Title of the Final Project

First Author
firstauthor@i1.org

Second Author
secondauthor@i2.org

Third Author
thirdauthor@i2.org

Fourth Author
fourthauthor@i2.org

Abstract

Brief summary of the work developed as well as the main results and contributions.

1. Introduction

Here, the problem to be solved is described (what do we want to do?), the motivation (why is it relevant to do it?), and the goals (what specific objectives are we going to address in order to solve the problem?).

We use the LaTeX format of the CVPR conference. This document can be written either in English or Spanish.

What follows is a tentative approximation of the sections the document should have. If students consider that they need others, as well as subdividing the sections into different subsections, they can do it without problem.

Students can take inspiration from the cs231n final projects website: http://cs231n.stanford.edu/project.html, where different projects are presented and developed.

This whole final report can have from 6 to 8 pages (no more and no less).

2. Background

This section presents the fundamental concepts necessary to understand the work.

3. Related Works

It presents what has been done in the field previously, and what the best methods are currently. It is very important, in general, not only in this section, to adequately document the relevant literature. To do this, you must use the *.bib* file, in the way I show here: [**?**, **?**, **?**]

4. Methods

4.1. Modelos de Clasificación

En nuestro problema tenemos dos etapas de clasificación de imágenes. Primero, una clasificación de hojas en tipos de plantas, y luego una clasificación de las enfermedades presentes en las hojas de cada tipo de planta.

Estos son dos problemas bastante diferentes en cuanto a la complejidad de la tarea, principalmente porque en la primera etapa las diferencias entre clases son más notorias (muy diferentes tipos de plantas), mientras que en la segunda etapa las diferencias son más sutiles (diferentes enfermedades que pueden afectar a la misma planta, y que a veces pueden parecerse mucho entre sí). Además, nos enfretamos al problema de la diferencia de tamaños de las imágenes de las hojas y la de sus enfermedades.

Con esto en mente, tiene sentido estudiar una amplia gama de modelos de clasificación, y elegir los más adecuados para cada etapa del problema.

He dividido el estudio inicial de los modelos en dos partes: arquitecturas clásicas y arquitecturas modernas basadas en Transformers. En las tablas **??** y **??** se presentan las comparaciones detalladas de las diferentes arquitecturas estudiadas.

4.1.1 Modelos clásicos

En la tabla **??** se presentan las comparaciones detalladas de las diferentes arquitecturas clásicas estudiadas: GoogLeNet (Inception), ResNet, EfficientNetV1 y EfficientNetV2. Aquí se describen las características principales de cada modelo, sus ventajas y desventajas, y por qué se eligieron para este proyecto.

A lo largo del tiempo se han desarrollado técnicas interesantes que han llevado al avance de estos modelos. A continuación los describiré brevemente:

1. **GoogLeNet (Inception)**: Introdujo los módulos Inception que permiten capturar características a múltiples escalas mediante convoluciones paralelas de diferentes tamaños. Esto mejora la eficien-

Característica	GoogLeNet (Inception)	ResNet	EfficientNet (V1)	EfficientNetV2
Primera Publicación	2014 (Szegedy et al., Google)	2015 (He et al.)	2019 (Tan & Le, Google)	2021 (Tan & Le, Google)
Idea Principal	Módulos Inception con convoluciones paralelas de múltiples escalas	Conexiones residuales (skip) para permitir el entrenamiento de redes muy profundas	Escalado compuesto: escalar conjuntamente profundidad, anchura y resolución usando un coeficiente fijo	Escalado compuesto mejorado + entrenamiento más rápido: arquitectura más inteligente + regularización adaptativa
Bloque Básico	Módulo Inception: convoluciones paralelas 1×1, 3×3, 5×5 + max pooling, concatenadas	• BasicBlock (2× 3×3 convs) para ResNet18/34 • Bottleneck (1×1 → 3×3 → 1×1 convs) para ResNet50+	MBConv (Mobile Inverted Bottleneck): 1×1 exp → 3×3 depthwise → SE → 1×1 proj	Fused-MBConv (etapas iniciales): conv 3×3 regular en lugar de 1×1+depthwise MBConv (etapas posteriores) SE solo en bloques posteriores
Conexiones Skip	× No (pero versiones posteriores como Inception-ResNet sí)	✓ Sí (residual aditivo)	✓ Sí (MBConv usa residual cuando strides=1)	✓ Sí
Atención	× No	× No	✓ Squeeze-and-Excitation (SE) en cada bloque	✓ SE, pero solo en etapas posteriores (para reducir overhead)
Método de Escalado	Manual: aumento de módulos Inception apilados	Aumento manual de profundidad (18 → 34 → 50 → 101 → 152)	Escalado compuesto uniforme: Profundidad $\times\alpha$, Anchura $\times\beta$, Res $\times\gamma$	Escalado compuesto no uniforme + aprendizaje progresivo (imágenes pequeñas → grandes durante entrenamiento)
Resolución de Entrada	Fija (224×224)	Fija (usualmente 224×224), pero flexible	Escala con el tamaño del modelo: B0: 224 → B7: 600	Menor que V1 para la misma precisión: S: 384, M: 480, L: 480–512
Tamaños Típicos	• GoogLeNet (6.8M params) • Inception-v3 (23.8M) • Inception-v4 (42.7M)	• ResNet18 (11M params) • ResNet34 (21M) • ResNet50 (25M) • ResNet101 (44M)	• B0 (5.3M) • B3 (12M) • B5 (30M) • B7 (66M)	• S (21M) • M (54M) • L (120M)
Precisión (ImageNet)	• GoogLeNet: ∼69.8% • Inception-v3: ∼78.8%	• ResNet50: ∼76% • ResNet101: ∼78%	• B0: ∼77% • B3: ∼82% • B7: ∼84.4%	• S: ∼83.9% • M: ∼85.2% • L: ∼85.7%
Velocidad de Entrenamiento	Moderada	Moderada	Lenta (debido a entradas de alta resolución y profundidad)	Mucho más rápida (hasta 11× vs V1 con precisión similar)
Eficiencia de Inferencia	Buena; diseño eficiente en parámetros para su época	Buena para modelos pequeños (18/34); más pesada para 101+	Muy eficiente en parámetros, pero alta resolución afecta latencia	Mejor relación FLOPs/precisión; optimizado para TPU/GPU
Mejor Para	• Comprensión histórica de CNNs • Extracción de características multi-escala	• Modelos base • Transfer learning con datos limitados • Prototipado rápido	• Despliegue con recursos limitados • Cuando se necesita alta precisión con menos parámetros	• Sistemas de producción que necesitan velocidad + precisión • Entrenamiento desde cero o fine-tuning
fastai / PyTorch	✓ Nativo en torchvision (`googlenet`, `inception_v3`)	✓ Nativo en torchvision y fastai	✓ Vía `timm` (ej., `efficientnet_b0`)	✓ Vía `timm` (ej., `efficientnetv2_s`)
Debilidad Principal	Arquitectura obsoleta; menor precisión que modelos modernos	Eficiencia de parámetros subóptima; obsoleto vs modelos modernos	Entrenamiento lento; SE añade overhead computacional; alta resolución = mucha memoria	Ecosistema ligeramente menos maduro; no está en torchvision

Table 1. Comparación de las Arquitecturas GoogLeNet (Inception), ResNet, EfficientNetV1 y EfficientNetV2

cia computacional y la capacidad de extracción de características.

2. **ResNEt (Capas Residuales)**: Introdujo las conexiones residuales (skip connections) que facilitan el entrenamiento de redes muy profundas al mitigar el problema del desvanecimiento del gradiente. Esto permitió construir redes con cientos de capas.

3. **EfficientNetV1 (Escalado Compuesto)**: Propuso un método sistemático para escalar redes neuronales ajustando conjuntamente la profundidad, anchura y resolución de entrada mediante un coeficiente fijo. Esto llevó a modelos más eficientes en términos de precisión y tamaño.

4. **EfficientNetv2 (Progesive Learning)**: Mejoró la velocidad de entrenamiento mediante una arquitectura más inteligente y una regularización adaptativa.

Característica	ConvNeXt	Swin Transformer	MaxViT
Primera Publicación	2022 (Liu et al., Meta/FAIR)	2021 (Liu et al., Microsoft)	2022 (Tu et al., Google)
Idea Principal	Modernización de ResNet con técnicas inspiradas en Transformers, demostrando que CNNs puras pueden competir	Transformer jerárquico con atención en ventanas locales y desplazamiento para conexión entre ventanas	Combinación de convoluciones + atención local (block) + atención global diluida (grid) en cada bloque
Tipo de Arquitectura	CNN pura (sin atención)	Transformer puro (sin convoluciones en backbone)	Híbrido CNN + Transformer
Bloque Básico	Inverted bottleneck con depthwise conv 7×7, LayerNorm, GELU	Swin Transformer Block: Window-MSA → MLP → Shifted-Window-MSA → MLP	MBConv → Block Attention → Grid Attention
Mecanismo de Atención	× No usa atención (solo convoluciones)	✓ Multi-head Self-Attention en ventanas 7×7, con shift en capas alternas	✓ Block Attention (local en ventanas) + Grid Attention (global diluida)
Receptive Field Global	Solo en capas profundas (por apilamiento)	Gradual mediante shifted windows	✓ Desde las primeras capas (via Grid Attention)
Complejidad Computacional	O(n) lineal respecto a resolución	O(n) lineal (atención local en ventanas fijas)	O(n) lineal (block + grid attention)
Representación Jerárquica	✓ Sí (4 etapas con downsampling)	✓ Sí (patch merging entre etapas)	✓ Sí (estructura multi-escala)
Resolución de Entrada	224×224 (escalable)	224×224 (escalable a alta resolución)	224×224 a 512×512
Tamaños Típicos	• ConvNeXt-T (29M) • ConvNeXt-S (50M) • ConvNeXt-B (89M) • ConvNeXt-L (198M)	• Swin-T (29M) • Swin-S (50M) • Swin-B (88M) • Swin-L (197M)	• MaxViT-T (31M) • MaxViT-S (69M) • MaxViT-B (120M) • MaxViT-L (212M)
Precisión (ImageNet-1K)	• T: ∼82.1% • B: ∼83.8% • L: ∼84.3% • XL (21K): ∼87.8%	• T: ∼81.3% • B: ∼83.5% • L: ∼86.4% • L (21K): ∼87.3%	• T: ∼83.6% • B: ∼85.0% • L: ∼86.4% • L (21K): ∼89.5%
Velocidad de Entrenamiento	Rápido (operaciones de convolución optimizadas)	Moderado (overhead de atención)	Moderado-lento (múltiples tipos de atención)
Eficiencia de Inferencia	Excelente (convoluciones muy optimizadas en GPU)	Buena (atención local eficiente)	Buena (pero más complejo que los otros)
Backbone Universal	✓ Sí (detección, segmentación)	✓ Sí (muy popular para downstream tasks)	✓ Sí (clasificación, detección, segmentación)
Mejor Para	• Cuando se prefiere simplicidad de CNNs • Inferencia eficiente • Transfer learning	• Backbone para tareas densas • Estado del arte en detección/segmentación • Preentrenamiento a gran escala	• Máxima precisión • Cuando se necesita contexto global desde el inicio • Tareas que requieren relaciones de largo alcance
Disponibilidad	✓ torchvision + `timm`	✓ torchvision + `timm`	✓ `timm` (`maxvit_*`)
Debilidad Principal	Menor capacidad de modelar dependencias globales que Transformers	Conexión entre ventanas limitada por shifted windows	Mayor complejidad de implementación; menos maduro que Swin/ConvNeXt

Table 2. Comparación de Arquitecturas Modernas: ConvNeXt, Swin Transformer y MaxViT

5. **MBConv (Mobile Inverted Bottleneck)**: Utilizado en EfficientNet, este bloque combina convoluciones depthwise separables con una expansión y proyección de canales, lo que reduce significativamente el número de parámetros y operaciones computacionales. Se basa en que las capas más finas (bottlenecks) ya tienen la información necesaria, y las convoluciones depthwise son más eficientes.

6. **Fused-MBConv**: Introducido en EfficientNetV2, este bloque combina convoluciones regulares con depthwise en las primeras etapas de la red para mejorar la velocidad de entrenamiento sin sacrificar la precisión.

7. **Squeeze-and-Excitation (SE)**: Introdujo bloques de atención que recalibran dinámicamente las características de los canales, mejorando la capacidad del modelo para enfocarse en características relevantes.

Estas son algunas de las técnicas clave que han impulsado el desarrollo de modelos clásicos de clasificación de imágenes. Cada una de ellas ha contribuido a mejorar la precisión, eficiencia y capacidad de generalización de las redes neuronales convolucionales.

4.1.2 Modelos modernos

EL desarrollo de los Transformers ha revolucionado el campo del aprendizaje profundo, y su aplicación

a la visión por computadora ha llevado a la creación de modelos modernos que superan a las arquitecturas clásicas en muchas tareas. En la tabla **??** se presentan las comparaciones detalladas de las diferentes arquitecturas modernas estudiadas: ConvNeXt, Swin Transformer y MaxViT. Aquí se describen las características principales de cada modelo, sus ventajas y desventajas, y por qué se eligieron para este proyecto.

A continuación se describen brevemente las técnicas clave de estos modelos modernos:

1. **Convolución Depthwise Separable**: Fundamental en arquitecturas como ConvNeXt y EfficientNet (MBConv). Descompone una convolución estándar en dos operaciones más ligeras: una *depthwise* (que opera espacialmente canal por canal de forma independiente) y una *pointwise* (convolución 1×1 que mezcla la información de los canales). Esto reduce drásticamente la cantidad de parámetros y FLOPs, permitiendo aumentar el tamaño de la red sin disparar el costo computacional.

2. **LayerNorm (Normalización de Capa)**: A diferencia de BatchNorm, que normaliza a través del lote (batch), LayerNorm normaliza las estadísticas a través de la dimensión de los canales para cada muestra individualmente. Originalmente popular en NLP, se ha convertido en el estándar para Vision Transformers y ConvNeXt debido a su estabilidad durante el entrenamiento y su independencia del tamaño del batch, lo cual es crucial cuando se trabaja con imágenes de alta resolución donde los batches son pequeños.

3. **GELU (Gaussian Error Linear Unit)**: Es una función de activación más suave que la tradicional ReLU. Mientras que ReLU trunca abruptamente a cero los valores negativos, GELU los pondera por su magnitud basándose en una distribución normal probabilística. Esta curvatura suave facilita la optimización y el flujo de gradientes en redes muy profundas, siendo la activación por defecto en BERT, GPT, ViT y ConvNeXt.

4. **Atención por Ventanas Desplazadas (Shifted Window Attention)**: Es la innovación clave del Swin Transformer. Calcular la auto-atención global es costoso ($O(N^2)$). Swin limita la atención a ventanas locales fijas (costo lineal $O(N)$), pero para que las ventanas se comuniquen entre sí, desplaza (shiftea) la partición de las ventanas en capas alternas. Esto crea conexiones cruzadas que permiten que la información se propague globalmente a través de la imagen con una eficiencia computacional muy alta.

5. **Atención Multi-Eje (Multi-Axis Attention)**: Utilizada en MaxViT para resolver la falta de contexto global eficiente. Descompone la atención dispersa en dos pasos: *Block Attention* (local, que captura texturas y detalles cercanos) y *Grid Attention* (global y diluida, que atiende píxeles lejanos en una rejilla regular). Juntos, permiten un campo receptivo global desde las primeras capas con una complejidad lineal, superando las limitaciones de las ventanas puramente locales.

6. **Sesgo de Posición Relativa (Relative Position Bias)**: Los Transformers, al ser invariantes a la permutación, necesitan información posicional. En lugar de sumar embeddings de posición absolutos a la entrada (como en el ViT original), modelos como Swin y MaxViT inyectan un sesgo aprendible directamente en la matriz de atención basado en la distancia relativa entre píxeles. Esto mejora la generalización a diferentes resoluciones de imagen, permitiendo entrenar en un tamaño y testear en otro diferente.

7. **Stem "Patchify" y Downsampling Separado**: Las CNN clásicas reducían la dimensión agresivamente al inicio. Los modelos modernos imitan a los Transformers dividiendo la imagen en "parches" no superpuestos (usando una convolución, por ejemplo, de 4×4 con stride 4). Además, en lugar de usar *max pooling* dentro de los bloques, utilizan capas de *downsampling* dedicadas y separadas entre etapas (generalmente convoluciones con stride 2 y normalización) para estabilizar el entrenamiento profundo.

Estos avances representan la convergencia entre los mundos de las CNNs y los Transformers: las CNNs modernas (ConvNeXt) adoptan la "macroarquitectura" de los Transformers (GELU, LayerNorm, parches), mientras que los Transformers modernos (Swin, MaxViT) reintroducen la jerarquía y la localidad propias de las convoluciones.

4.2. Selección de Modelos Propuesta (Revisada)

Considerando la arquitectura del pipeline definido: *Identificación de Hoja (Fase 1)* → *Extracción de Lesión vía YOLO (Fase 2)* → *Identificación de Enfermedad en recorte (Fase 3)*, la selección de modelos debe adaptarse a la naturaleza de los datos de entrada en cada etapa.

4.2.1 Fase 1: Clasificación de Tipos de Planta

Entrada: Imagen completa de la hoja/planta (con fondo, tallo, etc.). **Modelo Elegido: EfficientNetV2 (Variante S)**

Justificación:

- **Reconocimiento de Formas:** Para distinguir entre tipos de plantas (ej. maíz vs. tomate), el modelo debe basarse en características geométricas de bajo y medio nivel (forma del borde, venación, estructura del tallo). Las redes convolucionales (CNNs) son naturalmente eficientes extrayendo estas características de forma.
- **Velocidad de Entrada:** Al ser el primer filtro, EfficientNetV2 ofrece la mejor latencia. Su uso de *Fused-MBConv* en las primeras capas procesa la imagen agresivamente sin perder información espacial relevante para la morfología de la planta.
- **Robustez al Fondo:** EfficientNetV2, gracias a su entrenamiento con aumentos progresivos, suele generalizar bien incluso si el fondo de la imagen varía, centrándose en la estructura principal del objeto.

4.2.2 Fase 3: Clasificación de Enfermedades (Sobre recortes)

Entrada: Imagen recortada ("macro") centrada específicamente en la lesión o síntoma, extraída previamente por YOLO. **Modelo Elegido: MaxViT (o ConvNeXt)**

Justificación del cambio de enfoque: Al trabajar con recortes (crops), eliminamos el ruido del fondo y la necesidad de buscar la enfermedad. El problema se convierte en distinción de texturas de alta frecuencia. Por ejemplo, distinguir entre *Tizón Temprano* (anillos concéntricos) y *Mancha Bacteriana* (puntos negros con halo amarillo).

Opción A: MaxViT (Recomendada por Precisión)

- **Híbrido Perfecto para Macro:** MaxViT inicia cada bloque con capas *MBConv* (convolucionales). Las convoluciones son excelentes detectores de texturas, bordes y gradientes de color, que son vitales en una fotografía "macro" de una enfermedad.
- **Relación Centro-Periferia:** Aunque el recorte es pequeño, la atención (*Block Attention*) permite al modelo relacionar el centro de la lesión (ej. tejido necrótico) con su periferia (ej. halo clorótico). Esta relación estructural es lo que define a muchas patologías, y MaxViT lo captura mejor que una CNN pura.

Opción B: ConvNeXt (Recomendada por Coherencia/Simplicidad)

- **Dominio de la Textura:** ConvNeXt es una arquitectura puramente convolucional pero modernizada. Históricamente, las CNNs superan a los Transformers puros (como ViT) en tareas donde la textura es más importante que la forma global. Al ser un recorte de la enfermedad, la "forma" importa menos que la "textura" del hongo o bacteria.
- **Invarianza:** Al trabajar con recortes de YOLO, la lesión puede no estar perfectamente centrada o tener distintos tamaños. ConvNeXt maneja muy bien estas variaciones sin necesidad de embeddings posicionales complejos.

4.2.3 Conclusión de la Selección

Se propone utilizar **EfficientNetV2** para la Fase 1 debido a su eficiencia en características morfológicas globales, y **MaxViT** para la Fase 3.

La elección de **MaxViT** sobre un Transformer puro (como Swin) para la Fase 3 es estratégica: al tener componentes convolucionales fuertes (MBConv), no pierde la capacidad de analizar la "rugosidad" y los detalles finos de la enfermedad, pero su mecanismo de atención le permite entender la estructura compleja de la lesión mejor que una CNN clásica. Es el modelo que mejor se adapta a la variabilidad visual de las enfermedades en imágenes de primer plano.

5. Datasets

En esta sección se detallan los conjuntos de datos utilizados para el entrenamiento y validación de los modelos propuestos. La correcta elección de estos datos es fundamental para el desempeño del sistema, dado que se busca integrar tareas de clasificación taxonómica y detección de patologías en un único pipeline.

5.1. Criterios de Selección

Para la conformación del dataset final, se han establecido una serie de requisitos estrictos derivados de la arquitectura del sistema (Clasificador + YOLO):

- **Compatibilidad con Arquitectura en Dos Etapas:** Se requieren datos que permitan entrenar tanto el modelo de clasificación de especies (Etapa 1) como el detector de objetos (Etapa 2). Esto implica la necesidad de imágenes con etiquetas a nivel de imagen (especie) y anotaciones a nivel de región (bounding boxes para lesiones).

- **Volumen y Diversidad:** Para entrenar arquitecturas basadas en Vision Transformers y CNNs profundas (como YOLO), es necesario un volumen de datos suficiente (en el orden de miles de imágenes por clase) para evitar el sobreajuste. Además, las imágenes deben presentar variabilidad en condiciones de iluminación, fondo y estadios de la enfermedad para garantizar la capacidad de generalización del modelo en entornos no controlados.

- **Calidad de las Anotaciones:** Es crítico que las anotaciones de las lesiones sean precisas y consistentes, ya que el rendimiento de YOLO depende directamente de la calidad de las cajas delimitadoras (ground truth) durante el entrenamiento.

5.2. Descripción de los Datasets

La búsqueda de conjuntos de datos adecuados para este estudio ha presentado desafíos significativos, principalmente debido a la escasez de datasets que cumplan simultáneamente con los requisitos de calidad de imagen, precisión en las anotaciones y especificidad taxonómica necesarios para nuestra arquitectura de dos etapas.

Para abordar esta limitación, se llevó a cabo una investigación exhaustiva en repositorios especializados como Roboflow y Kaggle, así como en la literatura científica reciente. La metodología de búsqueda se centró en identificar colecciones que no solo ofrecieran un volumen suficiente de imágenes, sino que también incluyeran anotaciones de tipo *bounding box* verificadas para tareas de detección de objetos, descartando aquellos datasets limitados únicamente a etiquetas de clasificación global que no permitirían el entrenamiento efectivo del modelo YOLO.

Tras este proceso de filtrado y validación, se han seleccionado los siguientes datasets públicos que mejor se alinean con los objetivos del proyecto. La Tabla **??** resume sus características principales.

Usando ambos datasets conjuntamente, se pueden cumplir los requisitos de compatibilidad con la arquitectura en dos etapas, ya que se dispone de imágenes con etiquetas a nivel de imagen (especie) y anotaciones a nivel de región (bounding boxes y segmentación para lesiones).

5.3. Análisis Exploratorio de Datos

Para garantizar la robustez de los modelos, se ha realizado un análisis de la distribución de clases y la variabilidad de las muestras.

5.3.1 Distribución de Clases

Como se observa en la Figura **??**, existe un desbalanceo significativo en ciertas clases de enfermedades, lo cual ha motivado el uso de técnicas de aumento de datos.

Figure 1. Histograma de frecuencia de imágenes por clase de enfermedad.

5.3.2 Análisis de Bounding Boxes

Para poder trabajar con el modelo YOLO, es necesario transformar las anotaciones de segmentación en bounding boxes. Para ello se ha usado la función findcontours de OpenCV https://docs.opencv.org/4.x/d4/d73/tutorial$_py_contours_begin.html(ta$ $//docs.pytorch.org/vision/main/generated/torchvision.ops.masks_t$

5.4. Preprocesamiento y Particionamiento

Las imágenes han sido redimensionadas a $X \times X$ píxeles para la etapa de detección y normalizadas utilizando la media y desviación estándar de ImageNet, ya que este dataset contiene una gran cantidad de imágenes naturales y por lo tanto ha generalizado correctamente los valores medios y desviaciones estándar de este tipo de imágenes.

El conjunto de datos se ha dividido siguiendo un esquema estratificado:

- **Entrenamiento (Train):** 70%
- **Validación (Val):** 15%
- **Prueba (Test):** 15%

6. Experiments

The data used, the experimental validation protocol, the metrics used, the experiments carried out, the results obtained, and their discussion are presented here.

6.1. Dataset

7. Conclusions

Section that presents, briefly and as a summary, the main conclusions of the work carried out. It also usually includes future possible works. That is, what are the most promising lines to continue with this work, as well as possible proposals for improvement. IMPORTANT: these are the scientific conclusions reached in the project; not your personal conclusions about the work you have done!

Table 3. Resumen técnico de los datasets seleccionados.

Dataset	**Imágenes**	**Especies**	**Clases (Enf.)**	**Anotación**
Apple Tree Leaf Disease Segmentation Dataset [**?**]	1641	1	5	Segmentation
Tomato Diseases [**?**]	3649	1	2	Bounding Box