Reporte Final: Clasificación de Pokémon con EfficientNet-B0 y Fine-Tuning

1. Introducción

Las Redes Neuronales Convolucionales (CNN) son el estándar para el reconocimiento de imágenes. En este proyecto, buscamos clasificar especies de Pokémon. Dado que los conjuntos de datos de Pokémon suelen ser limitados en cantidad, utilizamos Transfer Learning (Aprendizaje por Transferencia) sobre una arquitectura EfficientNet-B0 pre-entrenada en ImageNet. El proceso se dividió en dos etapas estratégicas: Feature Extraction (Extracción de Características) y Fine-Tuning (Ajuste Fino).

2. Arquitectura y Metodología

Se utilizó EfficientNet-B0 debido a su eficiencia en parámetros y alto rendimiento.

Fase 1: Feature Extraction (Entrenamiento Base)

- Configuración: Se congelaron todas las capas convolucionales base (features). Solo se entrenó el clasificador final (capa lineal personalizada).
- Optimizador: Adam con Tasa de Aprendizaje (Learning Rate) de 0.001.
- Resultado: El modelo aprendió a usar las características genéricas de ImageNet para distinguir Pokémon, alcanzando un límite de precisión cercano al 82%.

Fase 2: Fine-Tuning (La Mejora Clave)

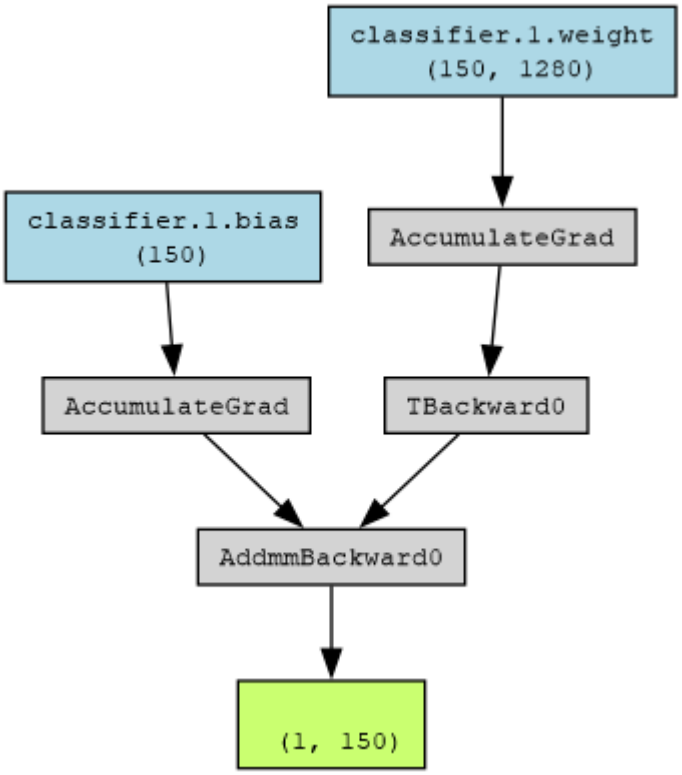
Para superar la barrera del 82%, implementamos una estrategia de ajuste fino:

- Descongelamiento: Se habilitaron los gradientes en todas las capas de la red base (model.features), permitiendo que los filtros internos de la red se adaptaran específicamente a las texturas y formas de los Pokémon.
- Ajuste de Hiperparámetros: Se redujo la tasa de aprendizaje drásticamente a 0.0001 (10 veces menor). Esto fue crucial para modificar suavemente los pesos sin destruir el conocimiento previo.
- Regularización: Se mantuvo el Data Augmentation (rotación, flip horizontal) y Dropout (p=0.2) para evitar el sobreajuste al entrenar la red completa.

Representación de la arquitectura de nuestra red (Revisar siguientes páginas)

```
==========================================================================================================================================
Layer (type (var_name))                                 Input Shape          Output Shape         Param #              Trainable
==========================================================================================================================================
EfficientNet (EfficientNet)                             [1, 3, 224, 224]     [1, 150]             --                   Partial
├─Sequential (features)                                 [1, 3, 224, 224]     [1, 1280, 7, 7]      --                   False
│    └─Conv2dNormActivation (0)                         [1, 3, 224, 224]     [1, 32, 112, 112]    --                   False
│    │    └─Conv2d (0)                                  [1, 3, 224, 224]     [1, 32, 112, 112]    (864)                False
│    │    └─BatchNorm2d (1)                             [1, 32, 112, 112]    [1, 32, 112, 112]    (64)                 False
│    │    └─SiLU (2)                                    [1, 32, 112, 112]    [1, 32, 112, 112]    --                   --
│    └─Sequential (1)                                   [1, 32, 112, 112]    [1, 16, 112, 112]    --                   False
│    │    └─MBConv (0)                                  [1, 32, 112, 112]    [1, 16, 112, 112]    (1,448)              False
│    └─Sequential (2)                                   [1, 16, 112, 112]    [1, 24, 56, 56]      --                   False
│    │    └─MBConv (0)                                  [1, 16, 112, 112]    [1, 24, 56, 56]      (6,004)              False
│    │    └─MBConv (1)                                  [1, 24, 56, 56]      [1, 24, 56, 56]      (10,710)             False
│    └─Sequential (3)                                   [1, 24, 56, 56]      [1, 40, 28, 28]      --                   False
│    │    └─MBConv (0)                                  [1, 24, 56, 56]      [1, 40, 28, 28]      (15,350)             False
│    │    └─MBConv (1)                                  [1, 40, 28, 28]      [1, 40, 28, 28]      (31,290)             False
│    └─Sequential (4)                                   [1, 40, 28, 28]      [1, 80, 14, 14]      --                   False
│    │    └─MBConv (0)                                  [1, 40, 28, 28]      [1, 80, 14, 14]      (37,130)             False
│    │    └─MBConv (1)                                  [1, 80, 14, 14]      [1, 80, 14, 14]      (102,900)            False
│    │    └─MBConv (2)                                  [1, 80, 14, 14]      [1, 80, 14, 14]      (102,900)            False
│    └─Sequential (5)                                   [1, 80, 14, 14]      [1, 112, 14, 14]     --                   False
│    │    └─MBConv (0)                                  [1, 80, 14, 14]      [1, 112, 14, 14]     (126,004)            False
│    │    └─MBConv (1)                                  [1, 112, 14, 14]     [1, 112, 14, 14]     (208,572)            False
│    │    └─MBConv (2)                                  [1, 112, 14, 14]     [1, 112, 14, 14]     (208,572)            False
│    └─Sequential (6)                                   [1, 112, 14, 14]     [1, 192, 7, 7]       --                   False
│    │    └─MBConv (0)                                  [1, 112, 14, 14]     [1, 192, 7, 7]       (262,492)            False
│    │    └─MBConv (1)                                  [1, 192, 7, 7]       [1, 192, 7, 7]       (587,952)            False
│    │    └─MBConv (2)                                  [1, 192, 7, 7]       [1, 192, 7, 7]       (587,952)            False
│    │    └─MBConv (3)                                  [1, 192, 7, 7]       [1, 192, 7, 7]       (587,952)            False
│    └─Sequential (7)                                   [1, 192, 7, 7]       [1, 320, 7, 7]       --                   False
│    │    └─MBConv (0)                                  [1, 192, 7, 7]       [1, 320, 7, 7]       (717,232)            False
│    └─Conv2dNormActivation (8)                         [1, 320, 7, 7]       [1, 1280, 7, 7]      --                   False
│    │    └─Conv2d (0)                                  [1, 320, 7, 7]       [1, 1280, 7, 7]      (409,600)            False
│    │    └─BatchNorm2d (1)                             [1, 1280, 7, 7]      [1, 1280, 7, 7]      (2,560)              False
│    │    └─SiLU (2)                                    [1, 1280, 7, 7]      [1, 1280, 7, 7]      --                   --
├─AdaptiveAvgPool2d (avgpool)                           [1, 1280, 7, 7]      [1, 1280, 1, 1]      --                   --
├─Sequential (classifier)                               [1, 1280]            [1, 150]             --                   True
│    └─Dropout (0)                                      [1, 1280]            [1, 1280]            --                   --
│    └─Linear (1)                                       [1, 1280]            [1, 150]             192,150              True
==========================================================================================================================================
Total params: 4,199,698
Trainable params: 192,150
Non-trainable params: 4,007,548
Total mult-adds (Units.MEGABYTES): 384.78
==========================================================================================================================================
Input size (MB): 0.60
Forward/backward pass size (MB): 107.88
Params size (MB): 16.80
Estimated Total Size (MB): 125.28
==========================================================================================================================================
```


```
==========================================================================================================================================
Layer (type (var_name))                                 Input Shape          Output Shape         Param #              Trainable
==========================================================================================================================================
EfficientNet (EfficientNet)                             [1, 3, 224, 224]     [1, 150]             --                   Partial
├─Sequential (features)                                 [1, 3, 224, 224]     [1, 1280, 7, 7]      --                   False
│    └─Conv2dNormActivation (0)                         [1, 3, 224, 224]     [1, 32, 112, 112]    --                   False
│    │    └─Conv2d (0)                                  [1, 3, 224, 224]     [1, 32, 112, 112]    (864)                False
│    │    └─BatchNorm2d (1)                             [1, 32, 112, 112]    [1, 32, 112, 112]    (64)                 False
│    │    └─SiLU (2)                                    [1, 32, 112, 112]    [1, 32, 112, 112]    --                   --
│    └─Sequential (1)                                   [1, 32, 112, 112]    [1, 16, 112, 112]    --                   False
│    │    └─MBConv (0)                                  [1, 32, 112, 112]    [1, 16, 112, 112]    (1,448)              False
│    └─Sequential (2)                                   [1, 16, 112, 112]    [1, 24, 56, 56]      --                   False
│    │    └─MBConv (0)                                  [1, 16, 112, 112]    [1, 24, 56, 56]      (6,004)              False
│    │    └─MBConv (1)                                  [1, 24, 56, 56]      [1, 24, 56, 56]      (10,710)             False
│    └─Sequential (3)                                   [1, 24, 56, 56]      [1, 40, 28, 28]      --                   False
│    │    └─MBConv (0)                                  [1, 24, 56, 56]      [1, 40, 28, 28]      (15,350)             False
│    │    └─MBConv (1)                                  [1, 40, 28, 28]      [1, 40, 28, 28]      (31,290)             False
│    └─Sequential (4)                                   [1, 40, 28, 28]      [1, 80, 14, 14]      --                   False
│    │    └─MBConv (0)                                  [1, 40, 28, 28]      [1, 80, 14, 14]      (37,130)             False
│    │    └─MBConv (1)                                  [1, 80, 14, 14]      [1, 80, 14, 14]      (102,900)            False
│    │    └─MBConv (2)                                  [1, 80, 14, 14]      [1, 80, 14, 14]      (102,900)            False
│    └─Sequential (5)                                   [1, 80, 14, 14]      [1, 112, 14, 14]     --                   False
│    │    └─MBConv (0)                                  [1, 80, 14, 14]      [1, 112, 14, 14]     (126,004)            False
│    │    └─MBConv (1)                                  [1, 112, 14, 14]     [1, 112, 14, 14]     (208,572)            False
│    │    └─MBConv (2)                                  [1, 112, 14, 14]     [1, 112, 14, 14]     (208,572)            False
│    └─Sequential (6)                                   [1, 112, 14, 14]     [1, 192, 7, 7]       --                   False
│    │    └─MBConv (0)                                  [1, 112, 14, 14]     [1, 192, 7, 7]       (262,492)            False
│    │    └─MBConv (1)                                  [1, 192, 7, 7]       [1, 192, 7, 7]       (587,952)            False
│    │    └─MBConv (2)                                  [1, 192, 7, 7]       [1, 192, 7, 7]       (587,952)            False
│    │    └─MBConv (3)                                  [1, 192, 7, 7]       [1, 192, 7, 7]       (587,952)            False
│    └─Sequential (7)                                   [1, 192, 7, 7]       [1, 320, 7, 7]       --                   False
│    │    └─MBConv (0)                                  [1, 192, 7, 7]       [1, 320, 7, 7]       (717,232)            False
│    └─Conv2dNormActivation (8)                         [1, 320, 7, 7]       [1, 1280, 7, 7]      --                   False
│    │    └─Conv2d (0)                                  [1, 320, 7, 7]       [1, 1280, 7, 7]      (409,600)            False
│    │    └─BatchNorm2d (1)                             [1, 1280, 7, 7]      [1, 1280, 7, 7]      (2,560)              False
│    │    └─SiLU (2)                                    [1, 1280, 7, 7]      [1, 1280, 7, 7]      --                   --
├─AdaptiveAvgPool2d (avgpool)                           [1, 1280, 7, 7]      [1, 1280, 1, 1]      --                   --
├─Sequential (classifier)                               [1, 1280]            [1, 150]             --                   True
│    └─Dropout (0)                                      [1, 1280]            [1, 1280]            --                   --
│    └─Linear (1)                                       [1, 1280]            [1, 150]             192,150              True
==========================================================================================================================================
Total params: 4,199,698
Trainable params: 192,150
Non-trainable params: 4,007,548
Total mult-adds (Units.MEGABYTES): 384.78
==========================================================================================================================================
Input size (MB): 0.60
Forward/backward pass size (MB): 107.88
Params size (MB): 16.80
Estimated Total Size (MB): 125.28
==========================================================================================================================================
```



3. Resultados del Experimento

El entrenamiento total consistió en 10 épocas de base seguidas de 5 épocas de ajuste fino. Los resultados finales tras la fase de Fine-Tuning fueron sobresalientes:

Métricas Finales (Época 5 del Fine-Tuning)

- Precisión de Prueba (Test Accuracy): 93.42% (Mejora de +11.4% respecto a la Fase 1).
- Precisión de Entrenamiento (Train Accuracy): 99.41%.
- Pérdida de Prueba (Test Loss): 0.2758 (Reducción significativa desde 0.8).

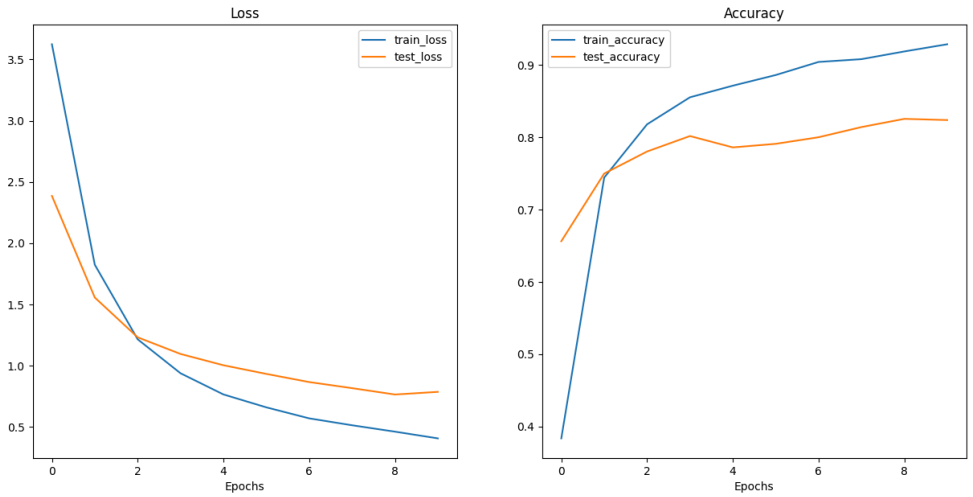
Análisis de Gráficas de Evolución

Basado en las curvas de aprendizaje obtenidas:

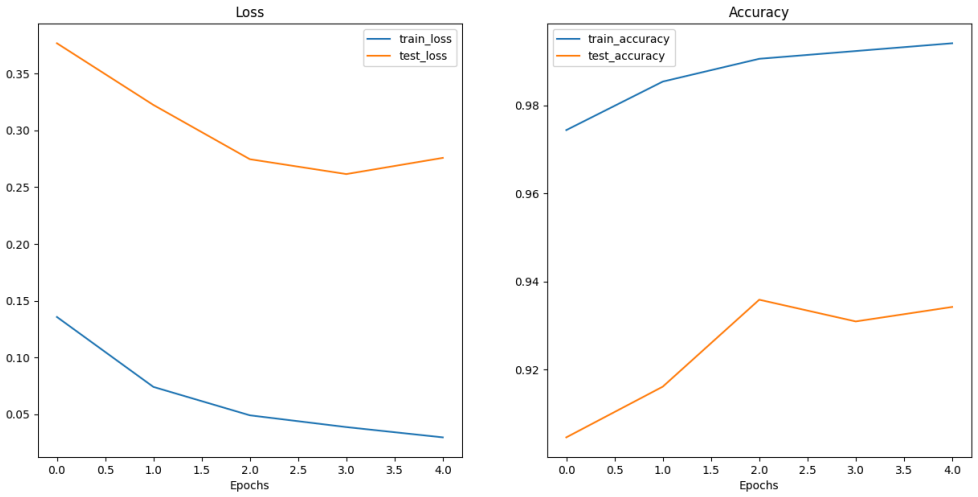
- Convergencia de Pérdida: La pérdida de prueba (línea naranja) continuó descendiendo durante el Fine-Tuning, pasando de ~0.37 a ~0.27. Esto indica que el modelo siguió aprendiendo generalización real y no solo memorización.
- Salto en Precisión: Inmediatamente al iniciar el Fine-Tuning (Época 0 del gráfico nuevo), la precisión saltó del 82% al ~90%, y continuó refinándose hasta el 93.4% en la época final.

- Estabilidad: A pesar de tener toda la red descongelada, las curvas son suaves, lo que confirma que la tasa de aprendizaje de 0.0001 fue la correcta.

Resultados entrenamiento inicial (Fase 1):



Resultados entrenamiento adicional con fine-tuning (Fase 2):



4. Conclusiones

La implementación de Fine-Tuning fue la estrategia determinante en este proyecto.

- Mientras que el Feature Extraction proporcionó una base sólida (82%), el modelo estaba limitado por "ver" el mundo a través de filtros entrenados en fotos cotidianas (perros, coches, frutas).

- Al descongelar las capas y permitir el Fine-Tuning, la red adaptó sus filtros internos para entender las características artísticas específicas de los Pokémon (trazos, colores planos, estilos de dibujo), elevando la precisión al 93.4%.
- El modelo final es altamente robusto y apto para implementación.