Histopathologic Cancer Detection

PROYECTO FINAL CURSO DEEP LEARNING

Iván Darío Gómez Marín.
C.C: 1.041.147.729

Docente: **Raúl Ramos Pollán**

UNIVERSIDAD
DE ANTIOQUIA
1 8 0 3

Facutad de ingeniería
Universidad de Antioquia

11 de noviembre de 2023

1 Proyecto Final

1.1. Contexto de la aplicación

La aplicación propuesta proviene de la competencia publicada en la plataforma Kaggle, que puede encontrarse en el siguiente link: histopathologic-cancer-detection.
El reto de la competencia consiste en **crear un algoritmo para identificar metástasis en pequeños parches de imágenes tomadas de exploraciones patológicas digitales de mayor tamaño**. Los datos para esta competición son una versión ligeramente modificada del conjunto de datos de referencia PatchCamelyon (PCam).PCam convierte la tarea clínica de detección de metástasis en una sencilla tarea de clasificación binaria de imágenes, similar a CIFAR-10 y MNIST.

El objetivo es crear un modelo de clasificación binario que dada una imagen permita discriminarla, indicado si hay presencia de tejido cancerígeno o no; 1 indica la presencia de tejido cancerígeno en la imagen y 0 la ausencia de este.

1.2. Dataset

El conjunto completo de imágenes disponible para el entrenamiento y validación del modelo, se puede acceder a través del siguiente link: data kaggle competition.
El conjunto de datos consiste en imágenes microscópicas de tejido de ganglios linfáticos. Cada imagen tiene una resolución de 96x96 píxeles, y la tarea consistirá en identificar tejido canceroso metastásico en una región central de la imagen de 32x32 píxeles. Según la descripción de la competencia de Kaggle, la identificación de al menos 1 píxel de tejido tumoral etiquetaría efectivamente la imagen como positiva, es decir, con cáncer. El conjunto de datos original de entrenamiento consta de **220.025 imágenes**.
Nota: *En nuestro caso de estudio, por cuestiones pedagógicas y por temas de capacidad computacional, haremos uso de un subconjunto de* ***50.000 imágenes****. En el conjunto de datos original, aproximadamente el 60 % de las imágenes no contienen tejido cancerígeno; el subconjunto de imágenes elegido aleatoriamente para el entrenamiento de los modelos contiene aproximadamente la misma proporción.*

La figura 1.1 muestran 3 imágenes elegidas aleatoriamente del dataset con su respectiva etiqueta (**1:** tejido cancerígeno; **0:** Tejido sin cáncer).

Figura 1.1: Imágenes ejemplo

1.3. Estructura del Notebook

La entrega final solo contiene un notebook llamado **1. Arquitectura de linea de base** debido a que los datos proporcionados por la competencia Kaggle no requerían preprocesamiento alguno. Este notebook contiene las siguientes secciones:

1. **ENCABEZADO:** Aquí se encuentran las subsecciones que sirven de introducción al problema a modelar, definición dle objetivo y métricas del modelo de machine learning, origen y disponibilidad conjunto de datos; las subsecciones son:
   - ***1.1 Contexto de la aplicación***
   - ***1.2 Objetivo de machine learning***
   - ***1.3 Conjunto de datos***
   - ***1.4 Métricas de desempeño***
2. **DATASET:** En esta sección se encuentra divida de la siguiente forma:
   - ***2.1 Carga del dataset:*** Aquí se cargan las librerías requeridas en el notebook y se hace uso de la API de Kaggle para descargar el conjunto de imágenes de la web de la competencia.
   - ***2.2 Generación del dataset para entrenamiento:*** En esta sección primero se define la función para extraer la muestra aleatoria del conjunto de imágenes original y posteriormente se extrae la muestra de 50.000 imagenes.
   - ***2.3 Visualización:*** En esta sección se visualizan 12 imágenes del subconjunto elegido previamente, adicionalmente se muestra la proporción de imágenes de cada clase que contiene el dataset elegido para entrenar y testear los modelos.
3. **MODELAMIENTO:** En esta sección se empiezan a construir los modelos y se encuentra divida así:
   - ***3.1 Definiciones previas:*** En esta subsección se encuentra a su vez dividida en:
     - *3.1.1 Funciones auxiliares:* Acá se crean tres funciones que sirven para graficar las métricas del modelo durante el entrenamiento, las curvas ROC y la matriz de confusión de cada modelo.
     - *3.1.2 Definición de métricas:* Haciendo uso de la API de Tensorflow se crea una lista con las métricas que serán monitoreadas durante el entrenamiento del modelo y también se define un Callback de tipo EarlyStopping para evitar el overfitting de los modelos.

    - *3.1.3 División del dataset:* Se divide el conjunto de datos en dos subconjuntos: uno para entrenamiento y otro para testeo en las proporciones habituales 80:20; posteriormente se visualizan la cantidad de imágenes que contiene cada subconjunto de cada clase.
    - *3.1.4 Creación de dataloaders:* Se crean dos funciones que hacen uso de la librería *tensorflow.keras.preprocessing.image* para cargar las imágenes a partir de un archivo csv que contiene las etiquetas y los nombres de las imágenes.
- ***3.2 Modelo 1: Modelo base:*** Aquí se crea el modelo inicial que sirve de base para la creación de otros modelos posteriormente en el análisis. Esta sección está conformada por:
    - *3.2.1 Arquitectura del modelo*
    - *3.2.2 Entrenamiento del modelo*
    - *3.2.3 Métricas del modelo*
- ***3.3 Modelo 2: Set Weights:*** Se propone un modelo con la misma arquitectura del creado en la sección 3.2 pero aquí ajustamos los pesos de cada clase en la muestra para lidiar con el desbalanceo en el conjunto de entrenamiento. La sección contiene:
    - *3.3.1 Arquitectura del modelo*
    - *3.3.2 Definición de los pesos*
    - *3.3.3 Entrenamiento del modelo*
    - *3.3.4 Métricas del modelo*
- ***3.4 Modelo 3: dataset balanceado:*** Se propone una red con la arquitectura del modelo base y se ajusta a un dataset que contiene el mismo número de imágenes con y sin cáncer. Esta sección se subdivide en:
    - *3.4.1 Elección de la muestra*
    - *3.4.2 Entrenamiento del modelo*
    - *3.4.3 Métricas del modelo*
- ***3.5 Modelo 4: Transfer Learning ResNet50:*** Se hace uso del proceso de transfer learning a partir del modelo Inception ResNet V2 y se propone uno nuevo que contiene el 50 % de la cantidad de datos de entrenamiento usado en los modelos anteriores; la sección se divide en:
    - *3.5.1 Elección de la muestra*
    - *3.5.2 Arquitectura del modelo*
    - *3.5.3 Entrenamiento del modelo*
    - *3.5.4 Métricas del modelo*

1.4. Descripción de la solución

El entorno de ejecución recomendado en Google Colab para la ejecución del notebook se muestra en la gráfica 1.2.La ejecución del notebook se hizo bajo la versión 2.14.0 de tensorflow. Se recomienda usar esta versión para que sea posible importar la métrica *tf.keras.metrics.F1Score.*

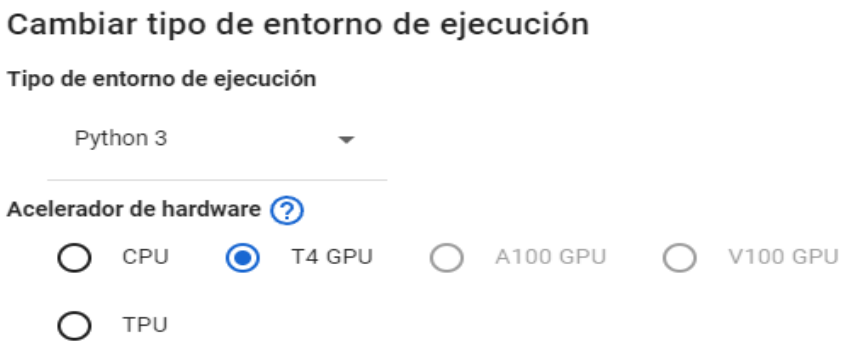


Figura 1.2: Entorno recomendado Google Colab

El interés se centró inicialmente en proponer modelos que ayudaran a manejar el desbalanceo de clases presente en la muestra seleccionada y posteriormente en el uso de la técnica de transfer learning para reducir la necesidad de un conjunto de imágenes mayor y a la vez reducir tiempos de computo. En la gráfica 1.3 se puede apreciar que el conjunto de datos elegido contiene 20.083 imágenes con tejido cancerígeno y 29.917 con tejido sano (el conjunto de entrenamiento y testeo conservan la misma proporción). En la literatura este desbalanceo no es considerado grave sin embargo se exploraron diferentes metodologías para intentar crear modelos capaces de generalizar correctamente a ambas clases.

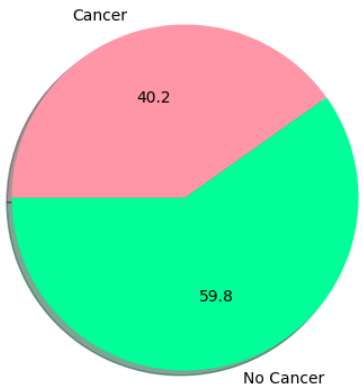


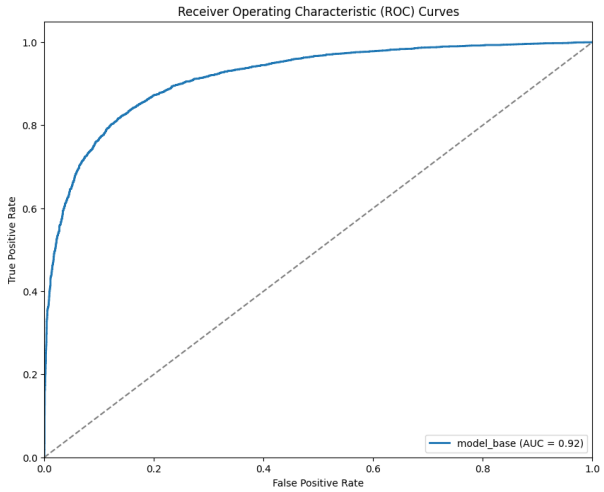
Figura 1.3: Proporción de imágenes por clase

1.4.1. Modelo base

Se propuso un modelo base con la arquitectura mostrada en la gráfica 1.4 la cual tiene 544.561 parámetros y se ajustó para ser entrenado durante 15 epochs. La gráfica 1.5 muestra las métricas obtenidas por el modelo durante el entrenamiento; Este modelo solo se entrenó durante 8 epochs debido a que el *EarlyStopping* ajustado detectó que la función de perdida no disminuía tal como si lo hacia en el conjunto de entrenamiento e interrumpió el proceso impidiendo así un posible sobreajuste. Podemos apreciar que el recall para la clase 1 es más bajo que el de la clase 0, indicando que la discriminación de este tipo de imágenes conlleva un reto mayor. A continuación se proponen otros modelos con la intención de mejorar esta métrica para las imágenes con cáncer.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 96, 96, 16)        448

 max_pooling2d (MaxPooling2  (None, 48, 48, 16)        0
 D)

 conv2d_1 (Conv2D)           (None, 46, 46, 64)        9280

 max_pooling2d_1 (MaxPoolin  (None, 23, 23, 64)        0
 g2D)

 conv2d_2 (Conv2D)           (None, 21, 21, 16)        9232

 max_pooling2d_2 (MaxPoolin  (None, 10, 10, 16)        0
 g2D)

 flatten (Flatten)           (None, 1600)              0

 dense (Dense)               (None, 300)               480300

 dropout (Dropout)           (None, 300)               0

 dense_1 (Dense)             (None, 150)               45150

 dense_2 (Dense)             (None, 1)                 151

=================================================================
Total params: 544561 (2.08 MB)
Trainable params: 544561 (2.08 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Figura 1.4: Arquitectura modelo base



(a) Curva ROC modelo base

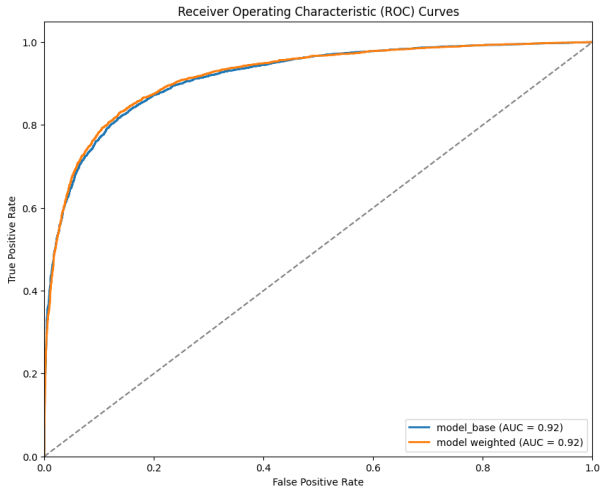
	precision	recall	f1-score	support
0	0.84	0.91	0.88	5993
1	0.85	0.75	0.79	4007
accuracy			0.85	10000
macro avg	0.85	0.83	0.84	10000
weighted avg	0.85	0.85	0.84	10000

(b) Reporte clasificación del modelo base

Figura 1.5: Resultados modelo base

1.4.2. Modelo 2: Set weights

Este modelo conserva la arquitectura de la red base (ver gráfica 1.4) pero en esta ocasión se asignaron pesos de forma inversamente proporcional a la cantidad de imágenes de cada clase en la muestra de entrenamiento. Las métricas del modelo se pueden apreciar en 1.6. Allí se puede apreciar que este modelo tiene un AUC igual que el modelo base pero discrimina mejor las imágenes con cáncer por lo que es ligeramente superior al modelo base.



(a) Curva ROC modelo 2

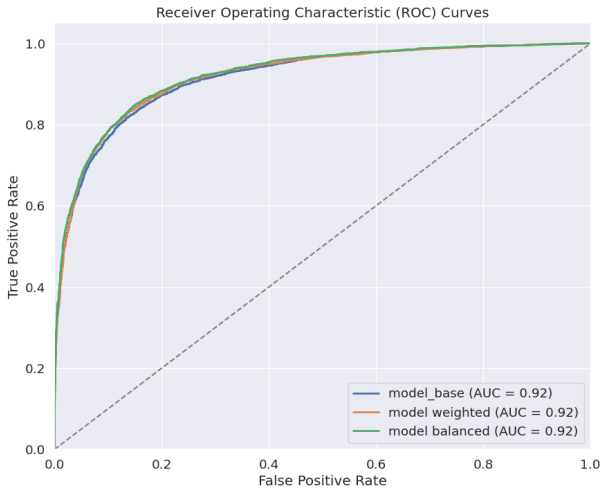
	precision	recall	f1-score	support
0	0.87	0.88	0.88	5993
1	0.81	0.81	0.81	4007
accuracy			0.85	10000
macro avg	0.84	0.84	0.84	10000
weighted avg	0.85	0.85	0.85	10000

(b) Reporte clasificación modelo 2

Figura 1.6: Resultados modelo 2

1.4.3. Modelo 3: dataset balanceado

Este modelo conserva la arquitectura de la red base (ver gráfica 1.4) pero el dataset de entrenamiento contiene el mismo número de imágenes con y sin cáncer. Las métricas del modelo se pueden apreciar en 1.7. Acá podemos apreciar que el AUC de los tres modelos es el mismo pero a diferencia de los anteriores modelos, éste tiene un mayor recall en la clase 1 que implicó una leve reducción en la precisión.



(a) Curva ROC modelo 3

	precision	recall	f1-score	support
0	0.89	0.85	0.87	5993
1	0.79	0.85	0.82	4007
accuracy			0.85	10000
macro avg	0.84	0.85	0.84	10000
weighted avg	0.85	0.85	0.85	10000

(b) Reporte clasificación modelo 3

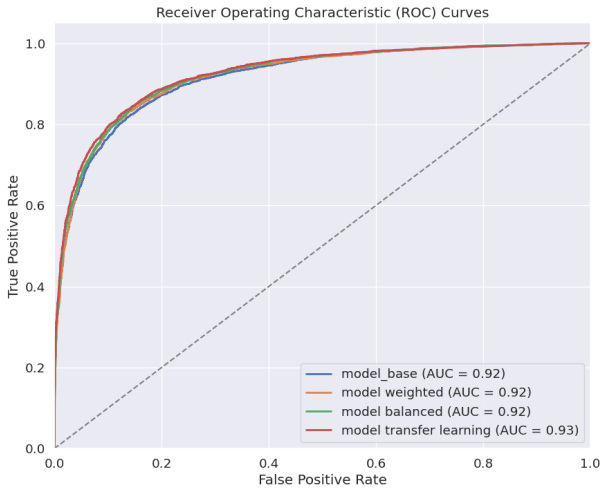
Figura 1.7: Resultados del modelo 3

1.4.4. Modelo 4: transfer learning

Este es un modelo que posee una arquitectura diferente a los tres primeros; en este caso se hizo uso del modelo pre entrenado **Inception Resnet V2**. La arquitectura del modelo propuesto se puede apreciar en la gráfica 1.8; allí se aprecia que el modelo tiene 55'975.574 parámetros de los cuales solo 100.301 fueron entrenados. Este modelo obtuvo un AUC superior a los tres modelos anteriores y proporciona además una reducción considerable en el tiempo de ejecución requerido respecto al modelo base y requiere un dataset mucho más pequeño haciéndolo muy eficiente en el uso de recursos.

```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 keras_layer (KerasLayer)    (None, 1001)              55875273

 dense_9 (Dense)             (None, 100)               100200

 dense_10 (Dense)            (None, 1)                 101

=================================================================
Total params: 55975574 (213.53 MB)
Trainable params: 100301 (391.80 KB)
Non-trainable params: 55875273 (213.15 MB)
_________________________________________________________________
```

Figura 1.8: Arquitectura modelo transfer learning



(a) Curva ROC modelo 4

	precision	recall	f1-score	support
0	0.88	0.87	0.88	5993
1	0.81	0.83	0.82	4007
accuracy			0.85	10000
macro avg	0.85	0.85	0.85	10000
weighted avg	0.85	0.85	0.85	10000

(b) Reporte clasificación modelo 4

Figura 1.9: Resultados modelo 4

1.4.5. Conclusiones

- El modelo con transfer learning utilizando ResNet Inception V2 fue superior a los otros modelos en términos de métrica AUC sugiriendo que la capacidad de transferir conocimientos de modelos preentrenados puede ser crucial en tareas de clasificación de imágenes microscópicas de ganglios linfáticos.

- Ajustar los pesos inversamente proporcional a la cantidad de imágenes de cada clase tuvo un impacto positivo en la capacidad del modelo para aprender patrones específicos relacionados con las imágenes cancerígenas. Sin embargo, el hecho de que no superara al modelo de transfer learning sugiere que la información aprendida durante el entrenamiento de transferencia puede ser más valiosa que simplemente ajustar los pesos de las muestras.

- La elección del modelo adecuado depende de las necesidades específicas y de los recursos disponibles. Aunque el modelo con transfer learning fue el más efectivo, es importante considerar la complejidad del modelo y la disponibilidad de datos al tomar decisiones prácticas sobre implementación.