Paddy Doctor: Paddy Disease Classification – Kaggle.

Ferley José Silva Jiménez, Jonatan Jair Leal González
ferley.silvaj@udea.edu.co, jonatan.leal@udea.edu.co
Ingeniería de Sistemas, Universidad De Antioquia, Medellín, Colombia

I. Introducción.

El presente informe ejecutivo resume los aspectos más relevantes del desarrollo e implementación de un modelo de clasificación de enfermedades del arroz mediante técnicas de aprendizaje profundo. Este trabajo se enmarca en el uso de herramientas tecnológicas para apoyar el sector agrícola, con el objetivo de mejorar los procesos de detección temprana y diagnóstico a partir del análisis automático de imágenes.

La solución propuesta se basa en la técnica de Transfer Learning utilizando la arquitectura VGG16, lo que permitió aprovechar redes pre entrenadas para alcanzar altos niveles de precisión con un volumen moderado de datos. Los resultados obtenidos evidencian el potencial de estas tecnologías para contribuir a una agricultura más eficiente, sostenible e inteligente, con impacto directo en la productividad y calidad de vida de los productores.

II. Estructura del Notebook.

El notebook sigue la siguiente estructura para abordar el problema de clasificación de imágenes:

Descarga y Preparación de Datos:

Se inicia con la descarga del dataset de Kaggle. Luego, se filtra el dataset original para incluir únicamente las cinco clases de interés: bacterial leaf blight, brown spot, hispa, tungro y normal. Las imágenes de estas clases se copian a un nuevo directorio. Posteriormente, se dividen aleatoriamente en conjuntos de entrenamiento (80%) y validación (20%), manteniendo la estructura de carpetas por clase para su uso con generadores de imágenes. Se verifica el conteo de imágenes por clase en cada conjunto.

Definición y Construcción del Modelo:

Se define un modelo de red neuronal convolucional utilizando Transfer Learning. Se emplea la arquitectura VGG16 pre entrenada con pesos de ImageNet, excluyendo la capa densa superior. Luego, se añaden capas personalizadas para la clasificación de las cinco clases específicas.

Entrenamiento del Modelo:

El modelo se compila con un optimizador y función de pérdida apropiados para clasificación multiclase. Se entrena usando los generadores previamente creados, iterando durante varias épocas. Se utiliza un callback que guarda automáticamente el modelo con la mejor precisión en el conjunto de validación.

Evaluación del Modelo:

Una vez completado el entrenamiento, el modelo se evalúa en el conjunto de validación mediante precisión global, reporte de clasificación por clase (precisión, recall, F1-score) y matriz de confusión. Se visualizan las curvas de precisión y pérdida para analizar el aprendizaje.

Inferencia (Ejemplo):
Se proporciona un ejemplo de cómo cargar el mejor modelo guardado y realizar una predicción sobre una imagen individual.

III. Arquitectura del Modelo: VGG16 con Transfer Learning.

La solución de modelado se basa en la arquitectura VGG16, una red convolucional profunda pre-entrenada con el dataset ImageNet.

Base del Modelo:

Se cargó la arquitectura VGG16 desde la API de Keras con weights=imagenet y include_top=False, lo que permite aprovechar sus pesos sin incluir las capas densas superiores (específicas para las clases de ImageNet).

Congelación de Capas Base:

Las capas convolucionales y de pooling fueron congeladas (base_model.trainable = False), es decir, sus pesos no se actualizaron durante el entrenamiento. Esto permite conservar la capacidad de extracción de características de VGG16.

Capas Adicionales:

Sobre la base congelada, se añadieron capas densas personalizadas:

- Flatten para aplanar la salida convolucional
- BatchNormalization para regularización
- Capas Dense con activación relu

- Dropout para prevenir sobreajuste
- Una capa Dense final con activación softmax para predecir las cinco clases.

IV. Preprocesamiento de Imágenes.

El preprocesamiento fue esencial para adaptar los datos al modelo VGG16:

Normalización:
Se utilizó ImageDataGenerator con rescale=1./255 para escalar los valores de píxel de [0, 255] a [0, 1], lo cual mejora el entrenamiento.

Generadores de Datos:
Se crearon train_generator y val_generator, los cuales leen imágenes desde las carpetas respectivas, redimensionan a (224, 224), y generan lotes con etiquetas codificadas en one-hot (class_mode=categorical).
El generador de entrenamiento usa shuffle=True para aleatorizar las imágenes en cada época.

V. Iteraciones de Entrenamiento.

El modelo se entrenó durante 15 épocas:

Épocas:

Se realizaron 15 iteraciones completas (epochs=15) sobre el conjunto de entrenamiento.

Monitoreo y Guardado:

Se empleó un callback ModelCheckpoint que monitorea la val_accuracy y guarda el modelo en /content/mejor_modelo_vgg16_gpu.h5 solo si hay mejora (save_best_only=True, mode='max').

Curvas de Entrenamiento:

Las curvas mostraron un rápido aumento en la precisión del entrenamiento (cercana al 100%) y mejora significativa en validación, aunque con fluctuaciones.
La pérdida en entrenamiento disminuyó constantemente; en validación también bajó, aunque con valores algo más altos.
Se observó un leve sobreajuste hacia el final, mitigado por el ModelCheckpoint.

VI. Resultados Clave.

La fase de evaluación del modelo permitió medir su desempeño al clasificar imágenes de hojas de arroz en cinco categorías: cuatro tipos de enfermedades y una clase correspondiente a hojas

sanas. Para esto se utilizaron métricas estándar en clasificación multiclase, como precisión global, precisión por clase, recall, y F1-score.

Precisión Global en Validación.

El modelo alcanzó una precisión del 93.47% sobre el conjunto de validación. Esto significa que, de todas las imágenes procesadas durante la evaluación, el modelo acertó en casi el 94% de los casos. Este resultado refleja un buen nivel de generalización del modelo, es decir, su capacidad para clasificar correctamente imágenes que no había visto durante el entrenamiento.

Desempeño por Clase.

A continuación, se detallan las métricas obtenidas para cada clase. Estas ayudan a entender mejor cómo se comporta el modelo frente a cada tipo de enfermedad:

Bacterial Leaf Blight

Precisión: 0.98 → El modelo acierta el 98% de las veces cuando predice esta enfermedad.

Recall: 0.82 → Detecta correctamente el 82% de los casos reales de esta enfermedad.

F1-score: 0.89 → Buen equilibrio entre precisión y recall, aunque el valor de recall indica que aún se escapan algunos casos.

Brown Spot

Precisión: 0.94, Recall: 0.93, F1-score: 0.94

Muy buen desempeño, con alta consistencia entre detección y precisión.

Hispa

Precisión: 0.94, Recall: 0.96, F1-score: 0.95

El modelo muestra una excelente capacidad para identificar esta clase con pocas omisiones y errores.

Normal (hojas sanas)

Precisión: 0.95, Recall: 0.97, F1-score: 0.96

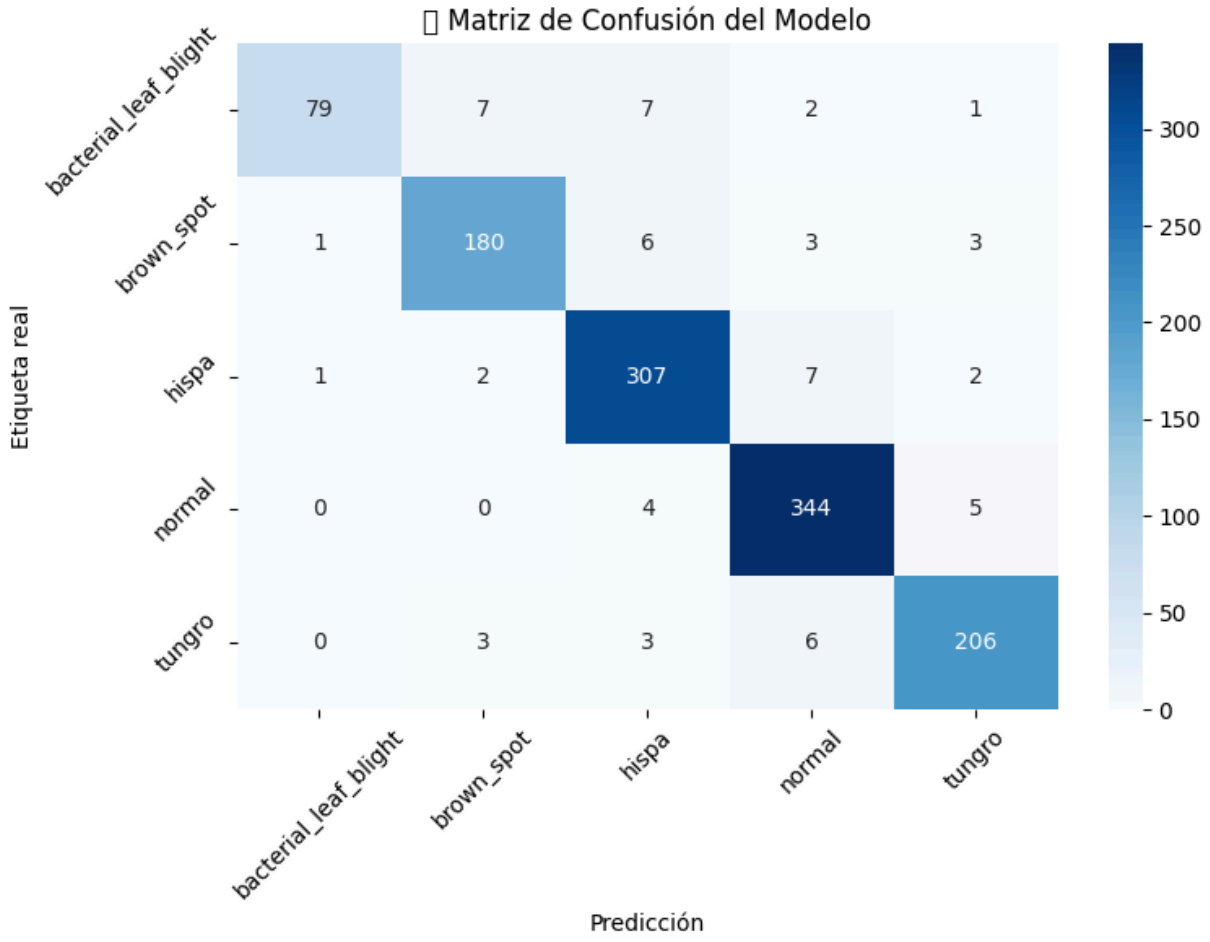
Se destaca el buen rendimiento en identificar hojas saludables, lo cual es esencial para evitar falsos positivos que podrían llevar a intervenciones innecesarias.

Tungro

Precisión: 0.95, Recall: 0.94, F1-score: 0.95

También presenta un desempeño equilibrado y confiable.

Matriz de Confusión.



La matriz muestra valores diagonales altos, reflejando una clasificación correcta en la mayoría de los casos. Los errores fuera de la diagonal indican confusiones entre clases específicas, útiles para análisis futuros.

Análisis General.

Los resultados muestran que el modelo no solo alcanza un alto rendimiento global, sino que también mantiene un desempeño robusto en todas las clases. Las diferencias observadas entre precisión y recall en algunas categorías (por ejemplo, bacterial leaf blight) sugieren que, aunque el modelo es muy preciso cuando identifica esta enfermedad, todavía hay algunos casos reales que no logra detectar. Esto puede deberse a similitudes visuales con otras clases o a variabilidad en las imágenes de entrada.

VII. Conclusiones.

El uso de Transfer Learning con VGG16, junto con un preprocesamiento adecuado, ha permitido desarrollar un modelo con un rendimiento sólido para clasificar cinco enfermedades del arroz (y hojas sanas), lo que lo convierte en una herramienta prometedora para aplicaciones en el sector agrícola.

VIII. REFERENCIAS

[1] F. Muthayya, J. Sugimoto, S. Montgomery, and G. Maberly, “An overview of global rice production, supply, trade, and consumption,” Annals of the New York Academy of Sciences, vol. 1324, no. 1, pp. 7–14, 2014.

[2] H. Ning, S. Liu, Q. Zhu, and T. Zhou, "Convolutional neural network in rice disease recognition: accuracy, speed and lightweight," *Frontiers in Plant Science*, vol. 14, p. 1269371, 2023. [Online]. Available: https://www.frontiersin.org/journals/plant-science/articles/10.3389/fpls.2023.1269371/full

[3] K. Kamilaris and F. X. Prenafeta-Boldú, “Deep learning in agriculture: A survey,” Computers and Electronics in Agriculture, vol. 147, pp. 70–90, Apr. 2018.

[4] R. Sladojevic, M. Arsenovic, A. Anderla, D. Culibrk, and D. Stefanovic, “Deep Neural Networks Based Recognition of Plant Diseases by Leaf Image Classification,” Computational Intelligence and Neuroscience, vol. 2016, Article ID 3289801, 2016.

[5] Paddy Doctor, Pandarasamy Arjunan (Samy), and Petchiammal. Paddy Doctor: Paddy Disease Classification. https://kaggle.com/competitions/paddy-disease-classification, 2022. Kaggle.