Percepción Computacional

Caso grupal 1: Implementación de un filtro espacial o morfológico

Albert Fernandez Lozano

Ander Lanas Alocén

Emili Bota Batlle

María de la Cruz Ramírez Trujillo

Olatz Urrutia Etxebarria

v. 02.02.2020 - 20:00:00

Objetivos

El objetivo de este trabajo grupal es desarrollar un filtro morfológico

Solución Implementada

El objetivo del software desarrollado es facilitar la localizacion de las pequeñas manchas algodonosas, tambien llamadas "cotton wool" spots, causadas por la retinopatía diabética eliminando los vasos sanguineos de la retina.

Además de detectar las pequeñas manchas algodonosas, tambien detecta los "hard exudates"que son los exudados duros son pequeños depósitos de color blanco o blanco amarillento con márgenes definidos. A menudo, aparecen cerosos, brillantes o relucientes. Están ubicados en las capas externas de la retina, en la profundidad de los vasos retinianos. Pueden disponerse como puntos individuales, parches confluentes, láminas, o en anillos o medias lunas que rodean zonas de edema retiniano o grupos de microaneurismas. Ocasionalmente se depositan exudados a lo largo de las venas retinianas.

Retinopatía diabética:

La retinopatía diabética es una enfermedad causada por la diabetes que afecta a los ojos. Esta enfermedad esta causada por altos niveles de azucar en sangre que causan hemorragias o taponamientos en los vasos sanguíneos de la retina y en ocasiones pueden causar la perdida de visión.

Dataset

El dataset utilizado proviene de una base de datos pública que contiene imagenes de pacientes con retinopatía diabética.

Algoritmo del filtro y eleccion de operadores

```
#imports
from skimage import data,io
from skimage.color import rgb2gray
from skimage.morphology import erosion, dilation, black_tophat, disk, diamond
from skimage.filters import gaussian, median, threshold_isodata
from skimage.exposure import equalize_adapthist
import matplotlib.pyplot as plt
import numpy as np
```

La función `plot_comp` es un metodo diseñado para comparar dos imágenes, una versión de la mostrada en la magistral correspondiente a los filtros morfológicos. Facilitamos así la apreciación de las alteraciones sobre la imagen original.

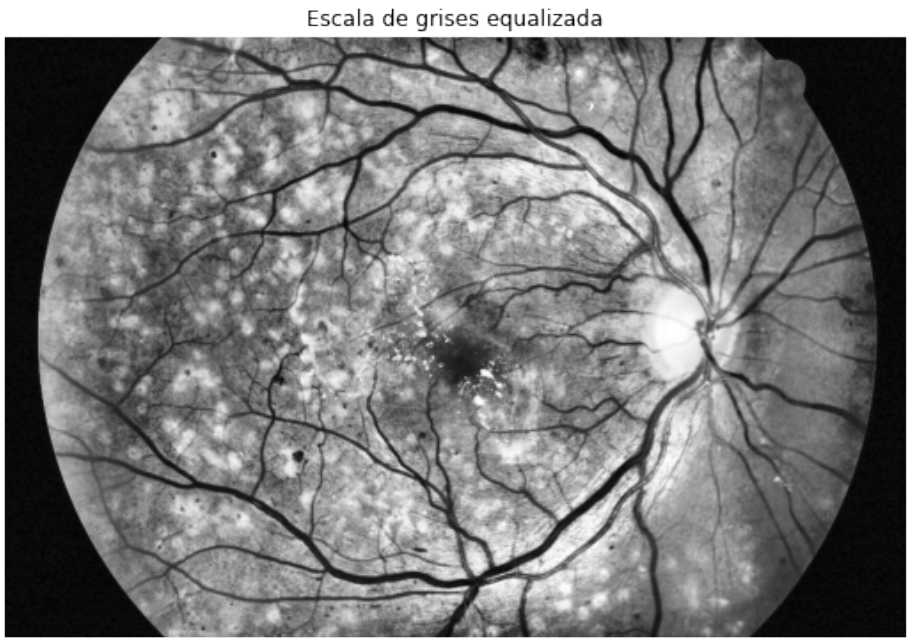
```
def plot_comp(original,filtered,filter_title,orig_title="Imagen Original"):
    """ Comparación de dos imagenes"""
    fig, (ax1, ax2) = plt.subplots(ncols=2, figsize=(20,10), sharex=True, sharey=True)
    ax1.imshow(original, cmap=plt.cm.gray)
    ax1.set_title(orig_title)
    ax1.axis('off')
    ax1.set_adjustable('box')
    ax2.imshow(filtered, cmap=plt.cm.gray)
    ax2.set_title(filter_title)
    ax2.axis('off')
    ax2.set_adjustable('box')
    plt.show()


#Importamos la/s imagen/es que queremos utilizar
images = [[]]
print('Start loading images...')
for i in np.arange(1,11):
    images.append(io.imread(f'retinopatia_diabetica/{i:02d}_dr.jpg'))
print('All images loaded!')

#Seleccionamos una de las imagenes para tratarla
selected = images[8]
```

```
Start loading images...
All images loaded!
```

Una vez importadas las imágenes, habrá que convertirlas a escala de grises para su manipulación. Se aprecia así mejor los retos y objetivos. Para apreciar mejor las manchas provocadas por la retinopatía diabética se deberá eliminar las venas, pero este claramente va a ser el reto del algoritmo. Como se puede ver, las manchas tienen una intensidad parecida a las venas, por lo que una umbralización no simplificaría esta tarea. Habrá que desarrollar métodos compuestos por varios tipos de transformación morfológica para llegar al objetivo propuesto.

Primero se realizó una equalización de la imagen, añadiendo contraste a la imagen en escala de grises. Se resaltan así mejor los puntos provocados por la retinopatía diabética.

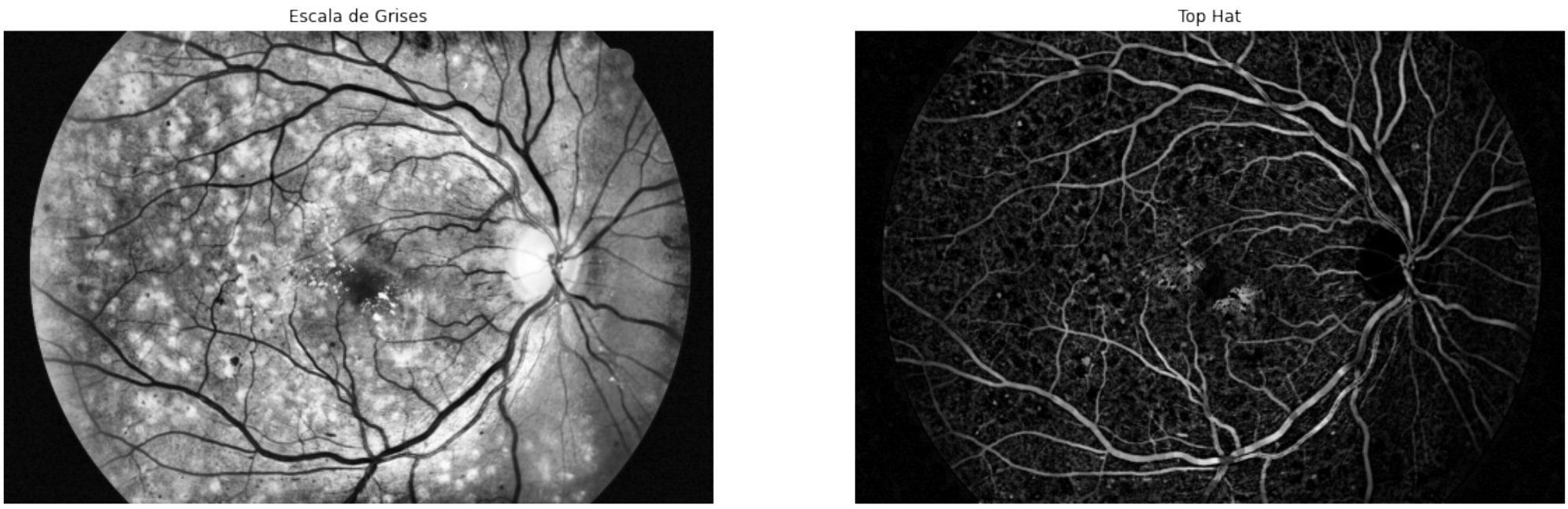
```
#Transformamos la imagen seleccionada a escala de grises
ojo = rgb2gray(selected)
#Equalizamos la imagen en escala de grises para tener un mejor contraste
ojo = equalize_adapthist(ojo, clip_limit=0.03)
plot_comp(selected,ojo,'Escala de grises equalizada')
```



Seguidamente, se aplica la operación morfológica de Top Hat. Esta operación trata de extraer los elementos menos apreciables de la imágen, a través de realizar la diferencia entre la imagen original y un elemento estructurante. En nuestro caso, este elemento se trata de un disco de tamaño de pixel 5. Los resultados se aprecian a continuación, donde se ve todas las ramificaciones y puntos mucho más nítidamente.

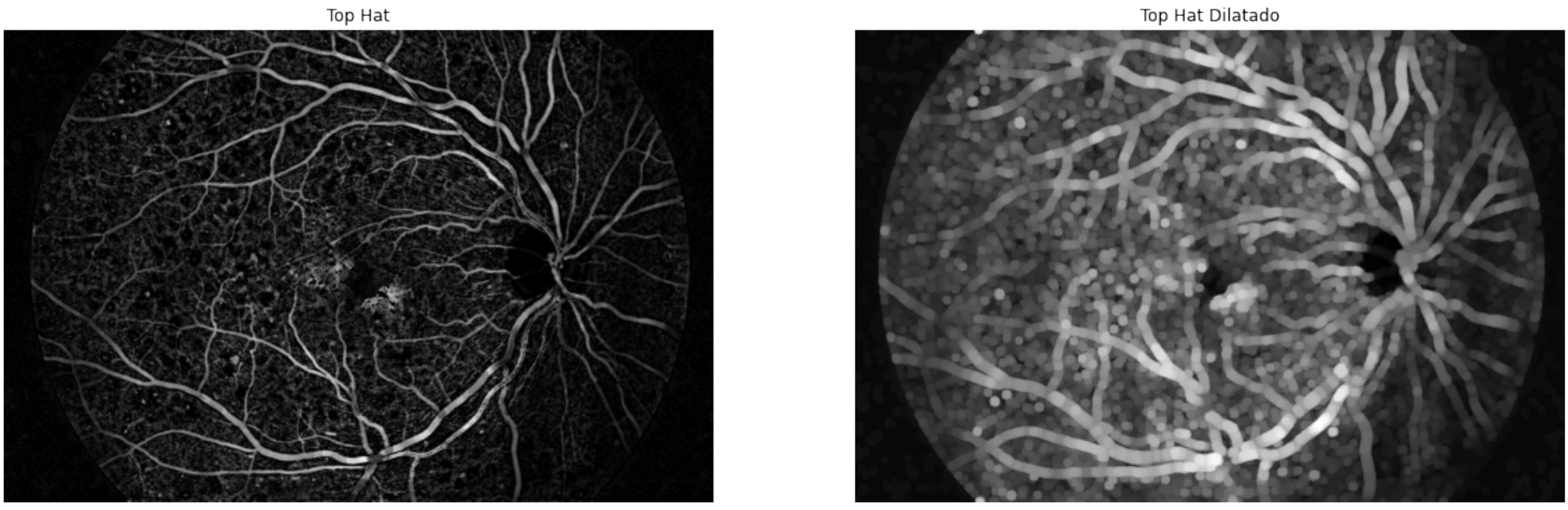
Este paso es fundamental, ya que para extraer la mayor cantidad de venas posible necesitamos extraer su extructura primero.

```
ee = disk(5)
bth = black_tophat(ojo,ee)
plot_comp(ojo,bth,'Top Hat','Escala de Grises')
```



Se aplica un proceso de dilatación para ensanchar las características extraidas con el Top Hat. Esto ayudará para, en procesos posteriores, extraer solo las venas.

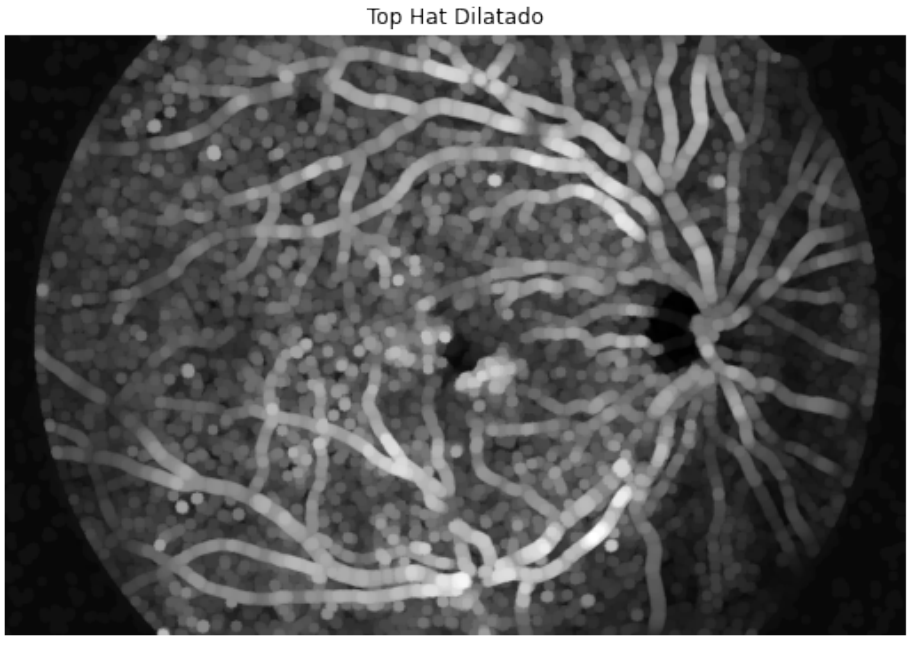
```
dbth = dilation(bth,ee)
plot_comp(bth,dbth,'Top Hat Dilatado','Top Hat')
```



Double-click (or enter) to edit

Se aplica un proceso de umbralización (Thresholded) en binario para obtener sólo las venas y las manchas.

```
thresh_iso = threshold_isodata(dbth)
binary_iso = dbth > thresh_iso
plot_comp(dbth,binary_iso,'Aplicación de Thresholding ISO','Top Hat Dilatado')
```

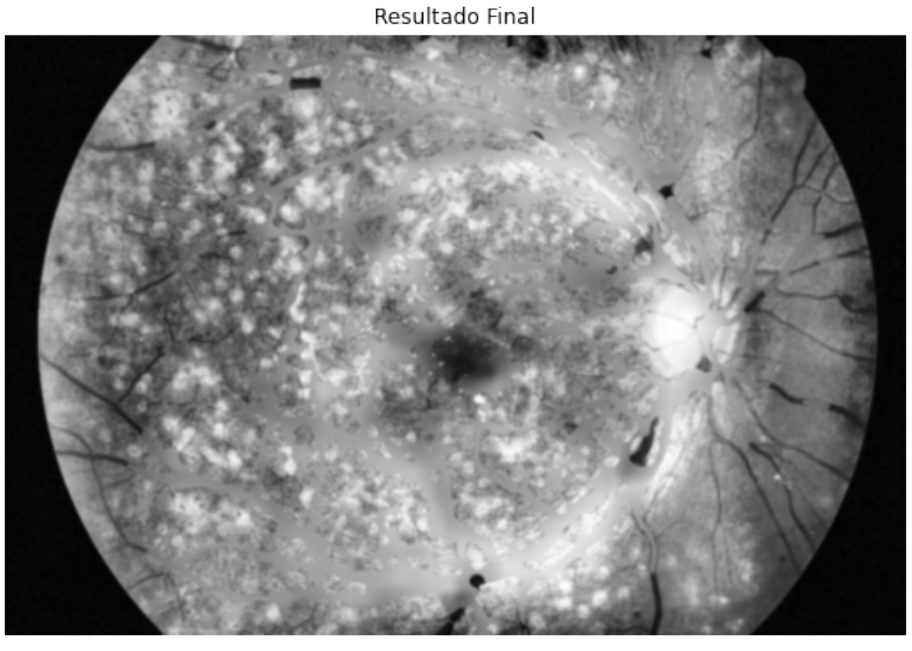


La imagen que obtenemos en la anterior operación obtiene mucho detalles y es difícil distinguir, para simplificar y ver lo que realmente interesa aplicamos la media aritmética de la erosión, que a su vez esta erosión obtiene la media aritmetica de una apertura (erosión seguida de una dilatación), obteniendo un umbral suavizado.

```
dmnds = [diamond(2),diamond(4)]
ebinary = median(erosion(median(dilation(erosion(binary_iso,dmnds[1]),dmnds[0]),dmnds[1]),dmnds[0]))
plot_comp(binary_iso, ebinary,'Threshold suavizado','Threshold ISO')
```

Para aproximarnos más a la hora de detectar diferentes tipos de manchas, se ha realizado un filtro personalizado. Este filtro consiste en devolver la media aritmética de los vecinos a una distáncia menor o igual a `cells` del pixel en la posición `(x,y)` los cuales no contengan una vena; los que si contienen estan enmascarados en la matriz que pasamos (`marray`), así la función `np.mean` no los tiene en consideración al hacer la media aritmética

```
def filtro_custom(marray,x,y,cells=20):
    """Box filter con restricciones; utiliza los vecinos dentro de la submatriz de medida cells*2 que no estén desact
    (lx,ly) = marray.shape
    xr = np.arange(x-cells,x+cells)
    if x < cells: xr = np.arange(x,x+cells)
    elif x >= (lx - (cells+1)): xr = np.arange(x-cells,lx-1)

    yr = np.arange(y-cells,y+cells)
    if y < cells: yr = np.arange(y,y+cells)
    elif y >= (ly - (cells+1)): yr = np.arange(y,ly-1)

    return np.mean(marray[np.ix_(xr,yr)])


#Establecemos lo que será la imagen final con un array de zeros
ojo_final = np.zeros(ojo.shape)

#Obtenemos un producto del binario del threshold con el ojo en escala de grises (asi dejando a 0 los elementos que co
ojo_combinacion = np.multiply(ojo,np.logical_not(ebinary).astype('int'))
#Creamos una mascara para que en la función np.mean no se tengan en cuenta los valores 0
masked = np.ma.masked_equal(ojo_combinacion,0)

for x in range(0,len(ebinary)):
    for y in range(0,len(ebinary[x])):
        ojo_final[x][y] = filtro_custom(masked,x,y) if ebinary[x][y] else ojo[x][y]
ojo_final = gaussian(ojo_final)
plot_comp(selected, ojo_final,'Resultado Final')
```



Como se puede observar en última instancia, el proceso de transformación realizado genera una imagen de salida con los detalles más definidos. El objetivo de esta transformación es la contribución al extenso mundo de la oftalmología, ofreciendo esta herramienta a los profesionales para ayudarlos en la prevención, diagnostico y tratamiento de enfermedades relacionadas directamente con el ojo. Observamos el potencial del tratamiento de imagen en casos como el que nos encontramos, las posibilidades de aplicación són impresionantes. Estas transformaciones bien adecuadas y contrastadas podrían llegar a contribuir a sistemas más complejos, como pueden ser modelos automáticos de detección de enfermedades en etapas tempranas o modelos de soporte para el correcto tratamiento de patologías.

Bibliografía

Lorina Leung, O. D. (s/f). *How does diabetic retinopathy cause vision loss?* Drlleung.com. Recuperado el 2 de febrero de 2021, de https://drlleung.com/blog/f/how-does-diabetic-retinopathy-cause-vision-loss

Pittu, V. P., Avanapu, S. R., & Sharma, J. (2013, enero 1). *Difference between Normal Retina and Diabetic Retinopathy.* Researchgate.net. https://www.researchgate.net/figure/Difference-between-Normal-Retina-and-Diabetic-Retinopathy_fig2_282609747

Boyd, K. (2020). *What is diabetic retinopathy?* Aao.org. https://www.aao.org/eye-health/diseases/what-is-diabetic-retinopathy

Lehrstuhl für Mustererkennung, Friedrich-Alexander-Universität Erlangen-Nürnberg. (s/f). *High-Resolution Fundus (HRF) Image Database.* Fau.de. Recuperado el 2 de febrero de 2021, de https://www5.cs.fau.de/research/data/fundus-images/