Instituto Tecnológico y de Estudios Superiores de Monterrey Campus Querétaro

CONCENTRACIÓN DE CIENCIA DE DATOS E INTELIGENCIA ARTIFICIAL AVANZADA

MÓDULO 2

IMPLEMENTACIÓN DE UN MODELO DE DEEP LEARNING

PORTAFOLIO DE IMPLEMENTACIÓN

JOSEMARÍA ROBLEDO LARA

A01612376

El modelo utilizado es una red neuronal convolucional (CNN), para clasificar imágenes de flores en cinco tipos; Dandelion, Rose, Tulip, Sunflower y Daisy. El modelo es capaz de recibir una imagen y determinar el tipo de flor que es.

Se utilizaron un aproximado de 3,500 imágenes como dataset, estas imágenes fueron divididas en cinco carpetas, cada una de su tipo correspondiente. A su vez los datos fueron divididos en 65% train, 25% validation y 10% test.

Para el manejo de datos y creación del modelo se utilizaron los siguientes frameworks:

- Manipulación de datos
  - Pandas
  - Numpy

- Visualización
  - Matplotlib
  - Seaborn

- Redes Neuronales
  - Keras
  - Tensorflow

Como fue mencionado, el modelo posee una **arquitectura** de CNN, por ende, consta de **capas convolucionales** y de **max pooling** intercaladas con capas completamente conectadas. Las capas convolucionales se utilizan para extraer características de las imágenes y las capas de max pooling reducen la resolución espacial. En la capa final de salida se clasifican las imágenes con la función de activación **softmax.**

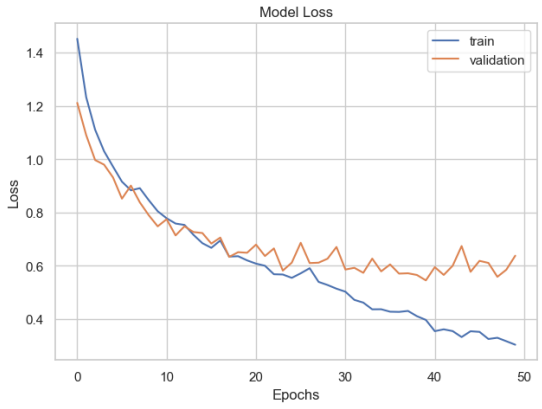
El tamaño de los hiperparámetros del **batch size** (tamaño de lote) y de las **epochs** (épocas) son de 128 y 50 respectivamente. Para el **callback** se utiliza **ReduceLROnPlateau**, que ajusta dinámicamente la tasa de aprendizaje durante el entrenamiento, con el fin de mejorar la convergencia del modelo y evitar el estancamiento en el entrenamiento.

Durante el proceso de entrenamiento, el modelo ajustará sus pesos y bias (sesgos), utilizando los datos de train y calculará las pérdidas y las métricas en los datos de train y validación en cada época. Capturándolos dentro de la

variable “History”, lo cual es importante para visualizar más adelante los resultados. El desempeño del modelo fue el siguiente:

```
Epoch 50/50
25/25 [==============================] - 66s 3s/step - loss: 0.3041 - accuracy: 0.8859 - val_loss: 0.6381 - val_accuracy: 0.7944
```

Se obtuvo un valor de pérdida (**loss**) de 0.3041, lo que significa que en train hay un bajo error promedio en las predicciones. En el caso de la exactitud (**accuracy**) es un valor de 0.8859, dicho en otras palabras, sería un 88.59% de exactitud en el entorno de entrenamiento. Al mismo tiempo se muestran en los resultados otros dos valores similares a los previamente mencionados. El “val_loss” y “val_acc”, siendo valores que describen los resultados con el conjunto de datos de validación, val_loss describe el valor de pérdida, que es 0.6381, y val_acc describe un valor de exactitud de 0.7944.



Las gráficas muestran el desempeño del modelo a través de las épocas, se visualiza que a medida avanzan las épocas la pérdida decrece y la exactitud aumenta, existiendo un mejor progreso en el entorno de train, esto puede deberse a la dificultad que representa predecir los nuevos datos y reducidos de validation.

Se hicieron algunos ajustes a los hiperparámetros para mejorar el modelo, se optó por aumentar el número de steps por epoch, de 25 a 28 y se obtuvieron los siguientes resultados:

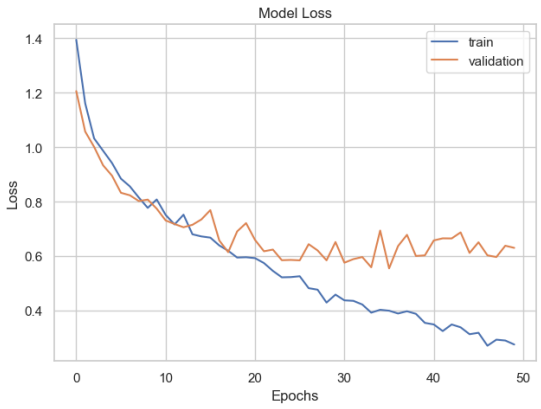
```
Epoch 50/50
28/28 [==============================] - 66s 2s/step - loss: 0.2756 - accuracy: 0.8964 - val_loss: 0.6304 - val_accuracy: 0.8056
```

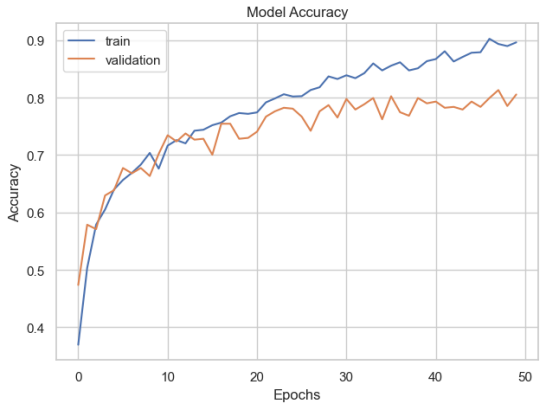
Los ajustes fueron beneficiosos para el modelo, obteniendo estas mejoras:

Magnitud	Primera Iteración	Iteración Final
Loss	0.3041	0.2756
Accuracy	0.8859	0.8964
Val_loss	0.6381	0.6304
Val_accuracy	0.7944	0.8056

Hubo mejoras en las cuatro medidas para calificar la calidad del modelo, reduciéndose la pérdida y aumentándose la exactitud, del mismo modo se mejoraron los resultados con respecto al conjunto de datos de validación. Aumentar en un gran número el hiperparámetro de los steps, podría generar overfitting.

Las gráficas finales son las siguientes:





Para finalizar y seguir probando el modelo, se guarda para ser ejecutado en un archivo separado donde se ingresa la imagen de una flor y se regresa un texto indicando el tipo de flor que es.