Tarea 4 - Clasificación de imágenes usando transfer learning

Claudia Lissette Gutiérrez Díaz
Licenciada en Ciencias Computacionales, FCFM, UANL

1. Introducción

La identificación precisa y rápida de objetos, personas, y patrones dentro de las imágenes es bueno para el desarrollo de tecnologías avanzadas. Para esto se pueden utilizar técnicas de aprendizaje automático o aprendizaje profundo para la clasificación, sin embargo, entrenar modelos de clasificación de imágenes desde cero puede ser complicado, debido a que se requieren muchas imágenes para esto, lo que implica el tener mucho tiempo y recursos computacionales para llevarlo a cabo.

En este reporte documentaremos el uso dos técnicas de transferencia de aprendizaje. Estas técnicas consisten en reutilizar un modelo previamente entrenado, adaptarlo a nuestras necesidades y probarlo en un conjunto diferente de información.

Abordaremos esas dos técnicas y realizaremos una comparación de sus estadísticas para definir cuál fue el mejor.

2. Descripción de los datos

Para este estudio, se decidió clasificar imágenes de pokemones. El conjunto de datos se obtuvo de Kaggle y este consiste en la recolección de aproximadamente entre 25 y 50 imágenes de los primeros 150 pokemones correspondientes a la primera generación. Estas imágenes vienen de diferentes fuentes, anime, videojuegos, cartas, dibujos o fanarts. Estas imágenes no tienen alta resolución para ayudar a aligerar el proceso y consumir menos recursos.

En la figura 1, se pueden observar algunos ejemplos del conjunto de datos seleccionado.

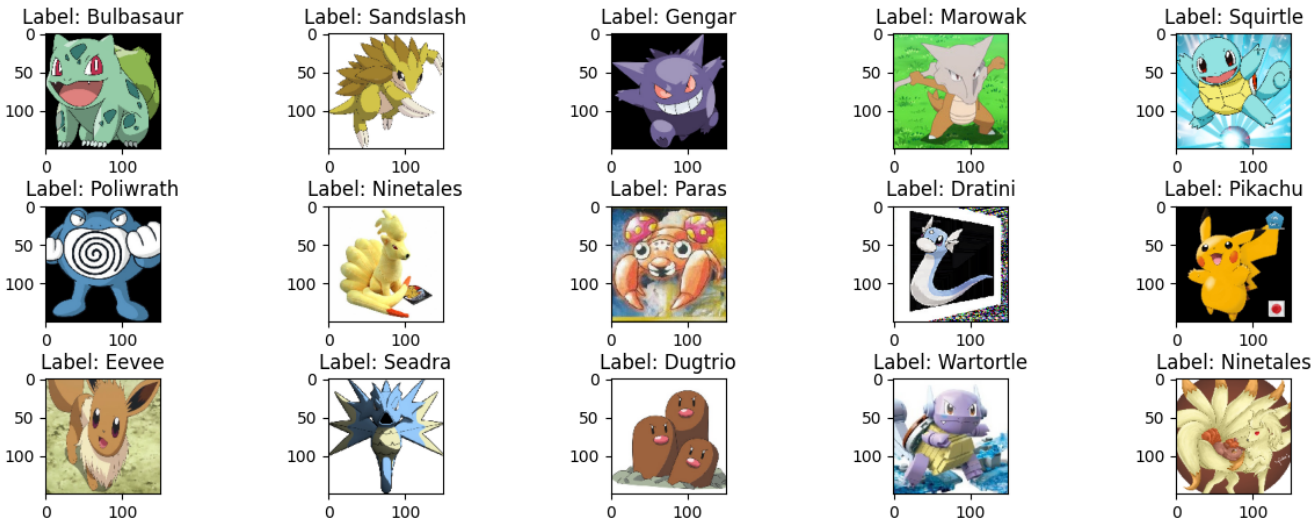

Figura 1. Muestra de pokemones del conjunto de datos seleccionado

3. Metodología y resultados

Utilizaremos y compararemos dos arquitecturas diferentes para comprobar cuál es más efectiva para nuestro conjunto de datos.

El primero de ellos es `DenseNet201`, el cual es una arquitectura de red neuronal convolucional (CNN) que pertenece a la familia de las Dense Convolutional Networks, comúnmente conocidas como DenseNets. Compuesta por 101 capas, se caracteriza por el uso de conexiones densas entre estas. En lugar de tener cada capa conectada solo a la siguiente, cada capa recibe como entrada las características de todas las capas anteriores y pasa sus propias características a todas las capas subsiguientes.

Esto resulta en un número masivo de conexiones entre capas, lo que permite una mejor reutilización de características y una propagación más eficiente de los gradientes durante el entrenamiento. Estas capas densas, facilitan la propagación de gradientes durante el entrenamiento, mitigando el problema del desvanecimiento del gradiente en redes profundas. (Huang et al., 2017)

La otra técnica que usaremos será la `VGG16`, la cuál es una de las arquitecturas más utilizadas para estos propósitos. Tiene 16 capas de pesos entrenables, que incluyen 13 capas convolucionales y 3 capas completamente conectadas. Todas las capas convolucionales utilizan filtros de tamaño 3x3 con un paso de 1 y padding de 1 pixel para preservar las dimensiones espaciales. Después de cada par o trío de capas convolucionales, se aplica una capa de pooling (max-pooling) de 2x2 con un paso de 2 para reducir las dimensiones espaciales a la mitad. (Simonyan & Zisserman, 2015) Esta arquitectura es de las más sencillas de comprender y es por eso que queremos ponerla a prueba en este estudio.

Para ambas técnicas se usó el mismo conjunto de datos, se eligió `Adam` como optimizador y se configuraron las redes para que se pudiera reducir la tasa de aprendizaje con cada `epoch`, que se configuraron 10 de estos y 32 `batch`. También se configuraron para el término temprano, por si llegaba al estado óptimo. Se medirá la pérdida con `categorical_crossentropy` y se medirá el avance con `accuracy`.

Para el , se configuraron capas adicionales de `pooling` por promedio y una capa de salida de 150 neuronas con la función de aplicación `softmax`.

Para el `VGG16`, se aplastaron los valores, se procesa una capa intermedia de 256 neuronas con función de activación `relU` y una capa de salida de 150 neuronas con la función de aplicación `softmax`.

En la tabla 1, se puede observar una comparación entre las métricas principales para modelos categóricos entre las dos arquitecturas mencionadas con anterioridad.

Arquitectura\Métrica	Precisión	Recall	F1-Score
DenseNet201	0.87	0.86	0.86
VGG16	0.68	0.63	0.63

Cuadro 1. Comparación de arquitecturas

`DenseNet201` muestra un rendimiento superior en todas las métricas comparadas con `VGG16`. Esto indica que `DenseNet201` es más efectivo en la clasificación de imágenes para el conjunto de datos. Este tiene una precisión del 87 %, lo cual es un número muy bueno si lo comparamos con el otro, y considerando que el número de epochs fue pequeño, es un muy buen indicador que la arquitectura funciona mejor para este caso particular. En cuanto a tiempos, DenseNet201 fue mucho más rápido que VGG16.

En las figuras 2 y 3, podemos observar los diferentes errores de categorización. Podemos ver, por ejemplo, que los errores del `DenseNet201`, son errores mínimos y en muchos se pueden observar que son referentes a confusiones entre las evoluciones, lo que hace entendible y justifica de cierta manera el error. Mientras que con la arquitectura `VGG16`, también hay errores del tipo de evolución, pero hay más errores un tanto aleatorios que no tienen justificación humanamente explicada.

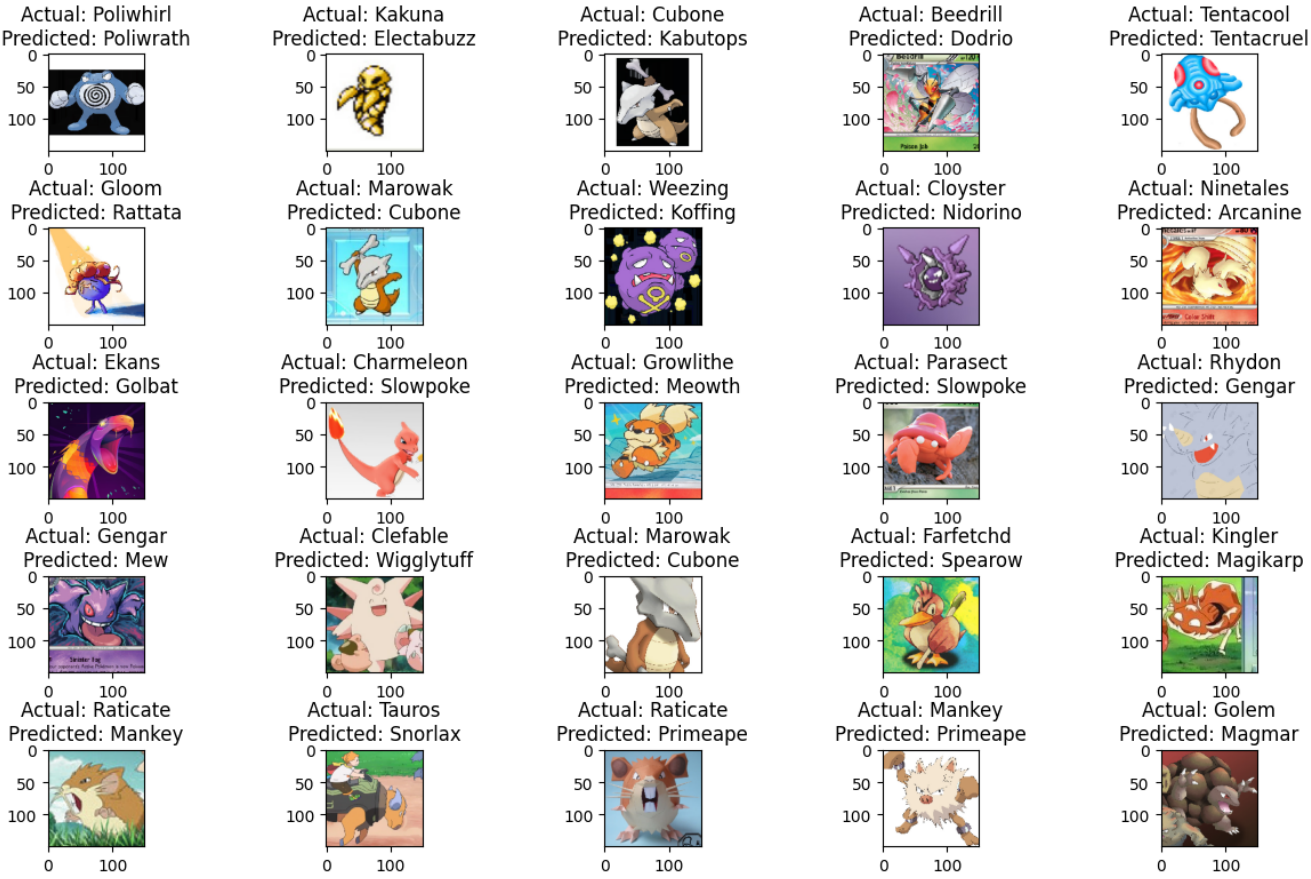

Figura 2. Errores aleatorios de la arquitectura DenseNet201

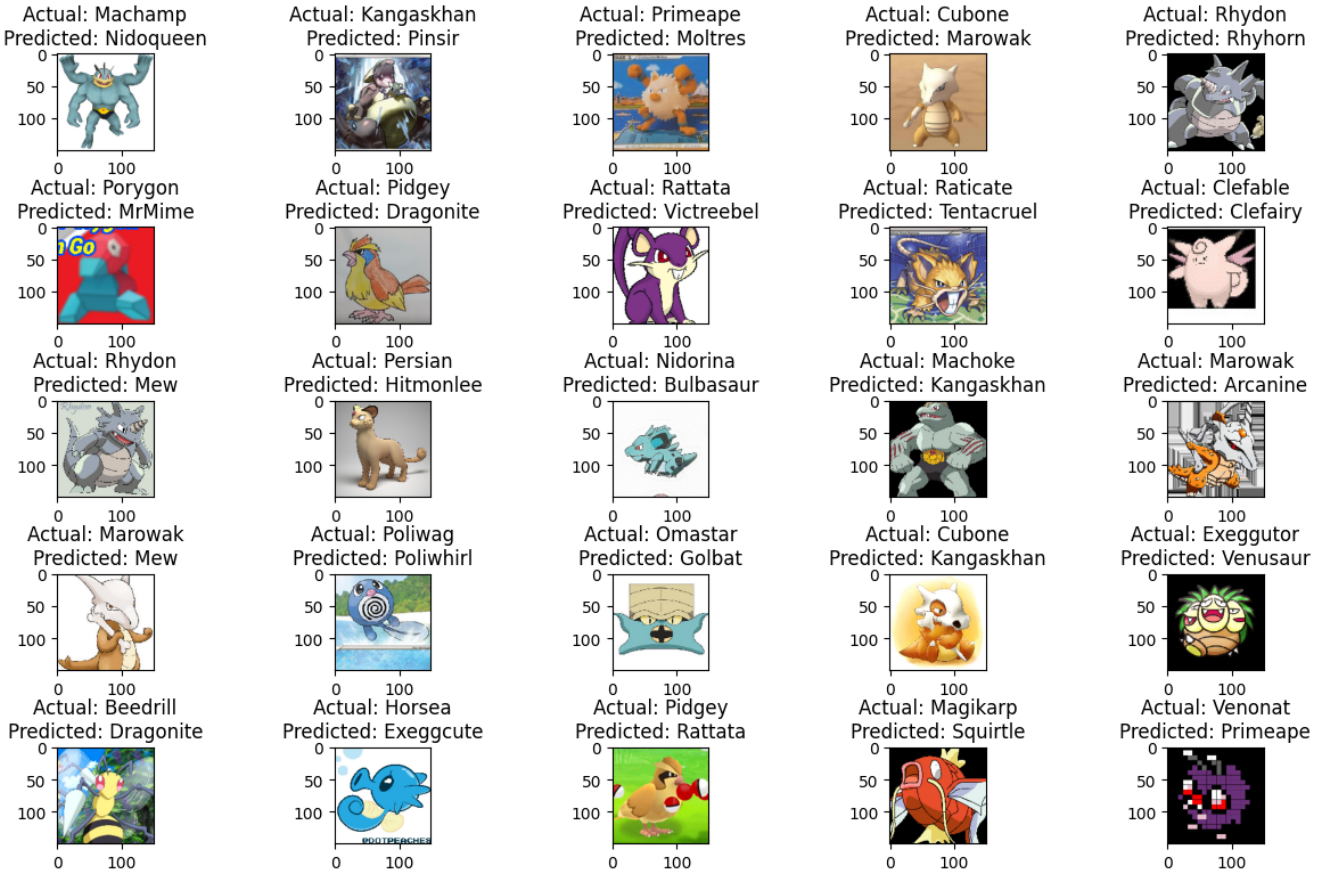

Figura 3. Errores aleatorios de la arquitectura VGG16

4. Conclusión

En este estudio, se compararon dos arquitecturas de redes neuronales convolucionales, DenseNet201 y VGG16, utilizando técnicas de transferencia de aprendizaje para la clasificación de imágenes de pokemones, siendo el primero el claro ganador de la comparación. Para casos reales, es importante hacer este tipo de diseños de experimentos para poder elegir una arquitectura adecuada que a largo plazo beneficie, ya que en ocasiones por el ahorro de tiempo se elige una arquitectura sencilla, pero que consume muchos recursos tanto computacionales como humanos. Para futuros análisis podemos utilizar otro tipos de arquitecturas y compararlos, o aumentar la muestra del conjunto de datos o el tamaño de epochs para aumentar su eficiencia.

Referencias

Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2261-2269. https://doi.org/10.1109/CVPR.2017.243

Simonyan, K., & Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition. *International Conference on Learning Representations (ICLR)*. https://arxiv.org/abs/1409.1556