MIARFID Máster en Inteligencia Artificial, Reconocimiento de Formas e Imagen Digital

José Javier Calvo Moratilla
2021/2022

Visión por computador

Prácticas laboratorio

Introducción

Para la realización de los laboratorios de la asignatura se han seleccionado cuatro ejercicios diferentes del temario de la asignatura:

1. Gender Classification by Image
2. Eyes Vein Segmentation
3. Colorize Gray Images
4. Car Model Classification

1. Gender Classification by Image

En el primer ejercicio de laboratorio se realiza una clasificación del género mediante el uso de imágenes. Se disponen de dos conjuntos, uno de train con unas dimensiones de (10585, 100, 100, 3) y uno de test con (2648, 100, 100, 3), que suman un total de 13233 imágenes para el experimento

Figura 01. Imágenes dataset

Primero se realiza un análisis para ver si las clases están desbalanceadas, en dicho sentido se obtienen los siguientes resultados:

	Hombres	**Mujeres**
X_train	8204	2381
X_test	2052	596

Se observa claramente que las clases están desbalanceadas, pero el desbalance no influye negativamente en el entrenamiento, por ello se realiza un experimento con todos los datos disponibles.

En primer lugar se define un Data Augmentation basado en la librería “Albumentations” para reducir el sobreaprendizaje en el entrenamiento. Los parámetros se eligen como recomendación de la propia librería para una clasificación en imágenes:

Parámetro	**Valores**
Horizontal Flip	p=0.5
Sift Scale Rotate	p=1.0, shift_limit=(-0.05, 0.05), scale_limit=(-0.05, 0.05), rotate_limit=(-10, 10), border_mode=4
RandomBrightnessContrast	p=0.2
Channel Dropout	Valores por defecto

En segundo lugar se define los callbacks utilizados en los proyectos de la asignatura de redes neuronales:

Callback	Parámetros
ReduceLROnPlateau	monitor='val_loss', factor=0.2, patience=5, min_lr=0.001
ModelCheckpoint	filepath=filepath, verbose=1, save_best_only=True
EarlyStopping	patience= 20, restore_best_weights=True, monitor="val_accuracy"

El modelo elegido ha sido ResNet50V2, utilizado con anterioridad.

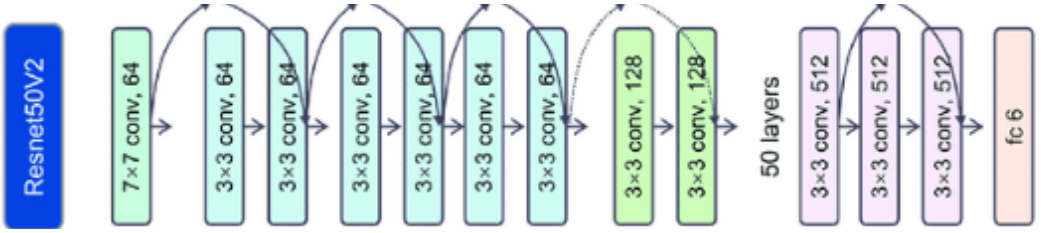


Figura 02. Arquitectura modelo Resnet50V2

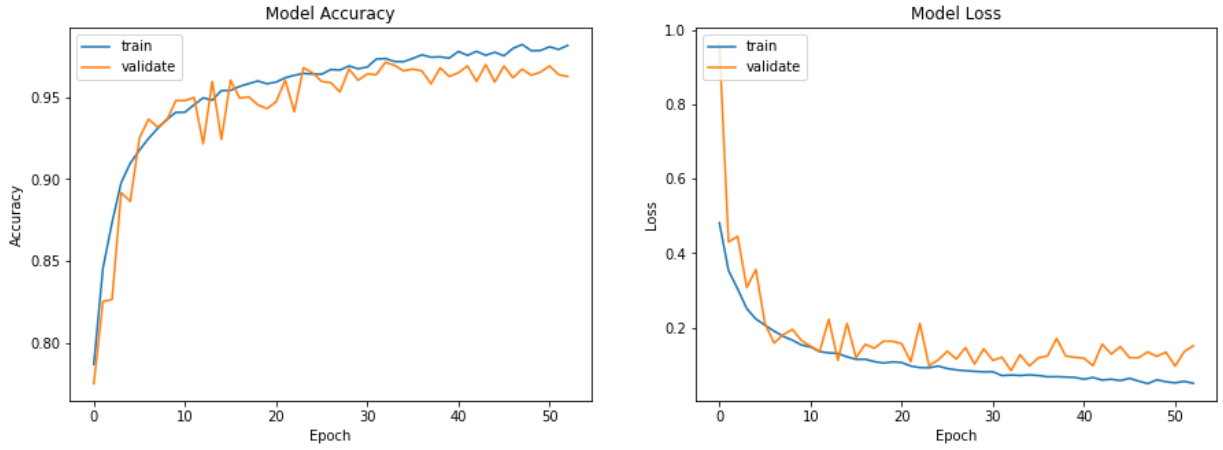
Para poder realizar la tarea se aplica un pequeño cambio a la salida de la red, donde se añade una densa de dos unidades, junto a una función de activación sigmoide, ya que las etiquetas tienen el formato “One hot encoding”.

Previamente se prueban diferentes parámetros para el “Data Augmentation”, definiendo los que mejoran el rendimiento para el experimento.

Una vez definidos los cambios se ejecuta el entrenamiento con los siguientes parámetros:

Parámetro	**Valor**
batch_size	128
epochs	100
núm classes	2
opt	"adam"

El experimento se realiza con las clases no balanceadas, obteniendo los siguientes resultados:




```
Test loss: 0.08395407348871231
Test accuracy: 0.9712991118431091
```

Los resultados han sido bastante óptimos, ya que se ha podido contener el sobreaprendizaje gracias al uso del “Data Augmentation”, logrando un 0.9713 de accuracy con el modelo Resnet50V2. Con dicho resultado se cumplen las exigencias mínimas como objetivo al utilizar una red con una cantidad alta de parámetros para entrenar.

2. Eyes Vein Segmentation

El segundo ejercicio realizado para los laboratorios de la asignatura consiste en la tarea de segmentación de las venas en ojos.

Inicialmente se dispone de una imagen de ojos tomada por equipos de electromedicina ocular con unas dimensiones de (584, 584, 3).

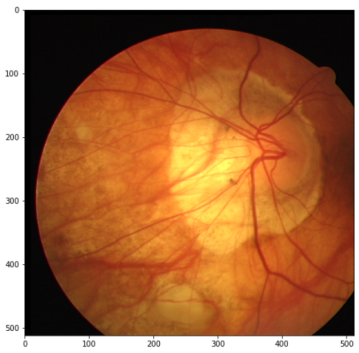


Figura 03. Imagen ocular

Aplicar en la tarea un “Data Augmentation” puede ayudar al rendimiento del modelo, ya que al utilizar sólo datos de entrenamiento va a sobreentrenar y se precisa de alguna técnica para intentar minimizar dicho problema. En dicho caso se utiliza la herramienta “Albumentations”, ya que permite utilizar transformaciones que no afectan a la correlación de las imágenes con su correspondiente máscara. Se prueba cambiar el brillo de las imágenes.

Para la tarea se ha implementado la red U-Net planteada por la asignatura realizando unas pequeñas modificaciones. Inicialmente el modelo no dispone de conexiones entre la zona de codificación y decodificación, por ello se prueba de incluir “Skip Connections” para mejorar el rendimiento en la segmentación de imágenes médicas [1].

No obstante, se decide experimentar incluyendo convoluciones en las “Skip Connections” para observar el rendimiento del modelo de arquitectura “UNet” para dicha tarea.

```
def ConcatBlock(layer, num, filters):
  l=layer

  for i in range(num):
    l=Conv2D(filters, (1, 1), padding='same')(l)
    l=Activation('relu')(l)
    l=Conv2D(filters, (1, 1), padding='same')(l)
    l=Activation('relu')(l)

  l=UpSampling2D((2,2))(layer)
  return l
```

Figura 04. Skip Convolutional Connection

Se define la arquitectura tomada como referencia para la implementación de la solución:

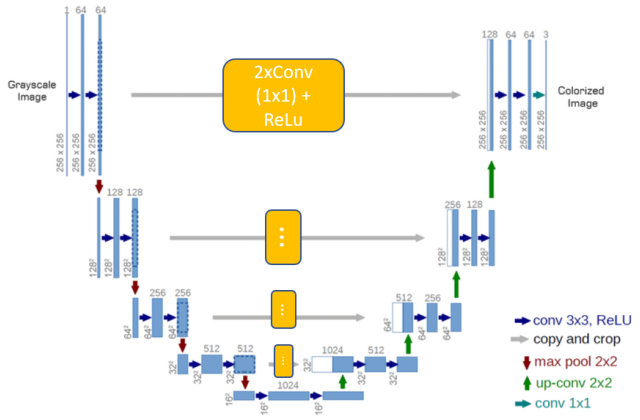


Figura 05. U-Net Model, skip connections, convolucionadas

El modelo se ejecuta con los siguientes parámetros:

Parámetro	**Valor**
Epochs	800
Learning rate	0.001
Optimizador	Adam
Métrica	Mean Squared Error

Una vez ejecutadas todas las epoch se obtienen los siguientes resultados:

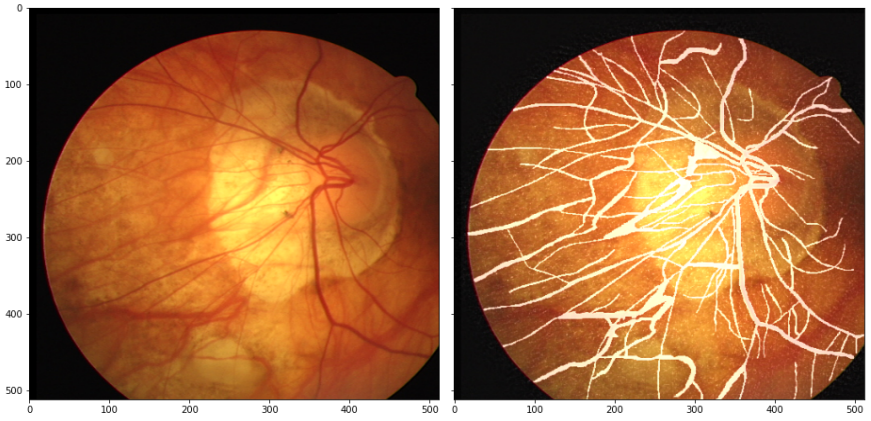


Figura 06. Resultados Segmentación U-Net

Resultado	0.0046 MSE

Todas las imágenes se han utilizado para el entrenamiento, por ello la imagen utilizada para la segmentación forma parte del conjunto de entrenamiento, pudiendo mostrar sobreaprendizaje en la predicción realizada, pero ha servido para poder mostrar el resultado en una tarea de segmentación mediante el uso de una red U-Net.

3. Colorize Gray Images

En el presente ejercicio se implementa la solución para la colorización de imágenes grises a color.

En primer lugar se ha buscado un dataset con imágenes a color en la páginas “Kaggle”, utilizando un subconjunto reducido de imágenes, ya que no se ha podido cargar una gran cantidad por las limitaciones de memoria que presenta la suscripción de Google Colab Pro. [2]

Figura 07. Imagen Dataset.

Una vez cargadas las imágenes se han redimensionado para la tarea, ya que el modelo está implementado para recibir a la entrada un array con unas dimensiones de (256, 256), siendo la dimensión de las imágenes original del dataset de (400, 400)

Una vez redimensionadas las imágenes se deben de normalizar para poder representar los colores en RGB.

Una vez normalizadas las imágenes se ejecuta una función para realizar la transformación RGB a (l_a_b), dicha transformación es necesaria para poder entrenar el modelo, por una parte el canal de brillo “l”que por una parte genera unas características para poder predecir el color de una imagen, reduciendo el MSE, junto a las características generadas por la red Inception, que ha sido pre entrenada con el dataset ‘imagenet’ para identificar lo que aparece en la imagen.

Al concatenar dichas características se puede entrenar una red que predice los colores que debe de tener una imagen en gris para ser coloreada, reduciendo el MSE obtenido.

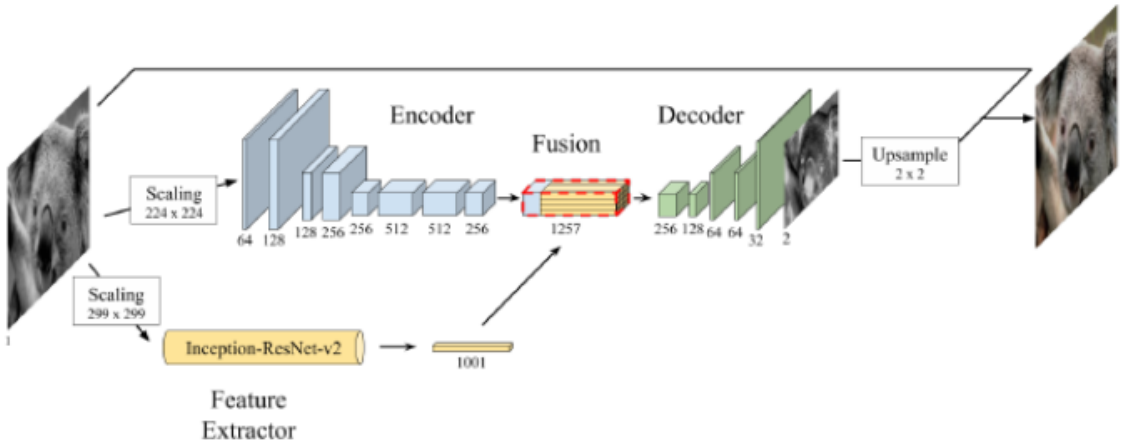


Figura 08. Arquitectura Deep Koalarization: Image Colorization using CNNs and Inception-Resnet-v2 [3]

En primer lugar se realiza un experimento con los siguientes parámetros:

Parámetro	Valor
Epochs	3000
Steps per epoch	5
imágenes de entrenamiento	5000

Uno de los inconvenientes más significativos ha sido la ocupación de memoria RAM en el sistema

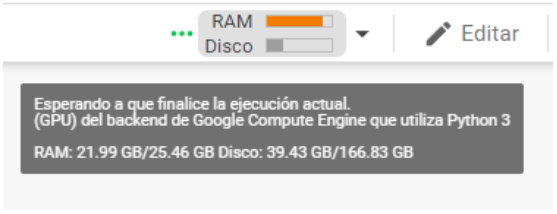


Figura 09. Ocupación alta de memoria RAM.

Cuando se cargan las imágenes de test el cuaderno se queda sin memoria y se reinicia, así que se opta por cargar los batches directamente desde la carpeta, sin la necesidad de cargar todas las imágenes de golpe, pudiendo ejecutar la el experimento, obteniendo los siguientes resultados:

Figura 10. Resultados imagen coloreada.

Como se observa en la imagen no se ha podido lograr una colorización de la imagen con los parámetros seleccionados.

4. Car Model Classification

En último lugar se implementa la solución para clasificar veinte modelos diferentes de coche mediante el uso de una red neuronal bilineal.

Para realizar la clasificación primero se cargan imágenes de modelos de coches de un dataset base aportado en el enunciado:

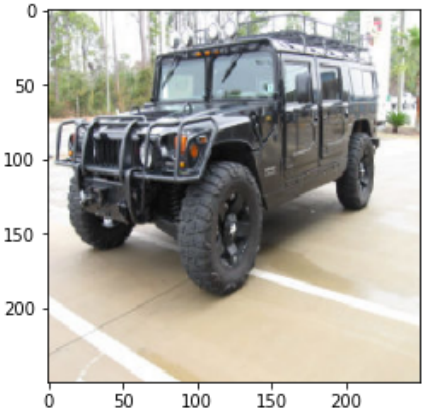


Figura 11. Muestra imagen modelo de coches.

Como modelo se plantea utilizar la red convolucional VGG16:

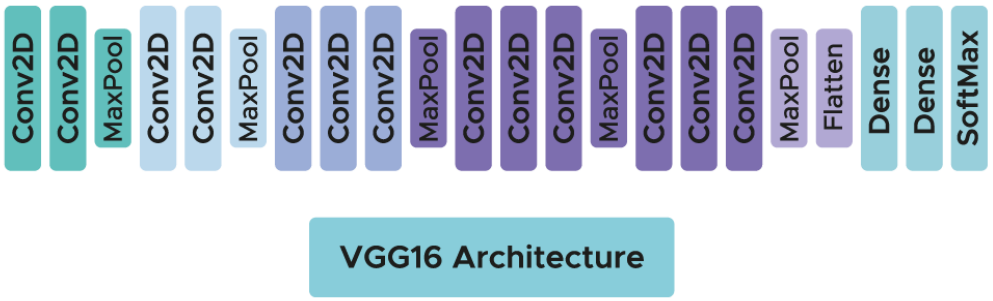


Figura 12. Arquitectura VGG16

Para la tarea se implementa una arquitectura bilineal en la que se utilizan dos arquitecturas idénticas en paralelo, dicha configuración es la elegida para realizar los experimentos correspondientes.

Se realiza el Data Augmentation en los datos para poder reducir el sobreaprendizaje y contribuir al correcto entrenamiento de la red, dado que al iniciar los experimentos se ha observado que el entrenamiento sobreentrenaba la red. Se ha intentado aplicar un Data Augmentation más duro para ver si se puede reducir el sobreaprendizaje.

Parámetro	**Valores**
Horizontal Flip	True
width_shift_range	0.5
height_shift_range	0.5
rotation_range	40
zoom_range	[1.0,1.2]
shear_range	0.3
fill_mode	'nearest'

Inicialmente se disponen sólo el conjunto de entrenamiento y test, por ello el conjunto de test se divide para disponer de un conjunto de validación con la mitad de los datos.

Para ejecutar la red se siguen las recomendaciones dadas por el enunciado, en las que primero se entrena la red pre entrenada por ‘imagenet’ unas 50 epochs con las

layer ‘no trainables’ con un learning rate bastante alto y después se procede a entrenar la red 250 epochs con las layers ‘trainables’ con un learning rate más reducido.

Parámetro	Valor
Epochs no train	50
Epochs train	250
batch_size	32
num_classes	20
optimizador	Adam {no trainable = 1e-3, trainable =1e-6}

Primero se muestran los resultados dónde se observa el entrenamiento en la fase de 50 epochs dónde las layers de la red bilineal no se entrenan:

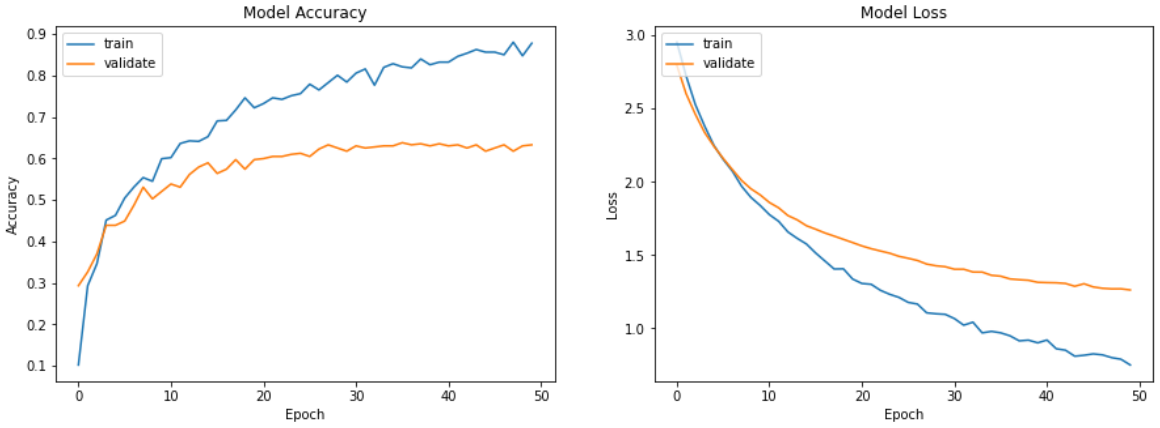


Figura 13. Evolución modelo primera fase

Se ha intentado que la red no sobreentrene excesivamente para pasar a la segunda fase, dónde se obtiene unos resultados del 60% de accuracy de validación para seguir el entrenamiento.

Una vez ejecutadas la primera fase se ejecuta la segunda con las layers descongeladas, observando en la siguiente imagen la evolución final del modelo y los resultados de test:

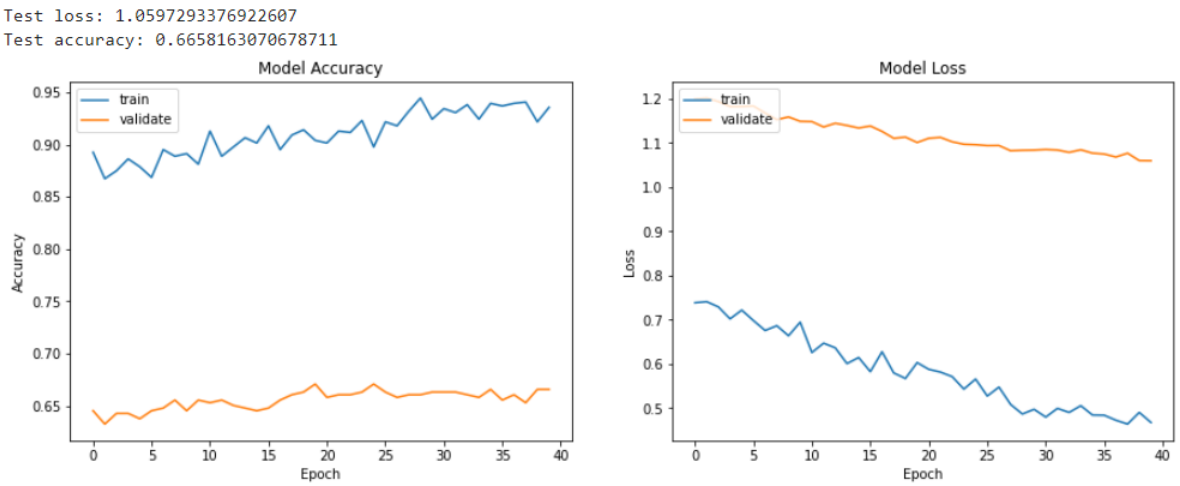


Figura 14. Evolución modelo segunda fase

El resultado obtenido en test es un accuracy de 0.6658% que supera el valor fijado por el enunciado del ejercicio.

Bibliografía

[1] The Importance of Skip Connections in Biomedical Image Segmentation Michal Drozdzal, Eugene Vorontsov, Gabriel Chartrand, Samuel Kadoury and Chris Pal. Imagia Inc., Ecole Polytechnique de Montréal, Université de Montréal, CHUM Research Center, Montreal Institute for Learning Algorithms, Montreal, Canada, 22 Sep 2016.

[2] Dataset Colorize:
https://www.kaggle.com/datasets/aayush9753/image-colorization-dataset

[3] Deep Koalarization: Image Colorization using CNNs and Inception-Resnet-v2. Federico Baldassarre KTH Royal Institute of Technology, Diego González Morín KTH Royal Institute of Technology, Lucas Rodés-Guirao {fedbal, diegogm, lucasrg} @kth.se Equal contributionKTH Royal Institute of Technology. 2017.