

SAMSUNG

Iris Eye

Iris Eye Team

AI Course

Samsung Innovation Campus

Together for Tomorrow!
Enabling People

Education for Future Generations



IrisEye

IRIS EYE

UNIT 1. Objetivo y contexto

- 1.1. Objetivo
- 1.2. Contexto
- 1.3. Proyecto

UNIT 2. Equipo

- 2.1. Coordinación y miembros
- 2.2. Organización y herramientas de colaboración

UNIT 3. Planteamiento desarrollo

- 3.1. Modelos y método de entrenamiento
- 3.2. Procesado de datos
- 3.3. Temporalización
- 3.4. Resultados

IRIS EYE

UNIT 4. Resultados primera iteración (prototipado)

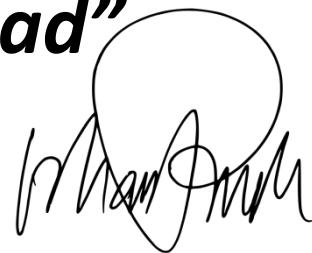
- 4.1. Estado actual planning desarrollo
- 4.2. Herramientas de coordinación y seguimiento
- 4.3. Plataforma de entrenamiento y random state y validación
- 4.4. Resultados modelos Machine Learning
- 4.5. Resultados modelos Deep Learning
- 4.6. Modelo seleccionado para desarrollo
- 4.7. Mejoras dataset original
- 4.8. Implementación web

IRIS EYE

UNIT 5. Modelado Final

- 5.1. Estado actual planning desarrollo
- 5.2. Herramientas y plataformas de desarrollo utilizadas
- 5.3. Datasets ampliación y preprocesamiento
- 5.4. Flujo de trabajo
- 5.5. Entrenamiento y optimización de hiperparámetros
- 5.6. Arquitectura modelo final
- 5.7. Resultados entrenamiento modelo final
- 5.8 Resultados test modelo final
- 5.9 Interface de usuario (interface)
- 5.10 Propuestas de mejora
- 5.11. Logros y beneficios
- 5.12 Conclusiones finales
- 5.13 Miembros del equipo

*“El color es el tacto del ojo,
la música de los sordos, una
palabra en la oscuridad”*



Orhan Pamuk
premio nobel literatura



UNIT 1. OBJETIVO Y CONTEXTO

1.1. Objetivo

OBJETIVO

El objetivo de nuestro proyecto es crear un **modelo basado en la inteligencia artificial** que ayude a las personas con dificultades visuales como el daltonismo o el monocromatismo a **identificar los colores**.



1.2. Contexto

EL COLOR GOBIERNA NUESTRA VIDA

Pensemos cómo sería orientarnos en el metro, reconocer un semáforo o simplemente elegir una camisa si no pudiésemos identificar el color?

El **90%** de las personas daltónicas necesitan **ayuda para comprar** ropa u otros objetos de color y el **41%** reconocen que **no se sienten integrados** socialmente.

Esta alteración de la visión afecta a **uno de cada diez hombres** y **una de cada 200 mujeres**. Hasta un 17% de los niños no descubren que tienen esta limitación hasta los 20 años.



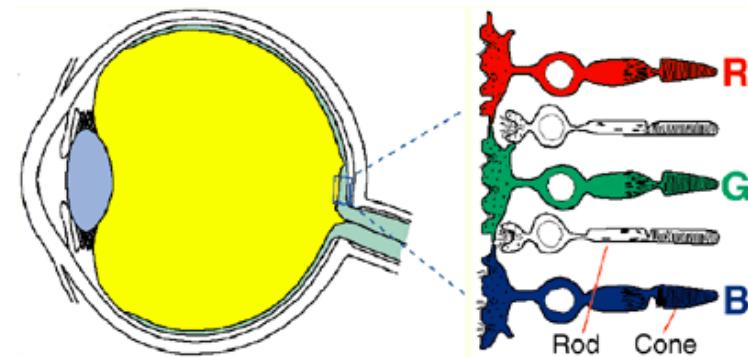
1.2. Contexto

I ¿QUÉ ES EL DALTONISMO?

El daltonismo es una alteración de **origen genético** que afecta a la capacidad de distinguir los colores.

El ojo humano tiene **tres tipos de células cónicas**. Estas células expresan diferentes tipos de genes de opsina, que son sensibles principalmente al **rojo, verde y azul**, respectivamente.

El daltonismo es la situación en la que la función de una de estas opsinas se pierde o se ve alterada.



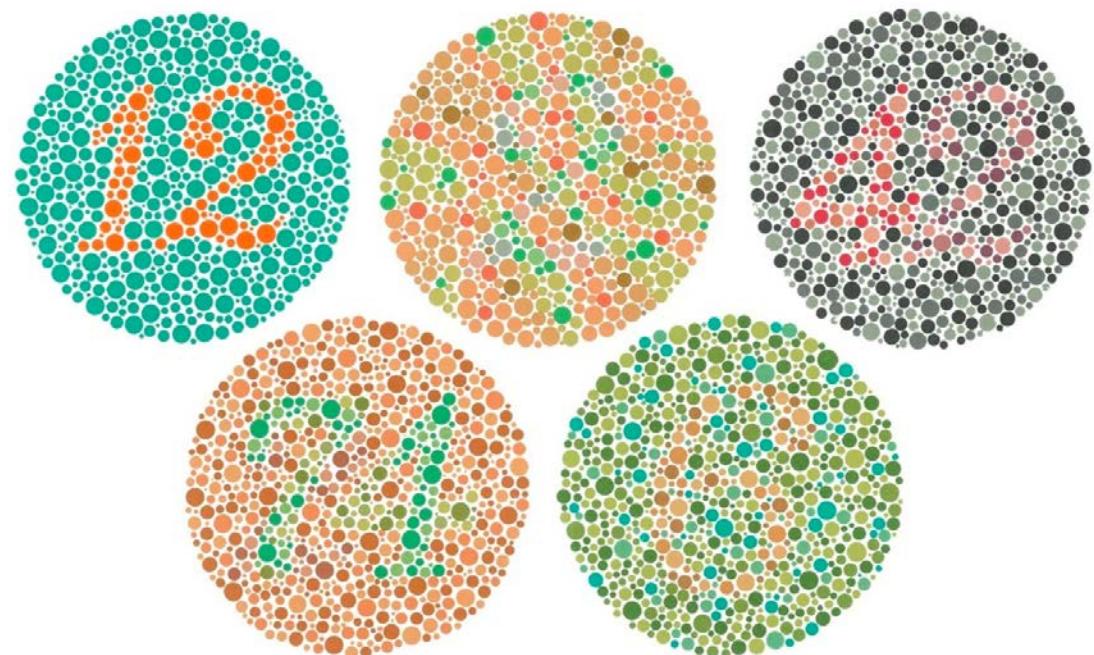
1.2. Contexto

■ ¿CÓMO SE DETECTA EL DALTONISMO?

Habitualmente el daltonismo se detecta en las revisiones médicas mediante pruebas específicas como el test de Ishihara.

Sin embargo, hay muchas personas que llegan a adultos sin saber que padecen este trastorno visual.

¿Puedes reconocer todos los números que parecen en la siguiente imagen?



1.2. Contexto

TIPOS DE DALTONISMO

Aunque existen muchos tipos de daltonismo, el **99 %** de los casos corresponden a **deuteranopia y protanopia**.

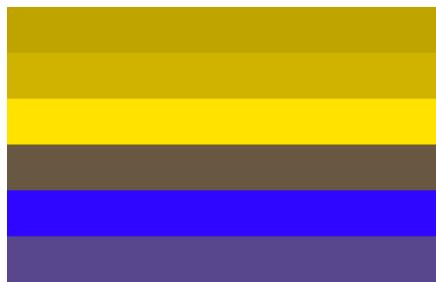
Las personas con el gen de la opsina roja mutante se llaman **protanope** y el gen de la opsina verde, **deuteranope**. Las personas con el problema en el gen de la opsina azul se llaman **tritanope**.

A la derecha podemos observar como perciben los colores las personas con estos tipos de alteración visual.

visión normal colores



protanopia



deuteranopia



tritanopia

1.3. Proyecto

PROYECTO

El proyecto se basa en entrenar un modelo para reconocer los colores con la mayor precisión y exactitud posible y utilizarlo a través de una aplicación web, de modo que al enfocar la cámara del móvil en un color, pueda identificarlo y clasificarlo en diferentes categorías predefinidas.



UNIT 2. EQUIPO

2.1. Coordinación y miembros

EQUIPO

Coordinadora

Lorena Jiménez Tejada

Miembros

Marina Jiménez Egea

Marta Freire Painceira

Marta Trasancos Yáñez

Mercedes Rodríguez Barbero



UNIT 2. EQUIPO

2.2. Organización y herramientas de colaboración

ROLES de EQUIPO

Organización

Hasta la primera iteración todo el equipo realiza pruebas de modelos y colabora en la elaboración de la documentación preliminar.

En una segunda fase nos organizaremos para que una parte del equipo optimice los modelos seleccionados y el resto implemente una aplicación web.

Herramientas de colaboración y asignación

Utilizamos la herramienta colaborativa

Trello para definir las tareas a realizar y llevar el seguimiento del estado de las mismas.



UNIT 3. PLANTEAMIENTO DESARROLLO

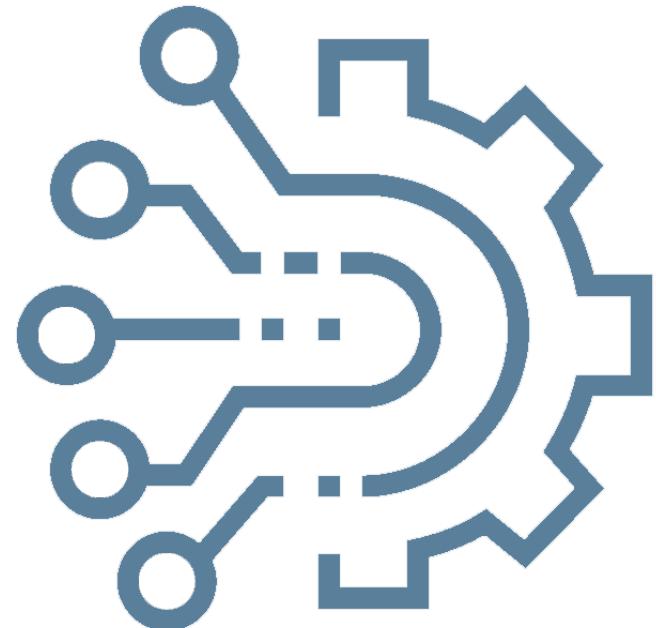
3.1. Modelos y métodos de entrenamiento

MACHINE LEARNING y DEEP LEARNING

Utilizaremos técnicas de machine learning y deep learning para comprobar cuál puede ser el acercamiento más efectivo a nuestro proyecto.

Dentro de machine learning probaremos KNN, Naive Bayes, Tree, Random Tree y SVC. Usaremos GridSearch para optimizar los hiperparámetros, validación cruzada para evaluar los resultados y ensemble para enlazar diferentes modelos.

En deep learning usaremos redes neuronales densas y convolucionales.



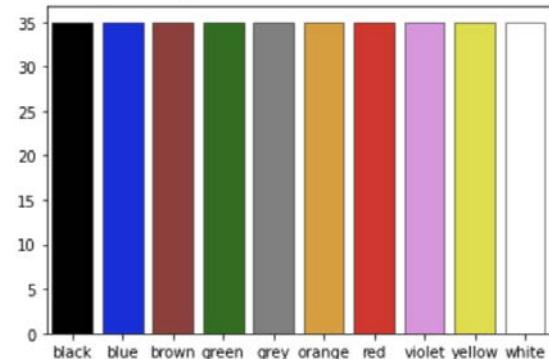
UNIT 3. PLANTEAMIENTO DESARROLLO

3.2. Procesado de datos

| SELECCIÓN Y PREPROCESADO DATASET

Nos hemos basado en dos conjuntos de datos obtenidos en Kaggle y, a partir de ellos, hemos generado nuestro propio dataset balanceando los datos y añadiendo nuevas imágenes.

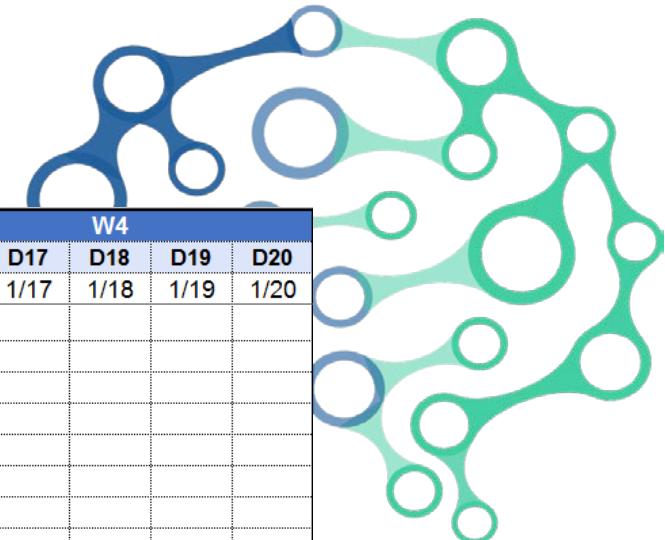
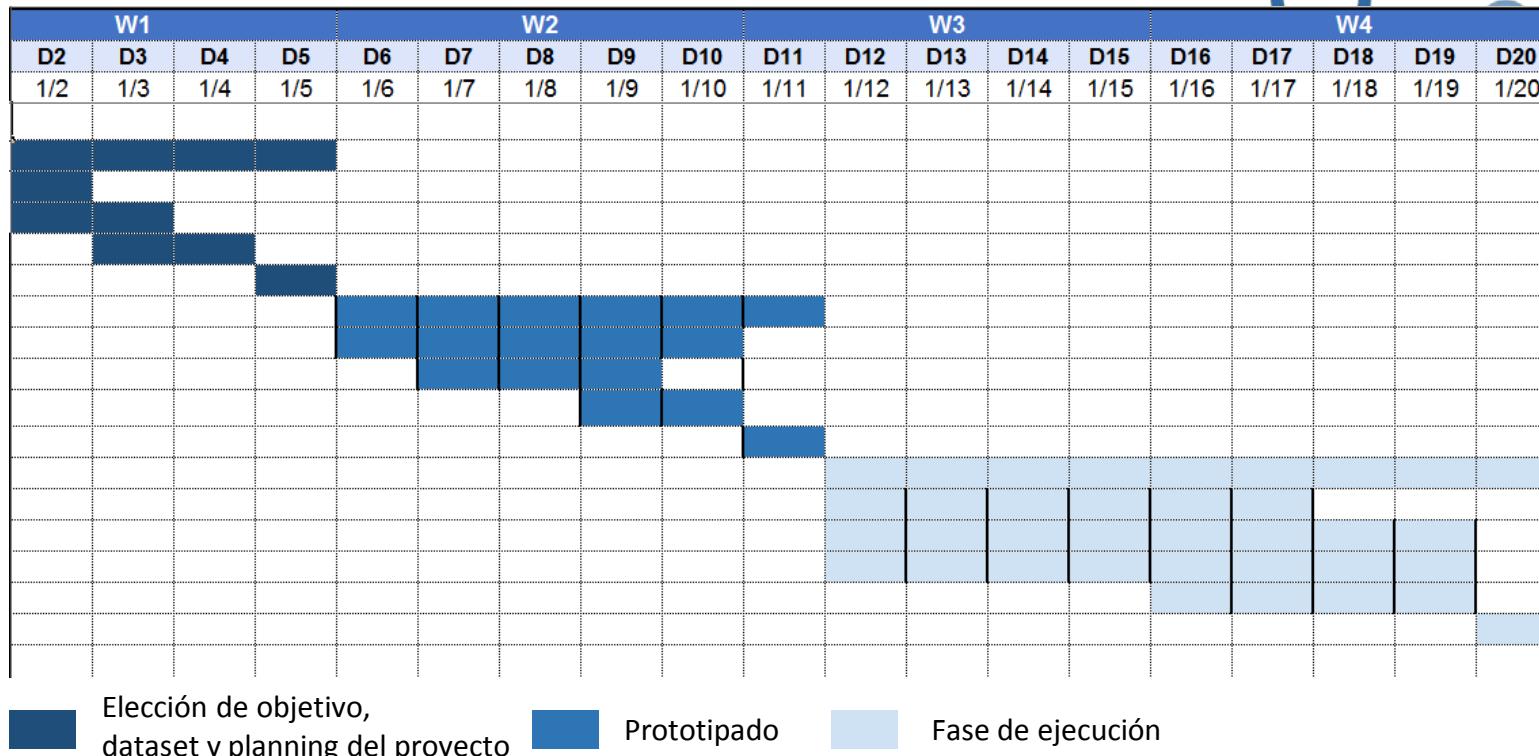
Están organizados en 10 clases diferentes y valoraremos la posibilidad de ampliar el número de imágenes de cada una de ellas si fuese necesario para mejorar el rendimiento del modelo.



UNIT 3. PLANTEAMIENTO DESARROLLO

3.2. Temporalización

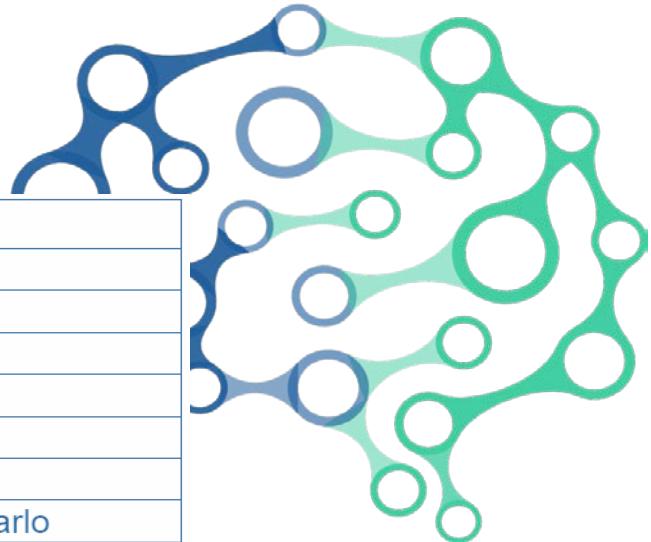
CAPSTONE BREAKDOWN STRUCTURE



3.2. Temporalización

Capstone Breakdown Structure

1. Proyecto		
1.1. Elección del objetivo del proyecto		
Clase preparatoria para la realización del proyecto		
búsqueda de datasets relacionados		
realización del planning del proyecto		
presentación preliminar		
1.2 Prototipando		
realizar pruebas para la mejora del modelo y testearlo		
investigación sobre la implementación web		
preparación del powerpoint para mostrar los avances		
presentación de la primera iteración		
1.3 Fase de ejecución		
mejoras del modelo		
aumento de los datos		
desarrollo la implementación web		
preparación del proyecto		
presentación del proyecto		



3.4. Resultados

RESULTADO ESPERADO

El objetivo es mejorar el rendimiento del modelo e implementar una web que pueda ser utilizada con la cámara de un dispositivo móvil, de modo que cuando una persona con dificultades visuales para reconocer los colores necesite distinguir uno pueda utilizarla.



18

SAMSUNG

resultados

PRIMERA ITERACIÓN

Iris Eye Team

AI Course

Samsung Innovation Campus

Together for Tomorrow!
Enabling People

Education for Future Generations

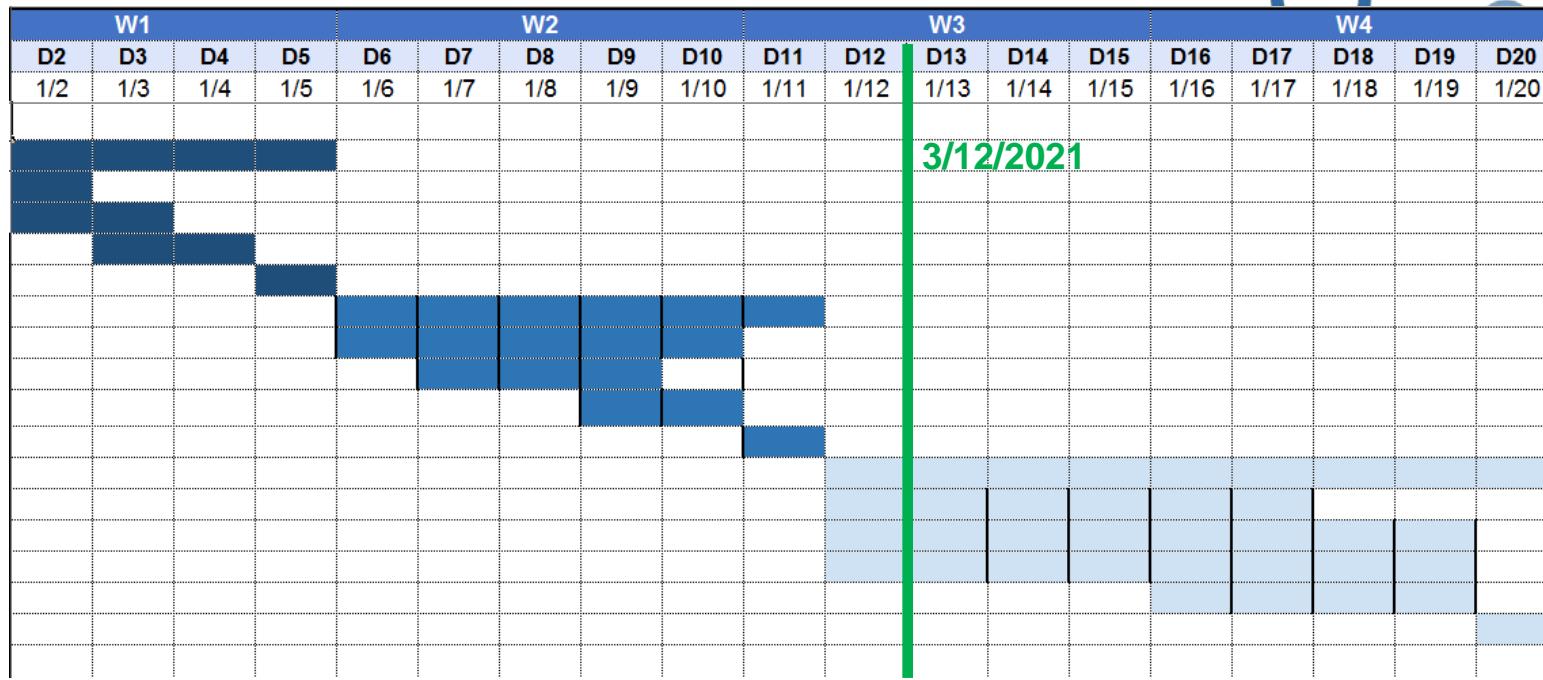


IRIS EYE

UNIT 4. Resultados primera iteración (prototipado)

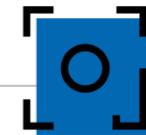
- 4.1. Estado actual planning desarrollo
- 4.2. Herramientas de coordinación y seguimiento
- 4.3. Plataforma de entrenamiento y random state y validación
- 4.4. Resultados modelos Machine Learning
- 4.5. Resultados modelos Deep Learning
- 4.6. Modelo seleccionado para desarrollo
- 4.7. Mejoras dataset original
- 4.8. Implementación web

CAPSTONE BREAKDOWN STRUCTURE



PROTOTIPO (REALIZADO)

F. EJECUCIÓN (en DESARROLLO)

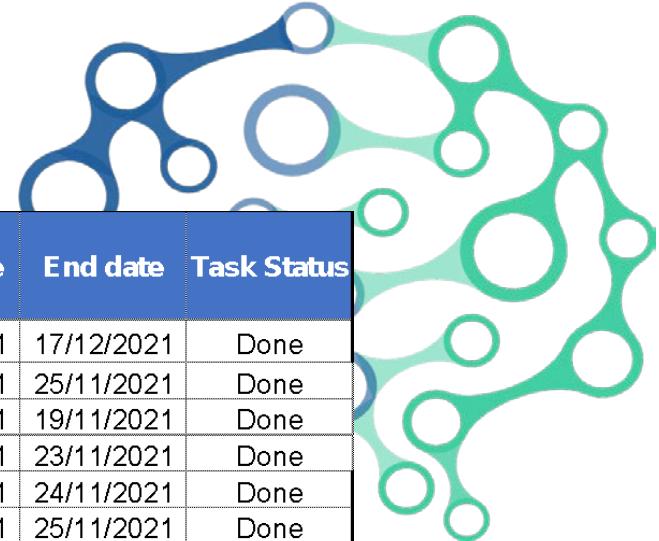


UNIT 4. RESULTADOS PRIMERA ITERACIÓN (PROTOTIPADO)

4.1. Estado actual planning desarrollo

Capstone Breakdown Structure

Task		Start date	End date	Task Status
1. Proyecto		19/11/2021	17/12/2021	Done
1.1. Elección del objetivo del proyecto		19/11/2021	25/11/2021	Done
	Clase preparatoria para la realización del proyecto	19/11/2021	19/11/2021	Done
	búsqueda de datasets relacionados	19/11/2021	23/11/2021	Done
	realización del planning del proyecto	23/11/2021	24/11/2021	Done
	presentación preliminar	25/11/2021	25/11/2021	Done
1.2 Prototipando		26/11/2021	2/12/2021	Done
	realizar pruebas para la mejora del modelo y testearlo	26/12/2021	1/12/2021	Done
	investigación sobre la implementación web	29/11/2021	1/12/2021	Done
	preparación del powerpoint para mostrar los avances	1/12/2021	2/12/2021	Done
	presentación de la primera iteración	3/12/2021	3/12/2021	Done
1.3 Fase de ejecución		6/12/2021	17/12/2021	In-progress
	aumento de los datos	6/12/2021	13/12/2021	Done
	mejoras en el modelo	6/12/2021	7/12/2021	In-progress
	desarrollo la implementación web	6/12/2021	16/12/2021	In-progress
	preparación del proyecto	10/12/2021	16/12/2021	In-progress
	presentación del proyecto	17/12/2021	17/12/2021	In-progress



UNIT 4. RESULTADOS PRIMERA ITERACIÓN (PROTOTIPADO)

4.2. Herramientas de coordinación y seguimiento

Lista de tareas

- Mejorar el modelo
- Elaboración informe
- Diapositivas
- Excel planificación tareas

En proceso

- Probar notebook con Dataset de 10 categorías(el de menor peso)
Accuracy=0.968 Fecha=23/11/2021
- redacción del diario del proyecto
- MODELOS A PROBAR
- convolutional neural network (706 - 707) acc_0.932
- investigar cómo desplegar el modelo en una web para hacerlo útil

En Revisión

- preparar presentacion 03/12

Hecho

- Limpiar dataset, quitar atributos
- Balancear imagenes dataset
- Probar notebook con dataset nuevo
- Averiguar cómo obtener la tabla de resultados del accuracy con nombres de los colores blue, white,.....en lugar del número de etiqueta (0,1,2,...)
- Buscando datasets para el proyecto
- Bagging ensemble: Classification by

ENLACES

- Drive
- Modelo base desde el que trabajar

La herramienta Trello ha resultado muy eficiente a la hora de asignar las diferentes tareas y poder realizar un seguimiento de las mismas.

4.3. Plataforma de entrenamiento, random state y validación

Plataforma de entrenamiento

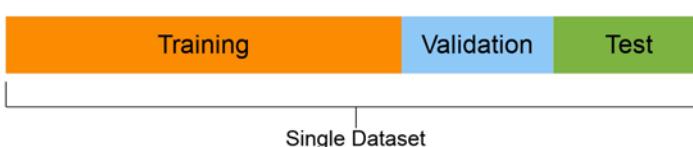


Se utiliza la plataforma google colaboratory para ejecutar el desarrollo y entrenamiento de los modelos. En el entrenamiento de redes CNN se reduce de forma importante el tiempo al poder disponer de un entorno de ejecución GPU.

Random State y validación



Se establece una semilla aleatoria en la partición del dataset en train y test para que selección aleatoria de muestras no impida la comparación de los resultados de los diferentes modelos y exista menos variabilidad a la hora reproducir resultados.

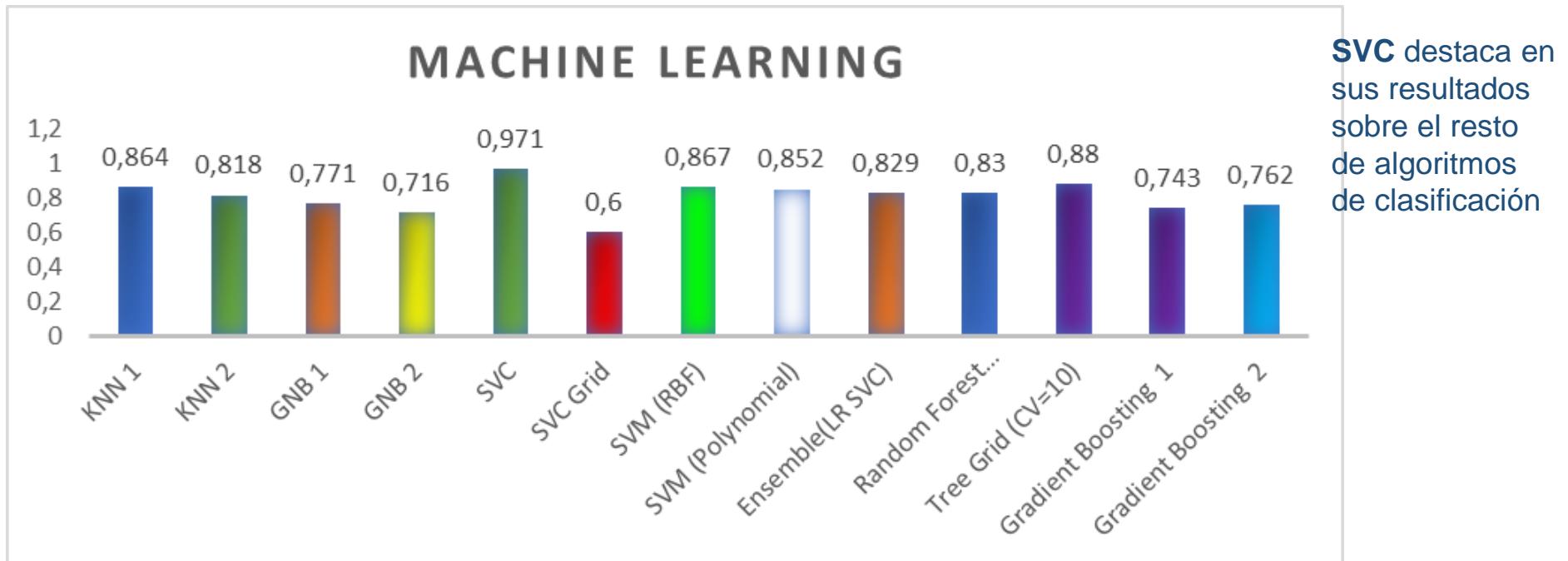


Se establece además un porcentaje de datos de validación en fase de entrenamiento.

UNIT 4. RESULTADOS PRIMERA ITERACIÓN (PROTOTIPADO)

4.4. Resultados modelos Machine Learning

- Accuracy alcanzado con modelos entrenados



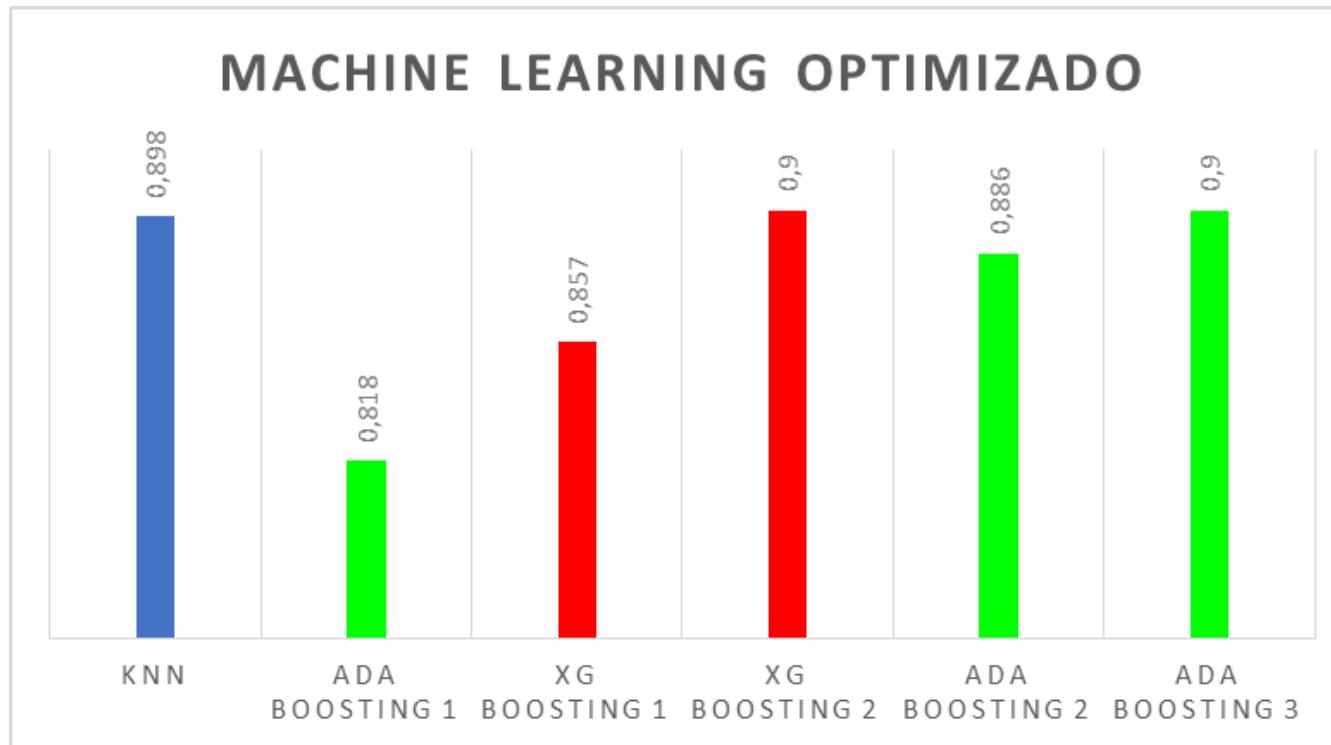
Optimización de hiperparámetros: NO

Dataset utilizado: Dataset original (35 img /clase, predominancia tonos color)

UNIT 4. RESULTADOS PRIMERA ITERACIÓN (PROTOTIPADO)

4.4. Resultados modelos Machine Learning

Accuracy alcanzado con modelos entrenados



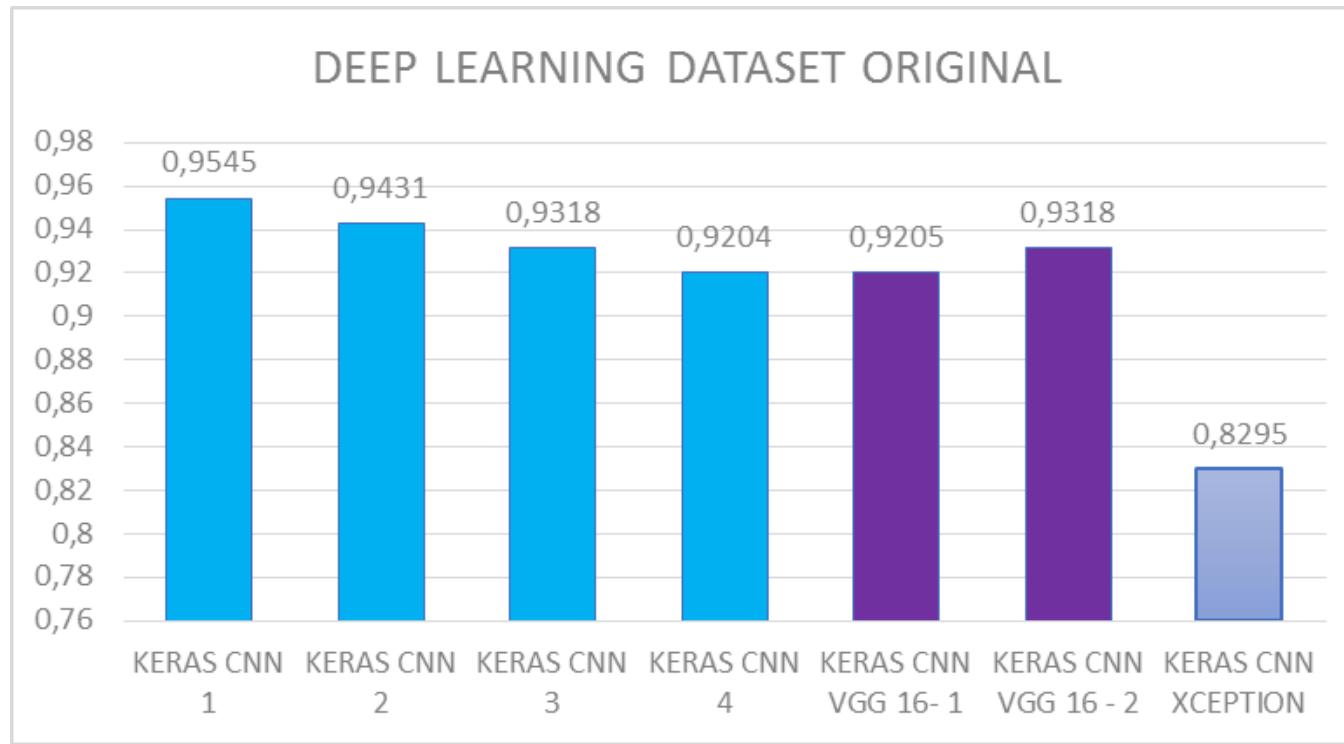
Optimizando hiperparámetros no se logra superar el resultado por **SVC**

Optimización de hiperparámetros: SI (GridSearchCV)
Dataset utilizado: Dataset original (35 img /clase, predominancia tonos color)

UNIT 4. RESULTADOS PRIMERA ITERACIÓN (PROTOTIPADO)

4.5. Resultados modelos Deep Learning

Accuracy alcanzado con modelos entrenados



Optimización de hiperparámetros: NO

Dataset utilizado: Dataset original ((35 img /clase, predominancia tonos color)

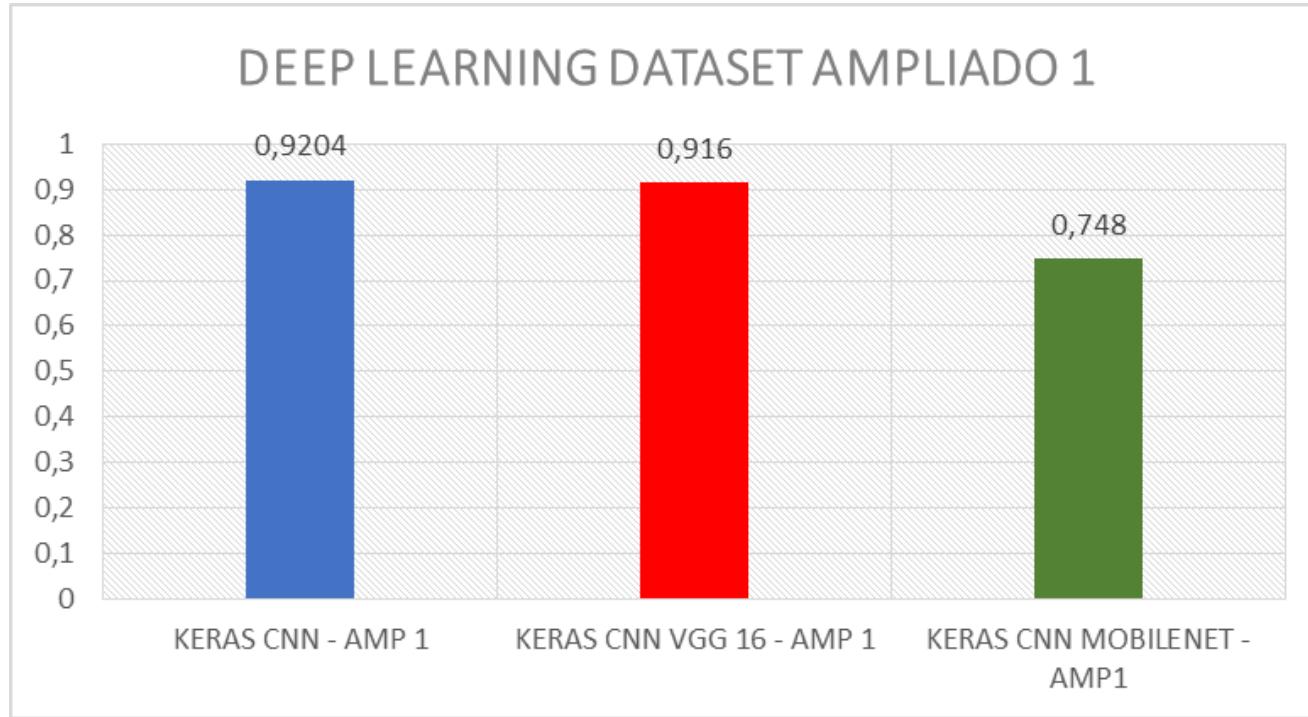
Las distintas redes neuronales convolucionales mejoran significativamente los resultados de machine learning.

Las redes ready-made **VGG16** y **Xception** ofrecen mejores resultados prescindiendo del fine-tuning, es decir, entrenando los pesos iniciales debido a la gran diferencia de tipología y tamaño entre nuestro dataset y el dataset original con el que han sido preentrenadas.

UNIT 4. RESULTADOS PRIMERA ITERACIÓN (PROTOTIPADO)

4.5. Resultados modelos Deep Learning

Accuracy alcanzado con modelos entrenados



Optimización de hiperparámetros: NO

Dataset utilizado: Dataset AMP1 (100 img /clase, predominancia objeto sobre fondo)

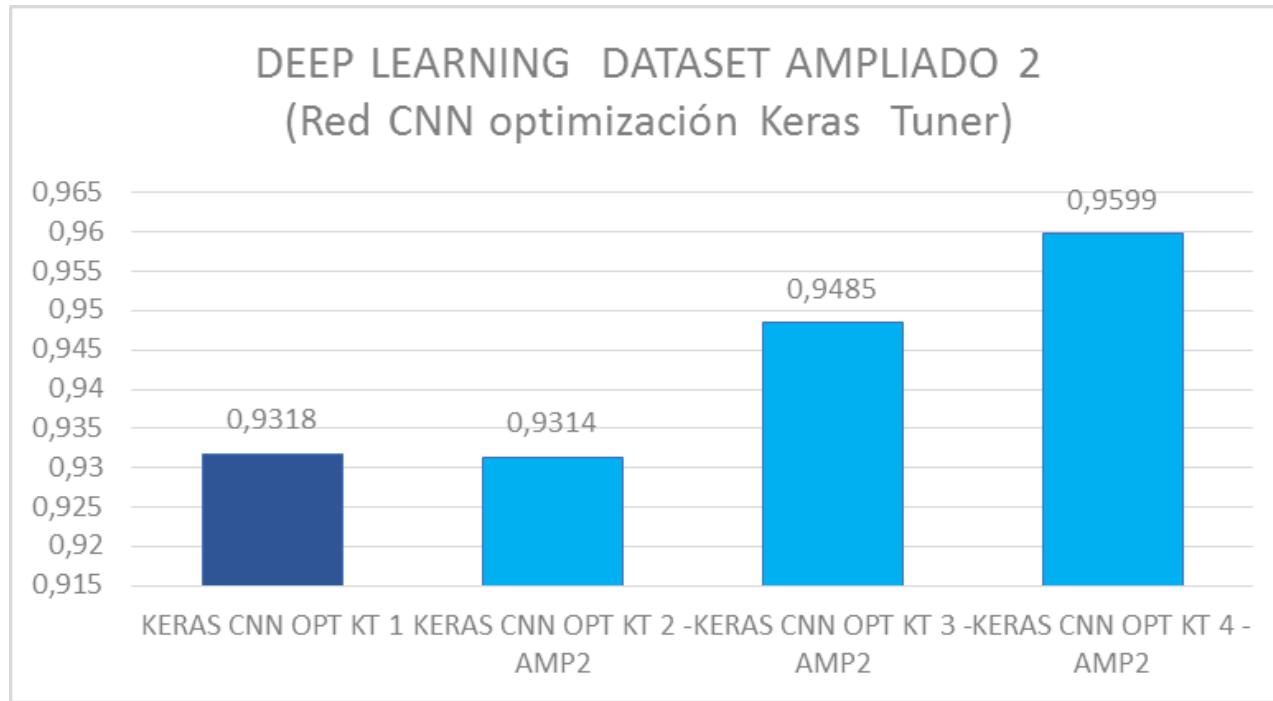
Ampliamos nuestro dataset original, estableciendo 100 imágenes por categoría (AMP1) y entrenando redes **VGG16** y **MobileNet**.

En las nuevas imágenes incorporadas predominan los **objetos del color de la categoría sobre un fondo**. En general se obtienen **peores resultados** ya que habría que implementar segmentación por áreas de reconocimiento para no introducir distorsiones en los cálculos de la red neuronal.

UNIT 4. RESULTADOS PRIMERA ITERACIÓN (PROTOTIPADO)

4.5. Resultados modelos Deep Learning

Accuracy alcanzado con modelos entrenados



Optimización de hiperparámetros: SI (Keras Tuner)

Dataset utilizados:

- CNN OPT KT 1: dataset original (35 img /clase, predominancia tonos color)
- Resto: dataset AMP2 (70 / clase con predominancia tonos color)

Ampliamos nuestro dataset original estableciendo 70 imágenes por categoría (AMP2), con imágenes de **tonos del color de la categoría** (sin objeto sobre fondo)

Como nuestro objetivo es reconocer un único color al enfocarlo con la cámara mejoran los resultados con este tipo de ampliación del dataset.

Se introduce optimización de hiperparámetros del modelo mediante Keras Tuner.

UNIT 4. RESULTADOS PRIMERA ITERACIÓN (PROTOTIPADO)

4.6. Modelo seleccionado para desarrollo

RED KERAS CNN OPTIMIZADA (Acc = 0.9599)

Layer (type)	Output Shape	Param #
conv2d_2 (Conv2D)	(None, 30, 30, 48)	1344
max_pooling2d_2 (MaxPooling 2D)	(None, 15, 15, 48)	0
conv2d_3 (Conv2D)	(None, 13, 13, 48)	20784
max_pooling2d_3 (MaxPooling 2D)	(None, 6, 6, 48)	0
flatten_1 (Flatten)	(None, 1728)	0
dense_2 (Dense)	(None, 224)	387296
dropout_1 (Dropout)	(None, 224)	0
dense_3 (Dense)	(None, 10)	2250
<hr/>		
Total params:	411,674	
Trainable params:	411,674	
Non-trainable params:	0	

Optimización hiperparámetros: Kera Tuner (Hyperband algorithm)

Dataset utilizados: AMP2 (70 img por clase, predominancia tonos color)

Tamaño entrada imágenes = 32x32 px.(resultados óptimos reduciendo peso computación)

Seleccionamos la red keras convolucional optimizada por sus esperanzadores resultados y potencial de desarrollo futuro.

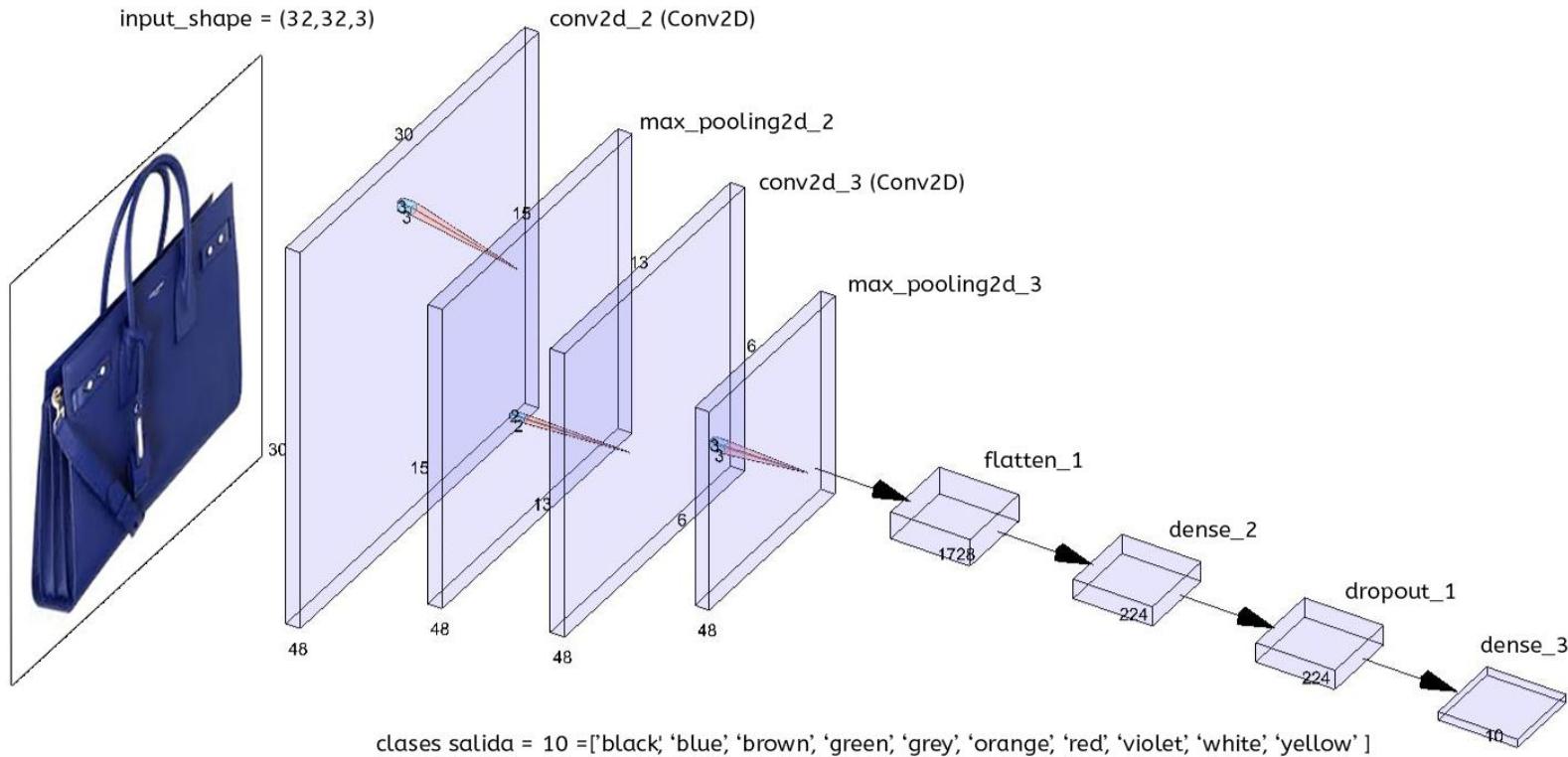
Si quiséramos implementar segmentación de imágenes para reconocer colores sobre fondos o varios colores en una misma imagen estar arquitectura podría ir aumentando su complejidad hasta lograrlo.

Continuamos con la afinación de hiperparametros con Keras Tuner.

UNIT 4. RESULTADOS PRIMERA ITERACIÓN (PROTOTIPADO)

4.6. Modelo seleccionado para desarrollo

RED KERAS CNN OPTIMIZADA (Acc = 0.9599)

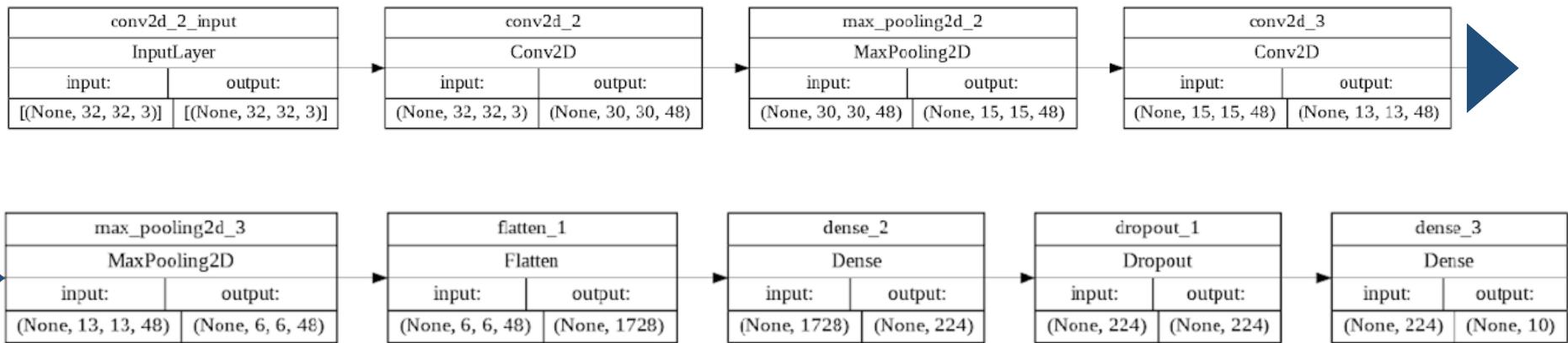


UNIT 4. RESULTADOS PRIMERA ITERACIÓN (PROTOTIPADO)

4.6. Modelo seleccionado para desarrollo

RED KERAS CNN OPTIMIZADA (Acc = 0.9599)

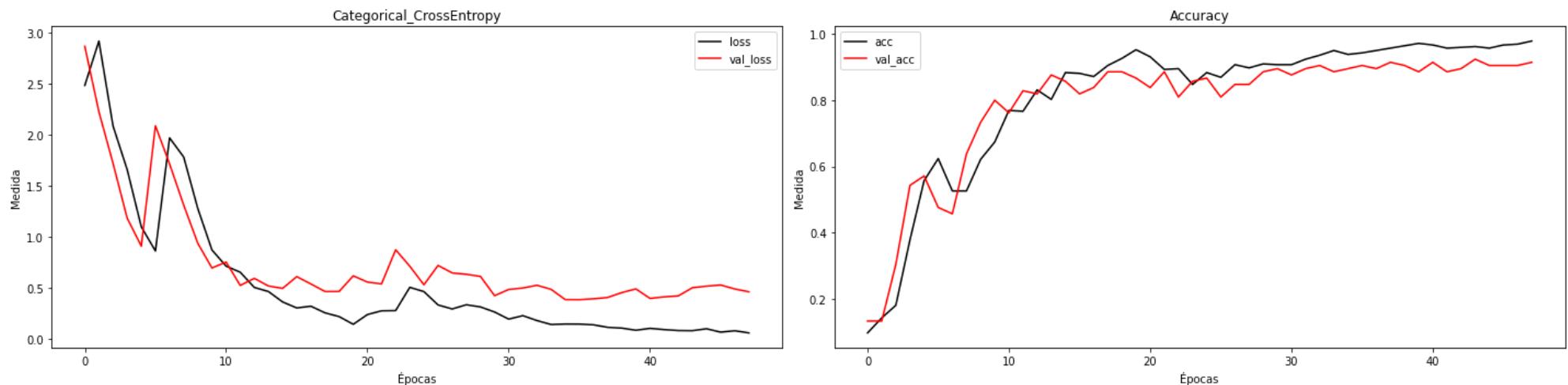
Esquema estructura capas modelo



UNIT 4. RESULTADOS PRIMERA ITERACIÓN (PROTOTIPADO)

4.6. Modelo seleccionado para desarrollo

RED KERAS CNN OPTIMIZADA (Acc = 0.9599)

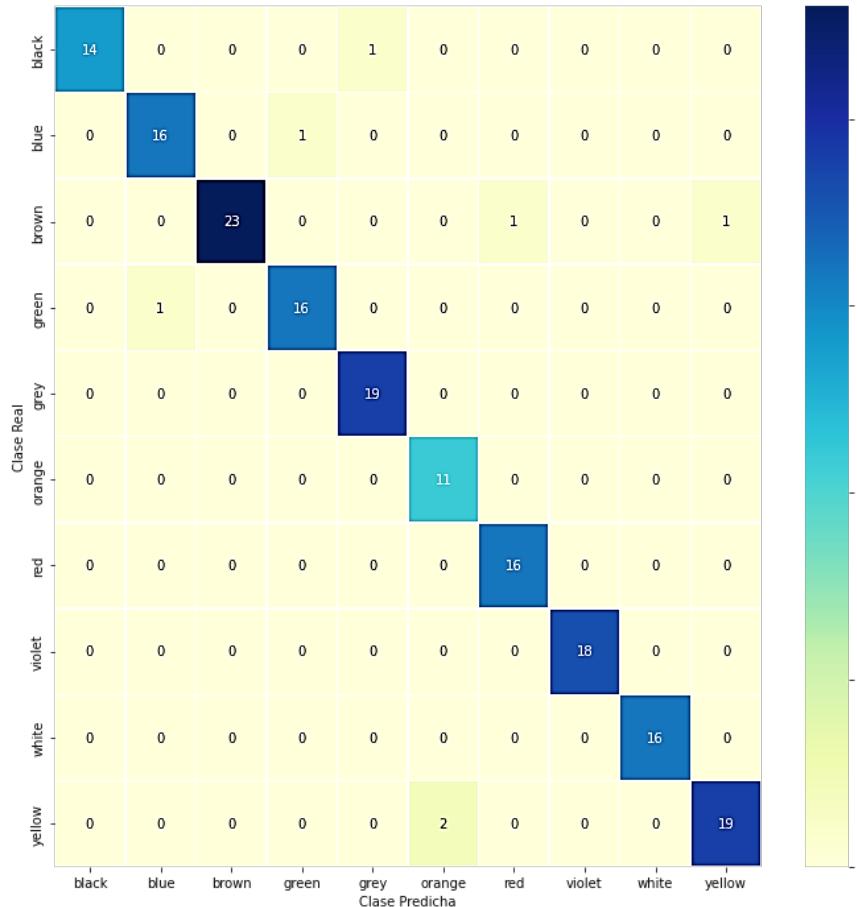


Resultados test (accuracy – categorial crossentropy)

UNIT 4. RESULTADOS PRIMERA ITERACIÓN (PROTOTIPADO)

4.6. Modelo seleccionado para desarrollo

RED KERAS CNN OPTIMIZADA (Acc = 0.9599)

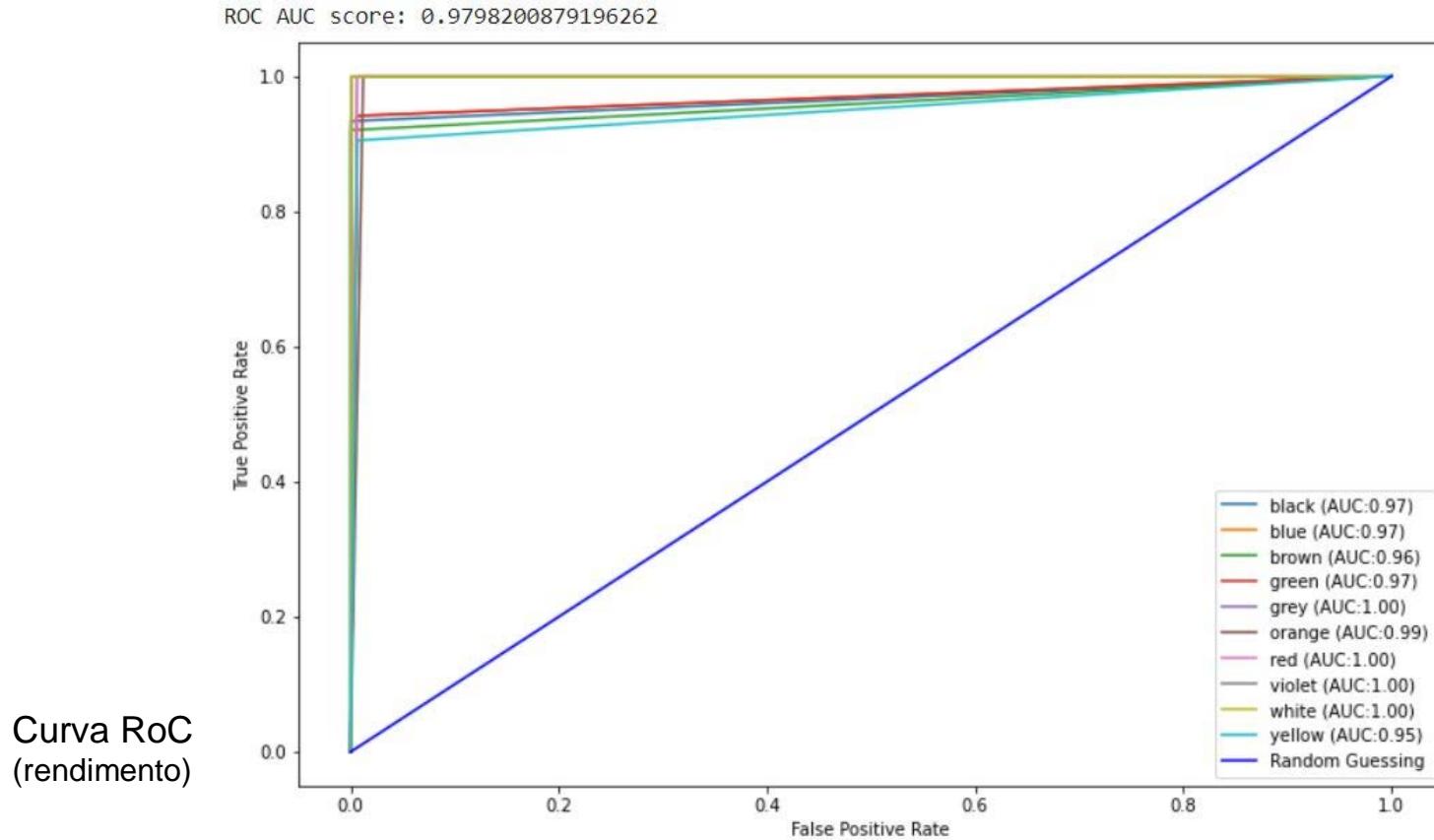


Matriz de confusión

UNIT 4. RESULTADOS PRIMERA ITERACIÓN (PROTOTIPADO)

4.6. Modelo seleccionado para desarrollo

RED KERAS CNN OPTIMIZADA (Acc = 0.9599)



UNIT 4. RESULTADOS PRIMERA ITERACIÓN (PROTOTIPADO)

4.6. Modelo seleccionado para desarrollo

RED KERAS CNN OPTIMIZADA (Acc = 0.9599)

Resultados de clasificación predicha y real

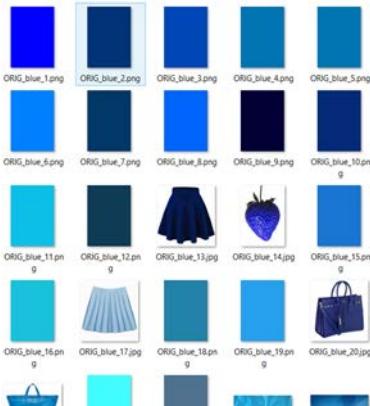
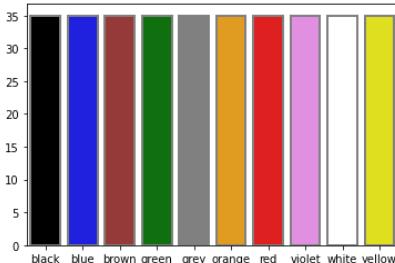
	precision	recall	f1-score	support
black	1.00	0.93	0.97	15
blue	0.94	0.94	0.94	17
brown	1.00	0.92	0.96	25
green	0.94	0.94	0.94	17
grey	0.95	1.00	0.97	19
orange	0.85	1.00	0.92	11
red	0.94	1.00	0.97	16
violet	1.00	1.00	1.00	18
white	1.00	1.00	1.00	16
yellow	0.95	0.90	0.93	21
accuracy			0.96	175
macro avg	0.96	0.96	0.96	175
weighted avg	0.96	0.96	0.96	175

UNIT 4. RESULTADOS PRIMERA ITERACIÓN (PROTOTIPADO)

4.7. Mejoras implementadas dataset original

Dataset Original*

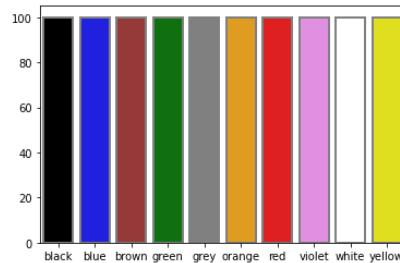
- Imágenes por clase = 35
- Predominancia = tonos color
- Total imágenes = 350



Ej. clase "azul"

Dataset AMP1

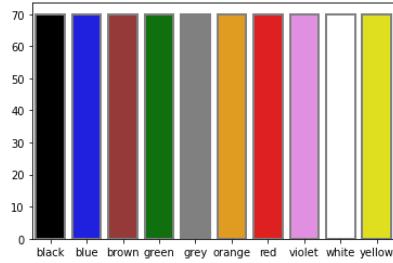
- Imágenes por clase = 100
- Predominancia = objeto/fondo
- Total imágenes = 1000



Ej. clase "azul"

Dataset AMP2

- Imágenes por clase = 70
- Predominancia = tonos color
- Total imágenes = 700

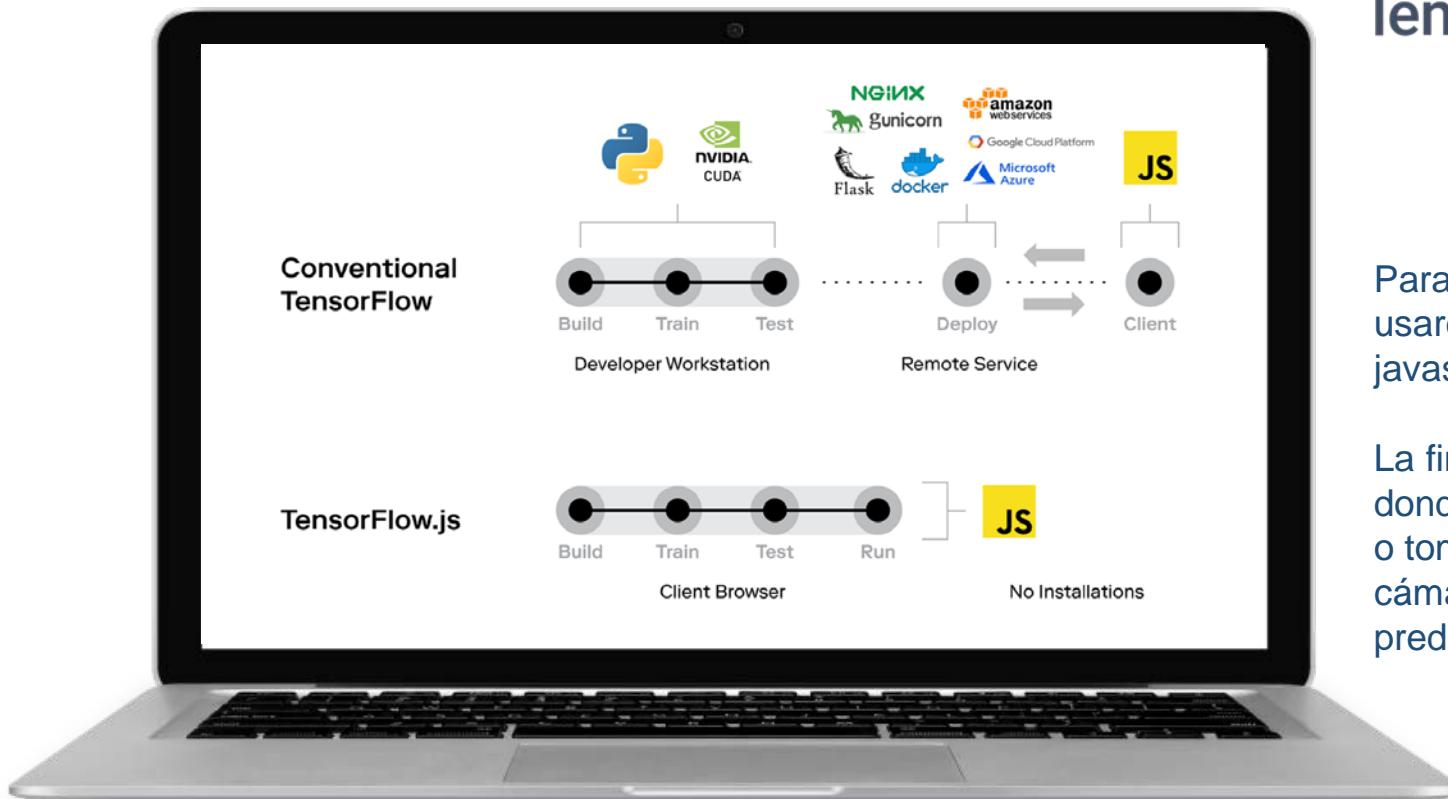


Ej. clase "azul"

*Dataset original creado a partir de dos dataset de kaggle, y aumentándolo con imágenes web para balancearlo

UNIT 4. RESULTADOS PRIMERA ITERACIÓN (PROTOTIPADO)

4.8. Implementación web



Para la implementación web usaremos tensorflow.js y javascript.

La finalidad es crear una api web donde se pueda subir una imagen o tomar una imagen con la cámara de un dispositivo móvil y predecir el color enfocado.

SAMSUNG

resultados

MODELADO FINAL

Iris Eye Team

AI Course

Samsung Innovation Campus

Together for Tomorrow!
Enabling People

Education for Future Generations

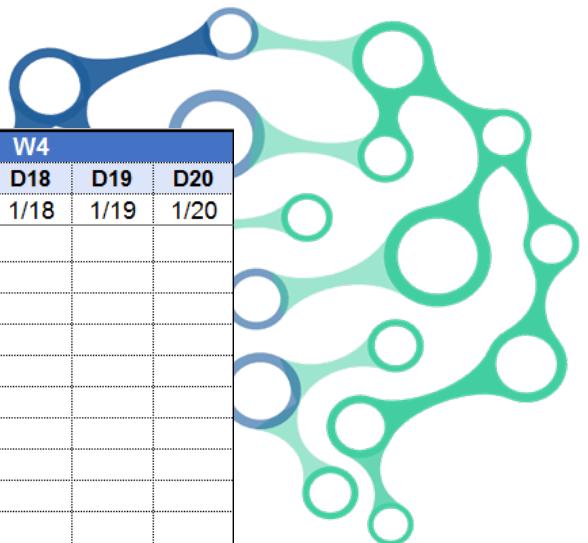
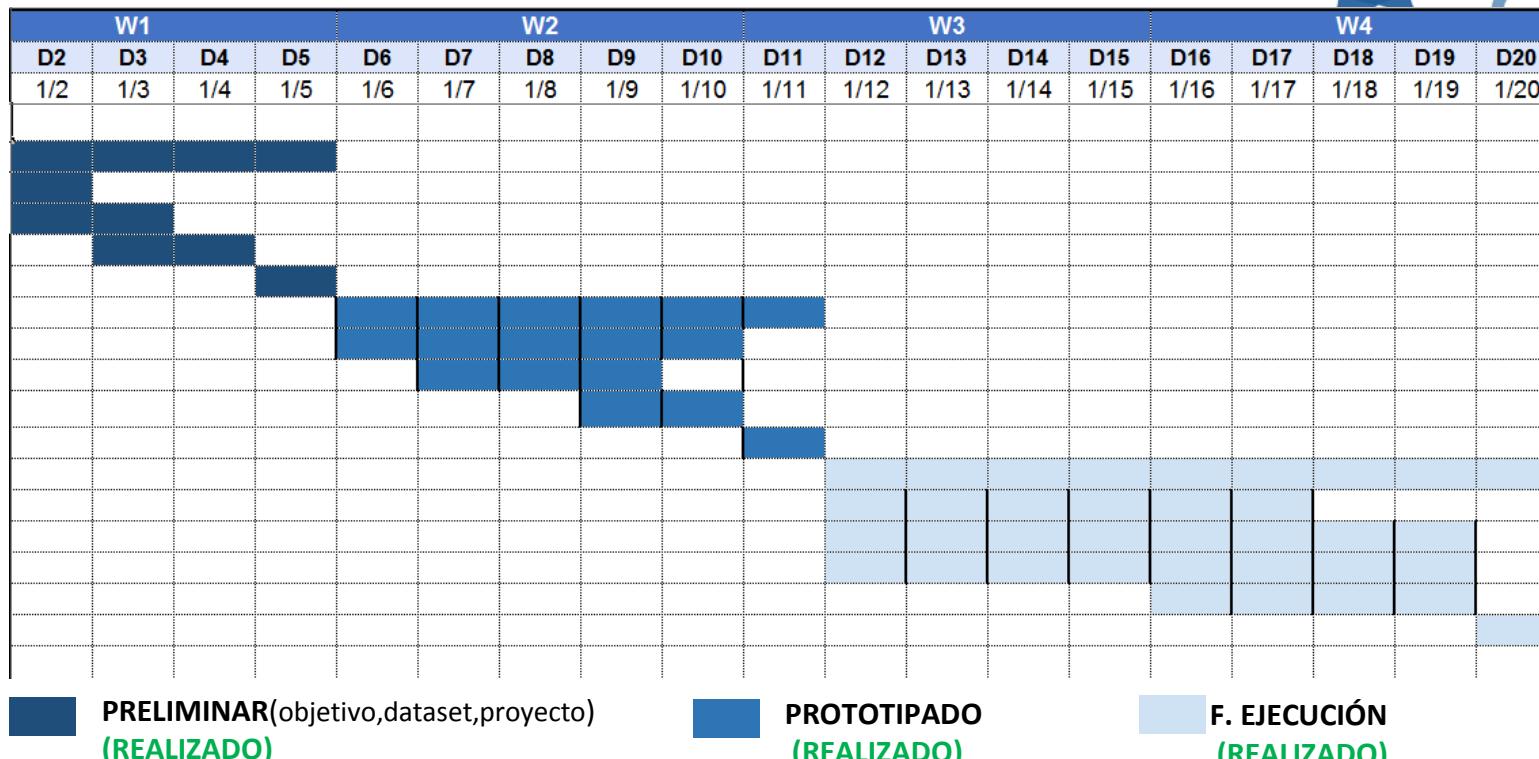


IRIS EYE

UNIT 5. Modelado Final

- 5.1. Estado actual planning desarrollo
- 5.2. Herramientas y plataformas de desarrollo utilizadas
- 5.3. Datasets ampliación y preprocesamiento
- 5.4. Flujo de trabajo
- 5.5. Entrenamiento y optimización de hiperparámetros
- 5.6. Arquitectura modelo final
- 5.7. Resultados entrenamiento modelo final
- 5.8 Resultados test modelo final
- 5.9 Interface de usuario (interface)
- 5.10 Propuestas de mejora
- 5.11. Logros y beneficios
- 5.12 Conclusiones finales
- 5.13 Miembros del equipo

CAPSTONE BREAKDOWN STRUCTURE



5.2. Herramientas y plataformas

I HERRAMIENTAS Y PLATAFORMAS DE DESARROLLO UTILIZADAS



UNIT 5. MODELADO FINAL

5.3. Roles asignados

Lorena Jiménez Tejada

Coordinación Equipo. Documentación y presentaciones. Adquisición dataset. Desarrollo, prueba y mejora de modelos Machine Learning. Desarrollo app web y adaptación a dispositivo móvil.

Marina Jiménez Egea

Documentación y presentaciones. Adquisición dataset. Desarrollo, prueba y mejora de modelos Machine Learning. Desarrollo app web y adaptación a dispositivo móvil. Investigación audio web.

Marta Freire Painceira

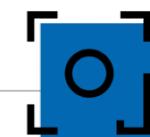
Documentación y presentaciones. Adquisición y mejora dataset. Desarrollo, prueba y mejora de modelos Machine Learning y Redes CNN hiperparámetros optimizados. Tablas de datos y gráficas.

Marta Trasancos Yáñez

Documentación y presentaciones. Adquisición y mejora dataset. Desarrollo, prueba y mejora de modelos Machine Learning y Redes CNN hiperparámetros optimizados. Diseño logo y presentaciones.

Mercedes Rodríguez Barbero

Documentación y presentaciones. Adquisición y mejora dataset. Desarrollo, prueba y mejora modelos Machine Learning. Desarrollo app web y adaptación a dispositivo móvil.

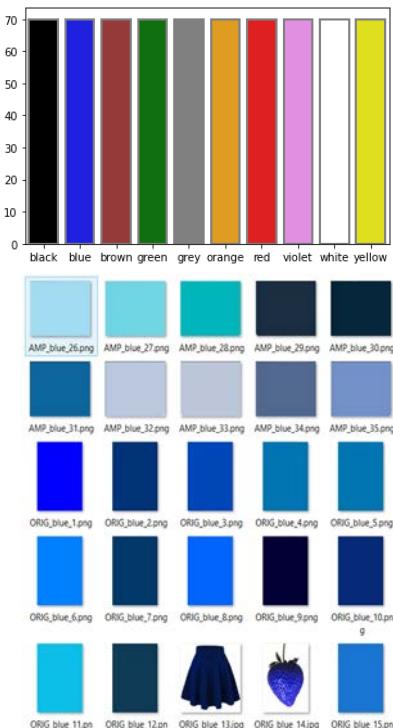


UNIT 5. MODELADO FINAL

5.4. Datasets ampliación y preprocessamiento

Dataset AMP2

- Nº de clases = **10**
- Imágenes por clase = 70
- Predominancia = tonos color
- Total imágenes = 700



Ej. clase “azul”

Dataset AMP3

- Nº de clases = **11**
- Imágenes por clase = 70
- Predominancia = tonos color
- Total imágenes = 770



Ej. clase “rosa”

Para entrenar el modelo final utilizamos en un primer momento el Dataset ampliado 2.

Una vez ajustado un modelo con buenos resultados de exactitud y rendimiento, decidimos mejorar nuestra clasificación inicial ampliando el dataset con una nueva clase (color rosa) y entrenando de nuevo el modelo para mejorar los anteriores resultados alcanzados con 10 clases.

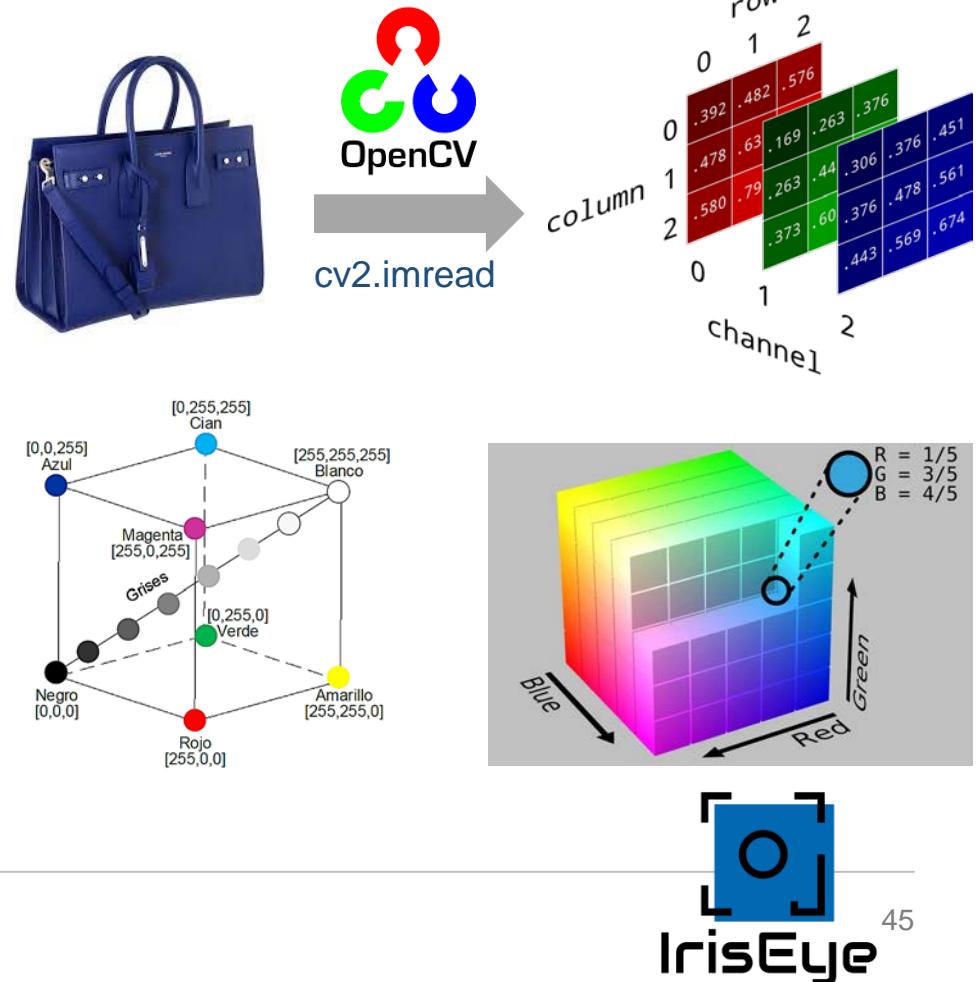
UNIT 5. MODELADO FINAL

5.4. Datasets ampliación y preprocessamiento

PREPROCESAMIENTO

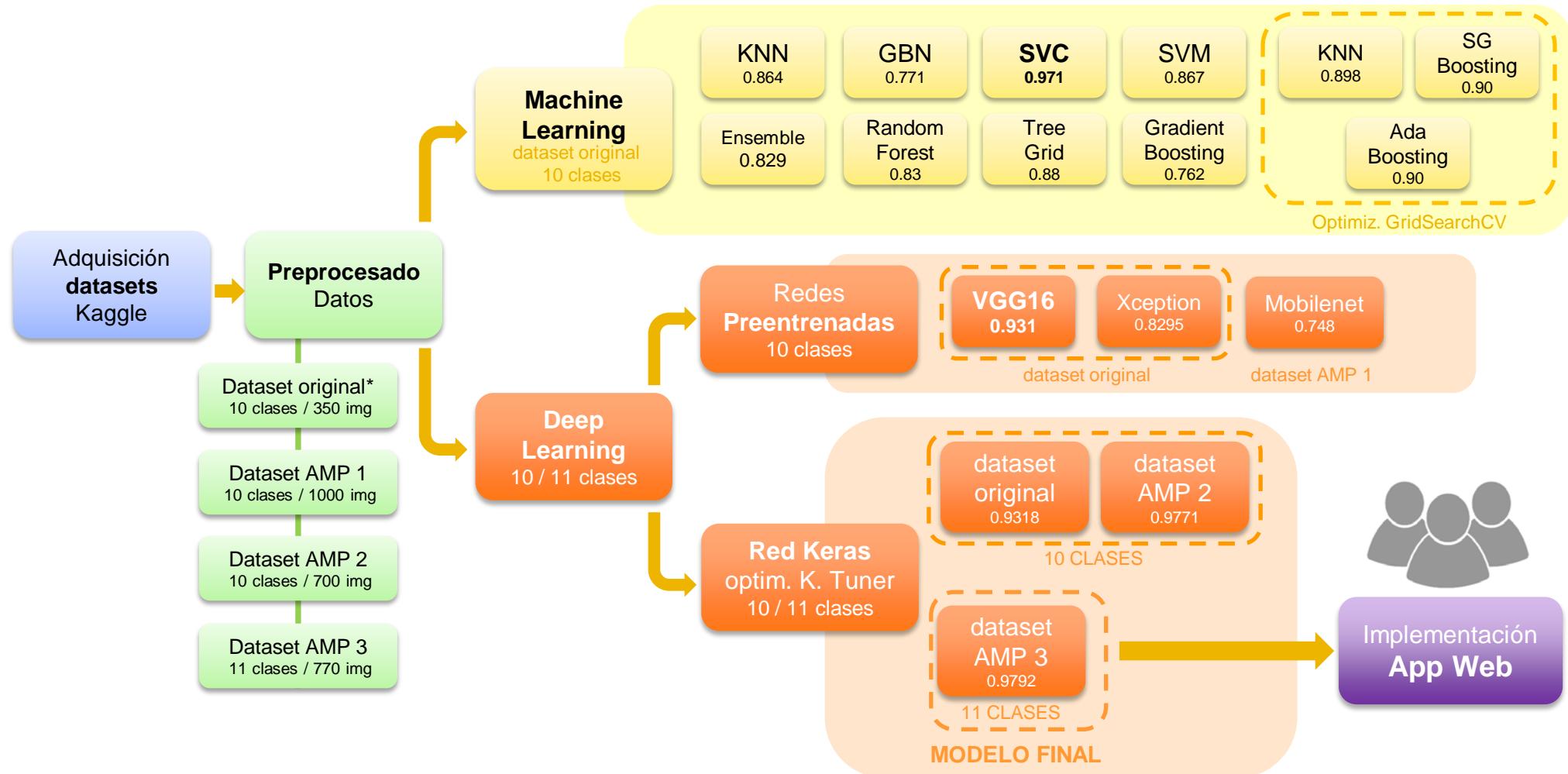
Previamente a la introducción de las imágenes en los modelos se realizan los siguientes pasos:

- Lectura de imágenes con librería OpenCv para convertir las imágenes originales en formato jpg/png en matrices dim x dim x 3 (modo **color bgr**), regularización su tamaño a 28x28 px o 32x32 px. dependiendo del modelo a probar.
- Se agrega una etiqueta de clasificación a cada imagen en función del nombre de su carpeta de almacenamiento: black, blue, brown, green , grey, orange, pink, red, violet, white, yellow.
- División del conjunto de datos en conjunto de train y test con proporción 75/25 y random state = 123.
- Normalización de los valores gbr dividiendo los datos de X_train y X_test por 255.0
- Codificación one-hot de las variables de salida Y_train, Y_test en 10 u 11 clases, dependiendo del dataset utilizado.



UNIT 5. MODELADO FINAL

5.5. Flujo de trabajo



UNIT 5. MODELADO FINAL

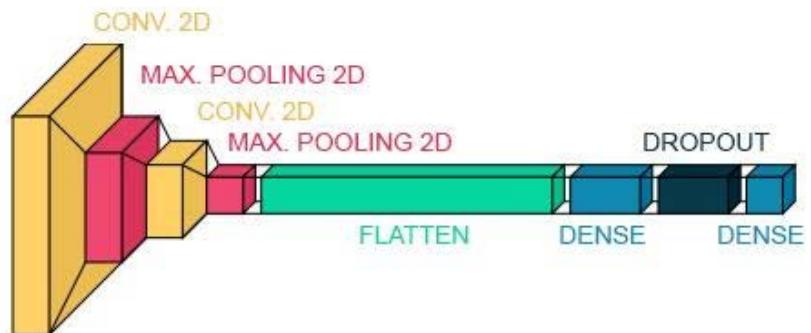
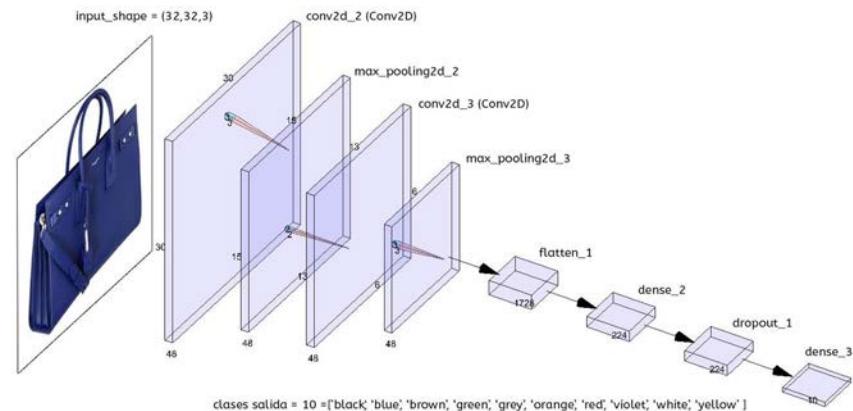
5.6. Entrenamiento y optimización hiperparámetros

CLASIFICACIÓN 10 COLORES*

Tras los excelentes resultados alcanzados en la primera iteración (accuracy = 0.9599) con la **red convolucional KERAS** optimizada con **KERAS TUNER**, decimos seguir entrenándola con el **dataset AMP 1** y los siguiente rangos de variación de hiperparámetros:

- N° filtros en capa convolucional 1: 32 a 64, paso:8
- Tamaño kernel en capa convolucional 1: 3 a 6
- N° filtros en capa convolucional 2: 32 a 128, paso:16
- Tamaño kernel en capa convolucional 2: 3 a 6
- N° unidades en capa Densa 1: 32 a 512, paso:16
- Valor ‘dropout’: 0.0 a 0.5, paso: 0.1
- Valor ‘learning rate’: 0.01 a 0.0001

*negro, azul, marrón, verde, naranja, rojo, violeta, blanco, amarillo



UNIT 5. MODELADO FINAL

5.6. Entrenamiento y optimización hiperparámetros

RESULTADOS ENTRENAMIENTO Y OPTIMIZACIÓN 10 COLORES*

	MODELO 01	MODELO 02	MODELO 03	MODELO 04	MODELO 05
ACCURACY	0.9314	0.9485	0.9599	0.9657	0.9771
OPT.KERAS TUNER	Sí	Sí	Sí	Sí	Sí
TAM. FILTRO	30x30x48 (Conv_2d 1) 13x13x48 (Conv_2d 2)				
TAM. KERNEL	3x3 (Conv_2d 1) 3x3 (Conv_2d 2)				
Nº UDS FLATTEN	1728	1728	1728	1728	1728
Nº UDS. DENSE	224 (Dense 1) 10 (Dense 2)				
TAM. BATCH	20	75	300	290	290
Nº ÉPOCAS	48	48	48	48	98

*negro, azul, marrón, verde, naranja, rojo, violeta, blanco, amarillo

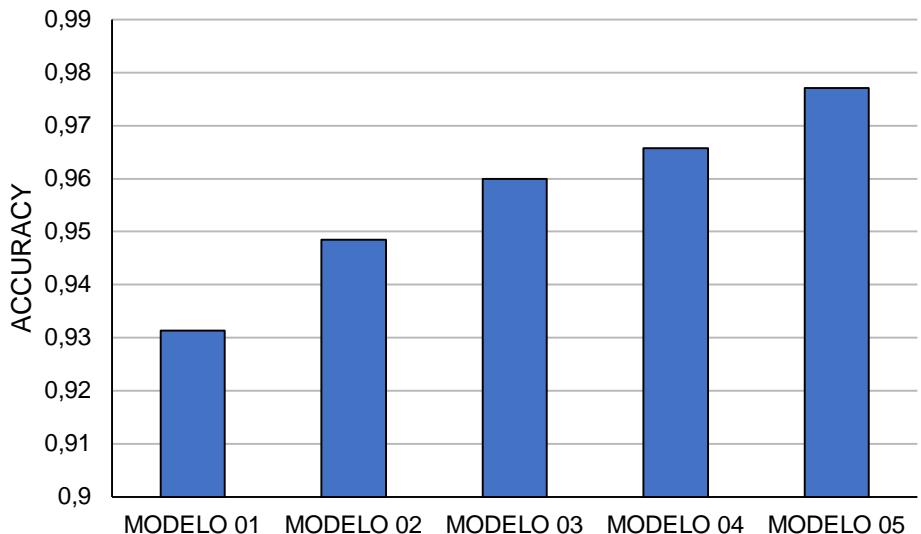
UNIT 5. MODELADO FINAL

5.6. Entrenamiento y optimización hiperparámetros

RESULTADOS CLASIFICACIÓN 10 COLORES*

Conseguimos diversos modelos con un alto grado de exactitud, siendo el mejor resultado alcanzado un 0,9771.

CLASIFICACIÓN EN 10 CLASES



*negro, azul, marrón, verde, naranja, rojo, violeta, blanco, amarillo

UNIT 5. MODELADO FINAL

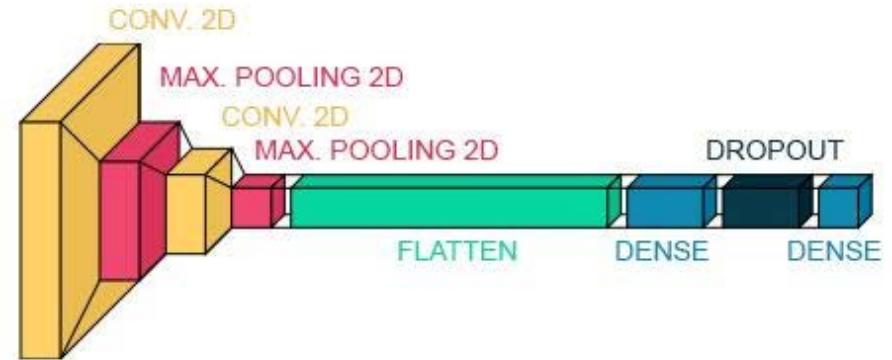
5.6. Entrenamiento y optimización hiperparámetros

CLASIFICACIÓN 11 COLORES*

Habiendo alcanzado muy buenos resultados en la clasificación para 10 colores, decidimos ampliar nuestro dataset de entrenamiento con una nueva clase, el color rosa, ya que al seleccionar, por ejemplo, prendas de ropa u otros objetos, nos parece importante poder diferenciarla del violeta, rojo o naranja.

Entrenamos ahora, nuestro modelo base con el **dataset ampliado 3** (70 imágenes por clase, **11 clases**) probando diferentes configuraciones de KERAS TUNER dentro de los siguientes rangos de variación para las diferentes capas del modelo:

- Nº filtros en capa convolucional 1: 32 a 64, paso:8
- Tamaño kernel en capa convolucional 1: 3 a 6
- Nº filtros en capa convolucional 2: 32 a 128, paso:16
- Tamaño kernel en capa convolucional 2: 3 a 6
- Nº unidades en capa Densa 1: 32 a 512, paso:16
- Valor 'dropout': 0.0 a 0.5, paso: 0.1
- Valor 'learning rate': 0.01 a 0.0001



*negro, azul, marrón, verde, naranja, rosa, rojo, violeta, blanco, amarillo

UNIT 5. MODELADO FINAL

5.6. Entrenamiento y optimización hiperparámetros

OPTIMIZACIÓN CLASIFICACIÓN 11 COLORES*

En algunos de los entrenamientos, implementamos una nueva mejora, desarrollando una función callback ('best weights training') que almacena los valores de los pesos en la época del entrenamiento donde se haya alcanzado una mayor exactitud ('accuracy').

Una vez entrenado, testeamos el modelo, tanto con los pesos resultantes del entrenamiento como con los mejores pesos almacenados consiguiendo, en algunos casos, mejorar la exactitud final (accuracy).

```
#Definimos el hipermodelo con los mejores parámetros
hypermodel = tuner.hypermodel.build(best_hps)

filepath="weights.best.hdf5"
checkpoint = ModelCheckpoint(filepath, monitor='val_accuracy', verbose=1, save_best_only=True, mode='max')
callbacks_list = [checkpoint]

# Reentrenamos el modelo en un número óptimo de épocas (best_epoch)
tamBatch =15
history = hypermodel.fit(
    X_train,
    y_trainFinal,
    epochs=best_epoch,
    batch_size=tamBatch,
    validation_split=0.2,
    callbacks=callbacks_list,
    verbose=2,
)
val_acc_per_epoch = history.history['val_accuracy']

eval_result = hypermodel.evaluate(X_test, y_testFinal)
print("[test loss, test accuracy]:", eval_result)

7/7 [=====] - 0s 6ms/step - loss: 0.1391 - accuracy: 0.9637
[test loss, test accuracy]: [0.13911795616149902, 0.9637305736541748]

#Probamos a realizar test cargando en el hipermodelo los mejores pesos del entrenamiento
hypermodel.load_weights("weights.best.hdf5")
eval_result_best = hypermodel.evaluate(X_test, y_testFinal)
print("[test loss, test accuracy]:", eval_result_best)

7/7 [=====] - 0s 5ms/step - loss: 0.1105 - accuracy: 0.9793
[test loss, test accuracy]: [0.11054679751396179, 0.9792746305465698]
```

*negro, azul, marrón, verde, naranja, rosa, rojo, violeta, blanco, amarillo

UNIT 5. MODELADO FINAL

5.6. Entrenamiento y optimización hiperparámetros

RESULTADOS ENTRENAMIENTO Y OPTIMIZACIÓN 11 COLORES*

	MODELO 06	MODELO 07	MODELO 08	MODELO 09	MODELO 10
ACCURACY	0.9585	0.9637	0.9637	0.9740	0.9792
OPT.KERAS TUNER	Sí	Sí	Sí	Sí	Sí
TAM. FILTRO	27x27x48 (Conv_2d 1) 11x11x48 (Conv_2d 2)	30x30x48 (Conv_2d 1) 10x10x64 (Conv_2d 2)	30x30x32 (Conv_2d 1) 13x13x64 (Conv_2d 2)	30x30x48 (Conv_2d 1) 10x10x64 (Conv_2d 2)	30x30x48 (Conv_2d 1) 10x10x64 (Conv_2d 2)
TAM. KERNEL	6x6 (Conv_2d 1) 3x3 (Conv_2d 2)	3x3 (Conv_2d 1) 6x6 (Conv_2d 2)	3x3 (Conv_2d 1) 3x3 (Conv_2d 2)	3x3 (Conv_2d 1) 6x6 (Conv_2d 2)	3x3 (Conv_2d 1) 6x6 (Conv_2d 2)
Nº UDS FLATTEN	1200	1600	2304	1600	1600
Nº UDS. DENSE	288 (Dense 1) 11 (Dense 2)	384 (Dense 1) 11 (Dense 2)	64 (Dense 1) 11 (Dense 2)	384 (Dense 1) 11 (Dense 2)	384 (Dense 1) 11 (Dense 2)
TAM. BATCH	10	20	400	15	15
Nº ÉPOCAS	65	131	64	131	131
BEST-WEIGHTS TRAINING	NO	SÍ	NO	SÍ	SI

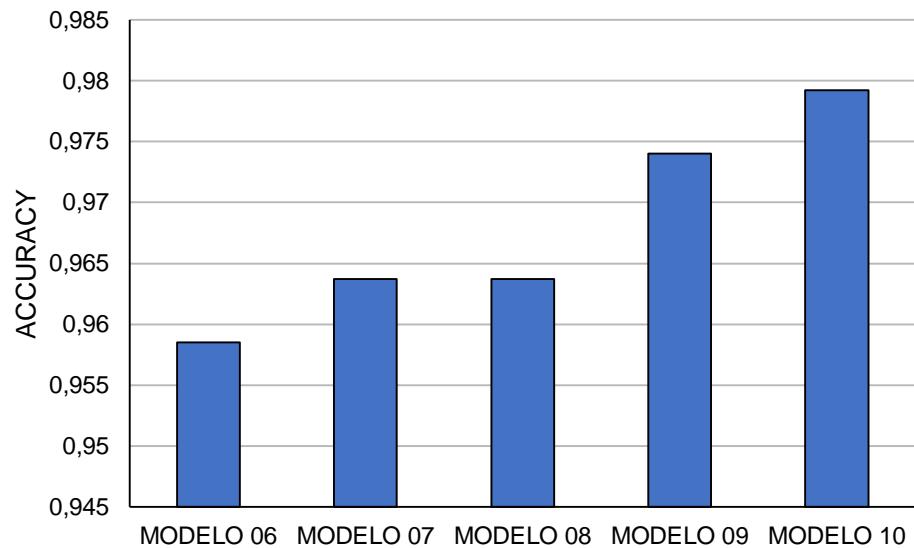
*negro, azul, marrón, verde, naranja, rosa, rojo, violeta, blanco, amarillo

5.6. Entrenamiento y optimización hiperparámetros

RESULTADOS CLASIFICACIÓN 11 COLORES*

Conseguimos, de nuevo, diversos modelos con un alto grado de exactitud, siendo el mejor resultado alcanzado un **0,9792**, que se corresponde con el **modelo seleccionado finalmente** y utilizado para **desarrollar la app web** de interacción con el usuario.

CLASIFICACIÓN EN 11 CLASES

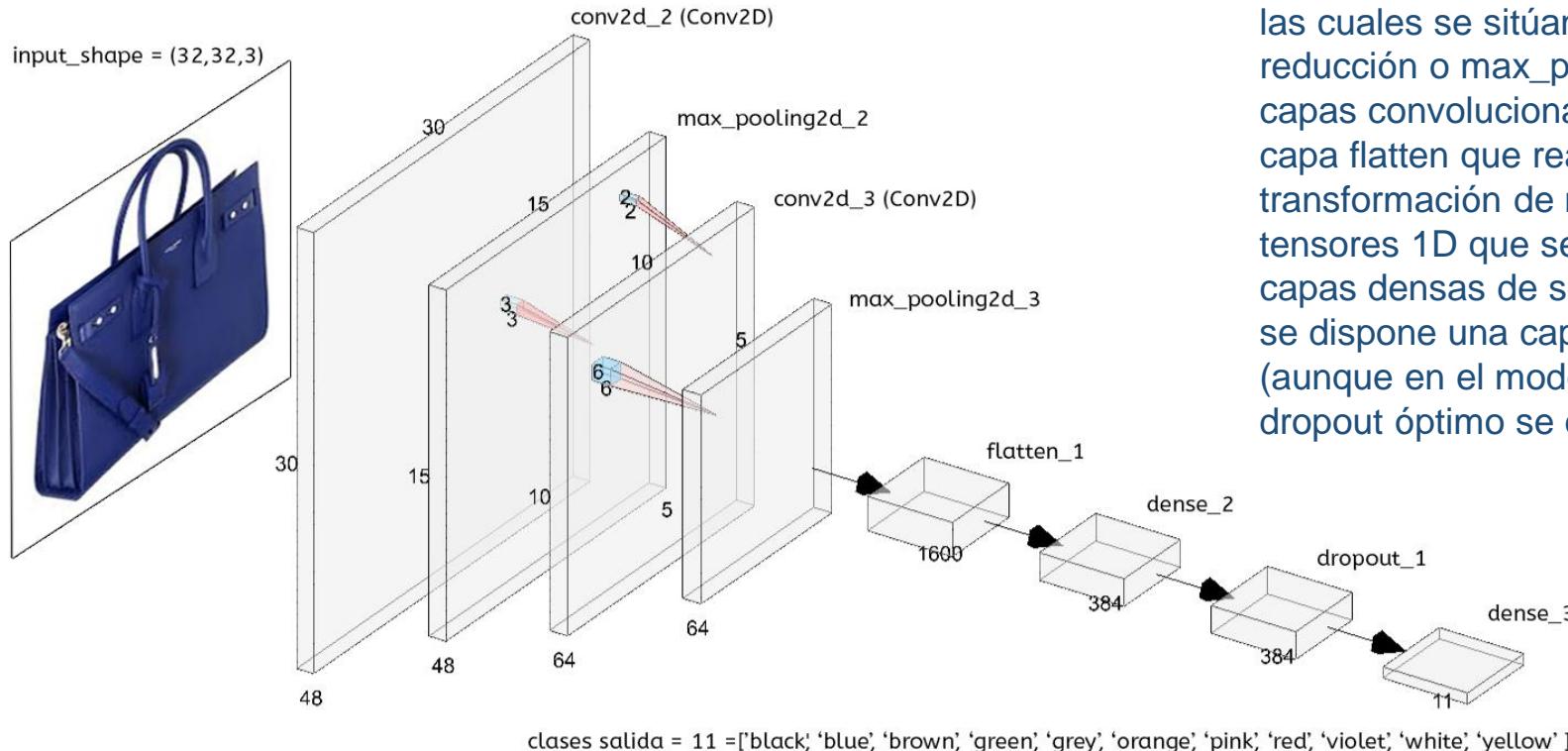


*negro, azul, marrón, verde, naranja, rosa, rojo, violeta, blanco, amarillo

UNIT 5. MODELADO FINAL

5.7. Arquitectura modelo final

CONFIGURACIÓN FINAL RED CONVOLUCIONAL KERAS



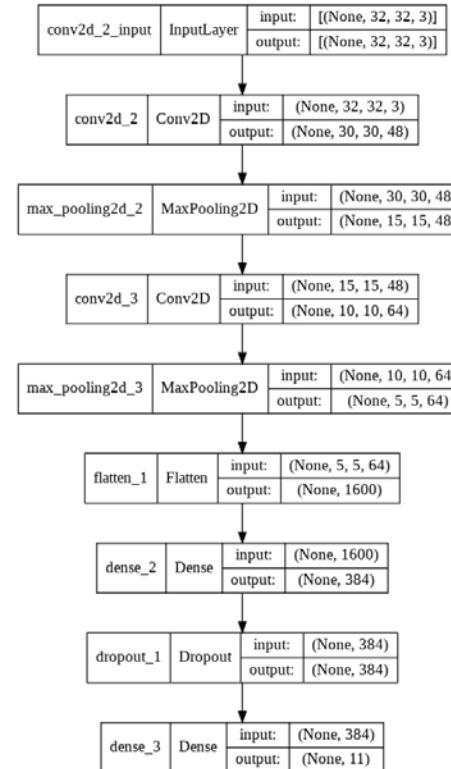
UNIT 5. MODELADO FINAL

5.7. Arquitectura modelo final

CONFIGURACIÓN FINAL RED CONVOLUCIONAL KERAS

Model: "sequential_1"

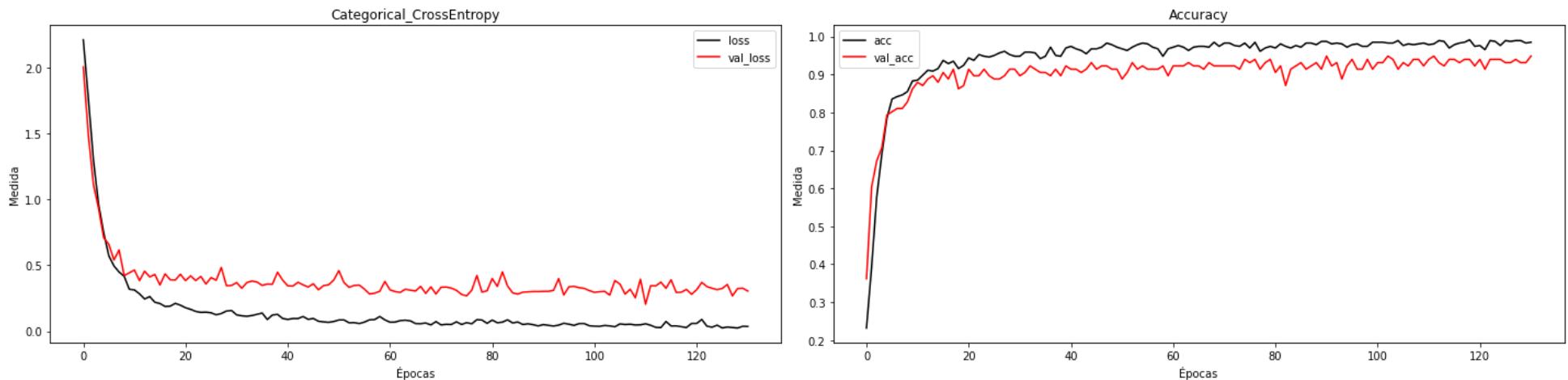
Layer (type)	Output Shape	Param #
<hr/>		
conv2d_2 (Conv2D)	(None, 30, 30, 48)	1344
max_pooling2d_2 (MaxPooling2D)	(None, 15, 15, 48)	0
conv2d_3 (Conv2D)	(None, 10, 10, 64)	110656
max_pooling2d_3 (MaxPooling2D)	(None, 5, 5, 64)	0
flatten_1 (Flatten)	(None, 1600)	0
dense_2 (Dense)	(None, 384)	614784
dropout_1 (Dropout)	(None, 384)	0
dense_3 (Dense)	(None, 11)	4235
<hr/>		
Total params:	731,019	
Trainable params:	731,019	
Non-trainable params:	0	



UNIT 5. MODELADO FINAL

5.8. Resultados entrenamiento modelo final

EXACTITUD Y FUNCIÓN DE PÉRDIDA EN ENTRENAMIENTO



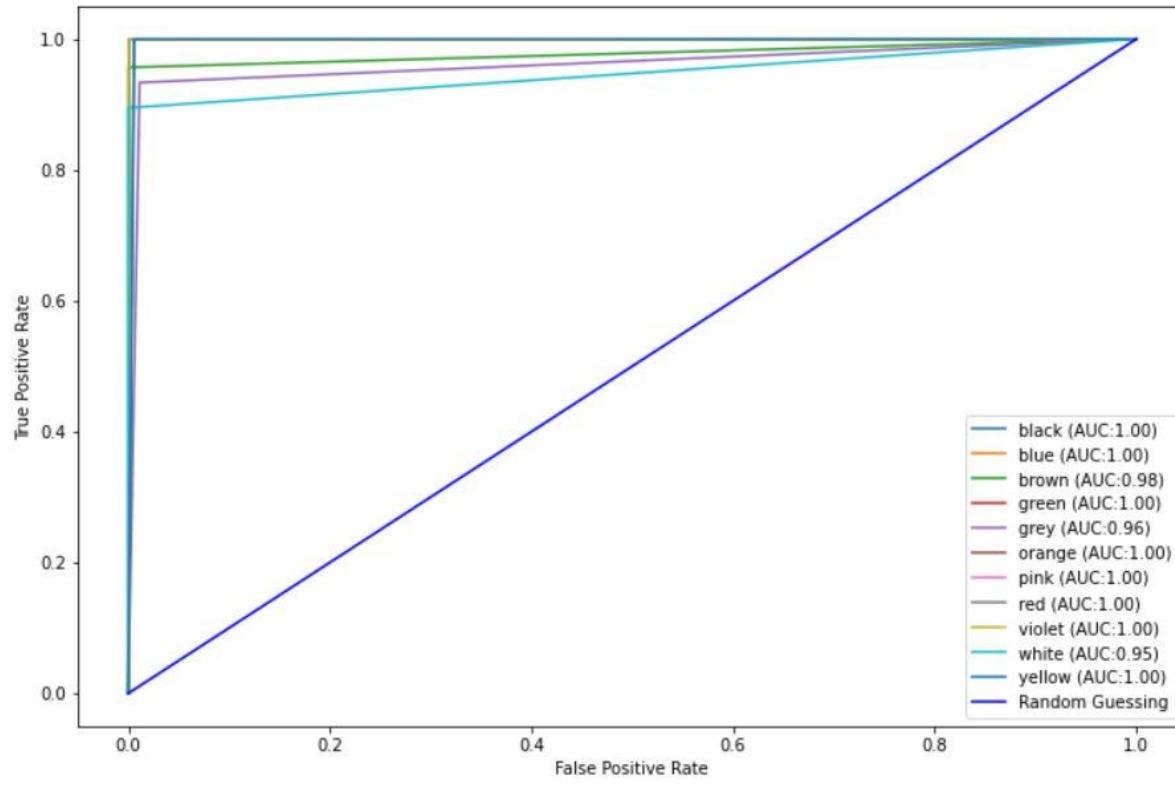
Resultados test (accuracy – categorial crossentropy)

Podemos observar como el modelo durante el entrenamiento se ajusta rápidamente desde las primeras épocas, sin producirse luego grandes oscilaciones ni sobreentrenamiento (overfitting).

5.8. Resultados test modelo final

CURVA ROC MODELO FINAL

ROC AUC score: 0.9891799934371247



La curva roc (Receiver Operating Characteristic) nos muestra que el modelo tiene un rendimiento muy bueno en la clasificación de los diferentes colores.

Esta gráfica enfrenta el **ratio de falsos positivos** (eje x) con el **ratio de falsos negativos** (eje y).

Los colores que presentan valores más bajos respecto a los demás son el blanco, el gris y el marrón.

UNIT 5. MODELADO FINAL

5.8. Resultados test modelo final

MATRIZ DE CONFUSIÓN MODELO FINAL

	precision	recall	f1-score	support
black	0.94	1.00	0.97	17
blue	1.00	1.00	1.00	14
brown	1.00	0.96	0.98	23
green	1.00	1.00	1.00	21
grey	0.88	0.93	0.90	15
orange	1.00	1.00	1.00	16
pink	1.00	1.00	1.00	10
red	1.00	1.00	1.00	17
violet	1.00	1.00	1.00	23
white	1.00	0.89	0.94	19
yellow	0.95	1.00	0.97	18
accuracy			0.98	193
macro avg	0.98	0.98	0.98	193
weighted avg	0.98	0.98	0.98	193

Precision (precisión)

Con esta métrica podemos medir la **calidad** del modelo la clasificación. Nos indica el ratio de predicciones positivas correctas respecto al total de observaciones predichas positivas.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Recall (Exhaustividad)

Esta métrica nos informa sobre la **cantidad** que el modelo es capaz de identificar. Es el ratio de predicciones correctas positivas respecto al total de observaciones reales de esa clase.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

F1-score

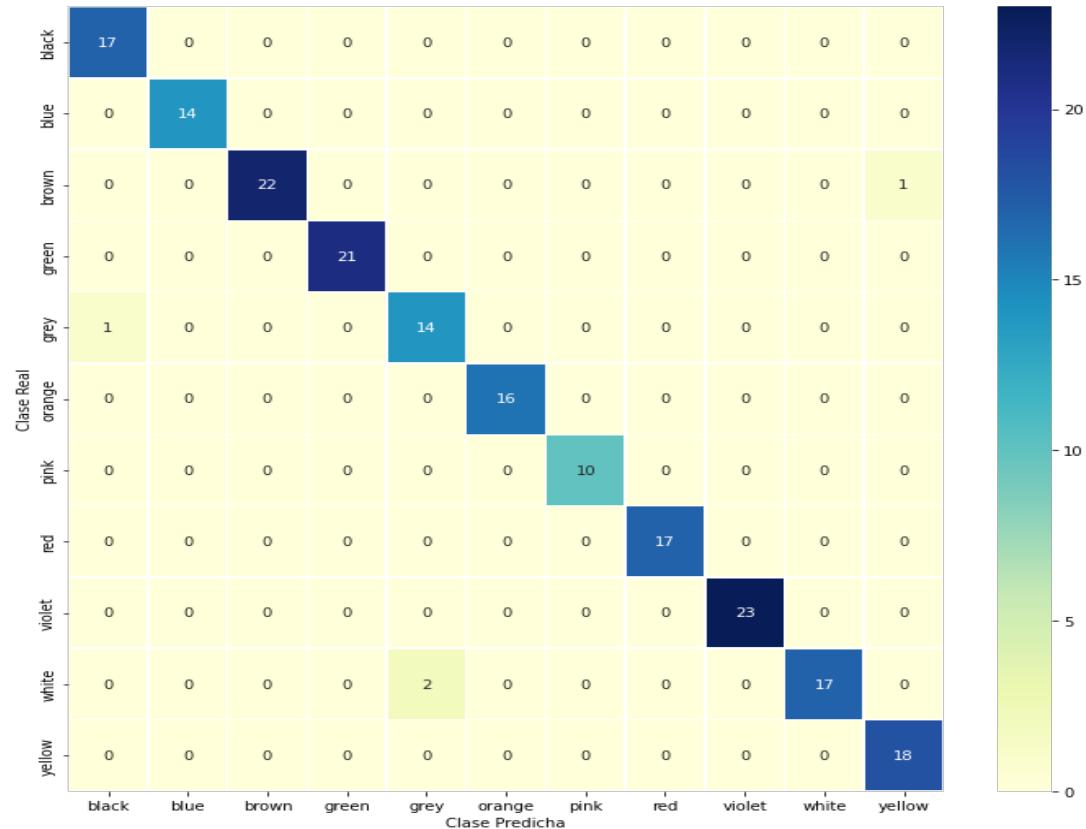
Esta métrica se utiliza para combinar las medidas de precision y recall en un sólo valor.

$$\text{F1-score} = \frac{2 \times (\text{Recall} + \text{Precision})}{(\text{Recall} + \text{Precision})}$$

UNIT 5. MODELADO FINAL

5.8. Resultados test modelo final

RESULTADOS MODELO FINAL



La matriz de confusión del modelo nos indica que presenta un alto nivel de exactitud a la hora de clasificar los diferentes colores, incluso aumentando su número a 11 clases.

La diagonal principal nos muestra las imágenes correctamente predichas para cada una de las clases (colores) del conjunto de test.

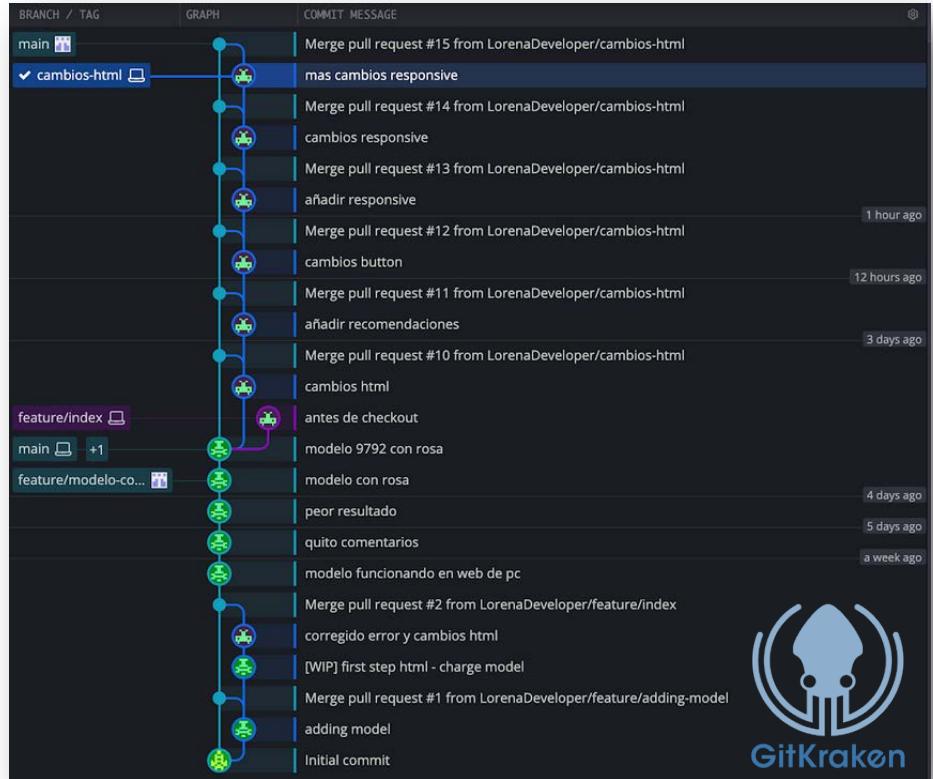
UNIT 5. MODELADO FINAL

5.9. Interfaz de usuario (interface)

IMPLEMENTACIÓN WEB

Para la implementación web investigamos y el modo que nos pareció más eficiente fue utilizar tensorflowjs_converter a partir del modelo guardado en formato h5.

Una vez convertido el modelo y descargado en local, lo cargamos en la web mediante el script de tensorflowjs, el cual permite hacerlo de forma asíncrona.



UNIT 5. MODELADO FINAL

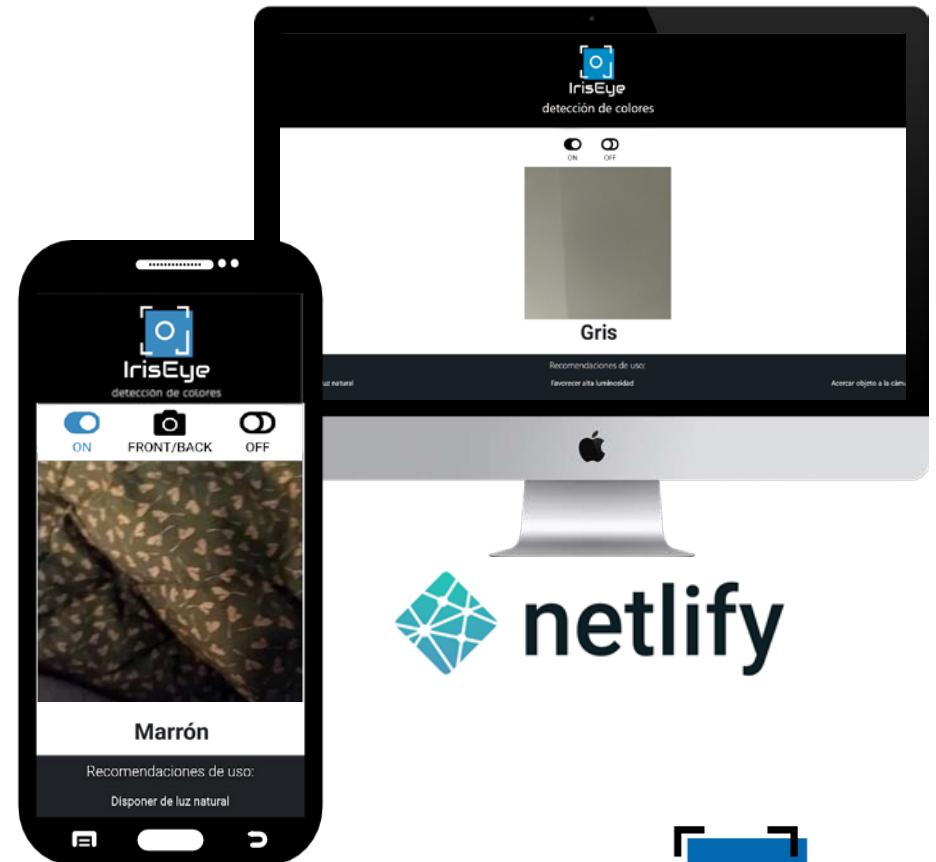
5.9. Interfaz de usuario (interface)

■ APPLICACIÓN MULTIDISPOSITIVO

Realizamos el acceso a la cámara mediante un mediaDevice en un elemento de vídeo y añadimos tres botones para dar las funcionalidades de apagar/encender cámara y cambio (en modo móvil) de cámara frontal a trasera y viceversa.

La imagen captada por la cámara, es convertida a un canvas de 32x32 y normalizada,luego se pasa a tf.tensor4d para darle el formato necesario de entrada a la función 'predict' del modelo.

La respuesta será la etiqueta de la clase que mayor probabilidad prediga y se mostrará por pantalla el nombre del color.



61

5.9. Interfaz de usuario (interface)

ACCESIBILIDAD

Una vez funcionando la web en local, utilizamos netlify para desplegarla online y acceder a ella desde cualquier navegador sin necesidad de instalar la app.

Se creó un código QR a través de una app online gratuita para poder acceder a la URL de la APP WEB con más facilidad.

<https://inspiring-payne-53333a.netlify.app/>



UNIT 5. MODELADO FINAL

5.9. Interfaz de usuario (interface)

TEST APLICACIÓN

Una vez que implementamos la aplicación en la Web y se podía acceder a ella desde diferentes dispositivos realizamos múltiples pruebas de reconocimiento de colores con resultados muy satisfactorios.

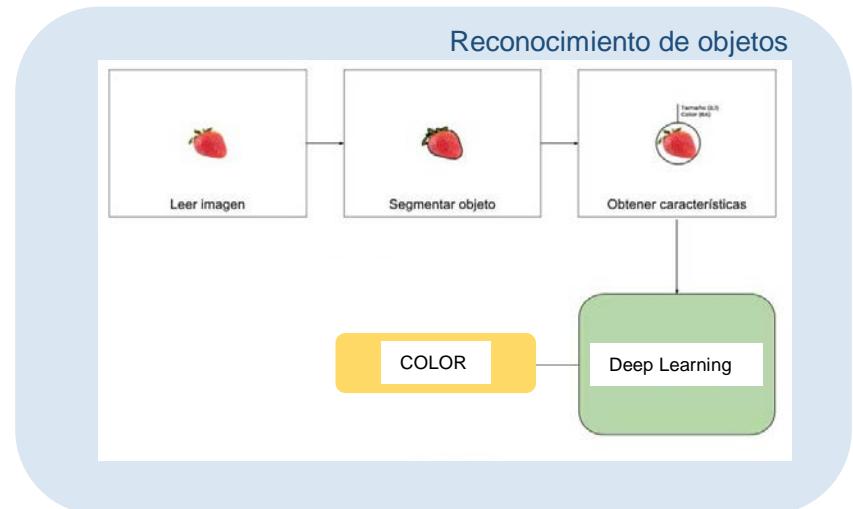


UNIT 5. MODELADO FINAL

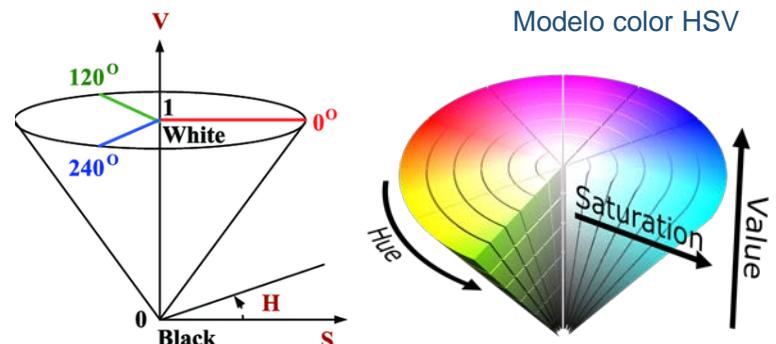
5.10. Propuestas de mejora

MODELO e INTERFACE

En un futuro se podría intentar mejorar el modelo implementando una función específica de detección por áreas de la imagen, para mejorar la detección de los colores cuando se trata de un objeto sobre un fondo.



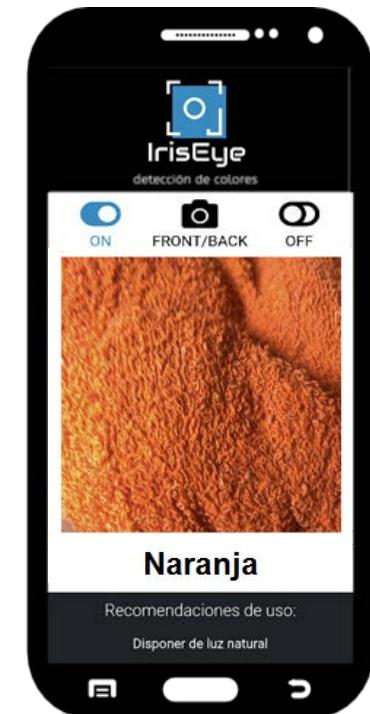
Otra posible mejora, sería realizar pruebas con otros modelos de color como HSV, para ver si se reduce la influencia de las condiciones de iluminación a la hora de detectar el color, realizando una separación de las clases basada en rangos de valores para cada uno de los diferentes parámetros (Hue, Saturation, Value).



5.10. Propuestas de mejora

■ MODELO e INTERFACE

A nivel de interacción con el usuario, nos gustaría lograr que la información sobre la clasificación no se mostrase sólo de manera gráfica en pantalla, sino a través de un audio, lo que haría mejorar la accesibilidad de la aplicación de cara a personas con deficiencias visuales.



UNIT 5. MODELADO FINAL

5.11. Logros y beneficios

COMPARTIENDO RESULTADOS

Hemos creado una aplicación web accesible a todos, la hemos publicado en un repositorio de **GitHub** y desplegado en **Netlify**.

Pretendemos con ello, un uso amigable y gratuito de nuestra aplicación para todo aquel que la necesite o que pudiera serle de ayuda.



UNIT 5. MODELADO FINAL

5.12. Conclusiones finales

CONCLUSIONES

Este proyecto nos ha ayudado a comprender el potencial que tiene la inteligencia artificial para cambiar y ayudar a las personas en su vida cotidiana.

Ha sido un curso muy intenso, donde hemos adquirido muchísimos conocimientos sobre los que poder seguir investigando el desarrollo y aplicación tanto de machine learning como deep learning ya que son campos muy extensos en constante cambio y evolución.

Partir de un simple conjunto de imágenes y llegar a desarrollar una interface que los usuarios puedan utilizar para ayudarles a reconocer los colores ha sido increíble!!



UNIT 5. MODELADO FINAL

5.13. Miembros del equipo



Marina
Jiménez
Egea



Marta
Freire
Panceira



Marta
Trasancos
Yáñez



Lorena
Jiménez
Tejada



Mercedes
Rodríguez
Barbero

El proyecto ha sido una experiencia fantástica de trabajo en equipo, donde nuestras diferentes competencias y ámbitos laborales han enriquecido el resultado aportando una visión multidisciplinar





Together for Tomorrow! Enabling People

Education for Future Generations

©2019 SAMSUNG. All rights reserved.

Samsung Electronics Corporate Citizenship Office holds the copyright of book.

This book is a literary property protected by copyright law so reprint and reproduction without permission are prohibited.

To use this book other than the curriculum of Samsung innovation Campus or to use the entire or part of this book, you must receive written consent from copyright holder.