Construcción de un modelo de IA para clasificar alimentos. Este proceso incluye la preparación de datos, el entrenamiento del modelo y su evaluación.

**Paso 1: Preparación de Datos**

-Recolección de Datos:

Se necesita un conjunto de imágenes de diferentes alimentos. Puedes obtener estas imágenes de bases de datos públicas como ImageNet.

Asegurar de tener varias imágenes para cada categoría de alimento que se dese clasificar.

-Etiquetado de Datos:

Etiquetar cada imagen con el nombre del alimento correspondiente.

Utilizar herramientas como LabelImg para facilitar el etiquetado.

-Preprocesamiento de Datos:

Redimensionar las imágenes a un tamaño uniforme, por ejemplo, 128x128 píxeles.

Normalizar los valores de los píxeles dividiéndolos por 255 para que estén en el rango [0, 1].

Dividir el conjunto de datos en entrenamiento (80%), validación (10%) y prueba (10%).

**Paso 2: Construcción del Modelo**

-Importar Librerías:

Python:

import tensorflow as tf

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense

-Definir el Modelo:

model = Sequential([

Conv2D(32, (3, 3), activation='relu', input\_shape=(128, 128, 3)),

MaxPooling2D((2, 2)),

Conv2D(64, (3, 3), activation='relu'),

MaxPooling2D((2, 2)),

Flatten(),

Dense(128, activation='relu'),

Dense(num\_classes, activation='softmax')

])

-Compilar el Modelo:

model.compile(optimizer='adam',

loss='sparse\_categorical\_crossentropy',

metrics=['accuracy'])

**Paso 3: Entrenamiento del Modelo**

-Preparar los Datos de Entrenamiento:

train\_dataset = tf.keras.preprocessing.image\_dataset\_from\_directory(

'path\_to\_train\_data',

image\_size=(128, 128),

batch\_size=32

)

validation\_dataset = tf.keras.preprocessing.image\_dataset\_from\_directory(

'path\_to\_validation\_data',

image\_size=(128, 128),

batch\_size=32

)

-Entrenar el Modelo:

history = model.fit(

train\_dataset,

validation\_data=validation\_dataset,

epochs=10

)

**Paso 4: Evaluación del Modelo**

-Preparar los Datos de Prueba:

test\_dataset = tf.keras.preprocessing.image\_dataset\_from\_directory(

'path\_to\_test\_data',

image\_size=(128, 128),

batch\_size=32

)

-Evaluar el Modelo:

test\_loss, test\_acc = model.evaluate(test\_dataset)

print(f'Test Accuracy: {test\_acc}')

-Visualización de Resultados:

Puedes graficar la precisión y pérdida del modelo durante el entrenamiento y la validación para entender mejor su rendimiento.

import matplotlib.pyplot as plt

acc = history.history['accuracy']

val\_acc = history.history['val\_accuracy']

loss = history.history['loss']

val\_loss = history.history['val\_loss']

epochs\_range = range(10)

plt.figure(figsize=(8, 8))

plt.subplot(1, 2, 1)

plt.plot(epochs\_range, acc, label='Training Accuracy')

plt.plot(epochs\_range, val\_acc, label='Validation Accuracy')

plt.legend(loc='lower right')

plt.title('Training and Validation Accuracy')

plt.subplot(1, 2, 2)

plt.plot(epochs\_range, loss, label='Training Loss')

plt.plot(epochs\_range, val\_loss, label='Validation Loss')

plt.legend(loc='upper right')

plt.title('Training and Validation Loss')

plt.show()