad del modelo en reconocer la maleza en la imagen. Es por ello que en el resto de las pruebas desarrolladas en esta sección, los modelos generativos son ajustados con imágenes donde se garantiza de forma manual que no haya oclusiones graves, o desenfoques no detectados por el filtro pertinente.

También es relevante observar el nivel de condicionamiento que se tiene sobre el modelo generativo una vez ajustado; esto es, que tanto se puede controlar la generación de la imagen según la descripción dada a la hora de generar la imagen. Esto se desarrolla en detalle en el anexo C considerando que no resulta crítico con los componentes actualmente utilizados, pero si es fundamental pensando en la viabilidad del trabajo futuro relacionado con el autoetiquetado vía el modelo generativo; en particular sobre técnicas como DiffuGen.

Se presentan en las figuras 5.6 y 5.7 ejemplos de imágenes generadas y reales para cada especie. Las imágenes son generadas vía el modelo difuso ajustado para su respectiva especie, utilizando la configuración previamente descrita. Notar que la primera columna contiene una imagen real de la maleza, con el fin de ser usada como referencia al contrastar su similitud con las generadas.

Se destaca que las imágenes generadas cumplen a simple vista con las principales características de cada especie, respetando la morfología, cantidad y distribución de las hojas. También se observa una correspondencia en cuanto al color de la maleza, aunque tienden a presentar un mayor brillo y saturación en

las malezas de las imágenes generadas.

Considerando el hecho que las malezas conforman gran parte de la imagen, se observa una gran diversidad en las diferentes instancias, tomando en cuenta la variedad de disposiciones adoptadas por las hojas, las distintas posiciones donde se sitúa la maleza en la imagen, los diferentes tipos de suelos, y elementos que estos contienen, como ramas, piedras o fisuras en la tierra. De todas formas la diversidad resulta menor que en otros trabajos como Deng y Lu (2025), donde las malezas al ocupar un menor tamaño en la imagen, dejan una mayor superficie de la imagen para el suelo, donde el modelo generativo presenta mejor diversidad al generarlo, dado su entrenamiento base.

5.4.2. Evaluación de imágenes generadas

Para evaluar la calidad de las imágenes generadas por los modelos difusos entrenados, se llevaron a cabo diferentes evaluaciones, tanto cuantitativas, mediante métricas específicas para este cometido, como cualitativas, en colaboración con expertas en el dominio.

FID e IS

Se utilizaron las dos métricas explicadas en 2.3.5, más específicamente, Distancia de Inception de Frechet (FID) y el Puntaje Inception (IS), ambas ampliamente empleadas en la mayoría de trabajos relacionados con la generación de imágenes sintéticas.

La evaluación fue realizada de forma individual para cada una de las clases de malezas presente en el conjunto de datos *CottonWeedDet12*. Para cada clase, se generó una cantidad de imágenes similar al número de instancias reales de cajas delimitadoras disponibles en el conjunto. De esta forma, se obtuvo, para cada tipo de maleza, dos conjuntos diferentes: uno compuesto por imágenes reales, y otro por imágenes generadas. A partir de estos conjuntos, se calcularon los valores de FID e IS; permitiendo evaluar la similitud entre las distribuciones reales y generadas, así como la diversidad dentro de cada conjunto de imágenes generadas.

En la figura 5.8 se detallan los resultados obtenidos por cada clase de maleza. En general, estos números no fueron satisfactorios, con valores de FID que van desde 60 hasta 170, y valores de IS entre 1,5 y 2. Estos valores indican que las imágenes generadas presentan una baja similitud con las reales y una diversidad limitada.

De todas formas, los resultados obtenidos para el FID no son concluyentes. En el trabajo original donde se presenta la métrica (Heusel y cols., 2017) se utilizaron 50.000 imágenes generadas, y 200.000 imágenes reales para realizar el calculo. Si bien no existe un umbral objetivo para poder calcular el FID de manera correcta, por lo general se recomienda contar con un mínimo de 10.000 imágenes reales. Por otro lado, para este trabajo, la especie de maleza de la cual se pueden encontrar más instancias en el conjunto de datos CottonWeedDet12,

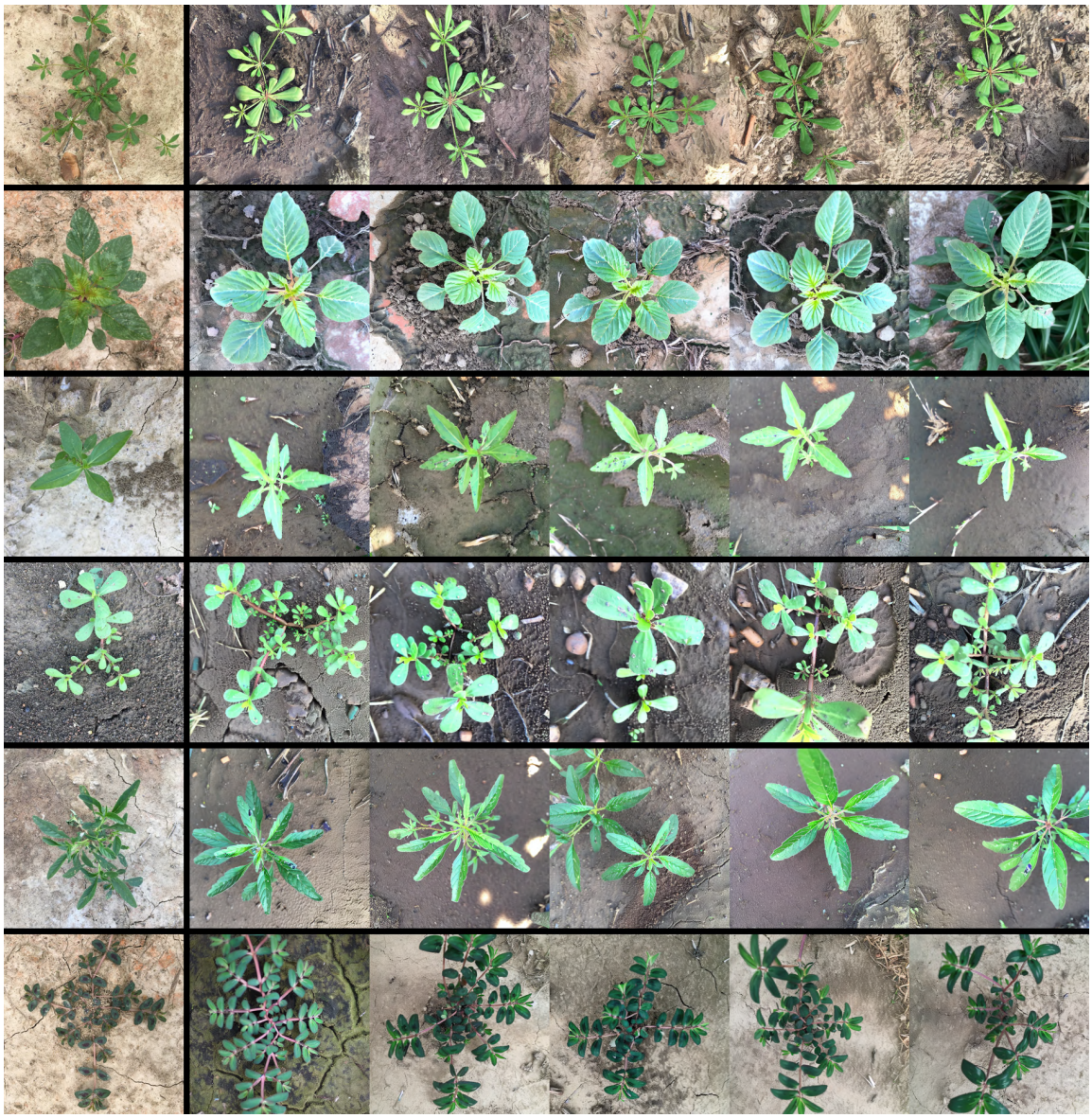


Figura 5.6: La columna inicial de la figura presenta imágenes reales de cada especie, mientras que las cinco columnas subsiguientes muestran imágenes generadas por el modelo difuso, ajustado a su especie correspondiente. Las filas se corresponden con las siguientes especies: *mollugo verticillata*, *amaranthus palmeri*, *eclipta*, *portulaca oleracea*, *amaranthus tuberculatus* y *euphorbia maculata* respectivamente.

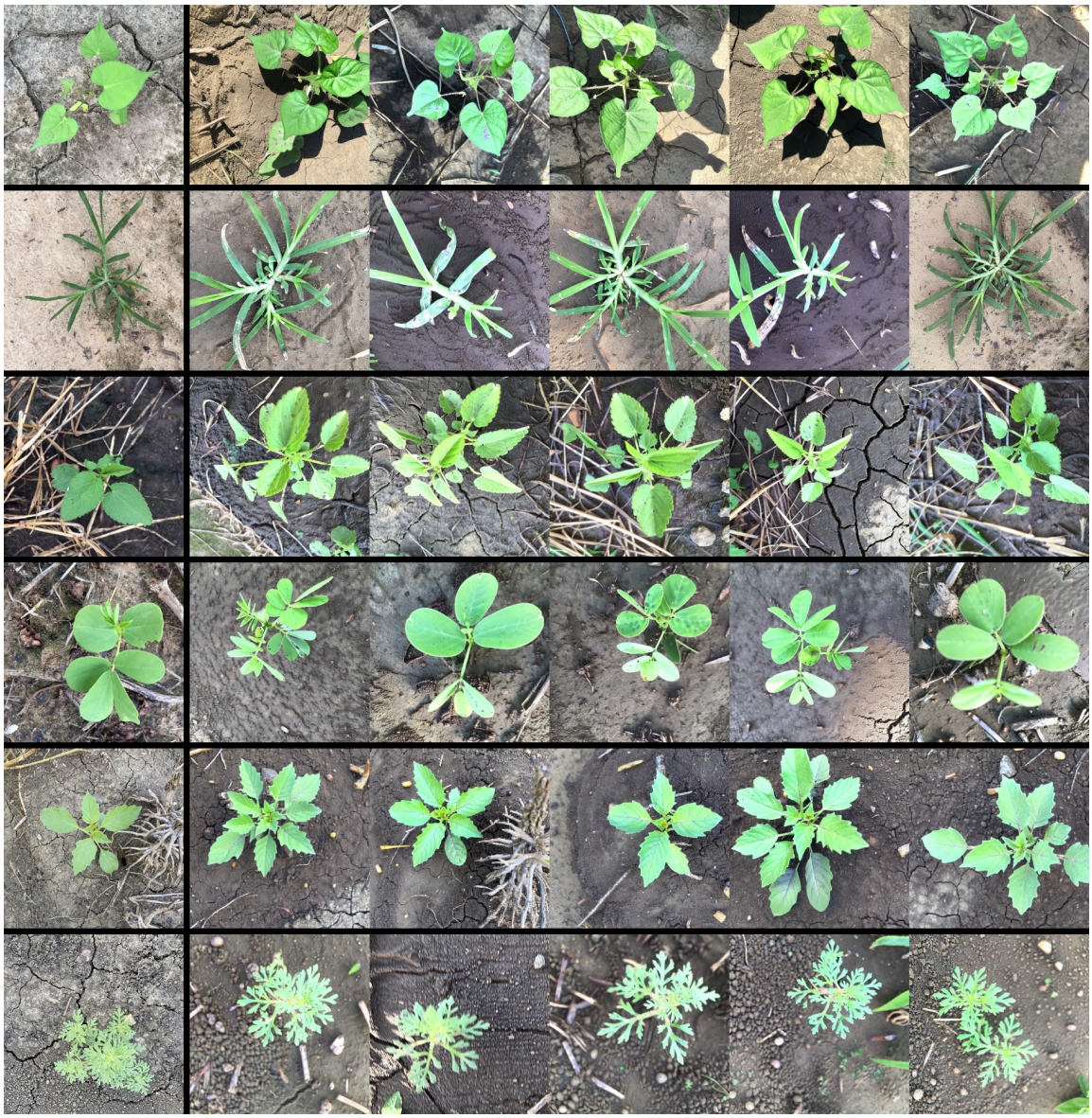


Figura 5.7: La columna inicial de la figura presenta imágenes reales de cada especie, mientras que las cinco columnas subsiguientes muestran imágenes generadas por el modelo difuso, ajustado a su especie correspondiente. Las filas se corresponden con las siguientes especies: *ipomoea indica*, *eleusine indica*, *sida rhombifolia*, *senna obtusifolia*, *physalis angulata* y *ambrosia artemisiifolia* respectivamente.

amaranthus tuberculatus, cuenta con tan solo 1.959 cajas delimitadoras, mientras que la que menos instancias posee, *physalis angulata*, tan solo tiene 123.

Por lo tanto, en un intento de obtener valores más acertados, se combinaron todas las imágenes reales en un único conjunto de datos, y se calculó la métrica al compararlo contra el conjunto completo de las imágenes generadas. En este caso, tomando en cuenta que la cantidad de muestras reales sigue siendo relativamente bajo, el FID obtenido se reduce a 43,3, indicando los valores de la figura 5.8 no son representativos.

Además, como se observará más adelante en la sección 5.4.4, esta afirmación es reforzada al demostrar resultados positivos al entrenar modelos de detección unicamente con imágenes generadas, lo que aparenta mostrar tanto fidelidad como diversidad. De igual modo, en la sección 5.4.3, se muestra como las imágenes generadas presentan claras mejoras en los resultados de detección al hacerlas parte del conjunto de entrenamiento de los modelos. Por otro lado, en relación al IS, estos valores se encuentran ligeramente por debajo de los obtenidos en Deng y Lu (2025), los cuales usaron un conjunto de datos similar al que se usa en este proyecto, con la diferencia de que, en este caso, se aplica el método de recorte de imágenes explicado en el anexo A, por lo que las imágenes generadas tienden a mostrar la maleza muy cercana a la cámara, ocupando la mayor parte de la imagen, lo que deja poco margen para la diversidad entre instancias, siendo esta una posible causa de la disminución de la métrica IS.

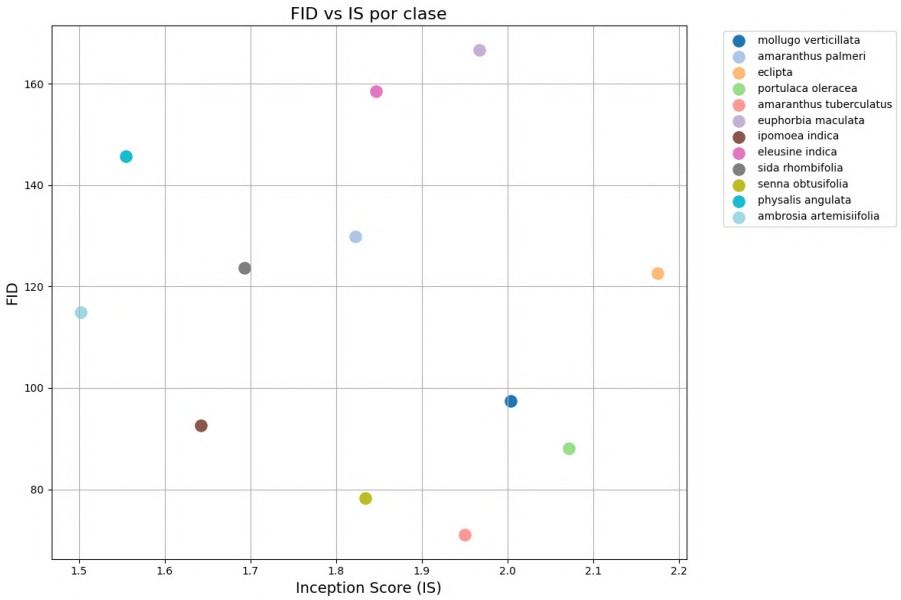


Figura 5.8: Métricas obtenidas al evaluar cada conjunto de imágenes falsas. En el eje X se detalla el IS (más alto significa mejor) mientras que en el eje Y se detalla el FID (más bajo significa mejor). Cada punto representa un tipo de maleza diferente.

Reunión con ingenieras agrónomas

Además de la evaluación mediante métricas, se organizó una instancia de evaluación cualitativa con un grupo de dos ingenieras agrónomas que se encuentran participando activamente en el proyecto del robot de detección y eliminación de malezas.

Durante la reunión, se les presentaron imágenes generadas correspondientes a diferentes clases, en algunos casos sin revelar si eran reales o sintéticas.

En la mayoría de los casos, las ingenieras coincidieron en que las imágenes generadas presentaban una morfología muy similar a la de los ejemplos reales, especialmente en características importantes del grupo al que pertenecen. Además, en la mayoría de los casos, visualizando unicamente las imágenes generadas, eran capaces de reconocer la especie con la que fue ajustado su modelo generativo correspondiente, mostrando así un suficiente nivel de detalle para tareas de reconocimiento.

Si bien esto no da ningún resultado numérico con el cual hacer comparaciones, si refuerza la validez de las imágenes generadas como complemento para los modelos de detección.

5.4.3. Integración de imágenes generadas al conjunto de datos

En esta sección se estudia como integrar las imágenes generadas junto con las imágenes de entrada del flujo de datos, previo al entrenamiento de los detectores de objetos. Notar que en esta fase del flujo de datos ya se cuenta con los modelos de difusión ajustados para cada especie del conjunto de datos. Existen varias posibilidades sobre cuantas imágenes de cada especie deberían ser generadas, específicamente en este trabajo con el fin de no generar disparidades entre las diferentes especies, se propone generar el mismo número de instancias para cada una; aunque otros enfoques pueden ser adoptados, como por ejemplo, balancear el número de imágenes por especie, generando un mayor número de instancias en las especies con escaso número de ejemplares.

En lo que respecta al número de imágenes artificiales, de igual forma que en Deng y Lu (2025) se estudian diferentes proporciones de imágenes sintéticas en relación al número de instancias del conjunto de datos de entrenamiento. Entonces se propone comparar los rendimientos utilizando 0,5, 1 y 2 de proporción, es decir, que la cantidad de imágenes generadas sea la mitad del número de imágenes del conjunto de datos, la misma cantidad, o el doble.

En relación con el proceso de experimentación, se utiliza un conjunto de entrenamiento de 500 imágenes base, y por lo tanto, son combinadas con 250 (proporción de 0,5), 500 (proporción de 1) y 1.000 (proporción de 2) imágenes generadas. Los resultados son presentados en la tabla 5.1, donde se observa que se obtienen mejores resultados utilizando una proporción de 2, aunque en la métrica mAP@50:95 no es concluyente. De todas formas se observa que los resultados son positivos siempre que las imágenes sintéticas son agregadas.

En el resto de la experimentación realizada en este capítulo, en todos los

Conjunto de entrenamiento	mAP@50	mAP@50:95
500 reales	0,851	0,788
500 reales + 250 generadas	0,878	0,817
500 reales + 500 generadas	0,877	**0,818**
500 reales + 1000 generadas	**0,883**	0,817

Tabla 5.1: Rendimiento del modelo YOLOv11m en mAP@50 y mAP@50:95 al variar el número de imágenes generadas agregadas al conjunto de entrenamiento formado por 500 imagen reales. Fijando el conjunto de imágenes reales, se prueba sin agregar ninguna imagen artificial, agregando 250, 500 y 1.000.

casos, se integran el doble de imágenes sintéticas frente a las escogidas por el componente de selección.

5.4.4. Impacto en el entrenamiento de modelos de detección

Finalmente se estudia el impacto que tiene las imágenes artificialmente generadas al entrenar modelos de detección. Existen múltiples formas de realizar este estudio, una de ellas sería entrenar detectores combinando imágenes reales con las imágenes falsas, de igual forma que se realizó en la anterior sección al variar la proporción de imágenes reales y generadas; mostrando mejoras en el desempeño los detectores.

En el caso particular de esta sección, se propone una prueba considerablemente más desafiante, la cual consiste en entrenar detectores unicamente con las imágenes sintéticas, y evaluarlos en el conjunto de evaluación, compuesto exclusivamente por imágenes reales. Notar que el grado de dificultad de esta prueba es considerablemente alto, ya que para tener éxito las imágenes generadas deben ser lo suficientemente fieles a sus correspondientes especies, como para que un modelo de detección aprendiendo exclusivamente de ellas, sea capaz de reconocerlas en imágenes autenticas; al mismo tiempo que deben ser diversas, ya que de lo contrario el detector no será capaz de generalizar el concepto, resultando en una pobre capacidad de detección.

En el trabajo de Deng y Lu (2025) se utiliza un conjunto de datos que comparte una gran cantidad de las imágenes del conjunto de datos utilizado en este trabajo, específicamente nueve de las doce especies aquí utilizadas también pertenecen al conjunto de datos del mencionado antecedente. Esto lo convierte en una gran oportunidad para poder comparar el desempeño de las imágenes generadas al entrenar detectores siguiendo las técnicas propuestas en ambos trabajos. En la tabla 5.2 son presentados los resultados de haber entrenado un modelo YOLOv8l durante 200.000 pasos (100 épocas) en este trabajo, y durante 300.000 pasos (24 épocas) en Deng y Lu (2025). Se debe considerar que en este trabajo el máximo número de imágenes etiquetadas manualmente es 1.000 (como se detalla en la sección 5.6), entonces al usar una proporción del doble de imágenes generadas que reales (como se estableció en la sección 5.4.3)

se cuentan con 2.000 imágenes generadas. Mientras que en el antecedente con el que se compara, cuentan con 12.800 imágenes generadas, ya que se utiliza una proporción de 1,5, y disponen de 8.500 imágenes reales etiquetadas. Es por ello que con el fin de evitar sobreajustes, en este trabajo se determina entrenar durante un menor número de pasos.

Especie	**Nuestro trabajo**	**Trabajo de referencia**
mollugo verticillata	**0,364** (**0,316**)	0,350 (0,284)
amaranthus palmeri	**0,572** (**0,530**)	0,301 (0,271)
eclipta	**0,465** (**0,412**)	0,346 (0,309)
portulaca oleracea	**0,509** (**0,396**)	0,287 (0,239)
amaranthus tuberculatus	0,553 (0,396)	**0,660** (**0,583**)
euphorbia maculata	0,470 (0,372)	**0,685** (**0,552**)
ipomoea indica	**0,780** (**0,620**)	0,656 (0,500)
eleusine indica	**0,808** (**0,521**)	0,627 (0,466)
sida rhombifolia	0,573 (0,510)	-
senna obtusifolia	0,874 (0,841)	-
physalis angulata	0,089 (0,081)	-
ambrosia artemisiifolia	0,397 (0,345)	**0,770** (**0,621**)
Total	**0,538** (**0,451**)	0,521 (0,425)

Tabla 5.2: Comparación entre las imágenes generadas en este trabajo y en Deng y Lu (2025) al entrenar un modelo de detección con el modelo YOLOv8l con 200.000 pasos y 300.000 pasos respectivamente. En la columna "Nuestro trabajo" se muestran los resultados obtenidos para la especie especificada en términos de AP@50 y entre paréntesis la AP@50:95; mientras para la columna "Trabajo de referencia" muestra los resultados obtenidos en el trabajo con el que se compara para aquellas especies que se tiene en común entre ambos conjuntos de datos. En la ultima fila se encuentra las mAP@50 y mAP@50:95 de cada trabajo.

Observar que los resultados mostrados marcan un similar desempeño en general entre ambas técnicas, donde en la ejecución realizada, el trabajo aquí presentado presenta una ventaja de 0,017 de mAP@50 y 0,026 de mAP@50:95, lo que no resulta concluyente para afirmar la superioridad del método de un trabajo sobre otro. Aunque se destaca que se obtienen mejores resultados en seis de las nueves especies comparadas.

Notar que las especies donde se obtiene un rendimiento elevado son comunes en ambos trabajos, aparentando que las características morfológicas propias de dichas especies son comúnmente más simples de reproducir. Aunque la especie *ambrosia artemisiifolia* resulta una excepción a la regla, donde en este trabajo su desempeño es sensiblemente menor al comparado. Se hipotetiza que el motivo sea la gran diferencia en la superficie ocupada por la maleza en las imágenes generadas en comparación con las reales; esto es producto de la técnica de recorte con la que fue generado el conjunto de entrenamiento con el que se ajustaron los modelos generativos.

Dado los resultados obtenidos, se observa que la especie con mayores dificultades es sin lugar a duda *physalis angulata*, donde el detector confunde sus instancias por la especie *senna obtusifolia*, la cual comparte varias de las características morfologías. Ademas se observa en la figura 5.7, que características de las hojas de la especie *physalis angulata* tienden a estar sobredimensionadas, como el dentado en los bordes de las hojas.

Por último, aunque los resultados en Deng y Lu (2025) son similares, en dicho antecedente la cantidad de trabajo manual es considerablemente mayor al trabajo aquí presentado. Por ejemplo las imágenes con la que se ajustan los modelos generativos son escogidas manualmente, de igual forma que se filtran las imágenes creadas por los modelos generativos. Ademas que las etiquetadas de las imágenes sintéticas son también y verificadas manualmente. Esto muestra que el proceso sin intervención humana propuesto en este trabajo, no aparenta incurrir en perdidas en lo que respecta al rendimiento de los modelos de detección de objetos al ser entrenados junto con imágenes sintéticas generadas.

5.5. Evaluación de mecanismos de autoetiquetado

Para finalizar la experimentación con los componentes individuales, es evaluado el último de estos que interviene en el proceso del flujo de datos: el componente de autoetiquetado.

Con el fin de comparar los métodos, se preparó un conjunto de datos compuesto únicamente por 887 imágenes sintéticas, las cuales son a las que se enfrentan los autoetiquetadores, y se etiquetaron de forma manual con el objetivo de comparar las mAP obtenida al comparar las etiquetas reales con las etiquetas devueltas por los autoetiquetadores. Luego, este conjunto de datos fue etiquetado con cada uno de los distintos métodos construidos.

En particular, el método basado en modelos de detección utiliza las mismas imágenes reales seleccionadas y posteriormente etiquetadas por el usuario para su entrenamiento. Por lo tanto, su desempeño puede variar significativamente en dependencia de la cantidad de imágenes reales etiquetadas, así como el método de selección utilizado, ya sea aleatorio o mediante PPAL. Por esta razón, es necesario evaluar dicho método al utilizar diferentes cantidades de imágenes reales para su entrenamiento, y seleccionadas de diferente forma.

Por otro lado, los métodos que hacen uso de los canales de colores no se ven afectados por la cantidad de imágenes reales etiquetadas, puesto que los mismos no requieren ser entrenados; por lo tanto, la mAP se calcula una única vez, sin variar esta cantidad. En las figuras 5.9 y 5.10 se presentan los resultados obtenidos.

En general, al comparar entre los modelos de detección, los resultados son los esperados; al aumentar la cantidad de imágenes con las cuales fue entrenado el modelo, o cambiar el método de selección de aleatorio a PPAL, tanto la mAP@50 como mAP@50:95 aumentan.

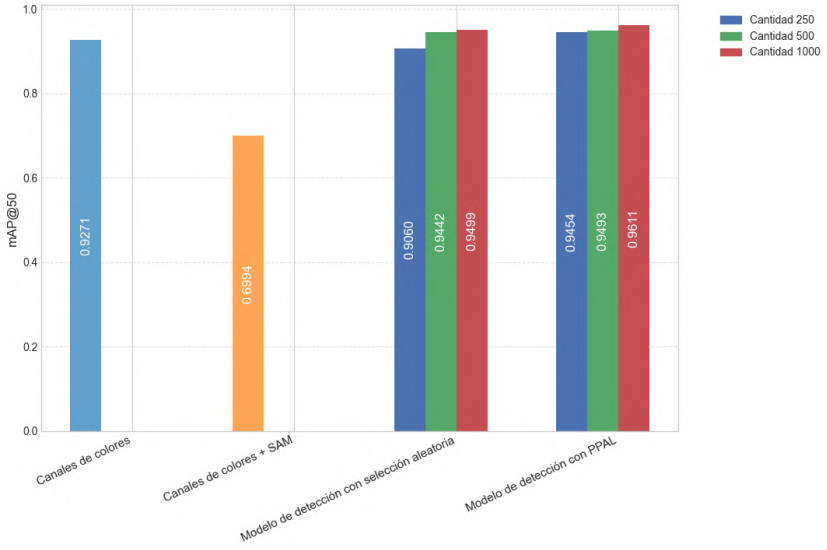


Figura 5.9: Resultados de mAP@50 obtenidos por cada método de autoetiquetado al etiquetar imágenes sintéticas generadas por un modelo generativo entrenado mediante Dreambooth. Los modelos de detección fueron entrenados con 250, 500 y 1000 imágenes.

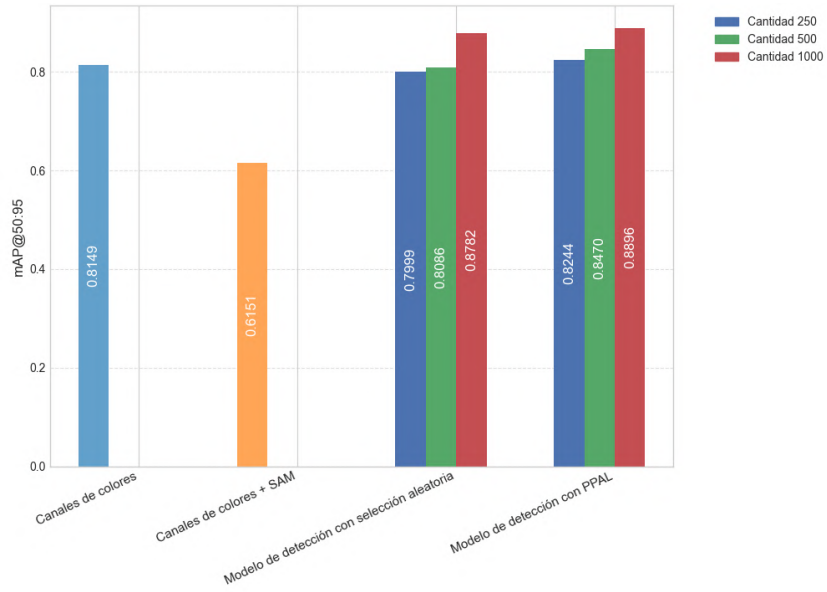


Figura 5.10: Resultados de mAP@50:95 obtenidos por cada método de autoetiquetado al etiquetar imágenes sintéticas generadas por un modelo generativo entrenado mediante Dreambooth. Los modelos de detección fueron entrenados con 250, 500 y 1000 imágenes.

Por otro lado, al observar los resultados obtenidos por los métodos que hacen uso de canales de colores, se observa como, al no utilizar SAM, el mismo es capaz de superar únicamente al modelo de detección con selección aleatoria usando 250 imágenes. Esto posiblemente se deba a que, como fue explicado en 5.4, las imágenes generadas son consistentemente más sencillas y con menos variación que las reales, facilitando así que los modelos de detección obtengan mejores resultados con un número reducido de imágenes.

Sin embargo, el agregado del modelo de segmentación SAM muestra claramente ser perjudicial. Al observar las etiquetas generadas por el mismo, si bien parece dar mejores etiquetas que su contraparte sin SAM al enfrentarse a imágenes con varias malezas, este tipo de imágenes son poco comunes dentro de las imágenes generadas, por lo que su agregado solo genera un posible punto de fallo más dentro del proceso en la mayoría de los casos.

5.6. Evaluación de las técnicas

Esta sección estará enfocada en la evaluación de las técnicas implementadas, presentando los resultados obtenidos en cada uno de los métodos, junto con un análisis comparativo en relación con la línea base.

La experimentación realizada consiste en tomar un porcentaje del conjunto de datos que deriva de elegir N imágenes del conjunto de imágenes total sin etiquetar U, donde se aplica cada técnica propuesta, que a partir de la entrada del conjunto de imágenes U realiza la elección de las N imágenes que se deberían etiquetar. Luego de obtener las N imágenes se entrena cada detector, y se evalúan sobre el conjunto de datos de evaluación obteniendo así la mAP, lo que deriva en la comparación entre el porcentaje de imágenes utilizado respecto al desempeño obtenido. Para el caso en el que la técnica involucre la utilización de generación de imágenes, se realiza un etiquetado de forma automática. Observar que como se encuentra en un proceso de evaluación de las técnicas, ya se cuenta con la etiqueta real de cada imagen sin necesidad de que un humano realice el etiquetado de forma manual, utilizando la etiqueta correspondiente para las N imágenes luego de que el modelo las eligiese.

A continuación se presentan las cuatro técnicas realizadas:

- **Aleatorio:** Correspondiente a seleccionar N imágenes de forma aleatoria.
- **PPAL:** Correspondiente a realizar la selección de N imágenes usando el componente PPAL.
- **Gen.:** Correspondiente a seleccionar N imágenes de forma aleatoria y a agregarles $2N$ imágenes generadas etiquetadas con un modelo de detección entrenado con las N imágenes elegidas. Salvo en el caso de $N = 250$, donde las imágenes generadas son etiquetadas con el método de canales de colores sin SAM.
- **PPAL + Gen:** Correspondiente a realizar la selección de N imágenes usando el componente PPAL y a agregarle $2N$ imágenes generadas eti-

quetadas mediante un modelo de detección YOLO entrenado con las N imágenes elegidas.

La configuración de los hiperparámetros de los modelos de evaluación se realizó con un tamaño de lote (batch) de cuatro para los tres evaluadores. En el caso de YOLO, al tratarse de un marco de trabajo que gestiona internamente la mayor parte de los hiperparámetros, la configuración se limitó a establecer la cantidad de épocas en 100, guardando el mejor modelo alcanzado durante ese periodo; para este estudio se utilizó el modelo YOLOv11m.

Para RetinaNet y Faster R-CNN, ambos evaluadores pertenecientes a la librería Torchvision, fue necesario realizar un ajuste más detallado de los hiperparámetros. En el caso de RetinaNet, la tasa de aprendizaje se estableció inicialmente en $0,01$, disminuyendo un $10\,\%$ en las épocas 50 y 55. Para Faster R-CNN, se utilizó una tasa de aprendizaje inicial de $2e-5$, la cual se redujo en un $10\,\%$ en las épocas 30, 40 y 50. En ambos casos, se entrenó durante un máximo de 60 épocas, también retornando el mejor modelo obtenido durante el entrenamiento. En cuanto a los optimizadores, se usa AdamW para Faster R-CNN, para RetinaNet se utiliza SGD, y para YOLOv11m se utiliza el optimizador por defecto el cual es AdamW.

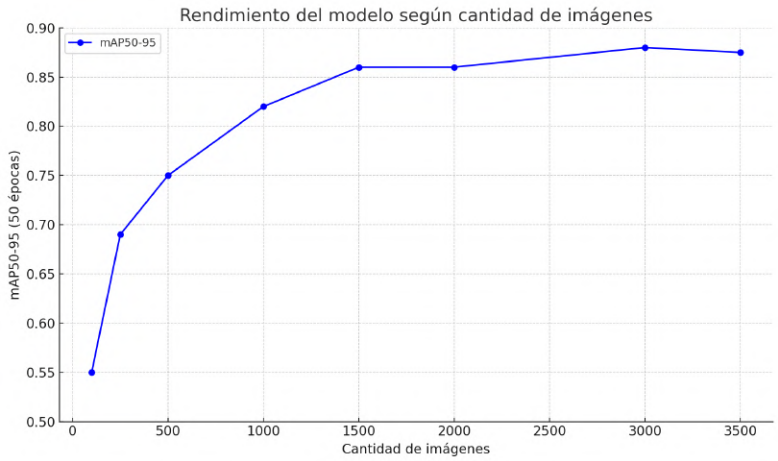


Figura 5.11: Rendimiento del modelo al variar la cantidad de imágenes, seleccionadas de manera aleatoria realizando el entrenamiento con YOLOv11m.

Para determinar los porcentajes del conjunto de datos a utilizar, primero se realiza un análisis preliminar para evaluar el desempeño del modelo al incrementar progresivamente la cantidad de imágenes etiquetadas usadas en el entrenamiento. Este estudio se muestra en la figura 5.11. Se parte del supuesto de que el conjunto completo contiene 5.000 imágenes. Entrenando YOLOv11m y utilizando diferentes tamaños de subconjuntos: 250, 500, 1.000 y hasta 3.500 imágenes, se mide el desempeño sobre el conjunto de evaluación.

El objetivo de este análisis es identificar los rangos más adecuados para la

experimentación. La idea es maximizar el rendimiento del modelo etiquetando la menor cantidad posible de imágenes. Si se seleccionan menos de 250 imágenes, la diferencia de rendimiento entre la selección aleatoria y las técnicas aplicadas es significativa, pero el desempeño general del modelo resulta insuficiente para ser útil. Por otro lado, usar más de 1.000 imágenes conlleva a un excelente desempeño, pero implica un gran esfuerzo humano y una posible mejora marginal al usar las técnicas, ya que el modelo se encuentra cerca de su máximo potencial.

Por este motivo, se eligen los valores de 250, 500 y 1.000 imágenes como los más representativos y provechosos para realizar los experimentos.

5.6.1. Resultados obtenidos

A continuación se presentarán los resultados obtenidos, la gráfica 5.12 corresponde a los resultados para Faster R-CNN, la gráfica 5.13 corresponde a los resultados para RetinaNet, y por último la gráfica 5.14 corresponde a los valores de YOLOv11m, además la tabla 5.3 muestra los resultados numéricos de cada gráfica. Como se puede apreciar los resultados son sumamente alentadores, en cada una de las gráficas se obtuvieron resultados por encima de la línea base aleatoria.

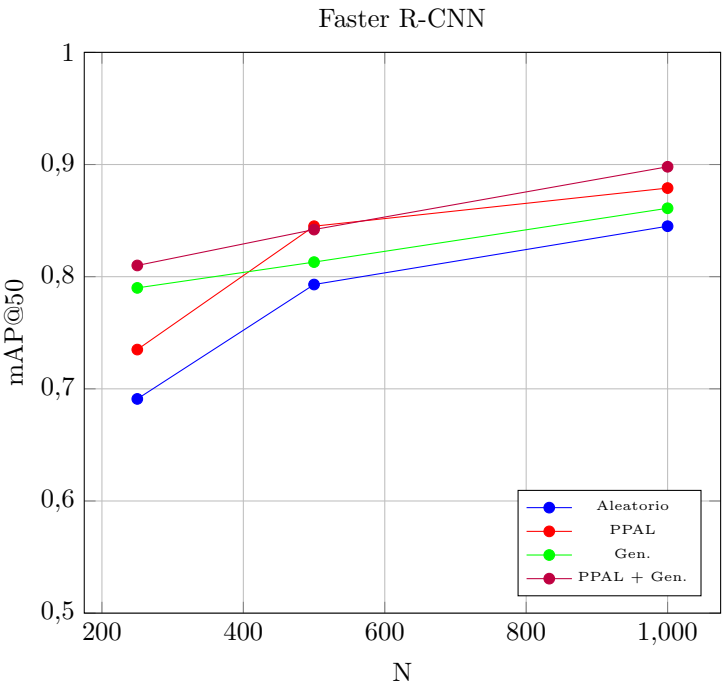


Figura 5.12: Comparación de estrategias - Faster R-CNN.

5.6.2. Análisis comparativo

En esta sección se procede a analizar los resultados obtenidos, realizando un análisis para cada detector, para luego dar una conclusión final sobre las técnicas realizadas.

Faster R-CNN

La estrategia combinada PPAL + Generación se posiciona como la más efectiva, superando claramente al resto de los métodos evaluados. En particular, con un conjunto de 1000 imágenes, esta estrategia alcanza una mejora de hasta 0,053 puntos en la mAP en comparación con la selección aleatoria. Por otro lado, la estrategia de Generación por sí sola muestra un mejor desempeño que PPAL cuando se utilizan solo 250 imágenes. Este resultado sugiere que, con un número reducido de datos, PPAL tiene dificultades para estimar de forma precisa la incertidumbre de las imágenes, mientras que la generación aporta ejemplos más informativos en esta etapa inicial.

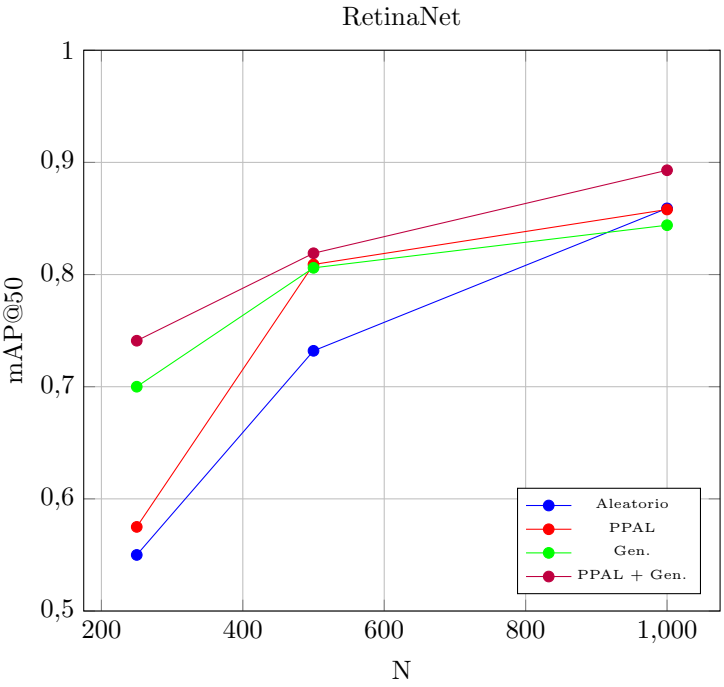


Figura 5.13: Comparación de estrategias - RetinaNet.

RetinaNet

En el caso del modelo RetinaNet, se observa una mayor sensibilidad a las técnicas aplicadas. Esto es especialmente evidente cuando se utilizan 250 imáge-

nes, donde tanto la estrategia de generación como la combinación PPAL + Generación alcanzan mejoras muy significativas, con incrementos de hasta 0,19 puntos en la mAP en comparación con la selección aleatoria. Estos resultados indican que RetinaNet se beneficia considerablemente de la calidad de los datos seleccionados o generados en etapas tempranas del entrenamiento.

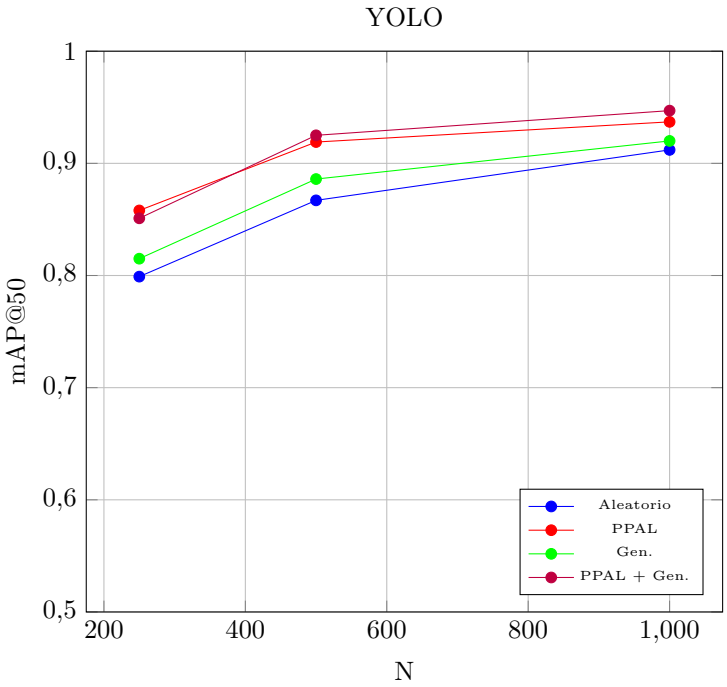


Figura 5.14: Comparación de estrategias - YOLOv11m.

YOLO

YOLOv11m es el modelo que muestra la mayor mejora en todas las estrategias evaluadas con respecto a la selección aleatoria. En particular, cuando se utilizan 1000 imágenes, la estrategia combinada PPAL + Generación alcanza un rendimiento de 0,947, lo que representa una mejora de 0,035 sobre el enfoque aleatorio. A diferencia de los otros modelos, PPAL y Generación muestran rendimientos similares entre sí en el rango de 500 a 1000 imágenes, lo que sugiere que YOLO es robusto frente a ambas técnicas aplicadas por separado.

Conclusiones clave de las evaluaciones

La estrategia PPAL + Generación se presenta como la más efectiva en todos los modelos evaluados, lo que demuestra que la combinación de selección mediante aprendizaje activo en conjunto con generación sintética para aumentación de datos permite obtener el mayor rendimiento. Incluso cuando se utiliza

Modelo	N	Aleatorio	PPAL	Aleatorio+Gen.	PPAL+Gen.
Faster R-CNN	250	0,691 (0,545)	0,735 (0,557)	0,790 (0,685)	**0,810 (0,683)**
	500	0,793 (0,664)	**0,845 (0,705)**	0,806 (0,715)	0,842 (0,752)
	1.000	0,845 (0,766)	0,879 (0,771)	0,861 (0,768)	**0,898 (0,799)**
RetinaNet	250	0,550 (0,313)	0,575 (0,393)	0,700 (0,531)	**0,741 (0,585)**
	500	0,732 (0,572)	0,809 (0,617)	0,806 (0,702)	**0,819 (0,709)**
	1.000	0,859 (0,774)	0,858 (0,793)	0,844 (0,786)	**0,893 (0,821)**
YOLOv11m	250	0,799 (0,724)	**0,858 (0,764)**	0.815 (0,734)	0,851 (0,762)
	500	0,867 (0,786)	0,919 (0,840)	0,886 (0,816)	**0,925 (0,857)**
	1.000	0,912 (0,837)	0,937 (0,877)	0,924 (0,855)	**0,947 (0,892)**

Tabla 5.3: Rendimiento por estrategia y cantidad de imágenes (N) para los modelos Faster R-CNN, Retina y YOLOv11m. En negrita se destacan los mejores valores por modelo y cantidad de imágenes. El valor fuera del paréntesis indica la mAP@50, mientras que el valor dentro del paréntesis indica, en el caso de Faster R-CNN, la mAP@75, y en el caso de YOLOv11m, indica la mAP@50:95.

por sí sola, la estrategia de generación ya aporta mejoras significativas, especialmente en contextos con bajo volumen de imágenes etiquetadas, como es el caso de $N = 250$. Al analizar el comportamiento de los distintos modelos, se observa que Faster R-CNN y RetinaNet muestran diferencias más marcadas entre los métodos aplicados, mientras que YOLO presenta un desempeño más estable y logra aprovechar de forma más eficiente cada estrategia. Esto puede atribuirse a que YOLO, al tratarse de un marco de trabajo completo, incluye optimizaciones propias que favorecen la obtención de mejores resultados.

En términos de desempeño, se alcanzan mejoras de hasta 0,191 en la mAP, lo cual representa un avance considerable en el contexto de modelos de redes neuronales. Para dimensionar esta mejora, es útil compararla con los resultados obtenidos en trabajos previos como Deng y Lu (2025), donde las redes generativas logran incrementos en el rango de solo 0,01 a 0,02 puntos de mAP.

Capítulo 6

Conclusiones y Trabajo Futuro

Este proyecto tuvo como objetivo principal investigar y desarrollar un sistema que integre diversos mecanismos que permitan reducir la carga del etiquetado de imágenes para tareas de detección de objetos, planteandose desde el contexto del desarrollo de un robot para la eliminación de malezas en sectores agrícolas, donde la escasez de datos etiquetados constituye un gran problema.

En particular, a lo largo de este trabajo, se aplicaron y combinaron diversas técnicas, entre las que se incluyen aprendizaje activo, generación de imágenes sintéticas mediante modelos generativos difusos ajustados mediante Dreambooth, y autoetiquetado de imágenes mediante canales de colores y modelos de detección.

En relación a los resultados alcanzados, se logró implementar una solución modular y fácilmente configurable, que permite, a partir de imágenes sin etiquetar, trabajar con diferentes componentes y flujos de datos, produciendo como salida conjuntos de datos más significativos para el entrenamiento de modelos de detección, y ampliados mediante datos sintéticos sin trabajo humano. Además, el proyecto se integra con CVAT (Computer Vision Annotatio Tool), facilitando el trabajo de los expertos al momento de utilizarlo.

La experimentación realizada permitió evaluar tanto cada componente por separado, como el sistema completo.

En lo que respecta al aprendizaje activo, la selección de imágenes mediante PPAL demostró ser útil para seleccionar las imágenes más relevantes, logrando claras mejoras de rendimiento, especialmente cuando la cantidad de imágenes a etiquetar es baja. Sin embargo, se observa que la selección inicial de imágenes, y las posteriores iteraciones del método puede tener un impacto notorio en el rendimiento final, por lo que métodos más avanzados de selección inicial, o un ajuste de hiperparámetros más sofisticado podrían mejor aún más los resultados finales.

En relación con la generación de imágenes sintéticas para ampliar el conjunto

de datos, el ajuste de modelos difusos está basado en el entrenamiento mediante Dreambooth sobre Stable Diffusion v1.5. El preprocesamiento automático de las imágenes de entrenamiento mediante diferentes criterios como el desenfoque, expansión de cajas delimitadoras y balance de brillo y contraste, permite crear conjuntos de datos con gran calidad y fidelidad a las muestras originales. Se emplea también el regularizador de Preservación Previa de la Clase para mantener la diversidad en el concepto aprendido. A pesar de que las imágenes generadas no presentaron buenos valores en las métricas FID e IS, las mismas, junto con el componente de autoetiquetado, también mostraron claras mejoras al ser utilizadas para el entrenamiento de modelos de detección. En cuanto a posibles mejoras a futuro de este componente, se podrían desarrollar métodos de detención automática para evitar el sobreajuste durante el entrenamiento de los modelos generativos, eliminando la necesidad de revisión manual, o permitiendo una revisión flexible por parte del usuario. Asimismo, podría explorarse la aplicación de filtros adicionales sobre el conjunto de entrenamiento para eliminar imágenes con oclusiones parciales de las malezas por otra vegetación cercana, rocas o troncos, lo que permitiría generar imágenes sintéticas más limpias, mejorando posteriormente los resultados de los métodos de autoetiquetado, especialmente de aquellos basados en canales de colores. Además, si bien en este trabajo se utilizó Stable Diffusion v1.5, por limitaciones de recursos, en futuras investigaciones se podría evaluar el desempeño al hacer uso de versiones más recientes como Stable Diffusion v3.5.

Respecto al autoetiquetado de imágenes generadas, se experimentó con un enfoque basado en canales de colores, y el modelo SAM. Estos métodos, al no requerir de entrenamiento, permiten etiquetar imágenes sintéticas sin intervención humana adicional. Aun así, métodos como el etiquetado a través de modelos de detección, en conjunto con filtros de cajas delimitadoras, mostraron resultados ligeramente superiores en la mayoría de los casos. Como mejora futura se podría explorar la implementación de técnicas como Diffugen, o una adaptación de HandsOff para modelos difusos, que aprovechan las capas intermedias del modelo al momento de generar una imagen para generar además sus etiquetas correspondientes.

También resulta prometedor el empleo de técnicas de detección de objetos semisupervisadas (SSOD), que permiten hacer uso no solo de los datos etiquetados manualmente, sino también el conjunto completo de imágenes reales no etiquetadas, aprovechando aun más los recursos disponibles. Estas técnicas no fueron implementadas debido a limitaciones de tiempo, aunque con el fin de impulsar trabajos futuros se incluye el anexo B el cual motiva el uso de estas técnicas para complementar el actual flujo de datos.

Finalmente, el análisis aplicado al comparar la combinación de las diferentes estrategias implementadas muestra que el flujo completo (PPAL + Generación mediante Dreambooth + Autoetiquetado mediante modelos de detección) resulta el más efectivo para mejorar el rendimiento. Aunque cabe destacar que los distintos modelos de detección aplicados (YOLOv11m, Faster R-CNN y RetinaNet) mostraron comportamientos diferentes ante las estrategias aplicadas. Además, desafortunadamente, dadas las limitaciones de tiempo, una posible

mejora consiste en aumentar la rigurosidad numérica a la hora de realizar los experimentos, replicandolos múltiples veces con el fin de que los mismos sean más robustos.

En resumen, el presente trabajo muestra el potencial de combinar aprendizaje activo, aumentación de datos mediante modelos generativos, y autoetiquetado en contextos de escasa disponibilidad de trabajo humano. A pesar de las limitaciones encontradas, los resultados obtenidos son alentadores y demuestran la efectividad de la integración de estas técnicas, que sientan bases sólidas para futuras investigaciones y mejoras, y abre posibles propuestas de trabajo que podrían extenderse a otros dominios más allá de la detección de malezas.

Referencias

Arjovsky, M., Chintala, S., y Bottou, L. (2017). Wasserstein generative adversarial networks. En *International conference on machine learning* (pp. 214–223).

Automatic. (s.f.). *Releases · AUTOMATIC1111/stable-diffusion-webui.* Descargado de https://github.com/AUTOMATIC1111/stable-diffusion-webui/releases

Bajwa, A. A., Walsh, M., y Chauhan, B. S. (2017). Weed management using crop competition in australia. *Crop Protection*, *95*, 8–13.

Bay, H., Tuytelaars, T., y Van Gool, L. (2006). Surf: Speeded up robust features. En *Computer vision–eccv 2006: 9th european conference on computer vision, graz, austria, may 7-13, 2006. proceedings, part i 9* (pp. 404–417).

Brust, C.-A., Käding, C., y Denzler, J. (2018). Active learning for deep object detection. *arXiv preprint arXiv:1809.09875*.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., y Zagoruyko, S. (2020). End-to-end object detection with transformers. En *European conference on computer vision* (pp. 213–229).

CEUTA. (2006). Agrotóxicos en uruguay: miradas desde los afectados. *Montevideo, Uruguay*.

Chawla, N. V., Bowyer, K. W., Hall, L. O., y Kegelmeyer, W. P. (2002). Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, *16*, 321–357.

Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., ... others (2019). Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*.

ClearML. (2024). *Clearml - your entire mlops stack in one open-source tool.* Descargado de https://clear.ml/ (Software available from http://github.com/clearml/clearml)

Cordeau, S., Triolet, M., Wayman, S., Steinberg, C., y Guillemin, J.-P. (2016). Bioherbicides: Dead in the water? a review of the existing products for integrated weed management. *Crop protection*, *87*, 44–49.

Dalal, N., y Triggs, B. (2005). Histograms of oriented gradients for human detection. En *2005 ieee computer society conference on computer vision and pattern recognition (cvpr'05)* (Vol. 1, pp. 886–893).

Dang, F., Chen, D., Lu, Y., y Li, Z. (2023a). Yoloweeds: A novel benchmark of yolo object detectors for multi-class weed detection in cotton production

systems. *Computers and Electronics in Agriculture*, *205*, 107655.

Dang, F., Chen, D., Lu, Y., y Li, Z. (2023b). Yoloweeds: A novel benchmark of yolo object detectors for multi-class weed detection in cotton production systems. *Computers and Electronics in Agriculture*, *205*, 107655.

Deng, B., y Lu, Y. (2025). Weed image augmentation by controlnet-added stable diffusion for multi-class weed detection. *Computers and Electronics in Agriculture*, *232*, 110123.

Dhariwal, P., y Nichol, A. (2021). Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, *34*, 8780–8794.

Donoser, M., y Bischof, H. (2006). Efficient maximally stable extremal region (mser) tracking. En *2006 ieee computer society conference on computer vision and pattern recognition (cvpr'06)* (Vol. 1, pp. 553–560).

Erdem, K. (2023). *Step by step visual introduction to diffusion models.* Step by Step Visual Introduction to Diffusion Models.-Blog by Kemal Erdem ....

Felzenszwalb, P. F., Girshick, R. B., McAllester, D., y Ramanan, D. (2009). Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, *32*(9), 1627–1645.

Gal, R., Alaluf, Y., Atzmon, Y., Patashnik, O., Bermano, A. H., Chechik, G., y Cohen-Or, D. (2022a). An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618*.

Gal, R., Alaluf, Y., Atzmon, Y., Patashnik, O., Bermano, A. H., Chechik, G., y Cohen-Or, D. (2022b). An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618*.

GeeksforGeeks. (2025, 7). *Artificial Neural Networks and its Applications.* Descargado de https://www.geeksforgeeks.org/artificial-neural-networks-and-its-applications/

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... Bengio, Y. (2020). Generative adversarial networks. *Communications of the ACM*, *63*(11), 139–144.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., y Courville, A. C. (2017). Improved training of wasserstein gans. *Advances in neural information processing systems*, *30*.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., y Hochreiter, S. (2017). Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, *30*.

Ho, J., Jain, A., y Abbeel, P. (2020). Denoising diffusion probabilistic models. *Advances in neural information processing systems*, *33*, 6840–6851.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., ... others (2022). Lora: Low-rank adaptation of large language models. *ICLR*, *1*(2), 3.

Jocher, G., y Qiu, J. (2024). *Ultralytics yolo11.* Descargado de https://github.com/ultralytics/ultralytics

Karras, T., Aila, T., Laine, S., y Lehtinen, J. (2017). Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint ar-*

Xiv:1710.10196.

Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., y Aila, T. (2020). Analyzing and improving the image quality of stylegan. En *Proceedings of the ieee/cvf conference on computer vision and pattern recognition* (pp. 8110–8119).

Kingma, D. P., Welling, M., y cols. (2013). *Auto-encoding variational bayes.* Banff, Canada.

Klidbary, S. H., Shouraki, S. B., Ghaffari, A., y Kourabbaslou, S. S. (2017). Outlier robust fuzzy active learning method (alm). En *2017 7th international conference on computer and knowledge engineering (iccke)* (pp. 347–352).

Kumari, N., Zhang, B., Zhang, R., Shechtman, E., y Zhu, J.-Y. (2023). Multi-concept customization of text-to-image diffusion. En *Proceedings of the ieee/cvf conference on computer vision and pattern recognition* (pp. 1931–1941).

LAION-Aesthetics — LAION. (2022). Descargado de https://laion.ai/blog/laion-aesthetics/

Lin, T.-Y., Goyal, P., Girshick, R., He, K., y Dollár, P. (2017). Focal loss for dense object detection. En *Proceedings of the ieee international conference on computer vision* (pp. 2980–2988).

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., ... Zitnick, C. L. (2014). Microsoft coco: Common objects in context. En *Computer vision–eccv 2014: 13th european conference, zurich, switzerland, september 6-12, 2014, proceedings, part v 13* (pp. 740–755).

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, *60*, 91–110.

Metz, L., Poole, B., Pfau, D., y Sohl-Dickstein, J. (2016). Unrolled generative adversarial networks. *arXiv preprint arXiv:1611.02163*.

Moreno, H., Gómez, A., Altares-López, S., Ribeiro, A., y Andújar, D. (2023). Analysis of stable diffusion-derived fake weeds performance for training convolutional neural networks. *Computers and Electronics in Agriculture*, *214*, 108324.

Nathancy. (2021). https://stackoverflow.com/a/56909036.

Nichol, A. Q., y Dhariwal, P. (2021). Improved denoising diffusion probabilistic models. En *International conference on machine learning* (pp. 8162–8171).

Nielsen, M. A. (2015). *Neural networks and deep learning* (Vol. 25). Determination press San Francisco, CA, USA.

Ojala, T., Pietikainen, M., y Maenpaa, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on pattern analysis and machine intelligence*, *24*(7), 971–987.

Packer, J. (2023, 1). Dreambooth Hyperparameter Guide - Jef Packer - Medium. Descargado de https://jefsnacker.medium.com/dreambooth-hyperparameter-guide-d8b7cd264245

Padrón, D., Estramil, S., y Núñez, N. (2025). *Informe estado del ar-*

te (Technical Report n.º 1). Montevideo, Uruguay: Facultad de Ingenieria Udelar. Descargado de https://gitlab.fing.edu.uy/daniel.padron/tesis-sistema-de-generacion-de-instancias-y-reduccion-de-carga-del-etiquetado-de-imagenes/-/wikis/Documento-del-estado-del-Arte

Papadopoulos, D. P., Uijlings, J. R., Keller, F., y Ferrari, V. (2017). Training object class detectors with click supervision. En *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 6374–6383).

Pech-Pacheco, J. L., Cristóbal, G., Chamorro-Martinez, J., y Fernández-Valdivia, J. (2000). Diatom autofocusing in brightfield microscopy: a comparative study. En *Proceedings 15th international conference on pattern recognition. icpr-2000* (Vol. 3, pp. 314–317).

Peng, H., Lin, S., King, D., Su, Y.-H., Abuzeid, W. M., Bly, R. A., ... Hannaford, B. (2024). Reducing annotating load: Active learning with synthetic images in surgical instrument segmentation. *Medical Image Analysis*, *97*, 103246.

Phil Wang. (2020). *Simple stylegan2 for pytorch.* https://github.com/lucidrains/stylegan2-pytorch?tab=readme-ov-file.

Pérez, S., Rodríguez, T., y Rován, E. (2023). *Detección de cultivos y malezas.* https://gitlab.fing.edu.uy/timag/2023/grupo-11-deteccion-de-cultivos-y-malezas/. (Accedido el 4 de mayo de 2025)

Redmon, J., Divvala, S., Girshick, R., y Farhadi, A. (2016). You only look once: Unified, real-time object detection. En *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 779–788).

Ren, P., Xiao, Y., Chang, X., Huang, P.-Y., Li, Z., Gupta, B. B., ... Wang, X. (2021). A survey of deep active learning. *ACM computing surveys (CSUR)*, *54*(9), 1–40.

Ren, S., He, K., Girshick, R., y Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, *28*.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., y Ommer, B. (2022). High-resolution image synthesis with latent diffusion models. En *Proceedings of the ieee/cvf conference on computer vision and pattern recognition* (pp. 10684–10695).

Ronneberger, O., Fischer, P., y Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. En *Medical image computing and computer-assisted intervention–miccai 2015: 18th international conference, munich, germany, october 5-9, 2015, proceedings, part iii 18* (pp. 234–241).

Rosebrock, A. (2024, 12). *Blur detection with OpenCV - PyImageSearch.* Descargado de https://pyimagesearch.com/2015/09/07/blur-detection-with-opencv

Ruiz, N., Li, Y., Jampani, V., Pritch, Y., Rubinstein, M., y Aberman, K. (2023). Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. En *Proceedings of the ieee/cvf conference on computer vision and pattern recognition* (pp. 22500–22510).

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., y Chen, X. (2016). Improved techniques for training gans. *Advances in neural information processing systems*, *29*.

Shenoda, M., y Kim, E. (2023). Diffugen: adaptable approach for generating labeled image datasets using stable diffusion models. *arXiv preprint arXiv:2309.00248*.

Song, J., Meng, C., y Ermon, S. (2020). Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*.

Suraj Patil, Pedro Cuenca y Valentine Kozin. (2022). *Training stable diffusion with dreambooth using diffusers.* https://huggingface.co/blog/dreambooth.

Sylabs. (2025, 3). *SingularityCE Documentation Hub — SyLabs.* Descargado de https://sylabs.io/docs/

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... Rabinovich, A. (2015). Going deeper with convolutions. En *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 1–9).

Tharwat, A., y Schenck, W. (2023). A survey on active learning: State-of-the-art, practical challenges and research directions. *Mathematics*, *11*(4), 820.

Uijlings, J. R., Van De Sande, K. E., Gevers, T., y Smeulders, A. W. (2013). Selective search for object recognition. *International journal of computer vision*, *104*, 154–171.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, *30*.

Viola, P., y Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. En *Proceedings of the 2001 ieee computer society conference on computer vision and pattern recognition. cvpr 2001* (Vol. 1, pp. I–I).

Wang, H. (2024, Dec.). Comparative analysis of gans and diffusion models in image generation. *Highlights in Science, Engineering and Technology*, *120*, 59–66. doi: 10.54097/9gba9v27

Wang, Y., Wu, C., Herranz, L., Van de Weijer, J., Gonzalez-Garcia, A., y Raducanu, B. (2018). Transferring gans: generating images from limited data. En *Proceedings of the european conference on computer vision (eccv)* (pp. 218–234).

Xu, A., Vasileva, M. I., Dave, A., y Seshadri, A. (2023). Handsoff: Labeled dataset generation with no additional human annotations. En *Proceedings of the ieee/cvf conference on computer vision and pattern recognition* (pp. 7991–8000).

Yang, C., Huang, L., y Crowley, E. J. (2024). Plug and play active learning for object detection. En *Proceedings of the ieee/cvf conference on computer vision and pattern recognition* (pp. 17784–17793).

Zhang, L., Rao, A., y Agrawala, M. (2023). Adding conditional control to text-to-image diffusion models. En *Proceedings of the ieee/cvf international conference on computer vision* (pp. 3836–3847).

Zhang, Y., Song, Z., y Li, W. (2021). Unsupervised data augmentation for object detection. *arXiv preprint arXiv:2104.14965*.

Zhou, Q., Yu, C., Wang, Z., Qian, Q., y Li, H. (2021). Instant-teaching: An end-to-end semi-supervised object detection framework. En *Proceedings of the ieee/cvf conference on computer vision and pattern recognition* (pp. 4081–4090).

Anexo A

Técnica de recorte, filtrado y corrección de imágenes para entrenamiento de modelos generativos

En este primer anexo se complementa la descripción realizada sobre el proceso de creación del conjunto de imágenes utilizada en el ajuste de los modelos difusos descrita en la sección 4.3.2. Particularmente se profundiza sobre los apartados de expansión de las cajas delimitadoras, filtrado de desenfoque y normalización de brillo y contraste.

Como se describe en la sección 4.3.2, el segundo paso del proceso descrito consiste en un filtro de desenfoque, con el fin de eliminar la mayor cantidad de instancias borrosas posible. Basándonos en la guía Rosebrock (2024) existen varias formas de detectar desenfoque, pero una de las técnicas más conocidas y veloces es utilizando la varianza del Laplaciano; en particular la implementación propuesta por Pech-Pacheco, Cristóbal, Chamorro-Martinez, y Fernández-Valdivia (2000).

El operador Laplaciano es utilizado para medir la segunda derivada de la imagen, en este contexto es utilizado con el fin de detectar zonas con cambios abruptos, como los bordes de los objetos que componen la imagen. La intuición detrás del método consiste en que si la varianza es elevada, entonces en la matriz se encuentran con una amplia gama de valores altos (indicado bordes claros), y valores bajos (indicado el interior de los objetos en la imagen), mostrando lo esperado en una imagen enfocada. Mientras tanto si la varianza es baja, indica poca presencia de bordes, típico de las imágenes desenfocadas. Se clasifican las imágenes como no borrosas cuando la varianza es superior a un umbral definido, el cual es configurable, aunque para el proceso de experimentación se definió dicho umbral como 100.

En particular, en la implementación utilizada (Pech-Pacheco y cols., 2000) se transforma la imagen a un único canal (escala de grises) y luego calcula el Laplaciano vía una operación convolucional, utilizando el siguiente núcleo (kernel):

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{A.1}$$

En lo que respecta a la expansión de las cajas delimitadoras, el objetivo es extenderlas de forma tal que la posición de la maleza en la imagen resultante no sea necesariamente el centro de la misma, sino que pueda estar situada en cualquier lugar. Ademas, la expansión es realizada de forma tal que se garantice no incluir otras malezas cercanas.

La estrategia consiste primero en definir una superficie alrededor de la caja delimitadora de la maleza que se busca expandir, la cual se definió que tenga como máximo un largo de $7r$, donde se define r como la mitad del lado más largo de la caja delimitadora. Esta área se entiende que es lo suficientemente grande como para que la maleza tome diversas posiciones en el recorte final, pero al mismo tiempo sea claramente visible. Luego se reduce la anterior superficie de forma tal que no se superponga con ninguna otra maleza, notar que en algunos casos, es posible que alguna otra planta también se encuentre dentro del recorte, debido a que unicamente se tiene información sobre la posición de las malezas etiquetadas, y no de objetos como el césped o pequeños brotes. Por último se escoge de forma aleatoria el recorte dentro de la anterior superficie, la cual tiene como máximo tamaño $5r$; este recorte sera una de las imágenes en el ajuste de los modelos difusos. Notar que las anteriores medidas puede ser modificadas, pero están definidas de forma tal que se garantiza que la maleza en su totalidad sea parte de la imagen final. En la figura A.1 se ilustra los principales componentes del proceso.

Finalmente se normaliza el brillo y contraste de las imágenes recortadas, para ello se utiliza un simple método propuesto por Nathancy (2021). En esencia, en este se busca aprovechar el espectro de colores de forma tal que la distribución de los valores más frecuentes se extienda a lo largo de todo el espectro.

Como se observa en el histograma azul de la figura A.2, todos los valores de pixeles (en escala de grises) más frecuentes se encuentran concentrados en su mayoría en una breve tramo del espacio de valores de pixeles (entre 0 y 255). Lo que se propone en este método es identificar este segmento, y expandirlo por todo el espacio. Para ello se define el inicio del anterior segmento (llamado g_{min}) como el primer valor del pixel que tiene una frecuencia mayor al 1 %, y el final del segmento (llamado g_{max}) de forma análoga.

Luego para todos los pixeles se aplica la siguiente operación:

$$\overline{img}(x, y) = \alpha.img(x, y) + \beta$$

donde $img(x, y)$ es el pixel de la imagen original, $\overline{img}(x, y)$ es el pixel de la imagen ajustada, $\alpha = \frac{255}{g_{max} - g_{min}}$ y $\beta = -g_{min} * \alpha$. Esto resulta en el histograma naranja de la figura A.2.

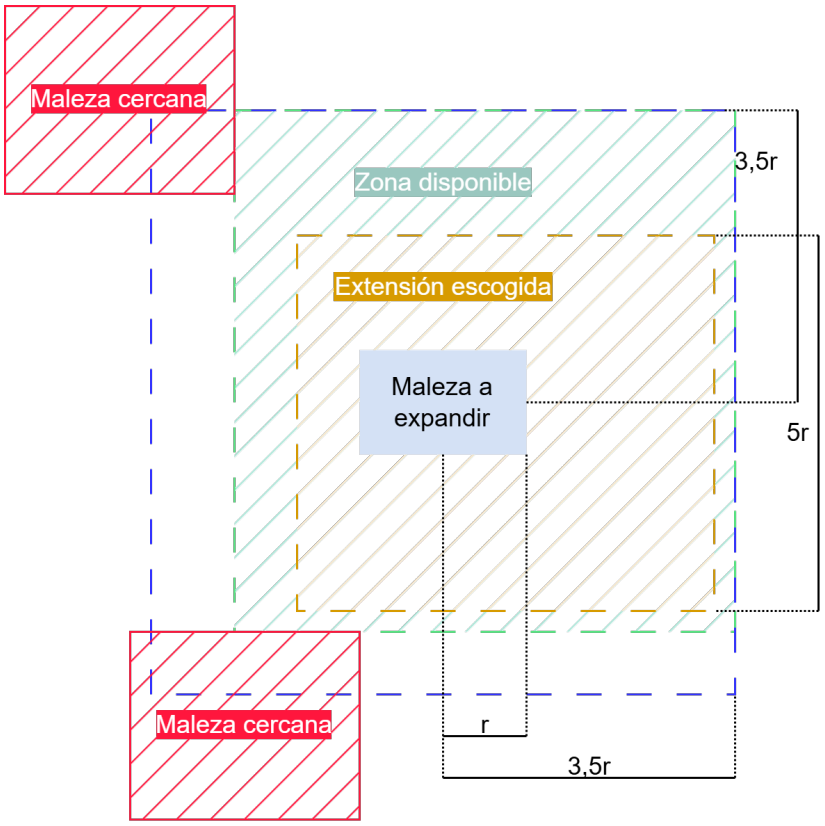


Figura A.1: Ilustración del proceso de expansión de las cajas delimitadoras en la creación del conjunto de datos utilizado en el ajuste de los modelos generativos. Se ilustra la caja delimitadora a expandir como "Maleza a expandir", donde su lado más grande mide $2r$. Se define el área máxima de expansión como la superficie de largo $7r$ centrada en "Maleza a expandir", ilustrada como el cuadrado con borde discontinuo azul. Dentro de la anterior superficie, se ilustra con el rectángulo de color verde llamado "Zona disponible" el área la cual no intercepta con ninguna de las malezas cercanas. Por último el recorte buscado de largo máximo $5r$ se encuentra dentro de "Zona disponible" bajo el nombre "Extensión escogida".

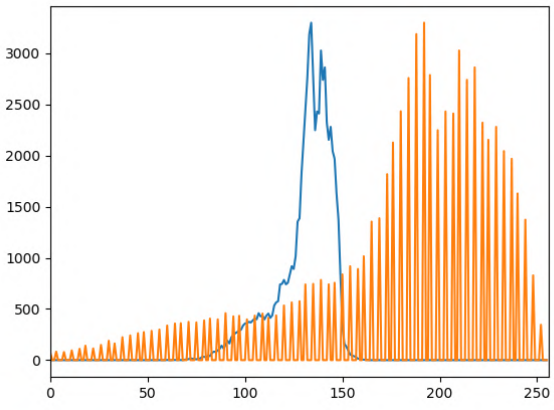


Figura A.2: En azul se muestra el histograma de valores de pixeles en escala de gris para una imagen sin modificar. En naranja se muestra el mismo histograma cuando se recorta la parte más significativa de primero, y se lo extiende en todo el espacio de pixeles.

En la figura A.3a se presenta un ejemplo de imagen sin ajuste de brillo ni constaste, mientras que en A.3b se presenta la misma imagen al aplicar el proceso descrito.

(a) (b)

Figura A.3: (a) Imagen de ejemplo de la especie *ambrosia artemisiifolia* donde se presentan de brillo y contraste. (b) es la misma imagen pero luego de haberle aplicado el proceso de balance de brillo y contraste antes descrito.

Anexo B

Uso de imágenes reales no etiquetadas

Como fue mencionado en la sección 4.1, el flujo de datos utilizado en este proyecto tiene como salida el conjunto de las imágenes generadas autoetiquetadas, junto con un subconjunto de las imágenes de entrada etiquetadas, donde este último está conformado por las imágenes elegidas por el componente de selección.

Una desventaja de lo anterior resulta evidente, ¿qué sucede con las imágenes de entrada no elegidas por dicho selector? Notar que en la actual solución, no se proporciona un mecanismo para etiquetar estas imágenes, y por ende no pueden ser parte de la salida del flujo.

Lo ideal sería obtener la localización y especie de cada maleza en las imágenes, de forma que las mismas puedan ser utilizadas en el entrenamiento de detectores. Pero al mismo tiempo, si no se cuenta con ninguna información o mecanismo de control extra, la anterior intención conlleva inexorablemente al mismo objetivo que tienen los detectores de objetos; esto es, a partir únicamente de la imagen, obtener la localización y clase de cada objeto.

En tal situación, se debería recurrir a los métodos de detección de objetos (relevados en sección 2.1) entrenados con las imágenes resultantes del actual flujo de datos. Entonces, ¿se debería utilizar dicho detector para etiquetar las imágenes restantes, y luego re-entrenar al detector incluyendo dichas imágenes? Aunque su uso es posible, si es utilizado sin consideraciones adicionales, esto solamente reforzaría los sesgos del detector. En otras palabras, el detector refuerza sus predicciones en lo que actualmente detecta y no detecta, causando que también se consoliden incluso las detecciones omitidas o incorrectamente realizadas.

Se propone entonces abordar el problema de las imágenes no etiquetadas vía técnicas de detección de objetos semi-supervisados (SSOD). La motivación de dichas técnicas subyace en el hecho de que una gran variedad de conjuntos de imágenes utilizados en entrenamientos de detectores actualmente cuentan

con una gran partición de imágenes sin etiquetar. Entonces, dado el carácter supervisado de los detectores de objetos, la anterior información no podría ser utilizada. El objetivo de las técnicas de SSOD es buscar mecanismos para utilizar dicho conjunto de imágenes sin etiquetar, de forma de obtener un beneficio en el rendimiento de los detectores.

Debido a esto, se propone en este proyecto entrenar un detector de objetos mediante un mecanismo de SSOD, donde se utiliza de forma supervisada las imágenes salientes del flujo de datos realizado, y de forma no supervisada las imágenes de la entrada del flujo no elegidas por el componente selector.

Restaría decidir si dicho detector debería ser utilizado para etiquetar las imágenes restantes y agregarlas a la salida del flujo de datos, o si debería ser utilizado en el entrenamiento de detectores finales (en particular, el detector utilizado en el robot eliminador de malezas, donde se encuentra el proyecto mayor).

Como fue previamente explicado, dado estrictamente a restricciones de tiempo, no fue posible la integración de dichas técnicas en la solución final, ni tampoco su respectiva experimentación. De todas formas, con el fin de motivar el trabajo futuro en lo que respecta a dichas técnicas, se incluye en el capítulo 4 del documento del estado del arte (Padrón y cols., 2025) un breve revelamiento sobre el estado del arte de SSOD y como podría ser aplicado en este proyecto.

Anexo C

Control sobre la generación de modelos generativos ajustados

Una de las razones por las que fue despriorizado y finalmente descartado el uso de DiffuGen (Shenoda y Kim, 2023) fue el poco control que se logra sobre la generación de imágenes mediante los modelos de difusión entrenados con Dreambooth; esto es, si bien los modelos son capaces de generar imágenes con alta fidelidad respecto a las originales, los mismos ciertamente no son capaces de aprender de forma precisa que regiones de la imagen se corresponden a la maleza, y cuales al fondo o al suelo.

(a) (b)

Figura C.1: (a) Imagen generada de la especie *ipomoea indica* utilizando el texto "a photo of [clase] seedling, pink background". Ciertamente, la misma no presenta un fondo rosado como se le indica. (b) Imagen generada de la especie *ambrosia artemisiifolia* utilizando el texto "a photo of three [clase] seedlings" .

Esta limitación se evidencia a través de la figura C.1b y especialmente la figura C.1a, las cuales presentan imágenes generadas con instrucciones más especificas que buscan comprobar si el modelo es capaz de entender y separar cual es el sujeto (la maleza) y cual es el fondo. En el caso de la figura C.1b, se observa que el modelo no logra generar tres instancias, sino que unicamente una, mostrando que no comprende que parte de la imagen se compone por la maleza. Por otro lado, en la figura C.1a se ve como el modelo no entiende que parte es el fondo, aunque si entiende el concepto de "rosado" (pink) puesto que es capaz de generar una de las hojas con este color, aunque esto no era lo esperado.

Anexo D

Taxonomía de estrategias de consulta para AL

Con el fin de complementar lo relevado en la QS, a continuación se desarrolla sobre la taxonomía de selección, en la figura D.1 se diagraman las principales categorías.

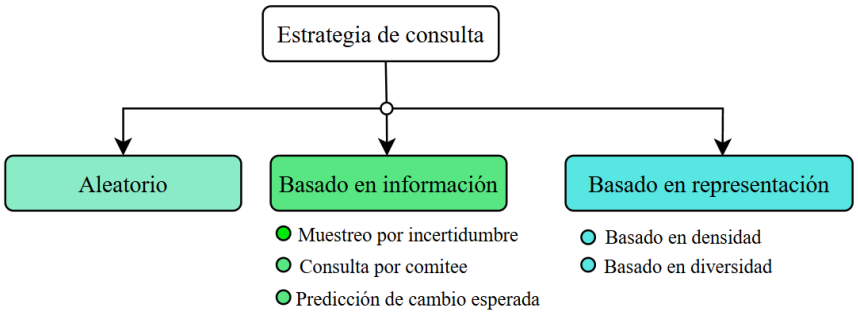


Figura D.1: Taxonomía correspondiente a las diferentes estrategias de selección en aprendizaje activo

A continuación se presentan algunas de las estrategias más relevantes:

- Basado en la información (Information-based): Estas estrategias se basan en cuánta información o incertidumbre tiene una muestra y qué tanto puede contribuir al modelo al ser etiquetada.
  - Muestreo por incertidumbre (Uncertainty Sampling): Selecciona las muestras sobre las cuales el modelo se encuentra más incierto.
  - Consulta por comité (Query by Committee): Diferentes modelos (comités) hacen predicciones y seleccionan las muestras donde hay mayor desacuerdo entre los modelos.

- Cambio de predicción esperado (Expected Prediction Change): Selecciona muestras que cambiarán significativamente las predicciones futuras del modelo.
- Basado en la representación (Representation-based): Estas estrategias consideran cómo las muestras representan el espacio de datos y buscan mejorar la diversidad y representatividad del conjunto etiquetado.
    - Basado en la densidad (Density-based): Selecciona muestras que están en áreas densamente pobladas del espacio de datos, para asegurar que los ejemplos comunes sean bien representados.
    - Basado en la diversidad (Diversity-based): Selecciona muestras que sean lo más diferentes posible entre sí, para cubrir una variedad de regiones del espacio de datos.

Anexo E

Configuración de método y modelos utilizados

Con el fin de facilitar la reproducibilidad de los experimentos presentados a lo largo de este documento, en el presente anexo se detallan las configuraciones empleadas en cada uno de los métodos y modelos utilizados.

Cabe aclarar que varios de estos métodos y modelos no corresponden a implementaciones propias y suelen contar con decenas de parámetros que no resultan relevantes para el presente trabajo. En dichos casos, se han utilizado los valores por defecto provistos por sus respectivas implementaciones. Para garantizar la trazabilidad, se especifica de manera precisa el nombre y la versión de cada método o modelo, de modo que los valores por defecto puedan consultarse en su documentación oficial en caso de ser necesario.

1. **Aprendizaje activo con PPAL**

   La tabla E.1 comprenden los parámetros utilizados durante el proceso de selección de imágenes con aprendizaje activo PPAL. El resto de los parámetros no especificados corresponden al método en sí, los mismos son explorados y descritos por los autores del método.

Parámetro	**Valor**
Número de ciclos	3
Conjunto inicial	40 %
Número de épocas de entrenamiento del detector por ciclo	35

Tabla E.1: Parámetros utilizados durante el proceso de selección de imágenes con PPAL.

2. **Preparación de conjunto de datos para el ajuste de los modelos generativos**

   Los parámetros utilizados en cada etapa del proceso de creación de los conjuntos de entrenamiento de los modelos difusos son descritas en la tabla E.2.

Etapa	Parámetro	Valor
Expansión cajas delimitadoras	Largo máximo de la zona disponible para la extensión de la caja delimitadora	7 veces el largo máximo de la caja delimitadora original
	Largo máximo de la extensión de la caja delimitadora	5 veces el largo máximo de la caja delimitadora original
	Máximo numero de instancias por especie	32
	Priorizar instancias por	Mayor resolución
Filtrado por desenfoque	Mínima varianza de Laplaciano	100
Filtro por tamaño	Mínimo largo de caja delimitadora	600 píxeles
Corrección de brillo y contraste	Porcentaje de frecuencia del histograma seleccionado y ampliado	Mayor al 1 %
Ajuste de resolución	Resolución final del recorte	680 píxeles

Tabla E.2: Parámetros utilizados en el proceso de creación de los conjuntos de entrenamiento de los modelos generativos.

3. **Generación de imágenes sintéticas**

   En la tabla E.3 se presentan los principales parámetros en lo que respecta a la etapa de generación de imágenes sintéticas.

Parámetro	Valor
Planificador de ruido	DDPMScheduler
Resolución	680
Número de iteraciones	50
Instrucción de generación	*a photo of sks seedling*

Tabla E.3: Parámetros utilizados en la generación de imágenes sintéticas utilizando los modelos generativos previamente ajustados.

4. **Ajuste de modelo generativo**

   Las configuraciones del ajuste de los métodos generativos aquí presentada corresponde únicamente a la técnica Dreambooth, ya que es la técnica escogida durante el proceso de experimentación. Los parámetros son presentados en la tabla E.4.

Parámetro	Valor
Pesos iniciales	stable-diffusion-v1-5
Resolución	680
Tamaño de lote	1
Acumulación de gradiente	1
Tasa de aprendizaje	e^{-6}
Planificador de tasa de aprendizaje	Constante
Pasos de calentamiento de tasa de aprendizaje	0
Numero de pasos	180 por instancia de entrenamiento
Entrenar codificador de texto	Si
Utilizar Gradient checkpointing	Si
Utilizar XFormers	Si
Utilizar Preservación previa específica de la clase	Si
Número de ejemplos de Preservación previa específica de la clase	500 por instancia de entrenamiento
Peso de Preservación previa específica de la clase en la función de perdida	1
Precisión mixta	fp16
Instrucción de entrenamiento	*a photo of sks seedling*

Tabla E.4: Parámetros utilizados en el ajuste de modelos generativos utilizando la técnica Dreambooth.

5. **Autoetiquetado**

   La tabla E.5 comprenden los parámetros utilizados por los métodos de autoetiquetado mediante canales de colores, canales de colores junto con SAM, y modelos de detección con el agregado de filtros.

6. **Detectores de objetos para evaluación de las técnicas**

   Por último, la Tabla E.6 muestra la configuración utilizada para los métodos Faster R-CNN, RetinaNet y YOLOv11m. Cabe aclarar que las implementaciones de Faster R-CNN y RetinaNet son provistas por la librería Torchvision, mientras que YOLOv11m corresponde a Ultralytics. Es importante señalar que únicamente se modificaron las configuraciones especificadas en la tabla, manteniendo los valores por defecto en el resto de los casos. Además, se destaca que solo durante el entrenamiento de YOLOv11m se aplica un aumento de datos adicional (integrado en su implementación), además del realizado mediante la incorporación de imágenes sintéticas.

Método	Parámetro	Valor
Mediante canales de colores.	Iteraciones de suavizado	3
	Iteraciones de dilatación	2
	Porcentaje de área mínima	0.5 %
Mediante canales de colores y SAM	Iteraciones de dilatación para máscaras de SAM	1
	Porcentaje de área mínima para máscaras de SAM	0.8 %
	Puntos guía por contorno	5
Mediante modelos de detección	Umbral de confianza mínima	40 %

Tabla E.5: Parámetros utilizados para el autoetiquetado mediante canales de colores, canales de colores con SAM, y modelos de detección.

Detector	Parámetro	Valor
YOLOv11m (Ultralytics)	Número de épocas	100
	Tamaño de lote	4
	Resolución de imagen máxima	680
Faster R-CNN (Torchvision)	Pesos iniciales	FasterRCNN_ResNet50_FPN_Weights (COCO_V1)
	Número de épocas	60
	Tamaño de lote	4
	Resolución de imagen máxima	1333
	Planificador de tasa de aprendizaje	Con descensos al 10 % en las épocas 50 y 55
	Tasa de aprendizaje inicial	2e-5
	Decaimiento de pesos	1e-2
	Optimizador	AdamW
RetinaNet (Torchvision)	Pesos iniciales	RetinaNet_ResNet50_FPN_V2_Weights (COCO_V1)
	Número de épocas	60
	Tamaño de lote	4
	Resolución de imagen máxima	1333
	Planificador de tasa de aprendizaje	Con descensos al 10 % en las épocas 30, 40 y 50
	Tasa de aprendizaje inicial	0.01
	Decaimiento de pesos	1e-4
	Impulso (momentum)	0.9
	Optimizador	SGD

Tabla E.6: Parámetros utilizados durante el entrenamiento de los detectores de objetos de evaluación Faster R-CNN, RetinaNet y YOLOv11m.