



CHARACTER CLASSIFICATION

Implementación de un modelo de deep learning

Inteligencia artificial avanzada para la ciencia de datos II

Osvaldo Del Valle

A01275702

2/11/2024

Índice

Introducción	3
Dataset	3
Modelo	4
Implementación	4
Resultados	5
Generación de predicciones	8
Nuevas clases	9
Modelo alternativo - CNN	12
Mejora	14

Introducción

Este proyecto se centra en el desarrollo de un modelo de deep learning capaz de clasificar imágenes de personajes específicos de la serie *The Quintessential Quintuplets*, donde cada una de las cinco protagonistas presenta rasgos físicos similares, para realizar esta clasificación se ha empleado un modelo basado en redes neuronales profundas utilizando la arquitectura EfficientNetB0, pre entrenada en *ImageNet* y ajustada para capturar las diferencias específicas entre los personajes.

Con un conjunto de datos de imágenes etiquetadas de cada una de las quintillizas, el modelo busca automatizar el proceso de identificación visual de cada hermana, optimizando así la precisión

Dataset

El dataset utilizado en este proyecto está compuesto por imágenes de cinco personajes principales de la serie *The Quintessential Quintuplets*, cada uno correspondiente a una de las hermanas, el conjunto de datos está organizado en cinco carpetas cada una etiquetada con el nombre de la respectiva hermana y contiene entre 600 y 700 imágenes aproximadas por carpeta, este diseño de carpetas facilita la carga de datos para la clasificación permitiendo que cada imagen esté directamente relacionada a una etiqueta de clase correspondiente al personaje

A continuación se muestra una imagen de los 5 personajes de la serie, que son los personajes que se busca que el modelo reconozca en base a fotos individuales.



En cuanto al procesamiento se ajustó el tamaño de las imágenes a dimensiones más pequeñas para un procesamiento más eficiente por el modelo durante el entrenamiento, se usó las dimensiones de 300 x 150 para a su vez conservar la relación de aspecto de las imágenes. Luego se dividió el conjunto de datos en tres subconjuntos: entrenamiento (train), validación (val) y prueba (test), añadiendo el parámetro para conservar la proporción de imágenes de cada clase en cada conjunto lo cual ayuda a mantener una representación proporcional de los datos a lo largo del entrenamiento y evaluación

Modelo

Se seleccionó EfficientNetB0 una arquitectura de red neuronal convolucional conocida por su eficiencia y precisión en tareas de clasificación de imágenes, el modelo se utiliza con *transfer learning* ya que se carga con pesos pre entrenados en *ImageNet* un conjunto de datos extenso que permite captar características visuales generales, al emplear esta técnica el modelo aprovecha las representaciones aprendidas previamente facilitando la diferenciación entre los personajes a partir de rasgos específicos, ajustando EfficientNetB0 específicamente para el conjunto de imágenes se busca lograr una clasificación precisa de cada personaje sin requerir una cantidad masiva de datos o un alto costo computacional

Implementación

En el modelo se han añadido capas adicionales a EfficientNetB0 para optimizar su rendimiento en la clasificación específica del dataset. Primero se incorpora una capa de *GlobalAveragePooling2D* que reduce la dimensionalidad de las características extraídas por EfficientNetB0 al condensar la información relevante de cada imagen buscando mejorar así la eficiencia computacional y evitando redundancias, se agregan capas densas (*Dense layers*) con 128 neuronas y la función de activación ReLU que permite al modelo aprender relaciones no lineales complejas en las características lo cual es esencial para captar las sutiles diferencias entre los personajes, en cuanto a una medida para mitigar el sobreajuste se implementan capas de *Dropout* con una tasa del 20% eliminando conexiones aleatorias durante el entrenamiento y mejorando la capacidad de generalización del modelo

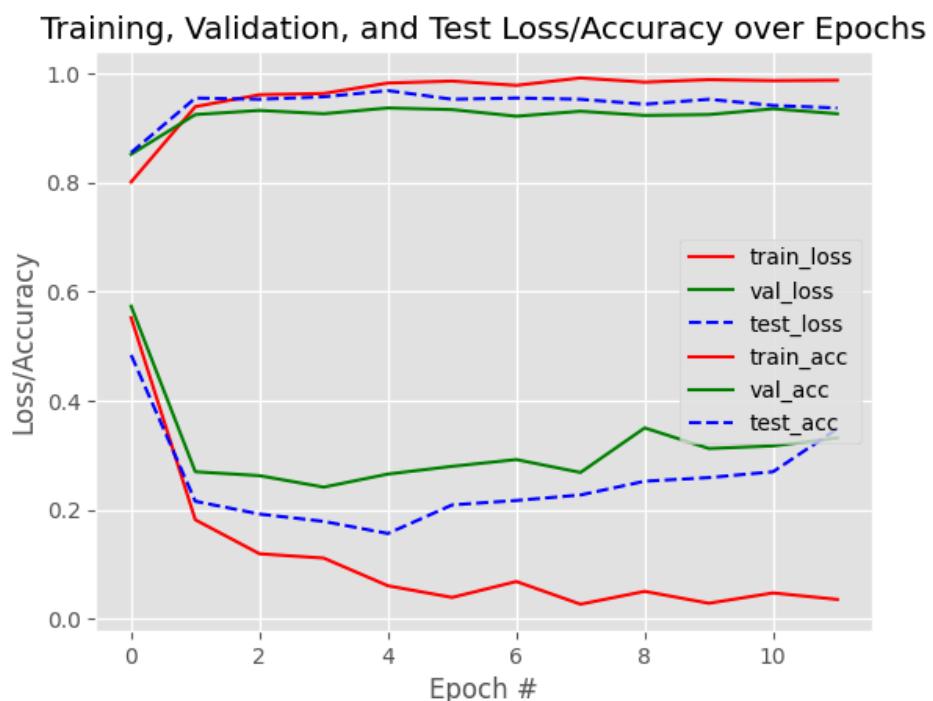
La ultima capa es una capa densa con 5 neuronas y activación *softmax* que convierte las salidas en probabilidades para cada clase, una neurona para cada clase, facilitando la interpretación de los resultados y la clasificación en una de las cinco categorías correspondientes a cada personaje. Además el modelo se entrena usando el optimizador *Adam* conocido por su capacidad de ajuste adaptativo de la tasa de aprendizaje lo cual acelera la convergencia sin necesidad de intervención manual. Finalmente la función de pérdida utilizada es la *categorical crossentropy*, que es buena para problemas de clasificación multicategoría lo que permite evaluar la precisión del modelo en la asignación correcta de probabilidades a cada clase

Layer (type)	Output Shape	Param #
efficientnetb0 (Functional)	(None, 10, 5, 1280)	4,049,571
global_average_pooling2d (GlobalAveragePooling2D)	(None, 1280)	0
dropout (Dropout)	(None, 1280)	0
dense (Dense)	(None, 128)	163,968
dropout_1 (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 5)	645

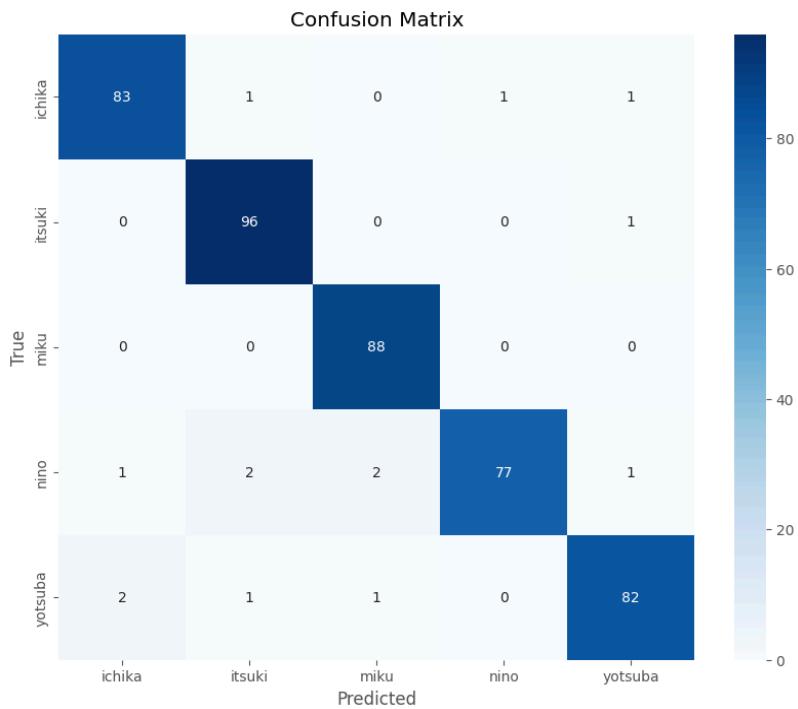
Resultados

En las 12 épocas que el modelo se entrenó mostró una tendencia constante de mejora tanto en el conjunto de entrenamiento como en el conjunto de prueba alcanzando una alta precisión en ambos rápidamente, en la primera época la precisión en el conjunto de entrenamiento fue del 67% con una pérdida de 0.84, mientras que el conjunto de prueba mostró una precisión del 85% y una pérdida de 0.48, lo cual indica que el modelo logró captar características clave de cada clase desde el inicio, y a medida que el entrenamiento avanzó la precisión en el conjunto de prueba alcanzó un pico del 96.8% en la época 5 mientras que el conjunto de validación se mantuvo alrededor del 93% mostrando un ligero sobreajuste en las épocas posteriores, a partir de la época 6 las pérdidas comenzaron a aumentar levemente mientras que la precisión se mantuvo estable especialmente en el

conjunto de prueba, por lo que las últimas épocas observadas la pérdida en el conjunto de prueba se incrementó hasta 0.34 con una precisión del 93% indicando que aunque el modelo sigue siendo muy preciso en la clasificación podría beneficiarse de técnicas adicionales de regularización para mejorar su capacidad de generalización y evitar una posible caída en la precisión en nuevas muestras, además considerar que el modelo pasó por un parado temprano que fue configurado para que si al paso de 7 épocas no existía una mejoría en el valor de precisión del conjunto de validación para el entrenamiento guardando los mejores parámetros obtenidos, esto para evitar un entrenamiento más largo que no estuviera generando mejoría y asegurar guardar los mejores pesos



Apoyando los resultados con la matriz de confusión se puede apreciar que en efecto el modelo cuenta con una buena precisión al clasificar las imágenes de los personajes con su clase correcta, lo que se puede apreciar observando que la diagonal es la que más peso tiene e identificando que los errores con otras clases son mínimos

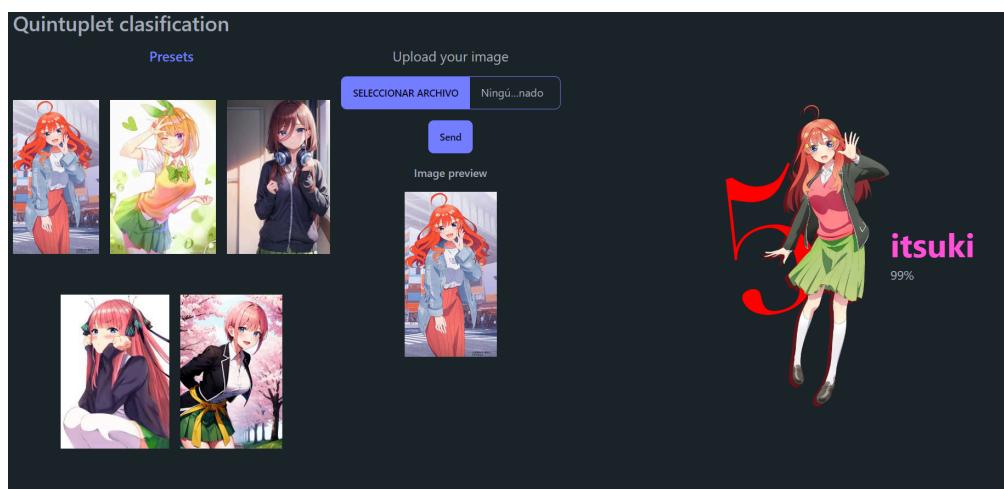


Generación de predicciones

Con el fin de poner a prueba el modelo generado, se realizó una prueba con diferentes imágenes del set de test y mostrando a su vez los resultados, teniendo así una versión más visual de los resultados que el modelo predice. Una vez realizada esta prueba se puede observar cómo efectivamente el modelo puede predecir en su mayoría la clase correspondiente a la imagen presentada

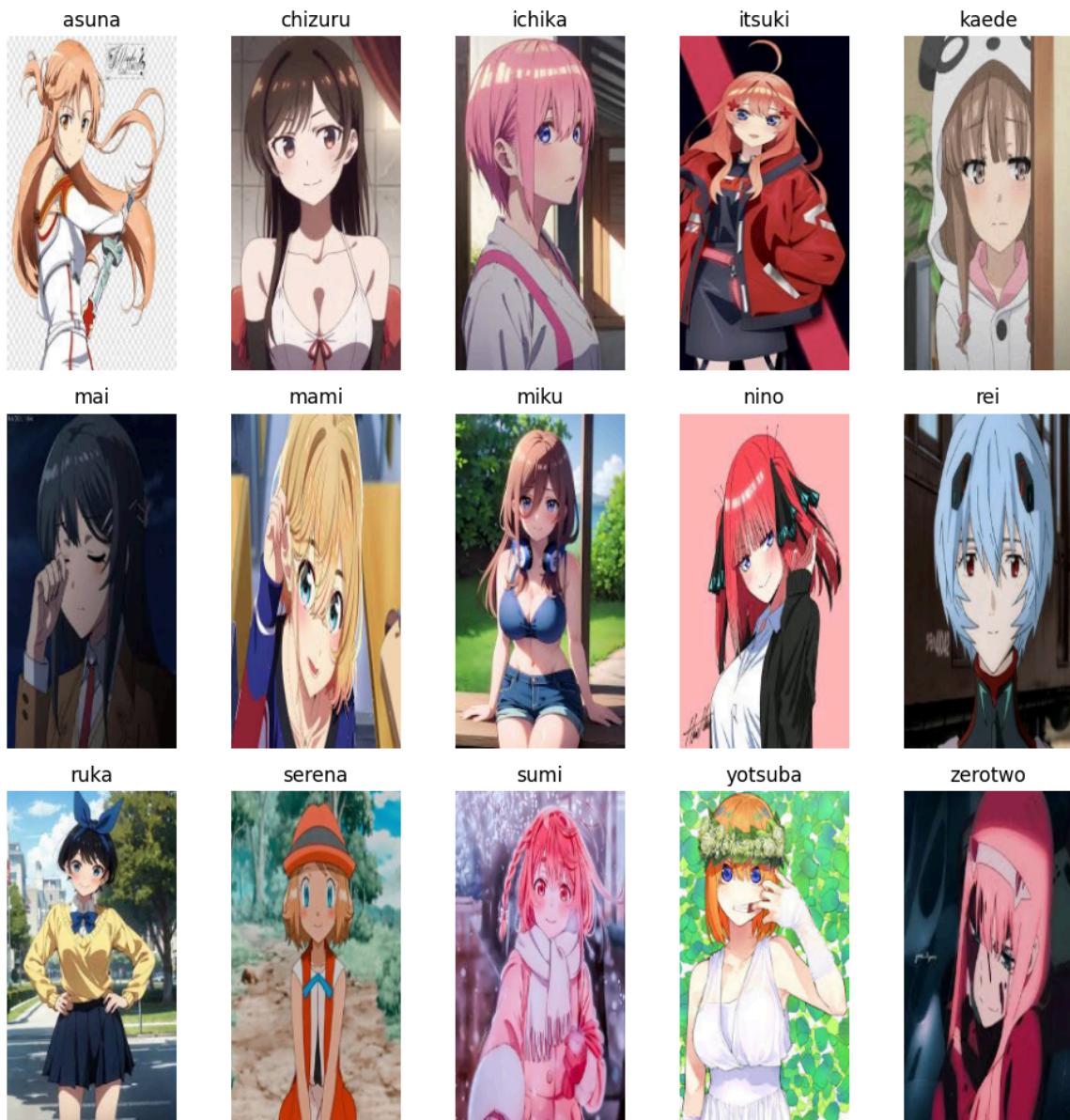


Así mismo se crea una interfaz de usuario interactiva donde se hace el uso del modelo para generar predicciones y mostrar resultados visuales en base a una de las imágenes precargadas seleccionada o bien una imagen proporcionada por el usuario, una vez realizó la predicción se muestra en la pantalla una imagen del personaje relacionado a la clase predicha como resultado acompañado del nombre de la clase (nombre del personaje) y el porcentaje con el que el modelo lo predijo



Nuevas clases

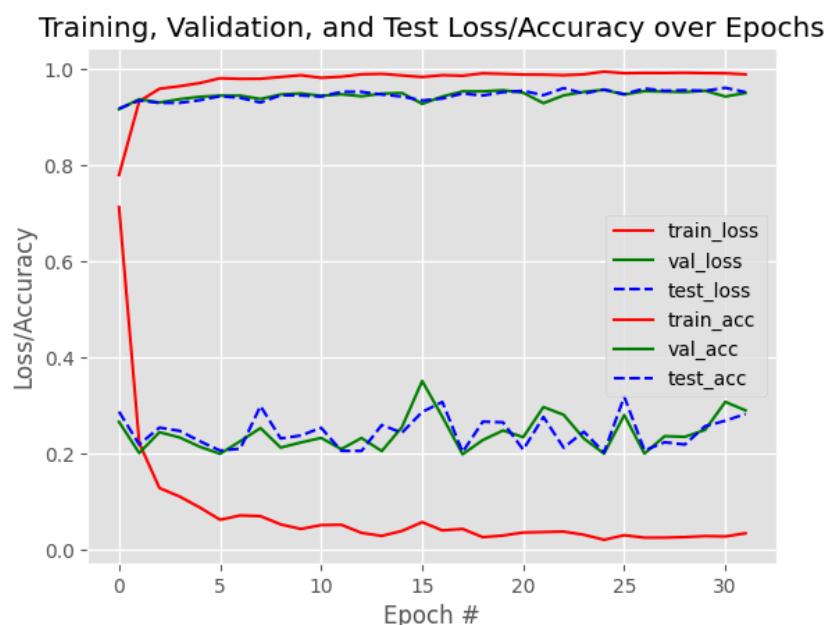
Con la finalidad de hacer pruebas con el modelo, se agregaron más clases de nuevos personajes al dataset, para ser un total de 15 personajes diferentes. Con este se busca observar si el modelo puede ser generalizable o si necesita de más ajustes. Para este nuevo entrenamiento se actualizó el dataset utilizado, agregando las 10 nuevas clases y actualizando las 5 existentes. Debido a que podría haber algún tipo de desviación si existían diferencias significativas entre clases, se optó por generar un dataset con 800 imágenes exactamente para cada clase. Estas imágenes fueron obtenidas con ayuda de un script de web scraping y una limpieza manual. A continuación se agrega una imagen de las clases que ahora considera el modelo con los nuevos personajes.



Para el entrenamiento del modelo se continúa usando la misma configuración de capas que anteriormente. A continuación se detallan las capas utilizadas.

Layer (type)	Output Shape	Param #
efficientnetb0 (Functional)	(None, 10, 5, 1280)	4,049,571
global_average_pooling2d_1 (GlobalAveragePooling2D)	(None, 1280)	0
dropout_2 (Dropout)	(None, 1280)	0
dense_2 (Dense)	(None, 128)	163,968
dropout_3 (Dropout)	(None, 128)	0
dense_3 (Dense)	(None, 15)	1,935

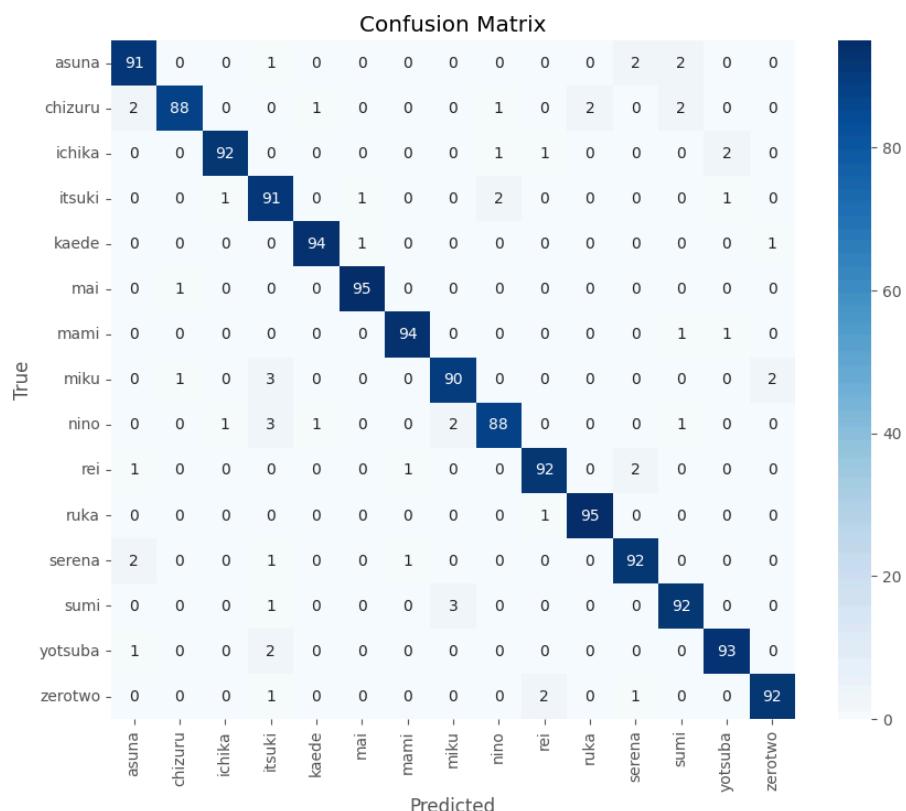
Como resultados en esta ocasión el modelo llegó a la época número 32 hasta que se detuviera por el early stopping configurado, lo cual se puede observar que tomó más épocas en entrenarse de manera correcta a comparación del primer modelo de 5 clases donde únicamente tomó 12 épocas. Esto se podría deber a la nueva cantidad de imágenes y clases agregadas al dataset. Aunque nuevamente se obtuvieron resultados satisfactorios con un valor de 99% en train, 95% en validation y 95% en test, por lo que se puede observar que el modelo continúa prediciendo de manera correcta la mayoría de veces las imágenes proporcionadas



Nuevamente se realizó una prueba con imágenes aleatorias del segmento de test de diferentes clases para realizar predicciones y observar gráficamente los resultados. Nuevamente se puede ver que el modelo predice correctamente la clase de las imágenes proporcionadas, respaldando el valor de precisión arrojado anteriormente



Analizando los resultados de la matriz de confusión se puede apreciar que el modelo cuenta con una buena precisión al clasificar las imágenes de los personajes con su clase correcta ya que se observa que la diagonal es la que más peso tiene e identificando que los errores con otras clases son mínimos, por lo que se aprecia que el modelo identifica correctamente todas las clases



Modelo alternativo - CNN

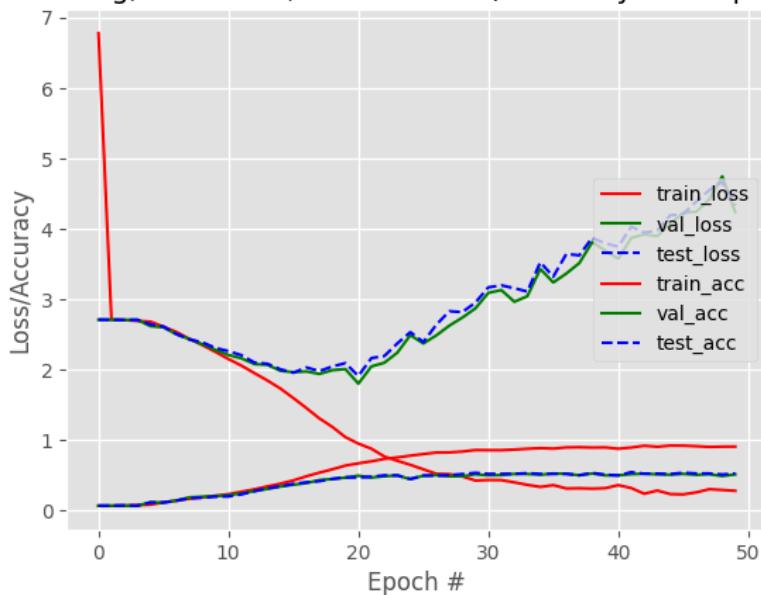
Además del modelo basado en EfficientNetB0 se desarrolló una alternativa utilizando una red neuronal convolucional (CNN) con el propósito de comparar el rendimiento entre el modelo generado antes y uno con una arquitectura diferente evaluandolos a través del valor de accuracy en los datos de test. Para este modelo de CNN se usan capas convolucionales con una función de activación relu que introduce no linealidad al modelo,además se aplica una operación de max pooling, también una capa de salida con 15 neuronas y una activación softmax genera las probabilidades de pertenencia a cada una de las clases permitiendo al modelo realizar predicciones multicategoría

Layer (type)	Output Shape	Param #
conv2d_6 (Conv2D)	(None, 98, 73, 10)	280
activation_6 (Activation)	(None, 98, 73, 10)	0
max_pooling2d_6 (MaxPooling2D)	(None, 49, 36, 10)	0
conv2d_7 (Conv2D)	(None, 47, 34, 10)	910
activation_7 (Activation)	(None, 47, 34, 10)	0
max_pooling2d_7 (MaxPooling2D)	(None, 23, 17, 10)	0
flatten_3 (Flatten)	(None, 3910)	0
dense_22 (Dense)	(None, 50)	195,550
dense_23 (Dense)	(None, 15)	765

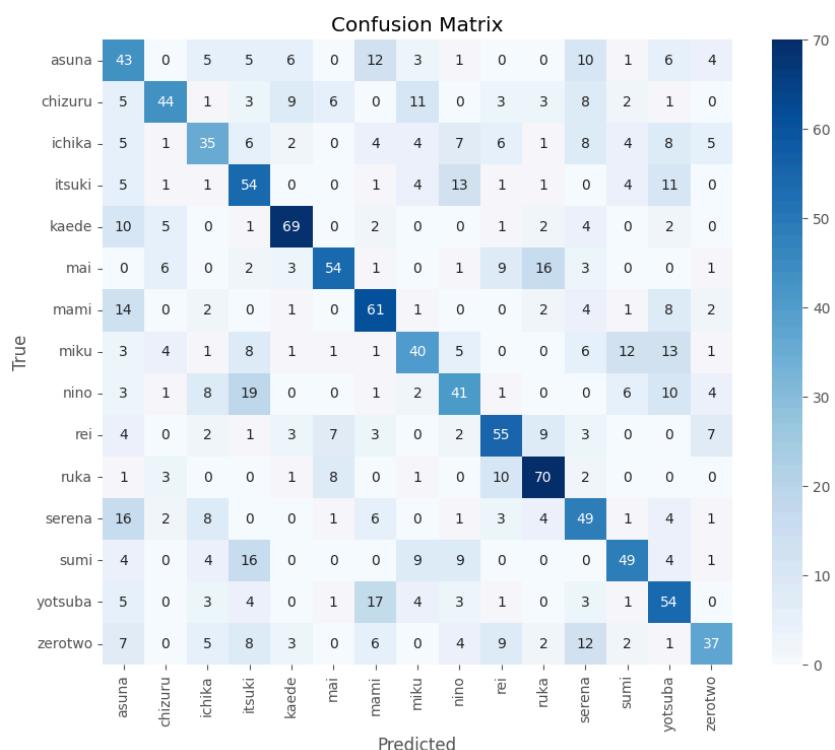
El modelo fue entrenado por 50 épocas, donde se noto un aprendizaje más lento comparado al modelo anterior, además de que se presenta un caso de overfitting ya que el modelo finaliza con un valor de precisión en train aproximado del 86% mientras que en el segmento de test tiene un valor del 52%, por lo que se observa un rendimiento inferior. Esto se puede observar en la siguiente gráfica, viendo la separación que existe entre el acc de test y train.

45/45 - 0s - 9ms/step - accuracy: 0.5243 - loss: 4.3328

Training, Validation, and Test Loss/Accuracy over Epochs



Para comprobar los datos arrojados por el modelo, se realizó una prueba en una matriz de confusión obteniendo los datos presentados a continuación. Se puede observar como aunque predomina la las predicciones correctas en la diagonal, existen varios casos donde se presenta predicciones erróneas entre clases de manera más recurrente



Mejora

Debido a que el modelo presenta una eficacia aún insatisfactoria, se realiza un ajuste en las capas y parámetros del modelo, para evitar principalmente el overfitting. Para ello se agregan dos capas intermedias de dropout para eliminar una parte del dataset ayudando a mejorar la capacidad para no obtener overfit. Además, se agrega el regularizador de L2 dentro de las capas densas para penalizar los valores grandes en los pesos ayudando a prevenir overfitting

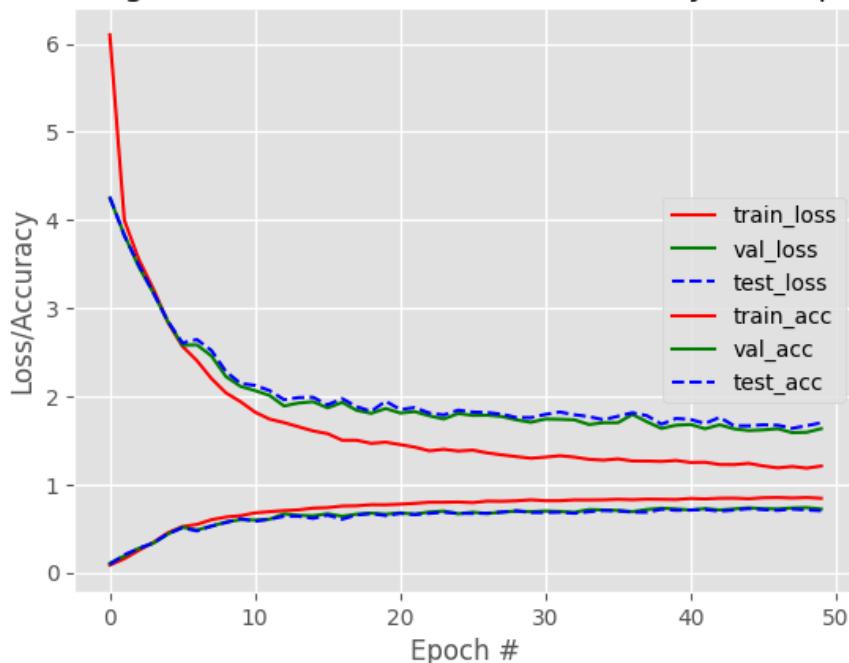
Layer (type)	Output Shape	Param #
conv2d_6 (Conv2D)	(None, 98, 73, 15)	420
activation_6 (Activation)	(None, 98, 73, 15)	0
max_pooling2d_6 (MaxPooling2D)	(None, 49, 36, 15)	0
dropout_6 (Dropout)	(None, 49, 36, 15)	0
conv2d_7 (Conv2D)	(None, 47, 34, 15)	2,040
activation_7 (Activation)	(None, 47, 34, 15)	0
max_pooling2d_7 (MaxPooling2D)	(None, 23, 17, 15)	0
dropout_7 (Dropout)	(None, 23, 17, 15)	0
flatten_3 (Flatten)	(None, 5865)	0
dense_12 (Dense)	(None, 50)	293,300
dense_13 (Dense)	(None, 50)	2,550
dense_14 (Dense)	(None, 50)	2,550
dense_15 (Dense)	(None, 15)	765

Así mismo este ajuste se volvió a entrenar por 50 nuevas épocas, donde se observó que el nivel de overfitting bajó considerablemente ayudando a un mejor valor de precisión en test, lo cual es bueno ya que se pudo reducir el overfitting y mejorar la precisión en test que es la métrica que se busca evaluar. Hablando sobre los valores de precisión arrojados por el modelo al finalizar el entrenamiento se tiene 86% en train, 70% en test y 72 en validation, aunque de nuevo se observa que tarda más en

el aprendizaje del modelo y aun teniendo resultados inferiores al primer acercamiento del proyecto

45/45 - 0s - 7ms/step - accuracy: 0.7028 - loss: 1.6998

Training, Validation, and Test Loss/Accuracy over Epochs



Finalmente se realiza una matriz de confusión para comprobar la manera en que el modelo está generando predicciones. Se observa una mejora considerable a la primer versión de este modelo (52% acc) ya que en esta resalta más la diagonal, representando una correcta predicción al mismo tiempo que las confusiones entre clases aunque son existentes están son menos significativas por lo tanto se puede decir que hubo una mejora en el rendimiento

