

## DEFINICIÓN DE LA ARQUITECTURA DE LA RED NEURONAL

### IDENTIFICACIÓN DE PERSONAS

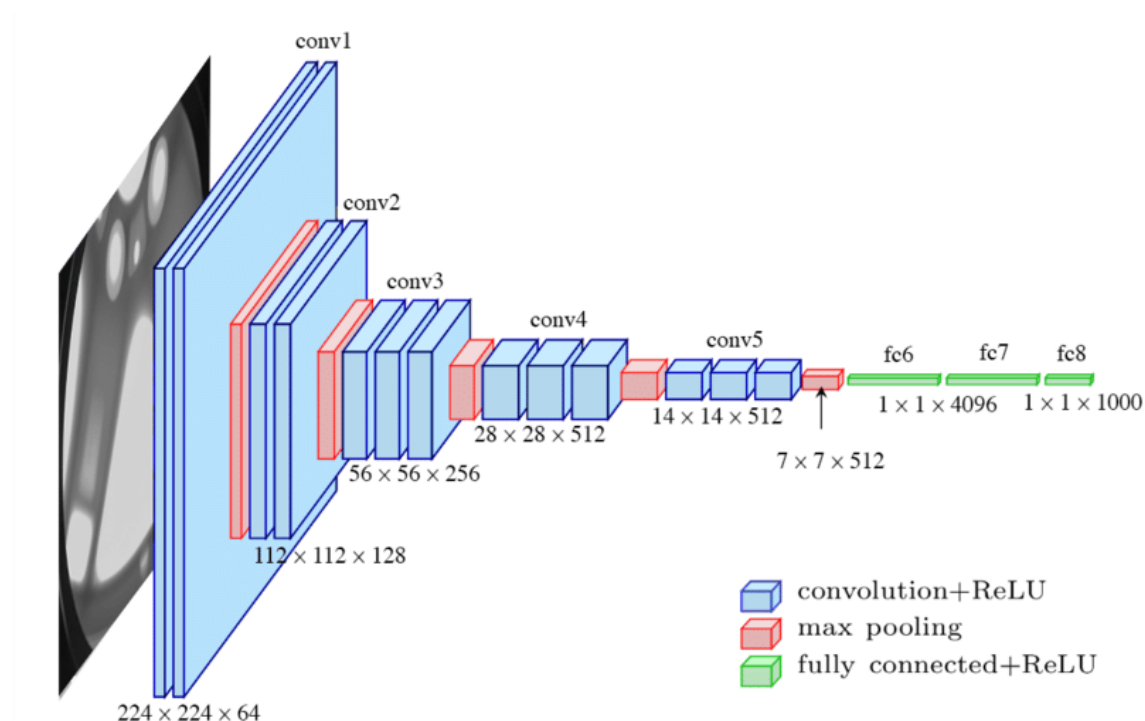
Se ha implementado una red neuronal convolucional (CNN) adaptada para la clasificación de imágenes con el objetivo de distinguir entre Nicolás Maduro, Hugo Chávez y ninguna de las dos figuras.

En ese contexto se plantea 2 modelo que nos puede ayudar a contrastar y demostrar ahorro computacional con la selección correcta del modelo.

### Modelo 1

#### Ejemplo de arquitectura CNN para la identificación de personas:

Se implementó una red neuronal convolucional (CNN) sencilla que recuerda a una versión reducida de VGG, a menudo referida como Tiny VGG.



Ejemplo de arquitectura VGG convolucional

**Componentes clave:**

- Capas Conv2D: Estas capas procesan las imágenes de entrada utilizando filtros convolucionales para detectar características locales en las imágenes.
- Capa MaxPooling2D: Esta capa reduce la dimensión espacial de la entrada, tomando el valor máximo de un conjunto de entradas.
- Capa Flatten: Aplana la entrada multidimensional a una dimensión única para prepararla para las capas densas.
- Capas Densamente Conectadas (Dense): Estas son capas estándar de perceptrones multicapa que procesan características y realizan la clasificación.

**Número de capas: 7 capas visibles.**

**Unidades por capa:**

- Capas Conv2D: 10 filtros.
- Capa Densa: 10 unidades.
- Capa Densa (Salida): 3 unidades.

**Funciones de activación:**

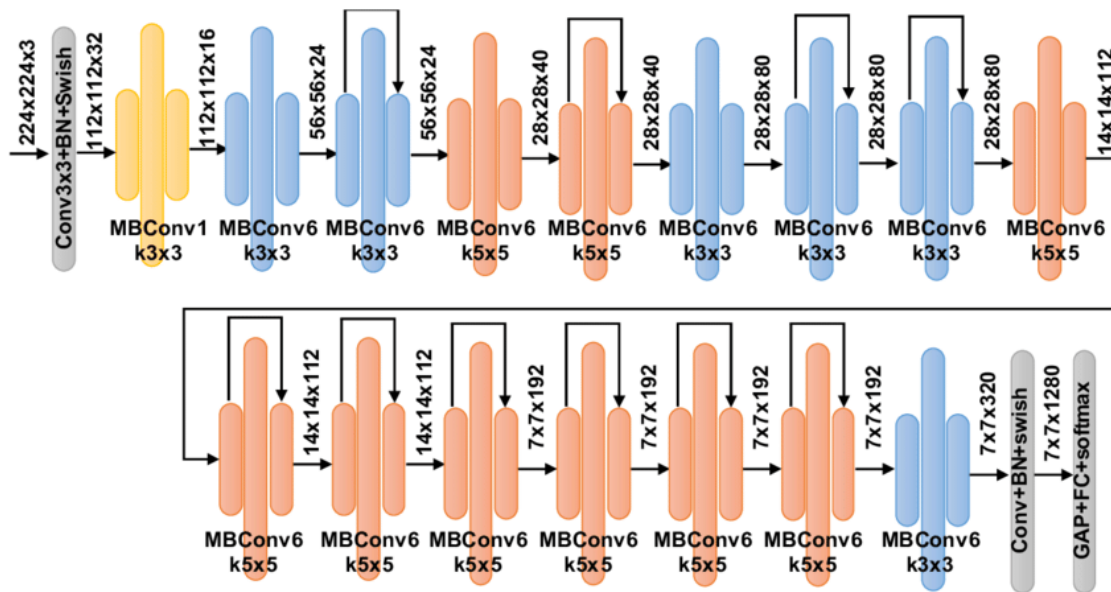
- Conv2D y Dense: 'relu'.
- Capa de salida: 'softmax'.

**Tipo de conectividad:** Convolucional y totalmente conectada.

Este modelo está diseñado para identificar si la persona en la imagen es Nicolás Maduro, Hugo Chávez o ninguna de las dos figuras, utilizando una arquitectura adaptada a la tarea específica de clasificación de individuos en imágenes.

## **Modelo 2**

Para la red se implementó la arquitectura EfficientNet B0 como base y agregando capas personalizadas para el problema específico de clasificación de tipos de rostro.



Ejemplo de arquitectura EfficientNet B0 general

#### Componentes clave:

- Capa de Extracción de Características (Feature Extractor): Implementa EfficientNet B0 y está preentrenada.
- Capa Densa (Salida): Una capa densa que realiza la clasificación final basándose en las características extraídas.

**Número de capas:** 2 capas visibles.

#### Unidades por capa:

- Capa Densa (Salida): 3 unidades.

#### Funciones de activación:

- Capa de salida: 'softmax'.

**Tipo de conectividad:** Convolutacional y totalmente conectada.

### MOTOR DE RECOMENDACIÓN

**Tipo de Arquitectura:** Se utiliza la técnica de transferencia de aprendizaje con EfficientNet, un modelo de red neuronal convolucional (CNN) preentrenado, como base para la extracción de características. Sobre esta base, se añaden capas personalizadas para adaptar el modelo al problema de clasificación de tipos de ROSTROS.

#### **Número de Capas:**

- Capa de extracción de características (proveniente de EfficientNet): No se especifica el número exacto de capas dentro de EfficientNet en el cuaderno, pero EfficientNet generalmente consta de varias capas convolucionales, de agrupación y otras capas auxiliares.
- Capa densa de salida.

#### **Unidades por Capa:**

- Capa de extracción de características: Al ser un modelo preentrenado, EfficientNet tiene un número específico de unidades en sus capas, pero esto no se especifica en el cuaderno.
- Capa densa de salida: 3 unidades (correspondientes a las 3 clases de tipos de rostros).

#### **Funciones de Activación:**

- Capa de extracción de características: No se especifica en el cuaderno, pero EfficientNet generalmente utiliza funciones de activación como ReLU o variantes de ReLU en sus capas internas.
- Capa densa de salida: softmax.

#### **Tipo de Conectividad:**

- Capa de extracción de características: Conectividad local en capas convolucionales, típica de cualquier CNN.
- Capa densa de salida: Totalmente conectada.

Adicional se mejora el modelo con una malla de hiperparámetros

El modelo utiliza la arquitectura EfficientNetB0 preentrenada en ImageNet como base, seguida de una capa de GlobalAveragePooling2D. Luego, agrega una serie de capas densas y de dropout, cuyo número y tamaño dependen de los hiperparámetros. Finalmente, agrega una capa densa de salida con 3 unidades

y activación 'softmax'. El modelo se compila con el optimizador Adam, cuya tasa de aprendizaje también es un hiperparámetro, y la pérdida de "categorical\_crossentropy"

## Mettricas de los modelo

### Modelo 1

Pérdida en datos de prueba: 0.4529256224632263  
Precisión en datos de prueba: 0.9266666769981384

### Modelo 2

Pérdida en datos de prueba: 0.4529256224632263  
Precisión en datos de prueba: 0.9266666769981384

Se selecciona el modelo 1, tiene menor costo computacional, fue el modelo más simple propuesto y además se obtiene mejores métricas.

### a) ¿Qué modelo eligió para la clasificación y por qué?

Se eligió una red neuronal convolucional (CNN) adaptada para la clasificación de imágenes. Las CNN son especialmente buenas para procesar imágenes debido a su capacidad para detectar características locales y espaciales en las imágenes a través de filtros convolucionales. En este caso, se ha adaptado una versión reducida de la arquitectura VGG, a menudo referida como Tiny VGG, que es eficiente y suficientemente potente para tareas de clasificación de imágenes de este tipo.

### b) ¿Qué parámetros tiene configurado el modelo de clasificación y por qué?

El modelo tiene varias capas Conv2D con 10 filtros cada una, que son las encargadas de detectar las características locales en las imágenes. Luego, se utiliza una capa MaxPooling2D para reducir la dimensión espacial de la entrada. Después de aplanar la entrada con una capa Flatten, se utilizan capas densas para procesar las características y realizar la clasificación. La última capa densa tiene 3 unidades, correspondientes a las tres clases de salida: Nicolás Maduro, Hugo Chávez y ninguna de las dos figuras. Las funciones de activación 'relu' y 'softmax' se utilizan en las capas Conv2D/Dense y de salida, respectivamente.

### c) ¿Son suficientes la cantidad de imágenes para el modelado?

Con 600 imágenes para entrenamiento y 150 para prueba, el conjunto de datos parece ser relativamente pequeño, pero la precisión obtenida en los datos de prueba es bastante alta (99.33%). Esto sugiere que el modelo está funcionando bien con la cantidad de datos disponibles. Sin embargo, tener más datos generalmente mejora el rendimiento del modelo

**d) ¿Qué sucedería si tengo más de una persona en una imagen, funcionaría el modelo?**

En la prueba que hice el modelo actual clasifica imágenes que contienen a más de una persona. Pero pierde efectividad, además no es el foco del entrenamiento, trabajar con múltiples personas, si hay más de una persona en la imagen, el modelo podría no funcionar correctamente, a pesar que en la prueba logro hacerlo sin problema, Para manejar imágenes con múltiples personas de una manera más optima, se podría considerar el uso de un enfoque de detección de objetos, que primero identifica y recorta cada persona en la imagen, y luego utiliza el modelo de clasificación para identificar a cada persona individualmente.

**e) Distorsione la imagen con un mecanismo de procesamiento de imágenes e intente predecir la clasificación de la misma, ¿funciona el modelo?**

Dependerá del tipo y grado de distorsión aplicada a la imagen. Pequeñas distorsiones, como rotaciones o cambios de escala, podrían no afectar significativamente el rendimiento del modelo,

Con la prueba que hice, si logro clasificar correctamente, pero como mencione depende del grado de distorsión y del tipo. Cuando exista distorsiones más grandes o no naturales podrían hacer que el modelo falle al clasificar la imagen. En este caso, se podría considerar el uso de técnicas de preprocesamiento de imágenes para corregir la distorsión antes de pasar la imagen al modelo.

## **Backend**

## **Aplicación en Python ( fastapi)**

Link

<https://web-production-7703.up.railway.app/>

<https://nolatechclasificacion.tuprofeestadistica.com/>

## Página de inicio

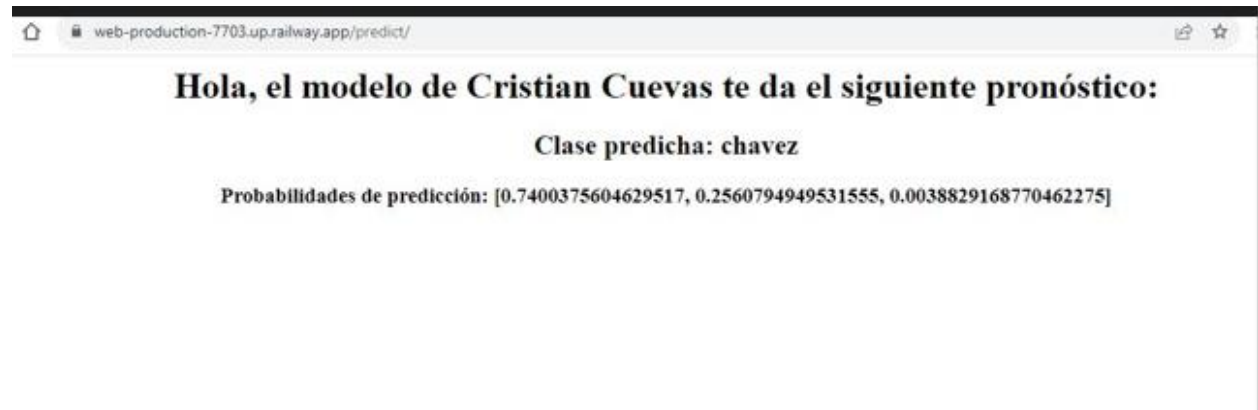


Subir imagen

Elegir archivo Browse

Subir

## Resultado



Hola, el modelo de Cristian Cuevas te da el siguiente pronóstico:

Clase predicha: chavez

Probabilidades de predicción: [0.7400375604629517, 0.2560794949531555, 0.0038829168770462275]