Taller 2

Juan Sebastián Peláez Pardo
Nicolás Tibatá

Ciencia de Datos Aplicada

Se tienen carpetas de datos con el set de datos ya dividido entre train, validación y test, por lo que no hace falta dividir el set de datos para el entrenamiento del modelo ya que se tiene dividido. Para el set de train se tiene 2407 imágenes, para el set de validación 291 y para el set de test 2266. Se definió que hay 43 clases de imágenes, las cuales son: Apple, Asparagus, Aubergine, Avocado, Banana, Cabbage, Carrots, Cucumber, Ginger, Juice, Kiwi, Leek, Lemon, Lime, Mango, Melon, Milk, Oat-Milk, Oatghurt, Onion, Orange, Passion-Fruit, Peach, Pear, Pepper, Pineapple, Pomegranate, Potato, Red-Beet, Red-Grapefruit, Satsumas, Sour-Cream, Soyghurt, Tomato, Yoghurt y Zucchini, y las cuáles tienen dimensiones variables pero con una mayor cantidad en 348 x 348 pixeles.

A partir de estos datos y clases se realiza el entrenamiento de los tres modelos. El primer es una red Convolucional con la siguiente estructura:

Layer (type)	Output Shape	Param #
Capa_Reescalado (Rescaling)	(None, 256, 256, 3)	0
Capa_Convolucional_1 (Conv2D)	(None, 254, 254, 64)	1,792
Max_Pool_1 (MaxPooling2D)	(None, 127, 127, 64)	0
Capa_Convolucional_2 (Conv2D)	(None, 125, 125, 128)	73,856
Avg_Pool_1 (AveragePooling2D)	(None, 62, 62, 128)	0
Capa_Convolucional_3 (Conv2D)	(None, 60, 60, 256)	295,168
Flatten_Layer (Flatten)	(None, 921600)	0
Capa_Densa_1 (Dense)	(None, 128)	117,964,928
Capa_Densa_2 (Dense)	(None, 64)	8,256
Capa_Respuesta (Dense)	(None, 36)	2,340

Total params: 118,346,340 (451.46 MB)
Trainable params: 118,346,340 (451.46 MB)
Non-trainable params: 0 (0.00 B)

Como se tienen 36 clases dentro del set de validación y train. Este modelo nos da como resultado en el set de train el siguiente rendimiento:

	precision	recall	f1-score	support
micro avg	0.87	0.87	0.87	2407
macro avg	0.89	0.85	0.85	2407
weighted avg	0.89	0.87	0.87	2407
samples avg	0.87	0.87	0.87	2407

Por otro lado, para test se tiene el siguiente rendimiento:

	precision	recall	f1-score	support
micro avg	0.38	0.38	0.38	2266
macro avg	0.30	0.27	0.26	2266
weighted avg	0.42	0.38	0.37	2266
samples avg	0.38	0.38	0.38	2266

Podemos observar que para train se tiene un buen rendimiento pero para test el rendimiento cae significativamente con rastros de sobreajuste a los datos.

El segundo modelo se realiza con búsqueda de hiperparámetros y se obtuvo la siguiente arquitectura:

Layer (type)	Output Shape	Param #
Reescalado (Rescaling)	(None, 256, 256, 3)	0
Convolucional_1 (Conv2D)	(None, 254, 254, 64)	1,792
MaxPooling_1 (MaxPooling2D)	(None, 127, 127, 64)	0
Convolucional_2 (Conv2D)	(None, 125, 125, 256)	147,712
MaxPooling_2 (MaxPooling2D)	(None, 62, 62, 256)	0
Flatten (Flatten)	(None, 984064)	0
Densa (Dense)	(None, 128)	125,960,320
Salida (Dense)	(None, 36)	4,644

Total params: 126,114,468 (481.09 MB)
Trainable params: 126,114,468 (481.09 MB)
Non-trainable params: 0 (0.00 B)

Este modelo nos da para el set de train el siguiente rendimiento:

	precision	recall	f1-score	support
micro avg	1.00	0.99	1.00	2407
macro avg	1.00	0.99	1.00	2407
weighted avg	1.00	0.99	0.99	2407
samples avg	0.99	0.99	0.99	2407

Para el set de test nos da el siguiente rendimiento:

	precision	recall	f1-score	support
micro avg	0.47	0.40	0.43	2266
macro avg	0.36	0.24	0.26	2266
weighted avg	0.45	0.40	0.40	2266
samples avg	0.40	0.40	0.40	2266

El análisis comparativo de los modelos muestra que el modelo 2 presenta un mejor desempeño en el conjunto de prueba, lo que indica una mayor capacidad de generalización, sin embargo seguimos viendo ese sobreajuste de los datos, indicando que tal vez pueda ser la estructura como tal de la red convolucional. Al examinar las métricas, se observa que el modelo 1 logra un recall promedio del 38%, lo que significa que identifica correctamente el 38% de las imágenes en cada categoría de alimento. Por otro lado, el modelo 2 mejora este valor, alcanzando un recall promedio del 40%.

En cuanto a la precisión, el modelo 1 clasifica correctamente, en promedio, el 38% de las imágenes asignadas a cada clase, mientras que el modelo 2 incrementa esta métrica a un 47%, mostrando una mejora considerable. Esto demuestra que el modelo 2 no solo tiene una mejor capacidad para identificar correctamente las imágenes, sino que también lo hace con mayor exactitud en cada categoría analizada.

Pero estos resultados no son del todo confiables por el sobreajuste que presentan, es por esto que se decidió crear un tercer modelo con arquitectura CNN + Data Augmentation permitiendo un entrenamiento con una mejor generalización de los resultados. Además se realizó un resize de las imágenes a 180 x 180 para no tomar características innecesarias de las imágenes y para un mejor procesamiento, además de una arquitectura no tan exhaustiva gracias a la generación de imágenes adicionales.

Resultados del modelo 3 para el conjunto train:

accuracy			0.83	2640
macro avg	0.83	0.81	0.81	2640
weighted avg	0.84	0.83	0.82	2640

Resultados del modelo 3 para el conjunto test:

accuracy			0.64	2485
macro avg	0.54	0.51	0.51	2485
weighted avg	0.64	0.64	0.62	2485

Como podemos ver el modelo 3 tiene menos indicios de sobre ajuste, pues tiene una precisión sobre train de 0.84 y sobre test de 0.64, es decir que clasifica correctamente el 64% de las imágenes de test. Además es una arquitectura menos exhaustiva y toma datos aumentados que promueven la generalización.

Por último, vamos a calcular la generación de valor del modelo. En primer lugar definimos que el tiempo asociado al registro de productos es de 4 minutos en promedio, el costo de tiempo asociado al registro de productos es de $10.000 por hora, el tiempo que se ahorra con el

modelo es de 3 minutos, en 30 días hay 1.000 productos registrados y el salario de un científico de datos es de $30.000 por hora.

a = ahorros esperados - ((1 - precisión calculada promedio)*costo de arreglar un error manualmente)

La ganancia por producto es de 1.56 minutos o aproximadamente 1560 minutos por 1.000 productos registrados al mes.

Por otro lado, el modelo está ahorrando $260.000 por mes:

```
[26] g_mes = g*1000
     g_mes_en_horas = g_mes/60
     ahorro_modelo = 10.000*g_mes_en_horas
     print(ahorro_modelo)

     260.0
```

Por último, si el modelo toma 2 semanas de trabajo, se tendrá un ROI de 0 luego de aproximadamente 9 meses de trabajo:

```
[27] s = 30.000*80
     roi_cero = s/ahorro_modelo
     print(roi_cero)

     9.23076923076923
```

Insights y conclusiones:

El análisis de los tres modelos implementados resalta la importancia de la optimización en la arquitectura, aumentación de las imágenes y los hiperparámetros para mejorar la capacidad de generalización. Aunque el modelo 1 mostró un desempeño adecuado en el conjunto de entrenamiento, su rendimiento en el conjunto de prueba disminuyó significativamente, evidenciando problemas de sobreajuste. Por el contrario, el modelo 2, optimizado mediante búsqueda de hiperparámetros, demostró un desempeño superior, con un incremento notable en métricas clave como precisión (47% frente a 38% en promedio) y recall (40% frente a 38% en promedio). Esto confirma que el modelo 2 no solo identifica con mayor exactitud las imágenes en cada clase, sino que también clasifica correctamente una mayor proporción de los datos de prueba. Sin embargo, el modelo 3 es el que tiene menos indicios de sobreajuste, cuesta menos computacionalmente e incluye aumentación de datos para aumentar su capacidad de generalización, consolidándose como la solución más robusta para el problema planteado.

Desde el punto de vista del impacto económico, el modelo ofrece un ahorro significativo en el tiempo de registro de productos, estimado en 26 horas mensuales, lo que equivale a $260,000 de ahorro mensual. Si bien el desarrollo del modelo requirió una inversión inicial, el retorno de la misma se alcanzará en aproximadamente 9 meses, demostrando su viabilidad económica a largo plazo. En conclusión, la implementación del modelo 3 no solo mejora la eficiencia técnica en la clasificación de imágenes, sino que también representa una inversión estratégica con beneficios tangibles en productividad y costos operativos.