Proyecto del curso #1

Carlos Andrés Capachero Martínez, Javier Enrique Huérfano Diaz, María Fernanda Izquierdo Aparicio

Introducción

En el ámbito de la inteligencia artificial y el aprendizaje profundo, la clasificación de imágenes representa un desafío esencial, con aplicaciones en múltiples sectores como la medicina, la agricultura y el comercio. El presente trabajo se centra en la clasificación de hojas de plantas en dos categorías: "saludables" y "enfermas". Esta tarea es vital para la agricultura contemporánea, ya que la detección temprana de enfermedades en las plantas puede influir significativamente en la productividad y la calidad de los cultivos.

Para abordar este desafío, se entrenaron y evaluaron cuatro modelos de redes neuronales convolucionales (CNN) con el objetivo de optimizar el rendimiento en términos de precisión, capacidad de generalización y reducción del sobreajuste. Cada modelo incorpora distintas técnicas y estrategias de regularización para mejorar su capacidad de generalización:

- **Modelo 1: CNN Sin regularización:** Este modelo básico se entrena sin aplicar técnicas de regularización, lo que permite evaluar el rendimiento de la CNN sin ninguna mitigación explícita del sobreajuste**.**

- **Modelo 2: Aumento de datos:** En este modelo se utiliza el aumento de datos (data augmentation), que consiste en crear nuevas muestras modificadas a partir de las imágenes originales, con el objetivo de incrementar la diversidad del conjunto de entrenamiento y mejorar la capacidad de generalización del modelo.

- **Modelo 3: Dropout y Aumento de datos:** En esta modelo, además del aumento de datos, se incorpora la regularización mediante dropout, que apaga aleatoriamente un porcentaje de las neuronas durante el entrenamiento para prevenir que la red se ajuste demasiado a los datos.

- **Modelo 4: Detención temprana y Aumento de datos:** Este modelo combina el aumento de datos con la técnica de detención temprana (early stopping), que interrumpe el entrenamiento cuando el rendimiento en el conjunto de validación deja de mejorar, evitando así el sobreentrenamiento.

A lo largo del presente trabajo, se abordarán los resultados obtenidos por cada modelo al responder preguntas planteadas, tambien se comparán sus rendimientos y se evaluará cuál de estas técnicas proporciona la mejor combinación de precisión y capacidad de generalización para clasificar hojas saludables y enfermas.

1. Preprocesamiento de Datos

Tarea:

Explique los pasos necesarios para preprocesar las imágenes antes de alimentarlas a una CNN. Esto incluye redimensionar, normalizar, aumentar datos y dividir los datos en conjuntos de entrenamiento, validación y prueba. Implemente al menos una tecnica de data augmentation

Los pasos que se realizaron para preprocesas las imágenes fueron (aplicados a 3 de los cuatro modelos realizados):

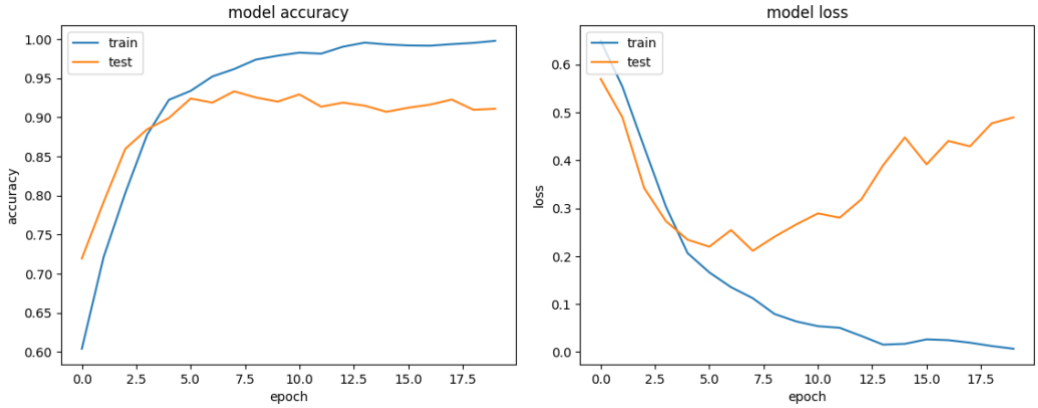
- **Redimensionado**: Todas las imágenes se redimensionan a 300×200 píxeles.
- **Normalización**: Los valores de los píxeles se reescalan entre 0 y 1 dividiendo por 255.
- **Data Augmentation**: Se utilizan técnicas como rotación, desplazamiento y volteo para generar más datos y evitar el sobreajuste en el modelo en las imágenes del entrenamiento del modelo.
- **División de los Datos**: El conjunto se divide en entrenamiento y validación utilizando train_test_split usando 80% para entrenamiento y 20% para validación.

Pregunta:

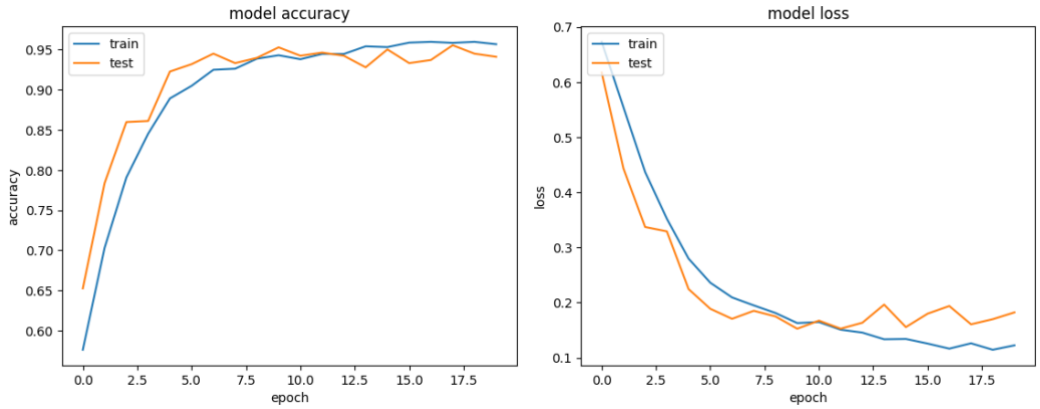
¿Qué impacto tiene la ampliación de datos (data augmentation), en el rendimiento de tu modelo CNN? Demuestre cuantitativamente.

El data augmentation mejora la generalización del modelo, reduciendo el sobreajuste. Esto se puede demostrar entrenando el modelo dos veces (con y sin data augmentation) y comparando las métricas como accuracy y pérdida en el conjunto de validación. Como se indica en la introducción, el primer modelo se realizó sin aplicar técnicas de data augmentation, mientras que el segundo ya las introduce. Estos dos modelos nos ayudan a evaluar el impacto de la ampliación de los datos, y en este caso se decide evaluar las métricas de accuracy y de pérdida. Los resultados fueron los siguientes:

Modelo 1: Sin Data Augmentation



Modelo 2: Con Data Augmentation



Con los resultados se puede evidenciar que el data augmentation tiene un impacto significativo en el rendimiento del modelo CNN, especialmente en su capacidad de generalización. Comparando los resultados se tiene que:

- **Modelo 1: Sin Data Augmentation:**
  - ***Accuracy:***
    - En el conjunto de entrenamiento, el modelo alcanza casi el 100% de accuracy, lo que es una señal clara de sobreajuste.
    - En el conjunto de prueba, el accuracy máximo es de aproximadamente 90%, pero a partir de la época 10, se observa una tendencia decreciente, lo que indica que el modelo no generaliza bien a los datos de prueba.
  - ***Pérdida (Loss):***
    - La pérdida en el conjunto de entrenamiento disminuye constantemente, mientras que en el conjunto de prueba,

comienza a aumentar después de la época 10, lo que sugiere sobreajuste.

- **Modelo 2: Con Data Augmentation:**
  - ***Accuracy:***
    - En el conjunto de entrenamiento, el accuracy es más estable, alcanzando un máximo cercano al 95%.
    - En el conjunto de prueba, el accuracy se mantiene cercana al accuracy del conjunto de entrenamiento, alrededor del 92%-93%, con menor tendencia al sobreajuste en comparación con el modelo sin data augmentation.
  - ***Pérdida (Loss):***
    - La pérdida en el conjunto de prueba disminuye significativamente y se mantiene baja, mientras que la pérdida en el conjunto de entrenamiento es también más estable y cercana a la pérdida del conjunto de prueba, lo que indica que el modelo generaliza mejor.

- **Impacto cuantitativo:**
  - ***Accuracy:*** El modelo con data augmentation mejora el accuracy en el conjunto de prueba en aproximadamente un 3% (de 90% a 93%), reduciendo el sobreajuste observado en el primer modelo.
  - ***Pérdida:*** La pérdida en el conjunto de prueba disminuye significativamente con data augmentation, manteniéndose baja y estable, mientras que en el modelo sin data augmentation, la pérdida aumenta notablemente después de la época 10.

En conclusión, la técnica de data augmentation mejora la capacidad del modelo para generalizar a los datos de prueba, reduciendo el sobreajuste y aumentando el accuracy de predicción en datos no vistos.

2. Diseño de la Arquitectura del Modelo

Tarea:
Diseñe una arquitectura de CNN adecuada para clasificar las imágenes de plantas en categorías de sanas o enfermas.

La arquitectura de la red neruonal convolucional de este proyecto tiene la siguiente estructura:

- **Capa 1: Conv2D con 32 filtros**
  La primera capa convolucional aplica 32 filtros de tamaño 3x3 con función de activación *ReLU* aplicada a las imágenes de entrada de tamaño 300x200 y 3

canales (RGB). Esta capa detecta características simples como bordes y texturas.
Se uso de una pequeña cantidad de filtros en esta primera capa ya que es una práctica adecuada para extraer características básicas sin sobrecargar la red con demasiados parámetros desde el inicio. El usar la función de activación *ReLU* añade no linealidad y permite que la red aprenda patrones complejos, al eliminar valores negativos.

- **Capa 2: MaxPooling2D**
  Esta segunda capa realiza una operación de pooling con un filtro 2x2 reduciendo la dimensionalidad espacial de la salida anterior de 298x198x32 a 149x99x32. Este capa de pooling reduce la resolución espacial, lo que ayuda a evitar el sobreajuste al reducir la cantidad de parámetros, manteniendo las características más importantes de las imágenes.

- **Capa 3: Conv2D con 64 filtros**
  Esta capa aplica 64 filtros de 3x3 a la salida de la capa de pooling. Con esta capa se espera que la red está capacitada para aprender características más complejas a partir de las características básicas extraídas anteriormente.
  Se aumenta en el número de filtros (de 32 a 64) ya que con esto la red logra capturar más patrones a medida que las características se vuelven más complejas, como formas y contornos de las plantas.

- **Capa 4: MaxPooling2D**
  De forma similar a la segunda capa de pooling, esta cuarta capa reduce la resolución espacial de 147x97x64 a 73x48x64. En esta capa se sigue reduciendo la dimensionalidad para evitar un exceso de parámetros y enfocarse en las características más importantes.

- **Capa 5: Conv2D con 64 filtros**
  En esta capa se agrega otra capa convolucional con 64 filtros tambien de 3x3 permitiendo a la red profundizar en características más complejas. Se continua con el uso de 64 filtros ya que es adecuado en este punto porque ayuda a consolidar las características intermedias extraídas previamente, permitiendo una mayor especialización en detalles más abstractos.

- **Capa 6: MaxPooling2D**
  Nuevamente se realiza pooling para reducir la dimensionalidad de la salida a 35x23x64. Con esta capa se mantiene el enfoque en la extracción de características importantes y se trata deevitar el sobreajuste, al tiempo que se simplifica el procesamiento posterior.

- **Capa 7: Conv2D con 128 filtros**
  Esta capa aplica 128 filtros de 3x3, lo que permite capturar aún más características complejas en la imagen. A medida que las imágenes se reducen dimensionalmente, aumentar el número de filtros nos permite que la red capture información más detallada y compleja, lo que es crucial en la clasificación de imágenes en dos categorías (planta saludable vs. Planta enferma).

- **Capa 8: MaxPooling2D**
  Se introduce esta capa para reducir la salida final convolucional a 16x10x128. Con esta capa se continúa reduciendo la dimensionalidad con el fin de preparar la salida para la capa densa final, manteniendo soloamente las características más representativas.

- **Capa 9: Flatten**
  Esta capa convierte la salida tridimensional en un vector unidimensional de tamaño 20480, para que pueda ser procesado por capas densas. Esto se realiza ya que es necesario aplanar los datos antes de pasarlos a una capa densa completamente conectada.

- **Capa 10: Dense con 64 unidades**
  Capa densa con 64 unidades completamente conectadas y función de activación *ReLU*. Esta capa aprende combinaciones complejas de características extraídas por las capas convolucionales anteriores y es crucial para la combinación de características, lo que permite a la red tomar decisiones más informadas sobre la clasificación.

- **Capa 11: Dense con 1 unidad**
  Esta es la capa final de una unidad con función de activación *sigmoid* para obtener una salida binaria (0 o 1) que indica si la hoja de la planta es "saludable" o "enferma". Se usa la función de activación *sigmoid* ya que es ideal para clasificaciones binarias a razón de que devuelve una probabilidad entre 0 y 1.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 298, 198, 32)      896

 max_pooling2d (MaxPooling2D  (None, 149, 99, 32)      0
 )

 conv2d_1 (Conv2D)           (None, 147, 97, 64)       18496

 max_pooling2d_1 (MaxPooling  (None, 73, 48, 64)       0
 2D)

 conv2d_2 (Conv2D)           (None, 71, 46, 64)        36928

 max_pooling2d_2 (MaxPooling  (None, 35, 23, 64)       0
 2D)

 conv2d_3 (Conv2D)           (None, 33, 21, 128)       73856

 max_pooling2d_3 (MaxPooling  (None, 16, 10, 128)      0
 2D)

 flatten (Flatten)           (None, 20480)             0

 dense (Dense)               (None, 64)                1310784

 dense_1 (Dense)             (None, 1)                 65

=================================================================
Total params: 1,441,025
Trainable params: 1,441,025
Non-trainable params: 0
```

Pregunta:

Justifique la elección del número de capas, tipos de capas y funciones de activación en la arquitectura de la CNN propuesta.

Justificación de la Arquitectura:

- **Número de Capas:** La elección de 4 capas convolucionales seguidas de capas de pooling está basada en la necesidad de extraer características de diferentes niveles de abstracción. Las primeras capas capturan patrones simples, mientras que las más profundas capturan patrones complejos.
- **Capas Convolucionales:** Las capas convolucionales con filtros de 3x3 nos permiten captar características locales en las imágenes. El incremento en el número de filtros en capas sucesivas ayuda a capturar una mayor variedad de patrones y detalles.
- **Max-Pooling:** Ayuda a reducir la dimensionalidad, mejorar la eficiencia del modelo y disminuir el riesgo de sobreajuste.
- **Capas Densas:** La primera capa densa con 64 neuronas aprende combinaciones complejas de las características extraídas, mientras que la capa final con 1 neurona y función de activación *sigmoid* es adecuada para la tarea de clasificación binaria.

- **Funciones de Activación:** Usar *ReLU* evita problemas de desvanecimiento del gradiente y acelera la convergencia, mientras que la función *sigmoid* en la última capa es adecuada para convertir la salida a una probabilidad entre 0 y 1.

Esta arquitectura se diseñó con el fin de que fuera equilibrada para la tarea de clasificación binaria de imágenes, optimizando el número de parámetros y garantizando una buena generalización sin ser excesivamente compleja.

3. Entrenamiento del Modelo CNN

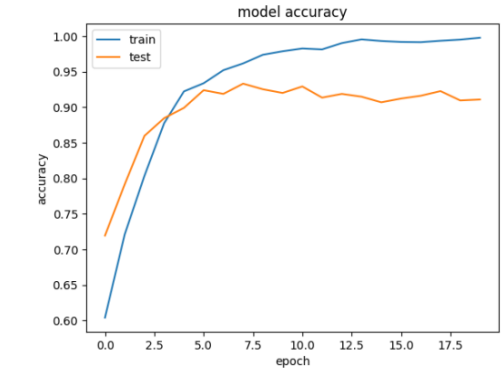
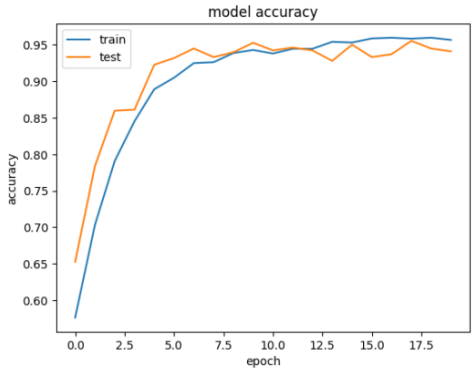
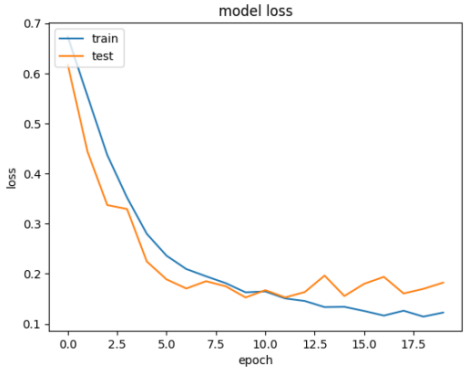
Tarea:
Entrene su modelo CNN en el conjunto de datos y monitoree el proceso de usando métricas. Justifique la metrica y funcion de perdida escogida.

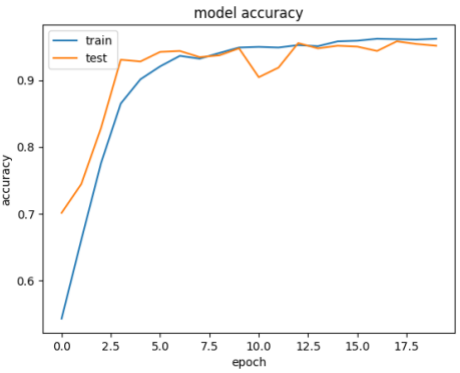
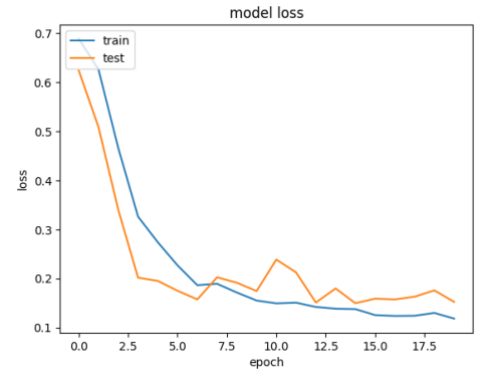
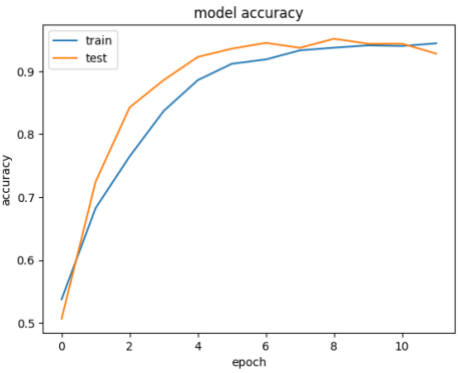
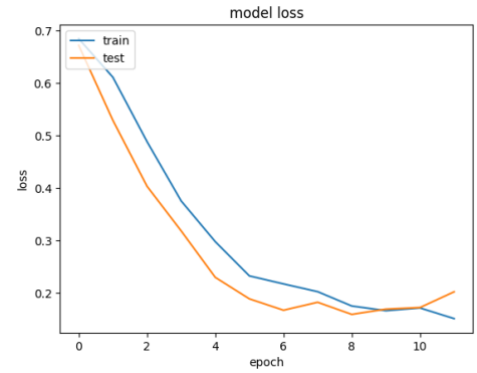
Justificación y evolución de la Métrica y Función de Pérdida:

- **Métrica**: El *accuracy* es la proporción de predicciones correctas con respecto al total de predicciones, es una métrica intuitiva y directa para evaluar modelos de clasificación binaria como en este caso, donde se intenta clasificar entre hojas saludables y enfermas. En este contexto, el accuracy es útil ya que queremos maximizar la cantidad de aciertos en la clasificación de ambas categorías (hojas saludables y enfermas).
    - Evolución a lo largo de los modelos:
        - En el Modelo 1 (sin regularización), el *accuracy* es alto en el conjunto de entrenamiento pero baja en el conjunto de prueba, lo que indica sobreajuste.
        - En el Modelo 2 (con data augmentation), el *accuracy* mejora en el conjunto de prueba, demostrando una mejor generalización del modelo.
        - En el Modelo 3 (Dropout + Data Augmentation), el *accuracy* se mantiene estable en ambos conjuntos, lo que indica un balance más adecuado entre la capacidad de ajuste y la generalización.
        - En el Modelo 4 (Early Stopping + Data Augmentation), el rendimiento mejora, deteniendo el entrenamiento antes de que el modelo comience a sobreajustarse.

- **Función de Pérdida:** Dado que se trata de un problema de clasificación binaria (hoja de saludable u hoja de planta enferma), se usa la entropía cruzada binaria (*binary crossentropy*), una función que mide la disimilitud entre las predicciones del modelo y las etiquetas verdaderas. Esta función penaliza predicciones erróneas y ayuda a que el modelo aprenda a ajustar las probabilidades a lo largo de las épocas.

- Evolución a lo largo de los modelos:
  - En el Modelo 1, la pérdida disminuye en el conjunto de entrenamiento, pero aumenta en el conjunto de prueba, indicando que el modelo se ajusta demasiado a los datos de entrenamiento (sobreajuste).
  - En el Modelo 2, la pérdida en el conjunto de prueba se estabiliza y disminuye, mostrando que el aumento de datos ayuda a mejorar la generalización.
  - En el Modelo 3, la combinación de dropout y aumento de datos mantiene la pérdida baja en el conjunto de prueba, lo que indica una mejor capacidad de generalización.
  - En el Modelo 4, el early stopping garantiza que la pérdida en el conjunto de prueba no se incremente, deteniendo el entrenamiento antes de que ocurra el sobreajuste.

Modelo	Accuracy	Binary Crossentropy
1		
2		

3		
4		

Pregunta:
¿Cómo cambia el rendimiento del modelo al ajustar hiperparámetros como la tasa de aprendizaje, el tamaño del lote y el número de épocas?

Impacto de Ajustar Hiperparámetros:

- **Tasa de aprendizaje (learning rate)**: La tasa de aprendizaje controla el tamaño de los pasos que el optimizador da al ajustar los pesos del modelo. Si es muy alta, el modelo puede converger demasiado rápido sin encontrar un óptimo local; si es muy baja, el entrenamiento puede ser lento y caer en mínimos locales.
    - Impacto:
        - Una tasa de aprendizaje alta podría causar fluctuaciones en el accuracy y la pérdida, mientras que una tasa demasiado baja puede hacer que el modelo tarde demasiado en aprender o no logre ajustarse adecuadamente.
        - Ajustando este hiperparámetro, especialmente con técnicas como early stopping, se logra un mejor balance entre la rapidez y la estabilidad en la convergencia.

- **Tamaño del lote (batch size)**: Un tamaño de lote más pequeño tiende a hacer que el modelo aprenda de manera más precisa en cada actualización de pesos, pero con mayor variabilidad, mientras que un tamaño de lote más grande tiende a generar actualizaciones de pesos más estables pero menos precisas.
    - Impacto:
        - En los modelos sin regularización (como el Modelo 1), un lote pequeño puede aumentar el sobreajuste, ya que el modelo se adapta demasiado a los pequeños cambios en los datos de entrenamiento.
        - En los modelos con técnicas de regularización (como Dropout y Data Augmentation), un tamaño de lote mayor puede generar un entrenamiento más estable.
- **Número de épocas (epochs)**: Controlar el número de épocas es crucial para evitar el sobreajuste. Si el entrenamiento se prolonga demasiado, el modelo podría ajustarse en exceso a los datos de entrenamiento.
    - Impacto:
        - En el Modelo 4, se utilizó early stopping, lo que permite optimizar el número de épocas deteniendo el entrenamiento cuando la métrica de validación deja de mejorar. Esto evita el sobreentrenamiento y mejora la generalización.

4. Sobreajuste y Regularización

Tarea:
Identifique signos de sobreajuste en tu modelo durante el entrenamiento.

Signos de Sobreajuste:

- Un alto accuracy en el conjunto de entrenamiento pero bajo en validación indica sobreajuste.
- Aumento del Lost en set de validación a medida que incrementan las epochs mientras el lost de entrenamiento disminuye

Pregunta:
¿Qué técnicas de regularización se pueden aplicar para mitigar el sobreajuste y cómo afectan estas técnicas al rendimiento del modelo? Implemente al menos una técnica.

Técnicas de Regularización:

- **Data Augmentation:** Al aumentar los datos de entrenamiento mediante transformaciones como rotaciones, volteos y zooms, el modelo se expone a una mayor variedad de datos, lo que le ayuda a generalizar mejor. En los Modelos 2,

3 y 4, esta técnica reduce el sobreajuste al hacer que el modelo aprenda a partir de un conjunto de datos más diverso.

- **Dropout:** Introducido en el Modelo 3, el dropout apaga aleatoriamente un porcentaje de las neuronas en cada capa durante el entrenamiento. Esto fuerza al modelo a no depender excesivamente de neuronas específicas, lo que mejora su capacidad de generalización. Aunque introduce cierta variabilidad, el dropout es efectivo para evitar el sobreajuste.
- **Early Stopping:** En el Modelo 4, se utiliza early stopping para detener el entrenamiento cuando la métrica de validación deja de mejorar, previniendo el sobreajuste. Este método es muy eficaz cuando se combina con data augmentation, ya que garantiza que el modelo no entrene en exceso y se ajuste únicamente a patrones relevantes.
- **L1/L2 Regularization**: Otra técnica que no se implementó en nuestro trabajo, pero que también es importante es la regularización de Lasso (L1) la cual penaliza los pesos grandes promoviendo la sparsidad, mientras que la de Rigde (L2) penaliza los pesos grandes en general, tendiendo a disminuirlos. Ambas evitan el sobreajuste en CNN al reducir la complejidad del modelo.

5. Evaluación del Modelo

Tarea:

Evalue el modelo entrenado en el conjunto de prueba utilizando métricas como precisión, exactitud, recall y F1-score.

Las métricas que se pueden extraer de los resultados proporcionados para cada uno de los modelos incluyen **precisión (precision)**, **exactitud (accuracy)**, **recall** y el **F1-score**. Obtenemos las siguientes métricas para cada modelo:

Modelo	Resultados
1	precision recall f1-score support 0 0.98 0.98 0.98 1904 1 0.98 0.98 0.98 1908 accuracy 0.98 3812 macro avg 0.98 0.98 0.98 3812 weighted avg 0.98 0.98 0.98 3812
2	precision recall f1-score support 0 0.94 0.98 0.96 1904 1 0.98 0.94 0.96 1908 accuracy 0.96 3812 macro avg 0.96 0.96 0.96 3812 weighted avg 0.96 0.96 0.96 3812

3	precision recall f1-score support 0 0.97 0.97 0.97 1904 1 0.97 0.97 0.97 1908 accuracy 0.97 3812 macro avg 0.97 0.97 0.97 3812 weighted avg 0.97 0.97 0.97 3812
4	precision recall f1-score support 0 0.95 0.96 0.95 1904 1 0.96 0.95 0.95 1908 accuracy 0.95 3812 macro avg 0.95 0.95 0.95 3812 weighted avg 0.95 0.95 0.95 3812

Modelo 1:

- **Precisión (precision)**: 0.98 tanto para la clase 0 (saludable) como para la clase 1 (enferma).
- **Exactitud (accuracy)**: 0.98.
- **Recall**: 0.98 para ambas clases.
- **F1-score**: 0.98 para ambas clases.

Este modelo muestra un buen rendimiento, con métricas muy altas para todas las clases. La precisión, recall y F1-score de 0.98 indican que el modelo tiene un alto nivel de exactitud y balance entre las predicciones verdaderas positivas y falsas negativas.

Modelo 2 (con Data Augmentation):

- **Precisión (precision)**: 0.94 para la clase 0 y 0.98 para la clase 1.
- **Exactitud (accuracy)**: 0.96.
- **Recall**: 0.98 para la clase 0 y 0.94 para la clase 1.
- **F1-score**: 0.96 para ambas clases.
-

Este modelo también tiene un buen rendimiento general, pero notamos una pequeña disminución en las métricas comparado con el Modelo 1. Es interesante ver que la precisión para la clase 0 es un poco más baja (0.94), lo que podría indicar que algunas hojas saludables se clasificaron erróneamente como enfermas.

Modelo 3 (Dropout + Data Augmentation):

- **Precisión (precision)**: 0.97 para ambas clases.
- **Exactitud (accuracy)**: 0.97.
- **Recall**: 0.97 para ambas clases.
- **F1-score**: 0.97 para ambas clases.

El rendimiento de este modelo es bastante balanceado. Al incluir dropout y data augmentation, vemos que las métricas son muy similares entre las dos clases, lo que indica que el modelo está generalizando bien y tiene un buen rendimiento en términos de precisión, recall y F1-score.

Modelo 4 (Early Stopping + Data Augmentation):

- **Precisión (precision)**: 0.95 para la clase 0 y 0.96 para la clase 1.
- **Exactitud (accuracy)**: 0.95.
- **Recall**: 0.96 para la clase 0 y 0.95 para la clase 1.
- **F1-score**: 0.95 para ambas clases.

Aunque el rendimiento sigue siendo bueno, el Modelo 4 muestra una ligera disminución en todas las métricas en comparación con los modelos anteriores. El uso de early stopping previene el sobreajuste, pero quizás detuvo el entrenamiento antes de que el modelo alcanzara su máximo potencial.

Pregunta:
¿Cómo se desempeña el modelo en las diferentes clases (saludable vs. enfermas)?

- **Clase 0 (Saludable):** En todos los modelos, la clase 0 tiene una alta precisión, pero el Modelo 2 (con data augmentation) presenta una precisión ligeramente más baja (0.94). Esto indica que el modelo tiene más dificultad para identificar hojas saludables correctamente en comparación con las hojas enfermas. Sin embargo, su recall es alto en todos los modelos, lo que significa que la mayoría de las hojas saludables son identificadas correctamente.
- **Clase 1 (Enferma):** La clase 1, que corresponde a las hojas enfermas, generalmente tiene un rendimiento mejor que la clase 0, con una precisión y recall cercanos a 0.98 en los modelos 1 y 3. Esto sugiere que el modelo es particularmente bueno en identificar hojas enfermas y en minimizar los falsos negativos.
- En general se puede ver que:
    - El Modelo 1 es el mejor en términos de precisión, recall y F1-score.
    - El Modelo 2 con data augmentation tiende a clasificar erróneamente algunas hojas saludables, pero sigue siendo muy preciso en la clasificación de hojas enfermas.
    - Modelo 3 es el más balanceado, con buen rendimiento en ambas clases gracias a la combinación de dropout y data augmentation.
    - Modelo 4, aunque previene el sobreajuste con early stopping, tiene un rendimiento levemente inferior en comparación con los otros modelos.

6. Análisis de la Matriz de Confusión

Tarea:

Genere una matriz de confusión para las predicciones de su modelo en el conjunto de prueba.

Matrices de confusión para cada modelo realizado:

Modelo	Matriz de Confusión
1	tf.Tensor([[1872 32] [36 1872]], shape=(2, 2), dtype=int32)
2	tf.Tensor([[1860 44] [121 1787]], shape=(2, 2), dtype=int32)
3	tf.Tensor([[1844 60] [64 1844]], shape=(2, 2), dtype=int32)
4	tf.Tensor([[1832 72] [103 1805]], shape=(2, 2), dtype=int32)

Pregunta:

Analice la matriz de confusión. ¿Qué se puede concluir sobre las fortalezas y debilidades del modelo?

Matriz Modelo 1:

- **Fortalezas**:
  - El modelo clasifica correctamente a la gran mayoría de las muestras de ambas clases.
  - *Verdaderos Positivos (TP):* 1872 enfermos clasificados correctamente como enfermos.
  - *Verdaderos Negativos (TN):* 1872 saludables clasificados correctamente como saludables.
- **Debilidades**:
  - *Falsos Positivos (FP):* 32 saludables mal clasificados como enfermos.
  - *Falsos Negativos (FN):* 36 enfermos mal clasificados como saludables.

Este modelo tiene muy pocos errores en ambas clases, lo que indica un buen rendimiento general. Sin embargo, hay una ligera tendencia a clasificar incorrectamente algunas muestras saludables como enfermas y viceversa.

Matriz Modelo 2:

- **Fortalezas**:
  - *Verdaderos Positivos (TP):* 1787 enfermos clasificados correctamente como enfermos.
  - *Verdaderos Negativos (TN):* 1860 saludables clasificados correctamente como saludables.
- **Debilidades**:
  - *Falsos Positivos (FP):* 44 saludables mal clasificados como enfermos.
  - *Falsos Negativos (FN):* 121 enfermos mal clasificados como saludables.

Este modelo tiene más errores de **falsos negativos** (121), lo que significa que tiene más dificultades en identificar correctamente las hojas enfermas. Este problema puede ser grave en un contexto real, donde detectar enfermedades es crucial.

Matriz Modelo 3:

- **Fortalezas**:
  - *Verdaderos Positivos (TP):* 1844 enfermos clasificados correctamente como enfermos.
  - *Verdaderos Negativos (TN):* 1844 saludables clasificados correctamente como saludables.
- **Debilidades**:
  - *Falsos Positivos (FP):* 60 saludables mal clasificados como enfermos.
  - *Falsos Negativos (FN):* 64 enfermos mal clasificados como saludables.

En este modelo, aunque el número de falsos negativos ha disminuido en comparación con el segundo modelo, todavía hay un número mayor de falsos positivos, lo que indica que el modelo es más propenso a clasificar hojas saludables como enfermas.

Matriz Modelo 4:

- **Fortalezas**:
  - *Verdaderos Positivos (TP):* 1805 enfermos clasificados correctamente como enfermos.
  - *Verdaderos Negativos (TN):* 1832 saludables clasificados correctamente como saludables.
- **Debilidades**:
  - *Falsos Positivos (FP):* 72 saludables mal clasificados como enfermos.
  - *Falsos Negativos (FN):* 103 enfermos mal clasificados como saludables.

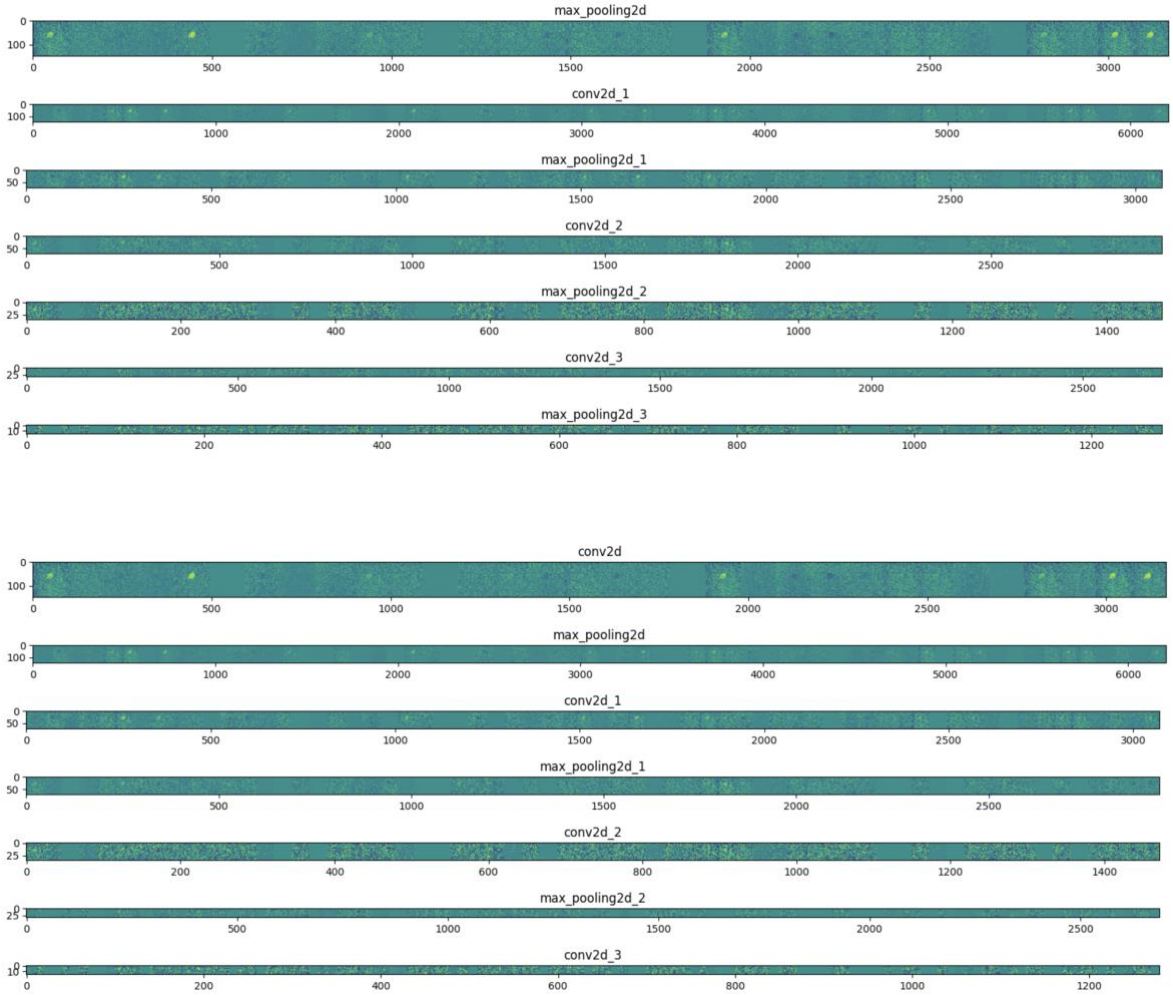
Este modelo tiene un número mayor de falsos negativos y falsos positivos en comparación con los modelos anteriores, lo que indica una menor capacidad para

distinguir correctamente entre hojas saludables y enfermas. En general, parece tener un peor desempeño que los otros modelos.

7. Visualización de Mapas de Características

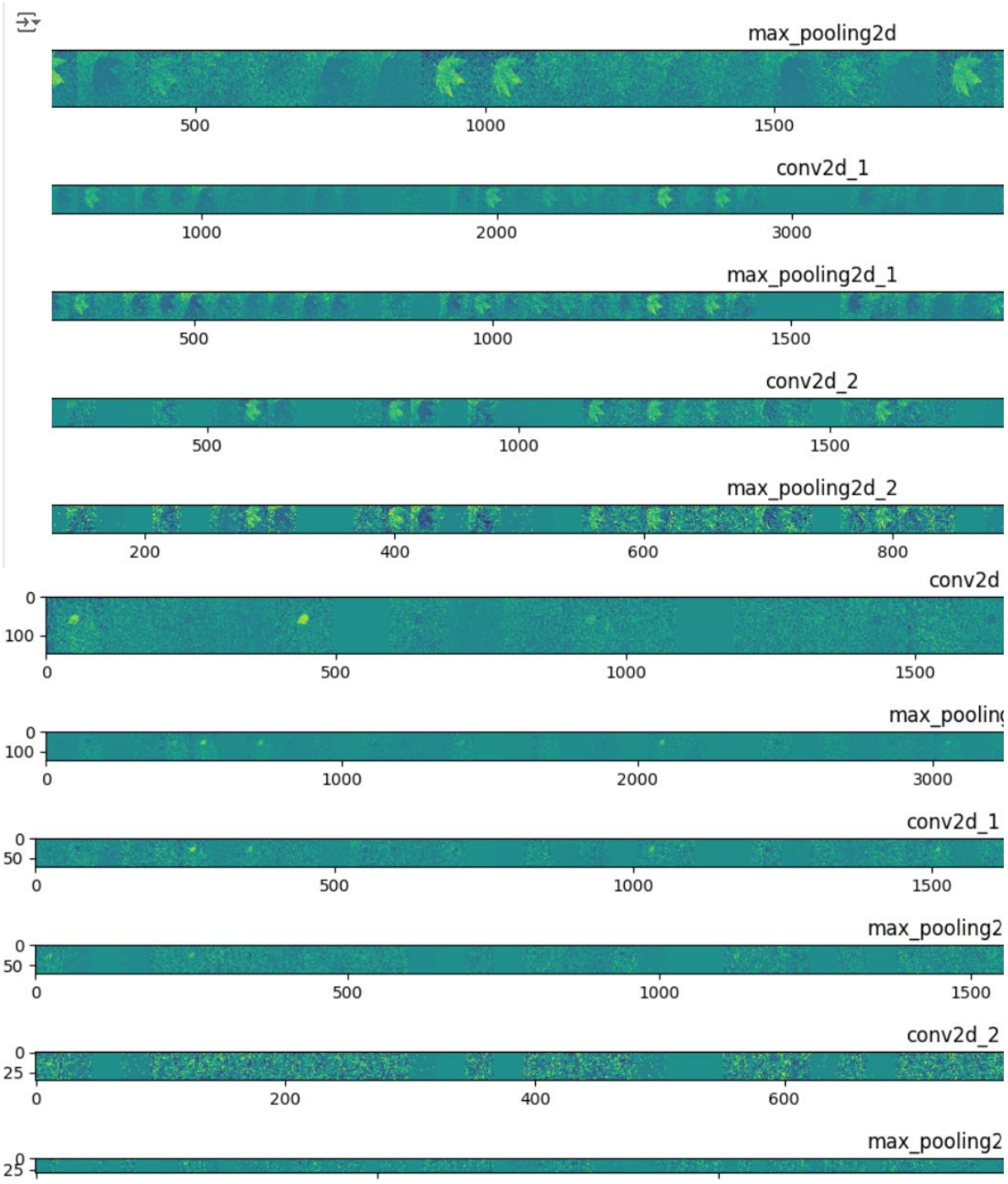
Tarea:

Visualice los mapas de características (salida de capas intermedias) de la CNN para algunas imágenes de muestra.



Pregunta:

¿Qué revelan los mapas de características sobre las características que la CNN está aprendiendo en diferentes capas?



En las primeras capas observadas se evidencia los mapas de características, en estos se observan activaciones en áreas que corresponden a bordes y contornos de la imagen. La imagen del archivo 10.jpg se evidencia claramente en la mayoria de las ventanas o filtros aplicados en la primera capa. A medida que se avanza a capas intermedias, los mapas resalnar texturas y formas básicas, aunque se sigue

evidenciando la forma de la hoja, algunas áreas de color homogéneo o pequeños patrones repetitivos pueden ser observados, en las capas más profundas, los mapas de características capturan información o patrones más abstractos y lo que se observaba en las primeras capas no resulta tan evidente en las capas profundas.

En este sentido, la visualización de la capas convolucionales y de maxpooling revelan que en efecto la red está aprendiendo y reconociendo formas básicas que posteriormente son refinadas en las capas más profundas de la red.

8. Consideraciones para el Despliegue

Tarea:
Discuta los pasos involucrados en el despliegue del modelo CNN entrenado en una aplicación del mundo real, como una aplicación móvil para agricultores.

Implementar un modelo de red neuronal convolucional en una aplicación del mundo real, como una app móvil para agricultores, requiere seguir varios pasos importantes que van desde la preparación del modelo hasta su integración final para que los usuarios puedan acceder a él. A continuación, detallamos los pasos que creemos son clave en este proceso:

1) **Preparación del Modelo para el Despliegue**
   Antes de lanzar el modelo, es esencial ajustarlo para que funcione en un entorno de producción:
   a) Conversión del modelo: El modelo entrenado en plataformas como TensorFlow o PyTorch debe transformarse a un formato más ligero y eficiente que pueda ejecutarse en dispositivos móviles. Herramientas como TensorFlow Lite o CoreML (para iOS) pueden ser útiles para optimizar el modelo y adaptarlo a hardware con recursos limitados.
   b) Compresión del modelo: Para disminuir el tamaño del modelo y acelerar su rendimiento, se pueden aplicar técnicas como cuantización o pruning. Estas técnicas permiten reducir la precisión de los pesos o eliminar conexiones innecesarias sin afectar significativamente la precisión general del modelo.

2) **Infraestructura en la Nube o Local**
   La ejecución del modelo puede realizarse en la nube o localmente, dependiendo de los recursos disponibles y la conectividad del usuario:
   a) Ejecución en la nube: Si el modelo es grande o necesita mucha capacidad de cómputo, se puede ejecutar en servidores en la nube. La aplicación móvil capturará imágenes de las plantas y las enviará a la nube para su procesamiento, recibiendo luego los resultados de clasificación (ya sea hoja sana o enferma)

b) Ejecución local: En áreas con conectividad limitada, es más conveniente ejecutar el modelo directamente en el dispositivo móvil (computación en el borde). Esto requiere un modelo optimizado para funcionar eficientemente en el hardware disponible (CPU o GPU móviles).

3) **Interfaz de Usuario (UI) y Experiencia de Usuario (UX)**
   a) Diseño de la interfaz: La interfaz debe ser simple e intuitiva, permitiendo a los agricultores capturar imágenes de las plantas y recibir respuestas inmediatas. Debe mostrar claramente el estado de la planta (saludable o enferma) y ofrecer recomendaciones o acciones correctivas.
   b) Manejo de imágenes: La aplicación debe gestionar adecuadamente el proceso de captura, carga y preprocesamiento de imágenes (como redimensionado y normalización). En este punto es crucial que las imágenes capturadas tengan calidad suficiente para que el modelo pueda clasificarlas correctamente.

4) **Consideraciones de Conectividad y Latencia**
   a) Conectividad intermitente: Dado que los agricultores pueden trabajar en áreas con poca o ninguna conexión a internet, la aplicación debe ser capaz de funcionar offline y realizar predicciones localmente, almacenando los resultados hasta que se restablezca la conexión.
   b) Latencia: Si el modelo se ejecuta en la nube, la latencia puede ser un problema. Es importante optimizar tanto la transmisión de imágenes como la respuesta del servidor para asegurar una experiencia fluida al usuario.

5) **Evaluación y Retroalimentación Continua**
   a) Actualizaciones del modelo: Con el tiempo, se puede mejorar el rendimiento del modelo al recopilar más datos reales (imágenes de plantas en diversas condiciones). Para esto, es necesario establecer un mecanismo para actualizar periódicamente el modelo basado en nuevos datos y así mejorar su capacidad de generalización.
   b) Manejo de errores: La aplicación debe incluir formas para que los agricultores den su opinión si hay errores en la clasificación (por ejemplo, si una planta se clasifica incorrectamente como enferma), esto permitirá enriquecer el conjunto de datos y mejorar el modelo con el tiempo.

6) **Consideraciones de Seguridad y Privacidad**
   a) Seguridad de los datos: Las imágenes tomadas por los usuarios pueden contener información sensible, por lo que es fundamental proteger la privacidad. Para ello, se deben implementar medidas como la encriptación de datos tanto en tránsito como en reposo.

b) Consentimiento de los usuarios: Es importante asegurarse de que los agricultores comprendan y acepten cómo se utilizarán las imágenes y datos que proporcionen.

7) **Pruebas y Validación en el Mundo Real**
   a) Pruebas de campo: Antes de un posible lanzamiento masivo, es necesario realizar pruebas exhaustivas con agricultores para garantizar que el modelo funcione correctamente bajo diversas condiciones y con diferentes dispositivos móviles.
   b) Métricas de rendimiento en producción: Se deben monitorear métricas como la tasa de errores, velocidad de inferencia y satisfacción del usuario para identificar problemas potenciales y optimizar la experiencia.

8) **Mantenimiento y Actualización del Sistema**
   a) Monitoreo continuo: Es crucial establecer un sistema para detectar posibles fallos tanto en la aplicación como en el modelo. Esto incluye identificar nuevas enfermedades o condiciones ambientales que no hayan sido contempladas durante el entrenamiento inicial.
   b) Actualizaciones periódicas: La aplicación y el modelo deben actualizarse regularmente para mejorar su rendimiento, añadir nuevas funcionalidades y mantener la seguridad.

Pregunta:
¿Qué desafíos podrían surgir al desplegar el modelo y cómo los abordaría?

1) **Rendimiento en Dispositivos de Bajo Costo**
   a) Desafío: Muchos agricultores utilizan dispositivos móviles con limitaciones de procesamiento, lo que puede afectar el rendimiento del modelo de CNN.
   b) Estrategia: Optimizar el modelo a través de cuantización (reducción de precisión de los pesos), pruning (eliminación de conexiones innecesarias) y técnicas de compresión y usar versiones ligeras del modelo con herramientas como TensorFlow Lite o CoreML permitirá un mejor rendimiento en hardware móvil.

2) **Conectividad Limitada o Inexistente**
   a) Desafío: La falta de internet en áreas rurales puede dificultar la transmisión de imágenes y la obtención de predicciones.
   b) Estrategia: Implementar inferencia local, ejecutando el modelo en el dispositivo móvil, y permitir que la aplicación funcione en modo offline, almacenando predicciones hasta que haya conexión. Esto implica pre-entrenar modelos ligeros que se puedan descargar y actualizar ocasionalmente.

3) **Recolección de Datos de Calidad**
   a) Desafío: Capturar imágenes de alta calidad puede ser difícil debido a condiciones ambientales como luz y clima.
   b) Estrategia: Integrar técnicas de preprocesamiento en la aplicación para ajustar automáticamente el contraste, brillo y enfoque. Además, entrenar el modelo con un conjunto diverso de datos que incluya diferentes condiciones ayudará a mejorar su precisión.

4) **Sobreajuste a Condiciones Específicas**
   a) Desafío: Un modelo puede funcionar bien en condiciones controladas pero fallar en situaciones reales con variaciones ambientales.
   b) Estrategia: Entrenar el modelo con un conjunto amplio y variado de datos que refleje diferentes entornos. De igual forma, continuar implementando técnicas de data augmentation para aumentar la diversidad del conjunto de datos también mejorará la generalización.

5) **Actualizaciones Continuas del Modelo**
   a) Desafío: El modelo necesita actualizarse regularmente para adaptarse a nuevos datos y cambios en las condiciones agrícolas.
   b) Estrategia: Establecer un sistema de aprendizaje continuo que utilice datos recopilados por los usuarios para mejorar el modelo periódicamente. Crear un pipeline robusto para recolectar, etiquetar y validar nuevos datos asegurará un retraining efectivo sin interrumpir el servicio.

6) **Seguridad y Privacidad de los Datos**
   a) Desafío: Manejar imágenes y datos sensibles presenta riesgos para la privacidad y seguridad.
   b) Estrategia: Implementar medidas de seguridad, como cifrado de extremo a extremo para las transmisiones y almacenamiento seguro en la nube o el dispositivo. Cumplir con normativas de privacidad y ser transparentes sobre el uso de los datos es fundamental.

7) **Mantenimiento y Escalabilidad**
   a) Desafío: A medida que crece la base de usuarios, la infraestructura debe ser capaz de manejar más imágenes y solicitudes.
   b) Estrategia: Adoptar una arquitectura en la nube que sea automáticamente escalable (por ejemplo, utilizando AWS o Google Cloud) permitirá gestionar un aumento en usuarios sin comprometer el rendimiento. Configurar sistemas de monitoreo y alertas automáticas ayudará a detectar problemas rápidamente.