Clasificación de Imágenes con Deep Learning

Segunda entrega

Vanessa Tocasuche Ochoa

Fundamentos de Deep Learning
Universidad de Antioquia
2025

1. Estructura de notebooks

En el notebook de ***exploracion de datos*** se realizó la inspección inicial del dataset y la construcción de un DataFrame principal. Primero se cargan las rutas de las imágenes utilizando glob, se calcula la cantidad total por clase y se organizan las rutas. Luego, se extraen las dimensiones de cada imagen.

En el notebook de ***preprocesado*** se aplica una división en train/val de 20 % para validación y estandariza el tamaño de las imágenes a 224×224 píxeles. El notebook incluye cuatro configuraciones de data augmentation (A1–A4), que incorporan transformaciones progresivamente más complejas como rotaciones, zoom, contraste, brillo, recortes aleatorios y ruido gaussiano.

En el notebook de ***Arquitectura de Línea Base*** se implementa la arquitectura base del modelo mediante transferencia de aprendizaje y se ejecutan experimentos sistemáticos con diferentes backbones y técnicas de data augmentation. El notebook comienza definiendo la estructura de red utilizando una función genérica que permite seleccionar entre ResNet50, MobileNetV2 y EfficientNetB0, incorporando además una capa de preprocesamiento específica para cada arquitectura. Los modelos se construyen inicialmente en modo de extracción de características (backbone congelado) y se añaden capas densas finales para la clasificación binaria. A continuación, se entrenan los modelos utilizando cada una de las variantes de data augmentation, registrando la accuracy de validación obtenida en cada experimento. Finalmente, se consolidan los resultados en un DataFrame y se generan visualizaciones comparativas mediante gráficos de barras que permiten evaluar el impacto del backbone y de la estrategia de aumento de datos sobre el desempeño del modelo.

2. Solución

2.1. Preprocesado de datos

- División train/val del 80/20 con: image_dataset_from_directory.
- Redimensionamiento de las imagenes a 224×224.
- Cuatro opciones de data augmentation.

- Normalización automática según cada Backbone del Transfer Learning.

2.2. Arquitectura del modelo

- **Modelo utilizado:** Se seleccionaron MobileNetV2, ResNet50 y EfficientNetB0 como backbones para evaluar diferentes compromisos entre precisión y costo computacional, permitiendo comparar arquitecturas ligeras frente a modelos más complejos.

- **Construcción del modelo base:** Cada backbone se empleó en modo de transferencia de aprendizaje, utilizando los pesos pre-entrenados en ImageNet. Se removió la capa clasificadora original (include_top=False) y se aplicó average pooling para obtener una representación vectorial del contenido de la imagen.

- **Capas añadidas:** Se añadió un clasificador compuesto por una capa densa de 128 neuronas con activación ReLU, un dropout del 30% para reducir sobreajuste y una capa final softmax de dos unidades para realizar la clasificación binaria.

- **Estado del Backbone:** Durante la primera fase se congeló el backbone (trainable=False) para usarlo únicamente como extractor de características, reduciendo el tiempo de entrenamiento y minimizando riesgos de sobreajuste en un dataset pequeño.

- **Técnicas de augmentación utilizadas:** Se definieron cuatro pipelines de data augmentation con niveles crecientes de complejidad. Esto permitió evaluar la influencia del aumento de datos en el rendimiento del modelo y mitigar el sobreajuste en un dataset reducido.
  1 (flip + rotación)
  2 (flip + rotación + zoom)
  3 (zoom, brillo, contraste)
  4 (zoom, shear, crop, ruido gaussiano)

- **Justificación de la elección del modelo:** La selección de los backbones se basó en criterios de robustez, eficiencia computacional y disponibilidad de pesos preentrenados. MobileNetV2 aporta rapidez y bajo costo computacional; ResNet50 proporciona mayor profundidad y generalización; y EfficientNetB0 ofrece una relación óptima entre precisión y eficiencia. Evaluar estas tres familias permitió seleccionar la arquitectura que mejor se ajusta al problema y a las restricciones de hardware.

- **Configuracion de entrenamiento:** El modelo se compiló con el optimizador Adam y la pérdida sparse_categorical_crossentropy, adecuados para un problema de

clasificación binaria con etiquetas codificadas como enteros. La métrica principal fue accuracy y cada experimento se entrenó durante 10 épocas utilizando un 20% de validación.

2.3. Estrategia de Entrenamiento

En este proyecto se implementó una estrategia de entrenamiento basada en transfer learning y en la comparación sistemática de diferentes configuraciones de data augmentation y distintos backbones.

Procedimiento:

- **Número de épocas**
  Cada modelo fue entrenado durante 10 épocas, manteniendo fija esta configuración para asegurar una comparación consistente entre arquitecturas y técnicas de aumento de datos.

- **Optimizador:**
  En todos los experimentos se utilizó el optimizador **Adam**, debido a su estabilidad, velocidad de convergencia y buen desempeño en entrenamiento con datasets moderados.

- **Función de pérdida:**
  La función de pérdida empleada fue sparse categorical crossentropy, adecuada para clasificación multiclase con etiquetas enteras y compatible con la salida softmax del modelo

- **Variaciones probadas**

Se evaluaron dos tipos principales de variaciones:

1. **Variación del modelo base (backbones):**
   - *MobileNetV2*
   - *ResNet50*
   - *EfficientNetB0*
2. **Variación de técnicas de data augmentation (A1–A4):**
   Cada técnica incorpora niveles crecientes de complejidad:
   - **1:** Transformaciones básicas (flip + rotación)
   - **2:** Aumentación moderada (flip + rotación + zoom)
   - **3:** Aumentación fuerte (brillo, contraste, zoom)

- **4:** Aumentación extrema (zoom, shear, crop, ruido gaussiano)

3. **Comparación de técnicas A1–A4**

   Para evaluar el impacto de cada técnica de aumentación, se entrenó cada backbone con cada una de las cuatro configuraciones A1–A4 manteniendo constantes los parámetros.

4. **Comparación de backbones**

   Los tres backbones se entrenaron bajo las mismas condiciones utilizando todas las técnicas A1–A4.

5. **Experimentos sistemáticos (A1–A4 × 3 backbones)**

   En total se realizaron 12 experimentos sistemáticos, combinando:

   - **4 técnicas de augmentación**
   - **3 backbones**

2.4. Métricas utilizadas

Accuracy

La métrica principal utilizada fue accuracy, tanto en entrenamiento como en el conjunto de validación. Esta métrica mide la proporción de predicciones correctas sobre el total de muestras procesadas.

Esta métrica se obtiene para cada una de los experimentos sistemáticos de data augmentation para finalmente comparar los resultados.

3. Iteraciones

Durante el desarrollo del modelo se llevó a cabo un proceso iterativo y sistemático con el objetivo de identificar la combinación óptima entre arquitectura base (backbone) y estrategia de *data augmentation*.

Las iteraciones pueden clasificarse en cuatro grandes grupos:

1. Construcción del modelo base
2. Incorporación progresiva de técnicas de aumento de datos
3. Evaluación de diferentes arquitecturas preentrenadas
4. Ajustes de hiperparámetros asociados al entrenamiento.

4. Resultados

El análisis comparativo entre las técnicas de data augmentation (A1–A4) y los tres backbones evaluados (MobileNetV2, ResNet50 y EfficientNetB0) muestra patrones consistentes y diferencias claras en rendimiento.

1. Efecto del Data Augmentation

- A1: Punto de partida estable; ofrece rendimientos medios en los tres modelos.
- A2: Produce una mejora visible en todos los backbones; Es el mejor equilibrio entre variabilidad y estabilidad.
- A3: En MobileNetV2 disminuye ligeramente, pero en ResNet50 y EfficientNetB0 genera uno de los rendimientos más altos.
- A4: Se observa caída en MobileNetV2 y mejora consistente en EfficientNetB0. ResNet50 mantiene estabilidad y mejora respecto a A1–A3.

2. Comparación entre Backbones

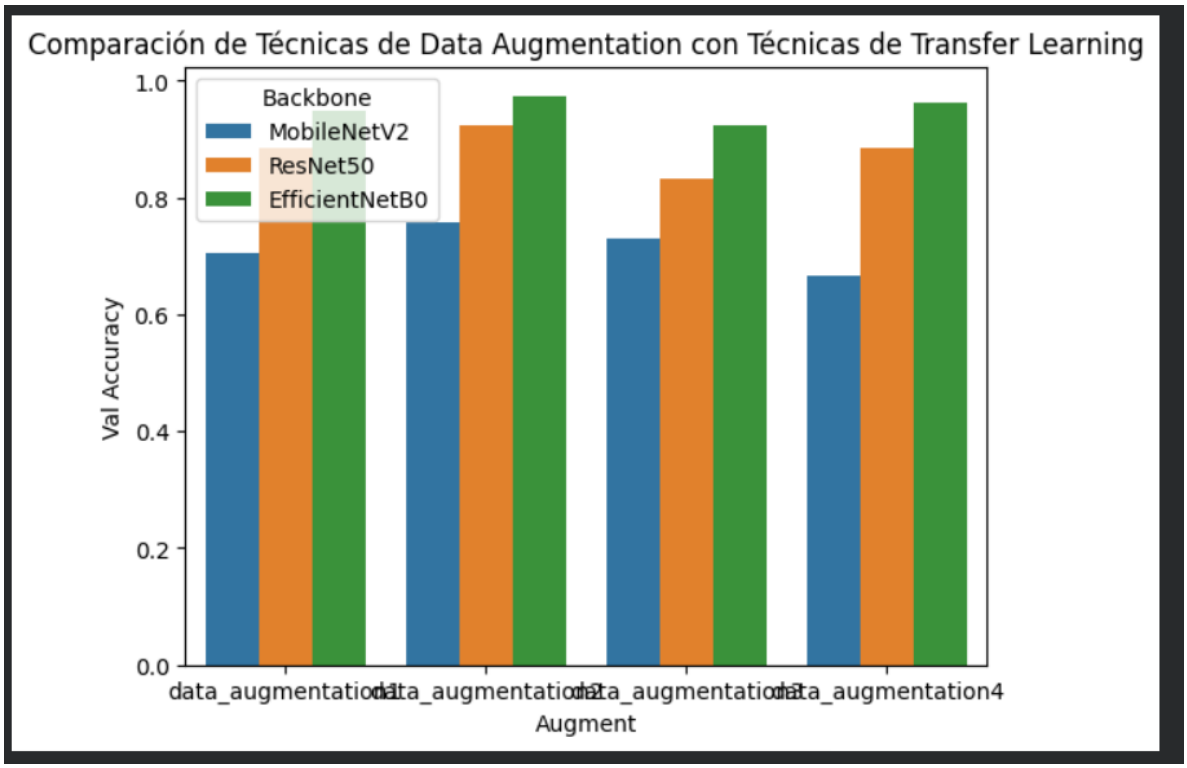
- **MobileNetV2:**
  Es el de menor rendimiento en todas las configuraciones.
  A2 y A3 mejoran ligeramente, pero A4 introduce demasiada variabilidad y el rendimiento cae.

- **ResNet50**:
  Modelo consistente y robusto frente a todas las técnicas.
  Muestra incrementos progresivos, logrando su mejor desempeño con A2 y A4.

- **EfficientNetB0**:
  El backbone más preciso en todos los casos.
  Alcanza las mejores accuracy globales, especialmente en A2 y A4.
  Maneja muy bien aumentaciones complejas sin perder estabilidad.

3. Mejor combinación observada

   EfficientNetB0 + dataAugmentation2 con un accuracy de 0.974359



	Augment	Backbone	Val Accuracy
0	data_augmentation1	MobileNetV2	0.705128
1	data_augmentation1	ResNet50	0.884615
2	data_augmentation1	EfficientNetB0	0.948718
3	data_augmentation2	MobileNetV2	0.756410
4	data_augmentation2	ResNet50	0.923077
5	data_augmentation2	EfficientNetB0	0.974359
6	data_augmentation3	MobileNetV2	0.730769
7	data_augmentation3	ResNet50	0.833333
8	data_augmentation3	EfficientNetB0	0.923077
9	data_augmentation4	MobileNetV2	0.666667
10	data_augmentation4	ResNet50	0.884615
11	data_augmentation4	EfficientNetB0	0.961538

5. Conclusiones

Tras evaluar las tres arquitecturas preentrenadas junto con las cuatro configuraciones de data augmentation, se observaron tendencias claras en el comportamiento del modelo. En términos de desempeño general, EfficientNetB0 fue la arquitectura que ofreció los mejores resultados, superando consistentemente a MobileNetV2 y ResNet50.

Respecto a las técnicas de augmentación, las configuraciones A2 y A4 resultaron ser las más efectivas. A2 mostró un equilibrio adecuado al introducir transformaciones moderadas sin alterar en exceso la estructura visual de las imágenes. Por otro lado, A4, que incorpora

augmentaciones más agresivas, potenció especialmente a modelos más complejos como EfficientNetB0 y ResNet50, mejorando aún más su capacidad de generalización.

De cara a futuras iteraciones, sería recomendable **aplicar fine-tuning a las últimas capas del backbone**, incorporar técnicas adicionales de regularización y experimentar con un **learning rate scheduler** para optimizar el proceso de entrenamiento. También sería útil entrenar durante más épocas para observar si el rendimiento continúa aumentando y complementar el análisis con métricas como precision, recall y F1-score, lo que permitiría evaluar mejor el comportamiento del modelo en cada clase.