

# Clasificación de Personajes de "One Piece" con Deep Learning

---

**Autor:** Alejandro Hernández Delgado

**Tecnología:** PyTorch, Python, Torchvision

**Dataset:** [One Piece Characters \(Kaggle\)](#)

---

## Descripción del Proyecto

Este proyecto tiene como objetivo desarrollar, entrenar y optimizar una **Red Neuronal Convolutacional (CNN)** capaz de identificar y clasificar imágenes de 5 personajes principales del anime *One Piece*: **Luffy, Nami, Sanji, Usopp y Zoro**.

El flujo de trabajo abarca desde la construcción de una CNN básica desde cero hasta la implementación de técnicas avanzadas como **Data Augmentation, Regularización y Transfer Learning** (utilizando EfficientNet-B0) para mejorar la precisión del modelo.

---

## Configuración del Entorno

### Requisitos

El proyecto utiliza las siguientes librerías principales:

- **torch & torchvision** (Framework de Deep Learning)
- **matplotlib & seaborn** (Visualización)
- **scikit-learn** (Métricas de evaluación)
- **kagglehub** (Descarga automática del dataset)

### Estructura de Datos

El dataset se ha dividido automáticamente garantizando conjuntos disjuntos para evitar el *data leakage*:

- **Train (60%):** Entrenamiento de los pesos.
  - **Validation (20%):** Ajuste de hiperparámetros y Early Stopping.
  - **Test (20%):** Evaluación final del rendimiento.
- 

## Metodología y Arquitecturas

Se han diseñado y comparado dos arquitecturas propias antes de pasar a modelos pre-entrenados:

### 1. SimpleCNN (Baseline)

Una red secuencial básica para establecer una línea base de rendimiento.

- 3 Bloques Convolucionales.
- Capas **Linear** densas al final.

- *Problema detectado:* Alta propensión al sobreajuste (overfitting) y baja capacidad de generalización.

2. ImprovedCNN (Arquitectura Mejorada)

Una evolución de la red anterior incorporando técnicas de regularización modernas:

- **Mayor profundidad:** 5 bloques convolucionales (hasta 512 filtros).
- **Batch Normalization:** Para estabilizar el aprendizaje en cada capa.
- **Global Average Pooling:** Reduce drásticamente los parámetros antes del clasificador.
- **Dropout (0.5):** Apagado aleatorio de neuronas para forzar robustez.

 Experimentos y Evolución (M1 - M4)

Se realizó un estudio ablativo incremental para medir el impacto de cada mejora técnica:

ID	Modelo	Técnica Clave	Accuracy (Test)	Observaciones
M1	SimpleCNN	Baseline	36.5%	El modelo apenas aprende, sufre overfitting rápido.
M2	ImprovedCNN	+ Batch Norm / Dropout	45.3%	Mejora notable en estabilidad gracias a la arquitectura.
M3	ImprovedCNN	+ Optimizador SGD	41.6%	SGD generaliza mejor teóricamente, pero convergió peor que Adam aquí.
M4	ImprovedCNN	+ Data Augmentation	43.8%	Se añadieron rotaciones y crops, reduciendo el overfitting pero dificultando el aprendizaje.

*Nota: Aunque las arquitecturas propias mejoraron la base, el límite del 50% de precisión indicaba la necesidad de un extractor de características más potente.*

 Estrategia Final: Transfer Learning (M5)

Para superar las limitaciones de las redes entrenadas desde cero con pocos datos, se implementó **Transfer Learning** utilizando **EfficientNet-B0** pre-entrenada en ImageNet.

Pipeline de Entrenamiento

1. **Preprocesamiento Avanzado:** `RandomResizedCrop`, `RandomRotation`, y `ColorJitter` para robustez.
2. **Fase 1 (Warmup):** Se congelaron las capas base y se entrenó solo el nuevo clasificador (Cabecera) por 7 épocas.
3. **Fase 2 (Fine-Tuning):** Se descongeló toda la red y se re-entrenó con un **Learning Rate diferencial muy bajo (1e-5)** para ajustar los filtros al estilo "anime" sin destruir el conocimiento previo.
4. **Scheduler:** Uso de `ReduceLROnPlateau` para reducir el LR si la validación se estancaba.

 Resultados Finales

El modelo final (M5) superó ampliamente a los intentos anteriores:

- **Accuracy en Test: 70.80%**
- **Mejora vs Baseline:** +34.3% puntos porcentuales.

### Gráficas de Entrenamiento

(Inserta aquí tu imagen de `plot_history` del modelo 5)

### Matriz de Confusión

El modelo distingue razonablemente bien a los personajes, aunque presenta algunas confusiones entre personajes con paletas de colores similares.

(Inserta aquí tu imagen de la Matriz de Confusión del modelo 5)

---

## 🏁 Conclusiones y Trabajo Futuro

### Conclusiones

1. **Arquitectura vs Datos:** Mejorar la arquitectura (M2) aportó más valor inicial que el aumento de datos (M4) en redes pequeñas.
2. **El poder de Transfer Learning:** Pasar de redes propias a EfficientNet supuso un salto cualitativo del 45% al 71% de precisión.
3. **Dificultad del Dataset:** Las imágenes de anime presentan alta varianza (dibujos de cuerpo entero, caras, estilos artísticos variados), lo que dificulta la clasificación sin un dataset masivo.

### Posibles Mejoras (Next Steps)

Para alcanzar el objetivo opcional del >85%:

- **Dataset:** Aumentar el número de imágenes (actualmente es escaso para 5 clases complejas).
- **Modelos más grandes:** Probar **ResNet50** o **EfficientNet-B4**.
- **Ensembling:** Combinar predicciones de varios modelos.
- **Test-Time Augmentation (TTA):** Realizar predicciones sobre varias versiones aumentadas de la misma imagen de test y promediar.