



UNIVERSIDAD DE GRANADA

PRÁCTICA 2

Sistemas Inteligentes para la Gestión de Empresas

ÍNDICE

1. Preparación del dataset.....	3
2. Análisis exploratorio.....	3
3. Entrenamiento.....	5
4. Aumento a 200 categorías.....	6
5. Mejoras implementadas.....	7
6. Explicabilidad.....	9
7. Conclusiones.....	9
8. Anexos.....	10

1. Preparación del dataset

Para adaptar las imágenes al formato requerido por las redes neuronales convolucionales, hemos aplicado distintas transformaciones durante la etapa de preprocesamiento. Entre ellas incluimos el redimensionado de las imágenes a 224×224 píxeles, la conversión a tensores y la normalización por canal utilizando las medias y desviaciones estándar habituales de los modelos preentrenados en ImageNet. Esta normalización permite mantener la coherencia estadística entre los datos de entrada y los parámetros aprendidos por redes como ResNet18 o ResNet50, mejorando así la estabilidad del entrenamiento y acelerando la convergencia.

En cuanto a la organización de los datos, se ha realizado una división aleatoria del conjunto original en dos particiones: un 80 % de las imágenes se ha destinado al entrenamiento y el 20 % restante a la validación. Este reparto garantiza que el modelo pueda aprender de una muestra representativa y, al mismo tiempo, evaluar su rendimiento sobre datos no vistos, permitiendo así detectar posibles problemas de sobreajuste.

Hemos creado *data loaders* independientes para cada partición, con un tamaño de lote (batch size) de 32 imágenes. En el caso del conjunto de entrenamiento, se ha activado el parámetro `shuffle=True`, lo que permite reordenar aleatoriamente los datos en cada época y evita que el modelo aprenda un orden implícito en el dataset. En cambio, para la validación hemos mantenido un orden fijo para asegurar la reproducibilidad y consistencia en la evaluación a lo largo de las distintas épocas de entrenamiento.

2. Análisis exploratorio

Hemos hecho un análisis exploratorio preliminar para entender la estructura del conjunto de datos. En primer lugar, verificamos que el subconjunto que usamos contiene 20 clases diferentes y un total de 1115 imágenes.

Después, hemos calculado la distribución de imágenes por clase. Aunque la mayoría de las categorías tienen en torno a 60 imágenes, hay cierta variabilidad en la representación de clases, con algunas categorías ligeramente infrarrepresentadas. Esto es especialmente relevante para anticipar posibles desbalances que puedan afectar al rendimiento del modelo durante el entrenamiento.

Para facilitar la visualización, hemos limpiado los nombres de las clases, eliminando los prefijos numéricos (e.g., 001.) y reemplazando los guiones bajos por espacios, lo que mejora considerablemente la legibilidad en los gráficos.

A continuación, hemos representado la distribución de imágenes por clase mediante un gráfico de barras. Esta visualización permite identificar de forma clara el reparto desigual entre clases, aunque en términos generales el conjunto puede considerarse razonablemente equilibrado.

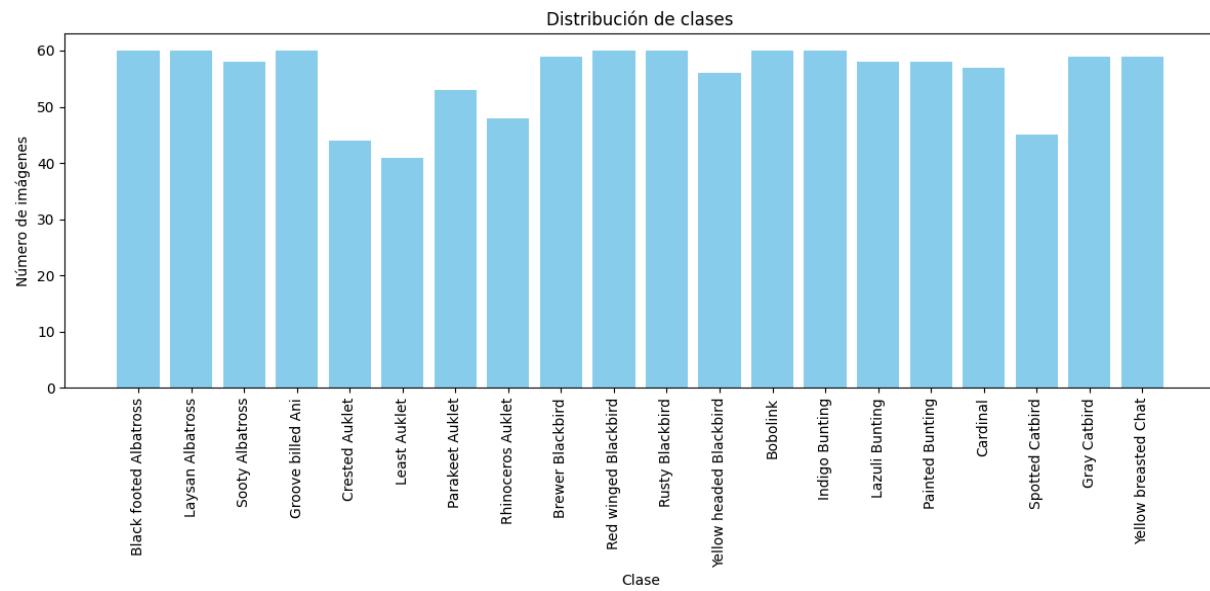


Figura 1. Distribución de imágenes por clase.

Finalmente, mostramos una imagen aleatoria de cada clase con el objetivo de familiarizarse con las características visuales distintivas de cada categoría. Esta revisión visual proporciona un contexto útil para el diseño y evaluación del modelo, especialmente al tratarse de un problema de clasificación de imágenes con alta variabilidad morfológica entre categorías.

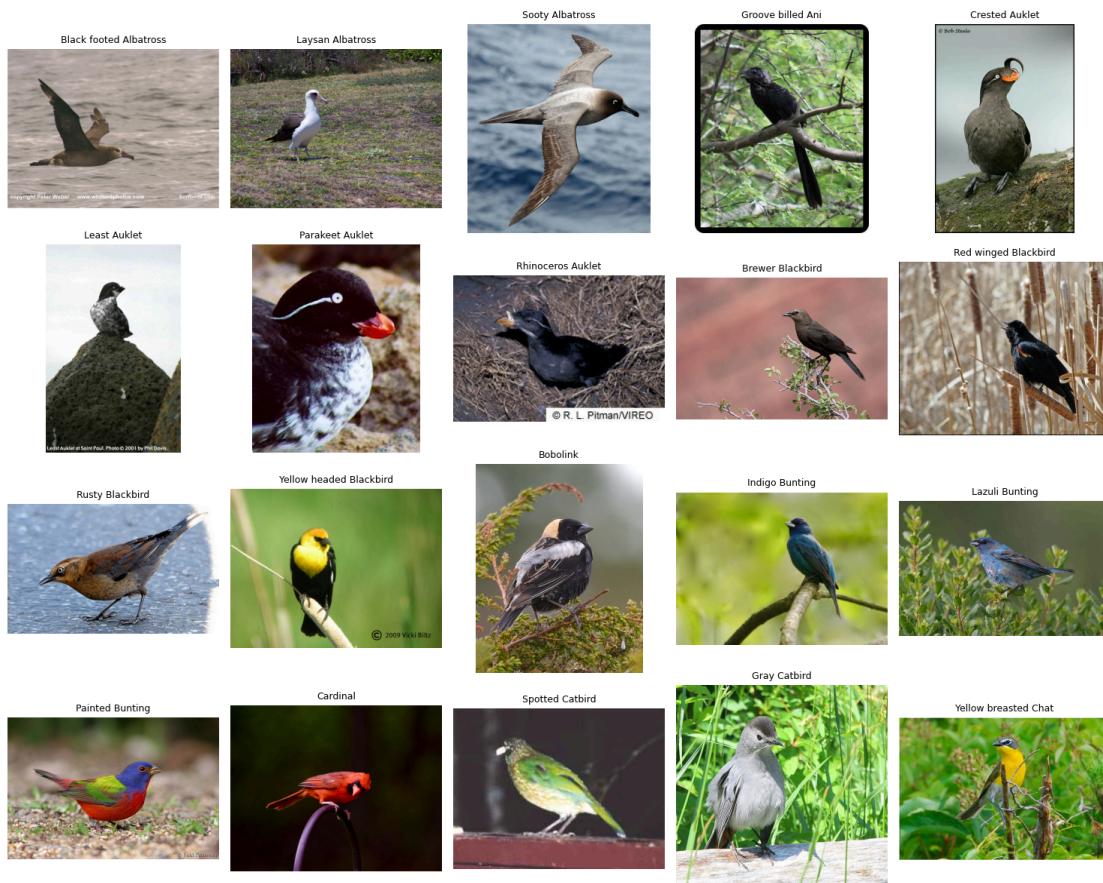


Figura 2. Ejemplo representativo de cada clase.

3. Entrenamiento

Para entrenar la red neuronal, hemos definido una función que encapsula tanto el bucle de entrenamiento como el de validación a lo largo de múltiples épocas. Esta función calcula en cada época tanto la pérdida (*loss*) como la precisión (*accuracy*) en ambos conjuntos, y registra el modelo que obtiene el mejor rendimiento sobre el conjunto de validación. Esta lógica permite retener los pesos óptimos del modelo y evita conservar configuraciones que hayan podido sobreajustarse al conjunto de entrenamiento.

Adicionalmente, hemos incorporado un scheduler de tipo `ReduceLROnPlateau`, que ajusta automáticamente la tasa de aprendizaje cuando no se observa mejora en la pérdida de validación durante un número determinado de épocas consecutivas. Esto permite afinar el proceso de optimización y evitar que el aprendizaje se estanke en mínimos locales.

Primero, hemos trabajado con un modelo ResNet18 preentrenado sobre ImageNet, aprovechando los beneficios del *transfer learning*. En esta fase, hemos congelado las capas convolucionales, que actúan como extractor de características, y las hemos reentrenado únicamente la capa final (*fc*), modificada para ajustarse al número de clases de nuestro conjunto de datos (20 categorías). Esta estrategia permite reducir el tiempo de entrenamiento y aprovechar representaciones visuales ya aprendidas en un dominio similar.

Después de 10 épocas de entrenamiento, el modelo ha conseguido una accuracy de validación del 85.20 %, mostrando una progresión estable tanto en la reducción de la pérdida como en la mejora de la precisión. Estos resultados indican que el modelo ha aprendido representaciones significativas y generaliza adecuadamente sobre datos no vistos.

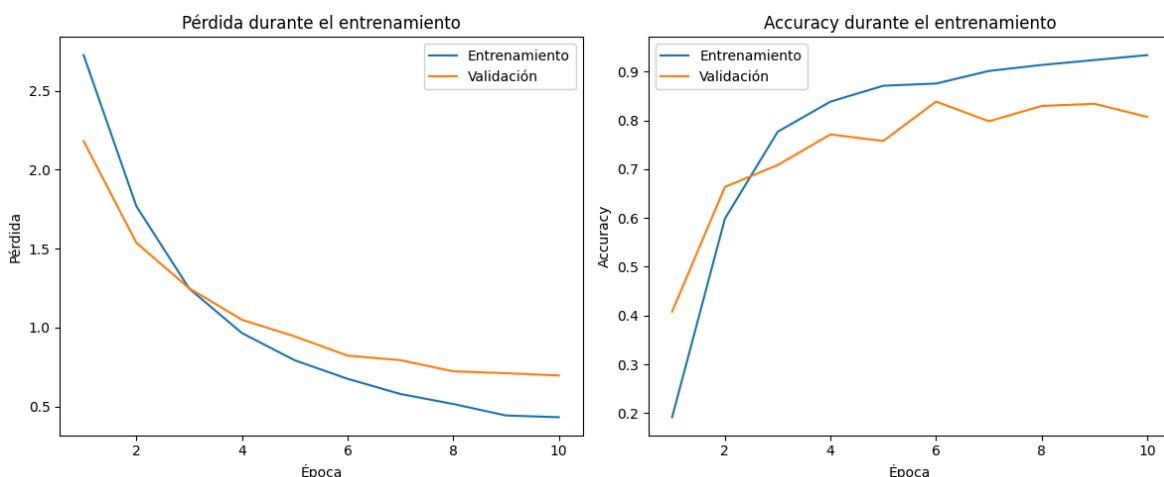


Figura 3. Evaluación de la pérdida y la precisión durante el entrenamiento de ResNet18

Tras la primera fase con ResNet18, hemos probado el mismo enfoque usando una arquitectura más profunda: ResNet50, también preentrenada sobre ImageNet. Esta red ofrece una mayor capacidad de representación, lo cual puede ser beneficioso para capturar detalles más complejos y sutiles en imágenes de aves, que presentan gran variedad morfológica y cromática.

El rendimiento del modelo con ResNet50 ha sido más alto, alcanzando una accuracy de validación del 86.55 % y mostrando una mejor estabilidad durante las últimas épocas. Esta mejora es coherente con el aumento en la capacidad del modelo y demuestra que, en este caso, la mayor profundidad ha sido beneficiosa.

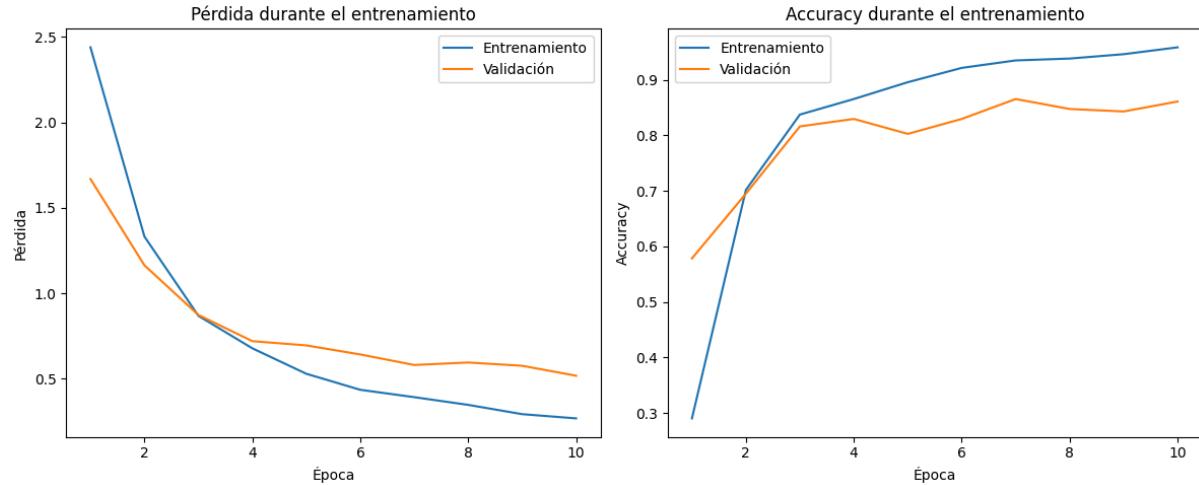


Figura 4. Evaluación de la pérdida y la precisión durante el entrenamiento de ResNet50

En ambos casos, la representación gráfica de las curvas de pérdida y precisión es importante para evaluar la evolución del modelo a lo largo del entrenamiento. Estas curvas permiten identificar de forma visual tanto la progresión del aprendizaje como posibles síntomas de sobreajuste (como una divergencia significativa entre entrenamiento y validación) o problemas de estancamiento (cuando no se ven mejoras en las métricas).

4. Aumento a 200 categorías

En una segunda fase del experimento, hemos planteado la ampliación del conjunto de datos para incluir las 200 clases completas del dataset CUB-200-2011, con el objetivo de abordar un problema de clasificación de mayor complejidad y evaluar la escalabilidad del enfoque usado.

Al igual que en los experimentos anteriores, hemos preparado una partición aleatoria del conjunto en 80 % para entrenamiento y 20 % para validación, asegurando un reparto equitativo entre clases. Hemos usado la arquitectura ResNet50 preentrenada, manteniendo la misma configuración de entrenamiento: *fine-tuning* parcial, técnicas de data augmentation, regularización mediante *label smoothing*, *early stopping* y *scheduler* de tasa de aprendizaje.

No obstante, debido a limitaciones computacionales en el entorno de ejecución, no ha sido posible completar el entrenamiento del modelo con las 200 clases. El aumento significativo del número de muestras y la complejidad del problema conlleva una mayor demanda de memoria y tiempo de cálculo, lo que supera los recursos disponibles en los dispositivos utilizados.

5. Mejoras implementadas

Para mejorar el rendimiento del modelo, hemos incorporado varias estrategias complementarias que actúan sobre distintas fases del entrenamiento. Estas técnicas han sido clave para mejorar la capacidad de generalización y evitar tanto el sobreajuste como el infraajuste.

Data Augmentation

Durante el preprocesamiento de las imágenes de entrenamiento, hemos aplicado un conjunto de transformaciones aleatorias, incluyendo recortes y escalados (`RandomResizedCrop`), volteos horizontales, variaciones de brillo, contraste y saturación (`ColorJitter`) y transformaciones afines ligeras (`RandomAffine`). Estas técnicas permiten simular distintas condiciones de captura sin necesidad de aumentar el volumen real de datos, generando un conjunto más diverso y robusto frente a la variabilidad del problema. Su uso es útil para reducir el sobreajuste y mejorar la capacidad del modelo para generalizar a nuevas muestras.

Early Stopping

Hemos implementado una estrategia de parada anticipada (`early stopping`) con una paciencia de 3 épocas, lo que significa que si no se observa una mejora en la precisión de validación durante tres épocas consecutivas, el entrenamiento se interrumpe automáticamente. Esta técnica permite evitar entrenamientos innecesariamente prolongados, previniendo así el sobreajuste y conservando el estado del modelo que ha ofrecido el mejor rendimiento en validación.

Label Smoothing

Para mejorar la generalización en contextos multiclase, hemos aplicado label smoothing sobre la función de pérdida (`CrossEntropyLoss`). Esta técnica consiste en suavizar las etiquetas verdaderas, de modo que el modelo no se vea forzado a asignar el 100 % de la probabilidad a una única clase. Esta estrategia reduce la confianza excesiva en las predicciones y mejora la calibración de las salidas, resultando especialmente útil en tareas con muchas clases y relaciones ambiguas entre categorías.

Análisis de las mejoras

La siguiente figura muestra la evolución de la pérdida y la precisión durante el entrenamiento y validación del modelo ResNet50, tras aplicar las técnicas de mejora implementadas.

En la gráfica de la izquierda, podemos ver una disminución más progresiva y estable en comparación con el entrenamiento previo sin mejoras. La pérdida de validación se mantiene en torno a 1.05 desde las primeras épocas, lo que indica que el modelo ha logrado evitar el sobreajuste a pesar de seguir optimizándose. Esto contrasta con la gráfica anterior, donde la pérdida de validación se estancaba más temprano.

En la gráfica de la derecha, se aprecia una diferencia clara: aunque el accuracy del entrenamiento sigue aumentando rápidamente, el accuracy de la validación alcanza un punto de estabilización más consistente alrededor del 83 %. Aunque existe una brecha con respecto al entrenamiento (lo esperado en modelos potentes como ResNet50), esta se mantiene relativamente constante y controlada.

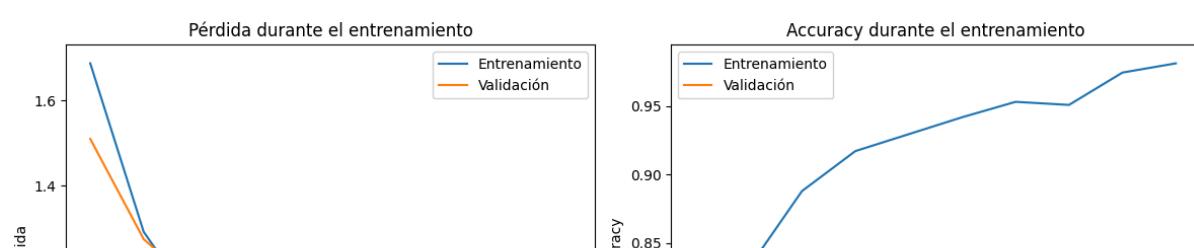


Figura 5. Evaluación de la pérdida y la precisión durante el entrenamiento tras aplicar mejoras

Registro de métricas con Weights & Biases (WandB)

Como apoyo adicional, hemos integrado la herramienta Weights & Biases para llevar a cabo un seguimiento exhaustivo del proceso de entrenamiento. Esta integración ha permitido registrar automáticamente las métricas por época, así como visualizar las curvas de evolución y los parámetros del modelo en tiempo real. Además, hemos utilizado el módulo `wandb.watch()` para monitorizar los gradientes y detectar posibles inestabilidades en la fase de ajuste.

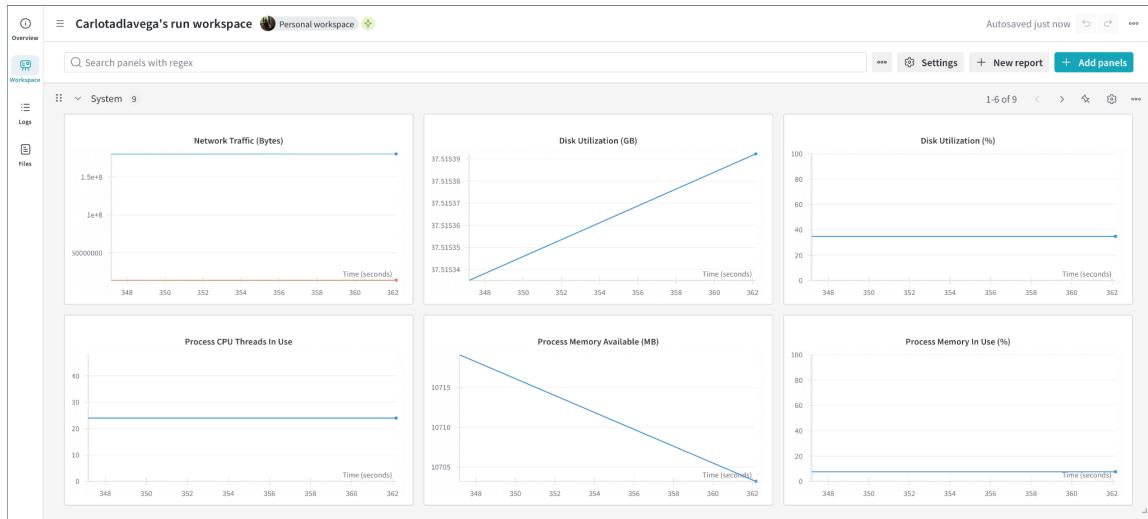


Figura 7. Captura de Weights & Biases

6. Explicabilidad

Para analizar qué regiones de la imagen tienen mayor influencia en la toma de decisiones del modelo, aplicamos Grad-CAM sobre la arquitectura ResNet50 entrenada. Esta técnica permite visualizar de forma cualitativa qué zonas activan más intensamente las últimas capas convolucionales justo antes de la predicción, proporcionando una aproximación visual a la interpretabilidad del modelo.

Grad-CAM genera mapas de calor que se superponen sobre las imágenes originales para indicar, mediante colores cálidos (rojos y amarillos), las zonas que han contribuido más a la predicción de una clase concreta. Para ello, se ha seleccionado aleatoriamente una imagen perteneciente a la clase "010.Red_winged_Blackbird", sobre las que se ha aplicado esta técnica utilizando activaciones de la capa layer3.

En la Figura 8, podemos ver cómo el modelo centra su atención en una región localizada donde se encuentra el ave, incluso cuando el entorno visual podría añadir ruido contextual. Las zonas activadas muestran un patrón claro que coincide con la forma y la posición del ave, indicando que el modelo ha aprendido a reconocer características visuales discriminativas relevantes.

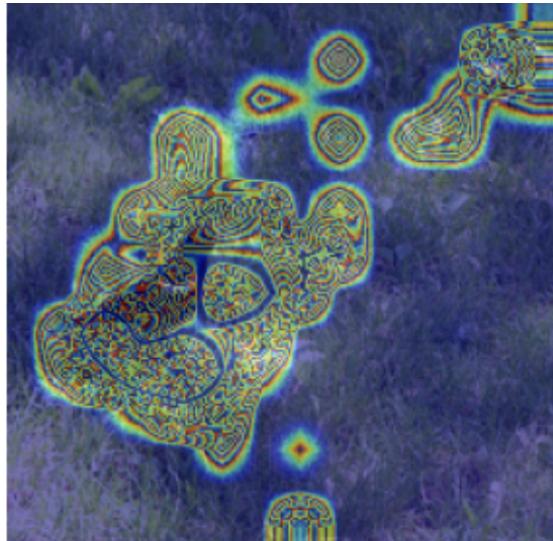


Figura 7. Grad-CAM aplicado sobre una imagen de la clase "Red_winged_Blackbird".

Esta visualización no solo confirma el correcto funcionamiento del modelo en términos de predicción, sino que también ofrece una herramienta de validación adicional, permitiendo detectar casos en los que el modelo podría estar fijándose en elementos irrelevantes o inconsistentes.

7. Conclusiones

A lo largo de este trabajo hemos desarrollado un sistema de clasificación multiclasa basado en aprendizaje profundo, aplicado al conjunto de datos CUB-200-2011. A partir de un subconjunto de 20 clases, hemos llevado a cabo un proceso completo de preparación, exploración y entrenamiento de modelos convolucionales, utilizando tanto ResNet18 como ResNet50 preentrenadas. Este enfoque nos ha permitido aplicar principios de transfer learning y evaluar cómo la profundidad del modelo influye en su rendimiento.

Hemos incorporado múltiples técnicas para mejorar la generalización, como data augmentation, label smoothing y early stopping, todas ellas reflejadas en la mejora cuantitativa de la precisión y en la estabilidad de las curvas de validación. Además, se ha integrado la herramienta Weights & Biases (WandB) para el seguimiento de métricas y se ha aplicado Grad-CAM como técnica de explicabilidad visual, permitiendo interpretar las decisiones del modelo de forma más transparente.

Pese a las limitaciones de recursos, que han impedido completar el entrenamiento sobre las 200 clases, el sistema ha mostrado una evolución sólida y lo hemos validado con resultados visuales y numéricos que reflejan un aprendizaje efectivo.

8. Anexos

Código completo disponible en: [clasificador.ipynb](#)