Red Neuronal Convolucional para la clasificación de plátanos

Este proyecto se basó en el uso de una red neuronal convolucional con tensorflow para clasifcar imágenes de plátanos dentro de 3 categorías:

->Fresh

->Rotten

->Unripe

El data set utilizado para el entrenamiento, la validación y las pruebas de testeo consta de más de 400 imágenes. Para mejorar el conjunto de datos y mejorar el rendimiento del modelo, se aplicaron técnicas de aumento de datos de imágenes. Se utilizaron las siguientes métricas de aumento:

->Reescalado

->Aumento y disminución de brillo

->Rotación de la imagen

->Zoom

->Giro horizontal

->Giro vertical

El propósito de estas transformaciones es evitar la deformación de la figura del plátano pero que a su vez, cree imágenes distintas entre los ejemplos base.

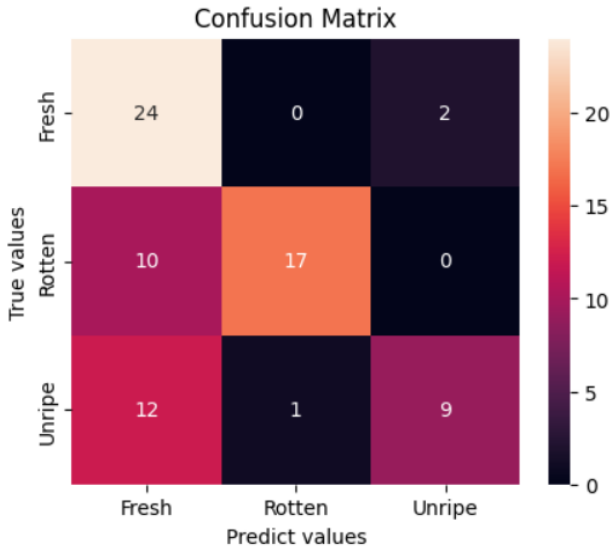
Mediante la incorporación de técnicas de aumento de datos y el uso de una combinación de múltiples conjuntos de datos, se tiene como objetivo que el modelo pueda mejorar su capacidad para clasificar las 3 clases de plátanos con mayor precisión

La estructura base de la red neuronal convolucional con la siguiente estructura:

```
 Layer (type)                Output Shape              Param #
=================================================================
 vgg16 (Functional)          (None, 7, 7, 512)         14714688

 flatten (Flatten)           (None, 25088)             0

 dense (Dense)               (None, 120)               3010680

 dense_1 (Dense)             (None, 3)                 363

=================================================================
```

Se escogió utilizar vgg16, por la clasificación de imágenes que tiene como entrenamiento, la capa de flatten o aplanamiento, es para que la información resultante del vgg16 pase a estar dentro de una sola dimensión, la capa de 120 neuronas es porque hay 3 categorías a clasificar y la última capa de 3 neuronas es para que cada neurona termine de clasificar los patrones encontradas y pueda dar un resultado escogido entre las 3 categorías.

Entrenado con 10 épocas, los resultados de la certeza durante el entrenamiento tienen un promedio de 0.55 , los relacionados a la parte de validación de datos tiene un promedio de 0.58 y el resultado al momento de probarlo con la parte del test fue 0.66. El tiempo que tardó en compilarse, entrenarse y mostrar resultados fue de 23.5min.



La matriz de riesgos presenta una tendencia muy clara al momento de clasificar de manera adecuada los plátanos frescos, sin embargo, en las otras 2 categorías flaquea mucho.

Para mejorar la anterior neurona y ver si se podía mejorar la clasificación de las otras 2 categorías se realizaron algunos ajustes dentro de los hiperparámetros.

Mejora de parámetros 1 (Doc. CNN2.ipynb):

Cambios realizados

-->Ajuste del tamaño de imagen de 250x250 a 512x512 pixeles

-->Épocas de training 10 -> 15

-->Se agrego una nueva capa de neurona

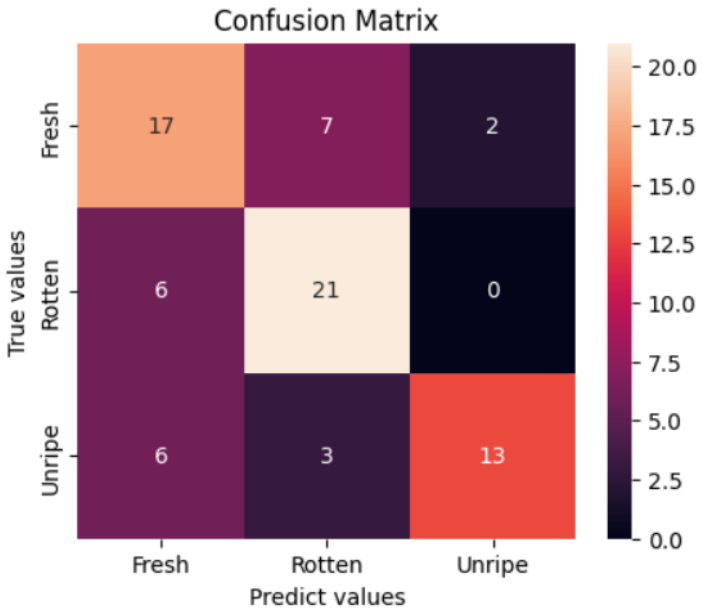
model.add(layers.Dense(40,activation='relu'))

Las decisiones del cambio de hiperparámetros se deben a que uno de los data set de donde saqué más imágenes y las procesaban a 512x512 pixeles[1], el incremento de épocas para ver si el modelo necesitaba más entrenamiento y la capa nueva, para ver si el modelo necesitaba algo más para ayudarle a seleccionar los patrones de las imágenes.

La neurona quedo construida de la siguiente manera:

Layer (type)	Output Shape	Param #
vgg16 (Functional)	(None, 16, 16, 512)	14714688
flatten (Flatten)	(None, 131072)	0
dense (Dense)	(None, 120)	15728760
dense_1 (Dense)	(None, 40)	4840
dense_2 (Dense)	(None, 3)	123

Sección de la comprobación	**Promedio de la certeza**
Train	0.52
Validación	0.59
Test	0.68
Tiempo de ejecución: 109.25 mins	



Lo obtenido dentro de este caso, es que al aumentarle el tamaño de las imágenes a procesar, el tiempo de ejecución y entrenamiento creció demasiado sin dar un aumento significativo dentro los resultados de certeza.

La matriz tiene buenos resultados para la categoría de Rotten, pero en las otras 2, hay problemas al momento de discernir a cuál categoría pertenecen.

Mejora de parámetros 2 (Doc. CNN3.ipynb):

Cambios realizados

-->Épocas de training 10 -> 15

-->Se agrego una nueva capa de neurona

model.add(layers.Dense(40,activation='relu'))

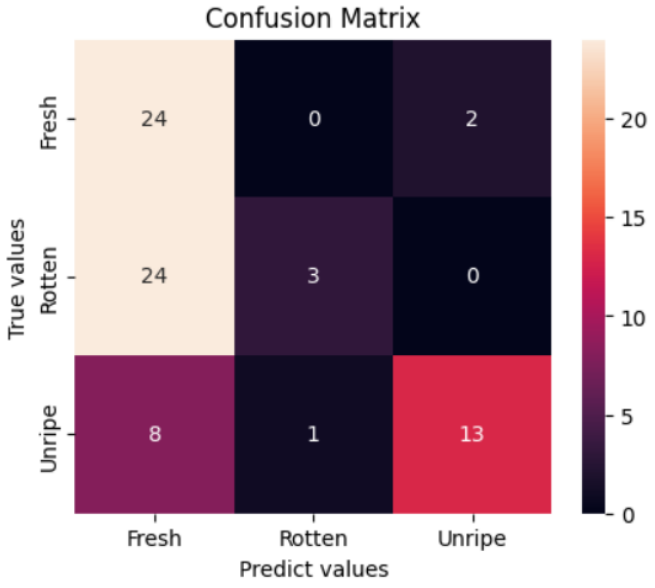
-->Batch size 8 -> 12 en train_generator

-->Batch size 6 -> 8 en val_generator

Para este siguiente experimento se quitó la opción de el tamaño de las imágenes dejándolo en el base de 250x250, para ver si hay un cambio significativo con la calidad de las imágenes al momento de entrenar, sin embargo, se aumentó el número de muestras que recibe el modelo al entrenarse y verificar con el ser de validación para ver si es que el modelo necesita más muestras para un mejor resultado.

Layer (type)	Output Shape	Param #
vgg16 (Functional)	(None, 7, 7, 512)	14714688
flatten_1 (Flatten)	(None, 25088)	0
dense_3 (Dense)	(None, 120)	3010680
dense_4 (Dense)	(None, 40)	4840
dense_5 (Dense)	(None, 3)	123

Sección de la comprobación	**Promedio de la certeza**
Train	0.62
Validación	0.59
Test	0.53
Tiempo de ejecución: 47.9 mins	



Este experimento presentó peores resultados que el base, pero, dentro de la matriz, pudo discriminar mejor los plátanos frescos que los de las otras categorías, lo cual da un indicio de cuales son los hiperparámetros a seguir y comprobando la hipótesis de que al meterle más imágenes al entrenamiento mejora el resultado.

Mejora de parámetros 3 (Doc. CNN4.ipynb):

Cambios realizados

-->Épocas de training 10 -> 15

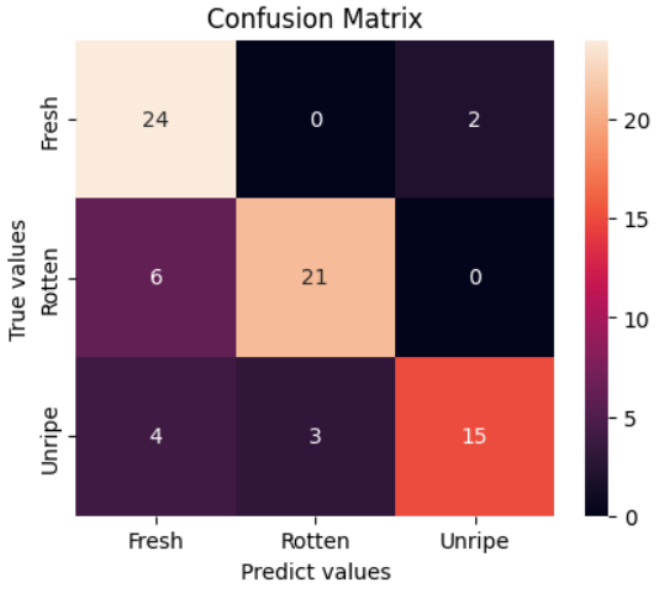
-->Batch size 8 -> 12 en train_generator

-->Batch size 6 -> 8 en val_generator

Para este experimento se quitó la capa de neurona, para ver si es que el modelo se está confundiendo al tener procesar los datos, lo que si se mantuvo fue el cambio de las épocas y el tamaño de las imágenes que recibe el modelo al entrenar y validar, con esto se quiere lograr que el modelo aprenda más los patrones presentes en cada categoría y pueda distinguirlos mejor.

Layer (type)	Output Shape	Param #
vgg16 (Functional)	(None, 7, 7, 512)	14714688
flatten (Flatten)	(None, 25088)	0
dense (Dense)	(None, 120)	3010680
dense_1 (Dense)	(None, 3)	363

Sección de la comprobación	**Promedio de la certeza**
Train	0.66
Validación	0.67
Test	0.80
Tiempo de ejecución: 46.1 mins	



En este caso, la matriz de confusión si presentó una mejora significativa en la clasificación de la clase de Rotten, conservando la eficacia de la clase Fresh. En la parte de Unripe, si hubo una mejora, pero no muy notoria, esto puede ser debido a que es la clase dentro del data set con mejor número de instancias.

Conclusión:

Como resultado en general, el experimento 3 es el que tiene mejor certeza en la clasificación de clases en general, pero, solo sería mejor utilizarlo para las clases de Rotten y Fresh, ya que termina habiendo menos probabilidad de que se equivoque y la clase más preocupante dentro de todas es la de Rotten, debido a que es la que puede terminar afectando a las demás frutas y ya va a ser apta para el consumo humano. El aumento en los pixeles de las imágenes, puede que no sea un valor determinante dentro de este data set y termine solo aumentando el tiempo de entrenamiento. Lo que más terminó afectando al modelo, fue que la clase de Unripe era la que contaba con menos instancias y termino perjudicando su clasificación.

Referencias:

[1] https://www.sciencedirect.com/science/article/pii/S2352340922007594?ref=pdf_download&fr=RR-2&rr=7c994fbbc8004865

[2] https://www.kaggle.com/datasets/sriramr/fruits-fresh-and-rotten-for-classification

[3] https://github.com/giovannipcarvalho/banana-ripeness-classification

[4] https://data.mendeley.com/datasets/y3649cmgg6