

#### ▼ Actividad 4

**Maestría en Inteligencia Artificial Aplicada**  
Curso: Proyecto Integrador  
**Institución:** Instituto Tecnológico de Monterrey  
**Profesor titular:** Dra. Gretchen Barroso Alonso  
**Profesor titular:** Dr. Luis Eduardo Falcon Morales  
**Profesora asistente:** Mtra. Verónica Sandra Guzmán de Valle

Avance V4

### Actividad Modelos alternativos

Nombre del estudiante: María Figueroa Bojano Matrícula: A01114853  
Nombre del estudiante: David Hernández Costellanos  
Matrícula: A01795964  
Nombre del estudiante: Francisco Javier Rímenz Arias Matrícula: A013

```
from google.colab import drive
drive.mount('/content/drive')

#Importando pacotes de DeepLearning, ou obtendo as dependências, via link para o GitHub
!git clone https://github.com/titu1994/DeepLearningFromScratch
```

## ✓ Introducción

El proyecto tiene como objetivo automatizar el reconocimiento morfológico de células sanguíneas a partir de imágenes microscópicas, una tarea que tradicionalmente se realiza en manera manual por profesionales clínicos. En concreto, se pretende desarrollar un sistema computacional capaz de poder recibir como entrada una imagen digital de una célula sanguínea individual y determinar automáticamente a qué tipo de célula pertenece. El sistema debe ser capaz de distinguir entre ocho tipos de células sanguíneas normales basándose exclusivamente en características morfológicas visibles de la imagen.

#### ▼ Preprocesamiento

Dado que los imágenes de datos son de colores centrados en el rojo y homogéneos, se debe observar la misma herencia, con la menor calidad, para el preprocesamiento de imágenes microscópicas se lleva a cabo una segmentación de nucleo dirigida para las regiones más relevantes de cada célula en sangre. En primer lugar, se aplica una normalización de imágenes en escala de grises para que las imágenes sean más apropiadas para el análisis. Luego, se aplica una transformación de histograma para transformar el espacio HSV, donde se definen masas específicas para identificar el color de la célula con criterios de tono, saturación y brillo, y otra masas más reducida para el citoplasma, restringida a la zona adyacente al núcleo. Estas masas se fusionan para formar una máscara binaria que indica la presencia de una célula. Finalmente, se aplica una máscara binaria dirigida correspondiente a la parte del citoplasma. La máscara se aplica a cada imagen original, generando una segmentación que elimina el fondo y artificiales perturbaciones. Esta imagen segmentada constituye el anel a través del cual se realizan las operaciones de procesamiento de imágenes. La máscara binaria es aplicada a cada imagen original, de modo que se fomrulan las técnicas de distinguir de manera automática entre las tipos de células sanguíneas normales.

#### 1. Segmentación basada en umbrales de color y operaciones morfológicas

```

CORTADAS
=====
# APPLIQUE MASCARA
for i in range(0, len(img)):
    for j in range(0, len(img[0])):
        if img[i][j] >= 255:
            img[i][j] = 255
        else:
            img[i][j] = 0
            mask[i][j] = 0
            mask[i][j] = 1
            img[i][j] = 255
            print("Aplicando mascara")
print("Aplicando mascara finalizada")
print("Finalizando segmentacion universal")

```

```

Finalizando segmentacion universal -maska...
Procesando clase: Symmetric
Clase Symmetric finalizada
Procesando clase: IG
Clase IG finalizada
Procesando clase: Parallel
Clase Parallel finalizada
Procesando clase: planar
Clase planar finalizada
Procesando clase: rectangular
Clase rectangular finalizada
Segmentacion universal finalizada

```

Se presentan ejemplos representativos de la segmentación obtenida para distintas clases celulares. Para cada clase se muestran imágenes originales junto con sus correspondientes imágenes segmentadas, permitiendo evaluar visualmente la identificación del núcleo y la consistencia del proceso de segmentación entre distintos tipos de células.

```

E VISUALIZAR EJEMPLOS DE SEGMENTACION (MULTICLASE)
=====
import os
import random
import cv2
import numpy as np
from skimage import io, img_as_ubyte
from skimage.segmentation import watershed
from skimage import measure
# CONFIRMAR RUTAS
=====
ruta_imagenes = "/content/drive/MyDrive/UMA/Proyecto Integrador/PROYECTO_FINAL/IMAGENES"
ruta_seguimiento = "/content/drive/MyDrive/UMA/Proyecto Integrador/SEGUIMIENTO"
# ESEGMENTACIONES
=====
nroClases = 3      # Numero de clases a mostrar
img_nroClase = 3   # Imagenes por clase
# E-----#
# SEGUIMIENTO CLASES
# E-----#
clases = [d for d in os.listdir(ruta_imagenes) if os.path.isdir(os.path.join(ruta_imagenes, d))]
clases = random.sample(clases, min(nroClases, len(clases)))
print("Clases seleccionadas:")
for c in clases:
    print(c)
print("Cargando...")
# E-----#
# SEGUIMIENTO
# E-----#
total_clases = nroClases * len(img_nroClase)
fig, axes = plt.subplots(1, total_clases, figsize=(10, 10))
fig.suptitle('Ejemplos de segmentación')
for i in range(total_clases):
    axes[i].imshow(img_nroClase * i)
    axes[i].set_title(f'{clases[i]}')
    axes[i].set_axis_off()
    axes[i].text(0, 0, f'{i+1}/{total_clases}', color='red', fontweight='bold')
# Si solo hay una fila
if total_clases == 1:
    axes = [axes]
    fig = axes[0]
# E-----#
# MOSTRAR IMAGENES
# E-----#
for clase in clases:
    ruta_img_clase = os.path.join(ruta_imagenes, clase)
    ruta_img_clase_seguimiento = os.path.join(ruta_seguimiento, clase)
    imagenes_clase = [f for f in os.listdir(ruta_img_clase) if f.endswith('.png')]
    imagenes_clase_seguimiento = [f for f in os.listdir(ruta_img_clase_seguimiento) if f.endswith('.png')]
    if len(imagenes_clase) > 0:
        continue
    imagenes_clase = random.sample(imagenes_clase, min(len(imagenes_clase), img_nroClase))
    for img_clase in imagenes_clase:
        img_clase = os.path.join(ruta_img_clase, img_clase)
        img_clase_seguimiento = os.path.join(ruta_img_clase_seguimiento, img_clase)
        if not os.path.exists(img_clase_seguimiento):
            continue
        img_clase = cv2.imread(img_clase)
        img_clase_seguimiento = cv2.imread(img_clase_seguimiento)
        if not os.path.exists(img_clase_seguimiento):
            continue
        img_clase_seguimiento = cv2.cvtColor(img_clase_seguimiento, cv2.COLOR_BGR2GRAY)
        img_clase_seguimiento = img_clase_seguimiento / 255
        img_clase_seguimiento = np.where(img_clase_seguimiento > 0.5, 1, 0)
        axes[img_clase_seguimiento].imshow(img_clase)
        axes[img_clase_seguimiento].set_title(f'{clase} {img_clase}')
        axes[img_clase_seguimiento].set_axis_off()
        axes[img_clase_seguimiento].text(0, 0, f'{img_clase}', color='red', fontweight='bold')
        axes[img_clase_seguimiento].text(0, 0, f'{clase}', color='red', fontweight='bold')
        axes[img_clase_seguimiento].text(0, 0, f'{img_clase}', color='red', fontweight='bold')
        axes[img_clase_seguimiento].text(0, 0, f'{clase}', color='red', fontweight='bold')
        axes[img_clase_seguimiento].text(0, 0, f'{img_clase}', color='red', fontweight='bold')
    fig.tight_layout()
plt.show()

```









Construcción de las matrices de imágenes y sus etiquetas numéricas a partir del dataset organizado en carpetas.

```
import os
import time
import random
import numpy as np
import cv2
import glob
from skimage import io
from skimage.feature import graycomatrix, graycoprops
from skimage.color import gray2rgb
from skimage.measure import regionprops, regionprops_tsd
from skimage.measure import label, regionprops_tsd
from skimage.io import imsave
from skimage.util import img_as_float
from skimage.util import img_as_ubyte
from skimage.util import img_as_int
from skimage.util import img_as_bool
from skimage.util import img_as_grey
from skimage.util import img_as_float32
from skimage.util import img_as_float64
from skimage.util import img_as_float16
from skimage.util import img_as_int16
from skimage.util import img_as_int32
from skimage.util import img_as_int64
from skimage.util import img_as_bool8
from skimage.util import img_as_grey8
from skimage.util import img_as_grey16
from skimage.util import img_as_grey32
from skimage.util import img_as_grey64
from skimage.util import img_as_grey128
from skimage.util import img_as_grey256
from skimage.util import img_as_grey384
from skimage.util import img_as_grey768
from skimage.util import img_as_grey1536
from skimage.util import img_as_grey3072
from skimage.util import img_as_grey6144
from skimage.util import img_as_grey12288
from skimage.util import img_as_grey24576
from skimage.util import img_as_grey49152
from skimage.util import img_as_grey98304
from skimage.util import img_as_grey196608
from skimage.util import img_as_grey393216
from skimage.util import img_as_grey786432
from skimage.util import img_as_grey1572864
from skimage.util import img_as_grey3145728
from skimage.util import img_as_grey6291456
from skimage.util import img_as_grey12582912
from skimage.util import img_as_grey25165824
from skimage.util import img_as_grey50331648
from skimage.util import img_as_grey100663296
from skimage.util import img_as_grey201326592
from skimage.util import img_as_grey402653184
from skimage.util import img_as_grey805306368
from skimage.util import img_as_grey1610612736
from skimage.util import img_as_grey3221225472
from skimage.util import img_as_grey6442450944
from skimage.util import img_as_grey12884901888
from skimage.util import img_as_grey25769803776
from skimage.util import img_as_grey51539607552
from skimage.util import img_as_grey10307921504
from skimage.util import img_as_grey20615843008
from skimage.util import img_as_grey41231686016
from skimage.util import img_as_grey82463372032
from skimage.util import img_as_grey164926744064
from skimage.util import img_as_grey329853488128
from skimage.util import img_as_grey659706976256
from skimage.util import img_as_grey1319413952512
from skimage.util import img_as_grey2638827905024
from skimage.util import img_as_grey5277655810048
from skimage.util import img_as_grey10555311620096
from skimage.util import img_as_grey21110623240192
from skimage.util import img_as_grey42221246480384
from skimage.util import img_as_grey84442492960768
from skimage.util import img_as_grey168884985921536
from skimage.util import img_as_grey337769971843072
from skimage.util import img_as_grey675539943686144
from skimage.util import img_as_grey1351079887372288
from skimage.util import img_as_grey2702159774744576
from skimage.util import img_as_grey5404319549489152
from skimage.util import img_as_grey10808639098978304
from skimage.util import img_as_grey21617278197956608
from skimage.util import img_as_grey43234556395913216
from skimage.util import img_as_grey86469112791826432
from skimage.util import img_as_grey172938225823652864
from skimage.util import img_as_grey345876451647305728
from skimage.util import img_as_grey691752903294611456
from skimage.util import img_as_grey1383505806589222912
from skimage.util import img_as_grey2767011613178445824
from skimage.util import img_as_grey5534023226356891648
from skimage.util import img_as_grey11068046453137783296
from skimage.util import img_as_grey22136092906275566592
from skimage.util import img_as_grey44272185812551133184
from skimage.util import img_as_grey88544371625022266368
from skimage.util import img_as_grey17708874325004453272
from skimage.util import img_as_grey35417748650008906544
from skimage.util import img_as_grey70835497300017813088
from skimage.util import img_as_grey141670994600035626176
from skimage.util import img_as_grey283341989200071252352
from skimage.util import img_as_grey566683978400142504704
from skimage.util import img_as_grey1133367956800285009408
from skimage.util import img_as_grey2266735913600570018816
from skimage.util import img_as_grey4533471827200140037632
from skimage.util import img_as_grey9066943654400280075264
from skimage.util import img_as_grey1813388730880560015056
from skimage.util import img_as_grey3626777461760120030112
from skimage.util import img_as_grey7253554923520240060224
from skimage.util import img_as_grey14507109847040480120448
from skimage.util import img_as_grey29014219694080960240896
from skimage.util import img_as_grey58028439388161920481792
from skimage.util import img_as_grey116056878776323840963584
from skimage.util import img_as_grey232113757552647681927168
from skimage.util import img_as_grey464227515105295363854336
from skimage.util import img_as_grey928455030210590727708672
from skimage.util import img_as_grey185691006042118145541744
from skimage.util import img_as_grey371382012084236291083488
from skimage.util import img_as_grey742764024168472582166976
from skimage.util import img_as_grey148552804833694516433392
from skimage.util import img_as_grey297105609667389032866784
from skimage.util import img_as_grey594211219334778065733568
from skimage.util import img_as_grey118842243866955613146736
from skimage.util import img_as_grey237684487733911226293472
from skimage.util import img_as_grey475368975467822452586944
from skimage.util import img_as_grey950737950935644855173888
from skimage.util import img_as_grey1901475901871289610347776
from skimage.util import img_as_grey3802951803742579220695552
from skimage.util import img_as_grey7605903607485158441391104
from skimage.util import img_as_grey1521180721497031682682208
from skimage.util import img_as_grey3042361442994063365364416
from skimage.util import img_as_grey6084722885988126730728832
from skimage.util import img_as_grey1216944571197625346145766
from skimage.util import img_as_grey2433889142395250692291532
from skimage.util import img_as_grey4867778284790501384583064
from skimage.util import img_as_grey9735556569581002768166128
from skimage.util import img_as_grey1947111313916200553633224
from skimage.util import img_as_grey3894222627832400106866448
from skimage.util import img_as_grey7788445255664800213732896
from skimage.util import img_as_grey1557689051132960042746576
from skimage.util import img_as_grey3115378102265920085493152
from skimage.util import img_as_grey6230756204531840170986304
from skimage.util import img_as_grey1246151240906320341977264
from skimage.util import img_as_grey2492302481812640683954528
from skimage.util import img_as_grey4984604963625281367909056
from skimage.util import img_as_grey9969209927250562735818112
from skimage.util import img_as_grey1993841985451120547163624
from skimage.util import img_as_grey3987683970902240104327248
from skimage.util import img_as_grey7975367941804480208654496
from skimage.util import img_as_grey15950735883608960417308992
from skimage.util import img_as_grey31901471767217920834617984
from skimage.util import img_as_grey63802943534435841669235968
from skimage.util import img_as_grey12760588706871168333847936
from skimage.util import img_as_grey25521177413742336667695872
from skimage.util import img_as_grey51042354827484673335391744
from skimage.util import img_as_grey10208470965496946667078348
from skimage.util import img_as_grey20416941930993893334156696
from skimage.util import img_as_grey40833883861987786668313392
from skimage.util import img_as_grey81667767723975573336626784
from skimage.util import img_as_grey16333553546951154667313568
from skimage.util import img_as_grey32667107093852309334627136
from skimage.util import img_as_grey65334214187704618669254272
from skimage.util import img_as_grey13066842835540923733858448
from skimage.util import img_as_grey26133685671081847467716896
from skimage.util import img_as_grey52267371342163694935433792
from skimage.util import img_as_grey10453474268432738967087584
from skimage.util import img_as_grey20906948536865477934175168
from skimage.util import img_as_grey41813897073730955868350336
from skimage.util import img_as_grey83627794147461911736700672
from skimage.util import img_as_grey16725558829492382347340144
from skimage.util import img_as_grey33451117658984764694680288
from skimage.util import img_as_grey66902235317969529389360576
from skimage.util import img_as_grey13380447063938955878672152
from skimage.util import img_as_grey26760894127877911757344304
from skimage.util import img_as_grey53521788255755823514688608
from skimage.util import img_as_grey107043576511511647029377216
from skimage.util import img_as_grey214087153023023294058754432
from skimage.util import img_as_grey428174306046046588117508864
from skimage.util import img_as_grey856348612092093176235017728
from skimage.util import img_as_grey171269722484186635247003556
from skimage.util import img_as_grey342539444968373270494007112
from skimage.util import img_as_grey685078889936746540988014224
from skimage.util import img_as_grey137015777987349308197602848
from skimage.util import img_as_grey274031555974698616395205696
from skimage.util import img_as_grey548063111949397232790411392
from skimage.util import img_as_grey1096126223898794465808226784
from skimage.util import img_as_grey2192252447797588931616453568
from skimage.util import img_as_grey4384504895595177863232907136
from skimage.util import img_as_grey8769009791187355726465814272
from skimage.util import img_as_grey1753801958237471145293762848
from skimage.util import img_as_grey3507603916474942290587525696
from skimage.util import img_as_grey7015207832949884581175051392
from skimage.util import img_as_grey1403041566589976916235010288
from skimage.util import img_as_grey2806083133179953832470020576
from skimage.util import img_as_grey5612166266359907664940041152
from skimage.util import img_as_grey11224332532719815329880082304
from skimage.util import img_as_grey22448665065439630659760164608
from skimage.util import img_as_grey44897330130879261319520329216
from skimage.util import img_as_grey89794660261758522638040658432
from skimage.util import img_as_grey179589320523577045276081316864
from skimage.util import img_as_grey359178641047154090552162633728
from skimage.util import img_as_grey718357282094308181104325267456
from skimage.util import img_as_grey143671456418861636220645534892
from skimage.util import img_as_grey287342912837723272441291069784
from skimage.util import img_as_grey574685825675446544882582139568
from skimage.util import img_as_grey114937165135089308976564279136
from skimage.util import img_as_grey229874330270178617953128558272
from skimage.util import img_as_grey459748660540357235906257116544
from skimage.util import img_as_grey919497321080714471812514233088
from skimage.util import img_as_grey183899464216142894362502846616
from skimage.util import img_as_grey367798928432285788725005693232
from skimage.util import img_as_grey735597856864571577450011386464
from skimage.util import img_as_grey147119571372942915490022677296
from skimage.util import img_as_grey294239142745885830980045354592
from skimage.util import img_as_grey588478285491771661960090709184
from skimage.util import img_as_grey117695657098354332320080141832
from skimage.util import img_as_grey235391314196708664640060283664
from skimage.util import img_as_grey470782628393417329280020556328
from skimage.util import img_as_grey941565256786834658560041112656
from skimage.util import img_as_grey188313051357366931712082225312
from skimage.util import img_as_grey376626102714733863424064450624
from skimage.util import img_as_grey753252205429467726848128901248
from skimage.util import img_as_grey150650441085933545369625780248
from skimage.util import img_as_grey301300882171867090739251560496
from skimage.util import img_as_grey602601764343734181478503120992
from skimage.util import img_as_grey1205203526865468362957062241984
from skimage.util import img_as_grey2410407053730936725914124483968
from skimage.util import img_as_grey4820814107461873451828248967936
from skimage.util import img_as_grey9641628214923746903656497935872
from skimage.util import img_as_grey1928325642984749380731295587176
from skimage.util import img_as_grey3856651285969498761462591174352
from skimage.util import img_as_grey7713302571938997522925182348704
from skimage.util import img_as_grey1542660514387799504850236469748
from skimage.util import img_as_grey3085321028775599009700472939496
from skimage.util import img_as_grey6170642057551198019400945878992
from skimage.util import img_as_grey12341284115102396038801891757984
from skimage.util import img_as_grey2468256823020479207760378351596
from skimage.util import img_as_grey4936513646040958415520756703192
from skimage.util import img_as_grey9873027292081916830401513406384
from skimage.util import img_as_grey1974605458416383366080302681276
from skimage.util import img_as_grey3949210916832766732160605362552
from skimage.util import img_as_grey7898421833665533464320210725104
from skimage.util import img_as_grey1579684366733106692660402145024
from skimage.util import img_as_grey3159368733466213385320804290048
from skimage.util import img_as_grey6318737466932426770641608580096
from skimage.util import img_as_grey1263747493864845354123201760192
from skimage.util import img_as_grey2527494987729690708246403520384
from skimage.util import img_as_grey5054989975459381416492807040768
from skimage.util import img_as_grey10109979950918762832985614081536
from skimage.util import img_as_grey20219959901837525665971228163072
from skimage.util import img_as_grey4043991980367505133194245632614
from skimage.util import img_as_grey8087983960735010266388491265228
from skimage.util import img_as_grey1617596791467002053376982531456
from skimage.util import img_as_grey3235193582934004106753965062912
from skimage.util import img_as_grey6470387165868008213507930125824
from skimage.util import img_as_grey1294077433173601642705860251648
from skimage.util import img_as_grey2588154866347203285411720503296
from skimage.util import img_as_grey5176309732694406570823441006592
from skimage.util import img_as_grey1035261946588801314166882001384
from skimage.util import img_as_grey2070523893177602628333764002768
from skimage.util import img_as_grey4141047786355205256667528005536
from skimage.util import img_as_grey8282095572710410513335056001108
from skimage.util import img_as_grey1656419114540202102667012002216
from skimage.util import img_as_grey3312838229080404205334024004432
from skimage.util import img_as_grey6625676458160808410668048008864
from skimage.util import img_as_grey1325135291632161682136096017728
from skimage.util import img_as_grey2650270583264323364272192035456
from skimage.util import img_as_grey5300541166528646728544384070912
from skimage.util import img_as_grey1060108233305729345708776014184
from skimage.util import img_as_grey2120216466611458691417552028368
from skimage.util import img_as_grey4240432933222917382835104056736
from skimage.util import img_as_grey8480865866445834765670208113472
from skimage.util import img_as_grey1696173173289166953134041622696
from skimage.util import img_as_grey3392346346578333906268083245392
from skimage.util import img_as_grey6784692693156667812536166490784
from skimage.util import img_as_grey1356938538631333562507232198152
from skimage.util import img_as_grey2713877077262667125014464396304
from skimage.util import img_as_grey5427754154525334250028928792608
from skimage.util import img_as_grey10855508309050678500578575841216
from skimage.util import img_as_grey21711016618101357001157151682432
from skimage.util import img_as_grey43422033236202714002314303364864
from skimage.util import img_as_grey86844066472405428004628606729368
from skimage.util import img_as_grey17368813294881085600925713329736
from skimage.util import img_as_grey34737626589762171201851406658732
from skimage.util import img_as_grey69475253179524342403702813317464
from skimage.util import img_as_grey13895050635904868480740562663496
from skimage.util import img_as_grey27790101271809736961481125326992
from skimage.util import img_as_grey55580202543619473922962250656984
from skimage.util import img_as_grey11116040588738957844592450131392
from skimage.util import img_as_grey22232081177477915689184900262784
from skimage.util import img_as_grey44464162354955831378369800525568
from skimage.util import img_as_grey88928324709811662756739601051136
from skimage.util import img_as_grey17785664941922333553459202102272
from skimage.util import img_as_grey35571329883844667106918404204544
from skimage.util import img_as_grey71142659767689334213836808409088
from skimage.util import img_as_grey14228531935378666847677616801816
from skimage.util import img_as_grey28457063870757333695355233603632
from skimage.util import img_as_grey56914127741514667390710467207264
from skimage.util import img_as_grey11382825548302933478142013441456
from skimage.util import img_as_grey22765651096605866956284026882912
from skimage.util import img_as_grey45531302193211733912568053765824
from skimage.util import img_as_grey91062604386423467825136017531648
from skimage.util import img_as_grey18212520877284693565027203506328
from skimage.util import img_as_grey36425041754569387130054407012656
from skimage.util import img_as_grey72850083509138774260108814025312
from skimage.util import img_as_grey14570016701877554852021762805064
from skimage.util import img_as_grey29140033403755109704043525610128
from skimage.util import img_as_grey58280066807510209408087051220256
from skimage.util import img_as_grey11656001301520401881614010244512
from skimage.util import img_as_grey23312002603040803763228020488024
from skimage.util import img_as_grey46624005206081607526456040976048
from skimage.util import img_as_grey93248010401203215052912081952096
from skimage.util import img_as_grey18649602082406423010582161690496
from skimage.util import img_as_grey37299204164813246021164016380992
from skimage.util import img_as_grey74598408329626492042328032761984
from skimage.util import img_as_grey14919681665953298408465665540392
from skimage.util import img_as_grey29839363331906596816931211080784
from skimage.util import img_as_grey59678726663813193633862422161568
from skimage.util import img_as_grey11935745332762638766772844432312
from skimage.util import img_as_grey23871485665525277533545688864624
from skimage.util import img_as_grey47742971331050555067091218729248
from skimage.util import img_as_grey95485942662101110134182437458496
from skimage.util import img_as_grey19097188532420222026836475491692
from skimage.util import img_as_grey38194377064840444053672950983384
from skimage.util import img_as_grey76388754129680888107345901966768
from skimage.util import img_as_grey15277750825936177621481803933536
from skimage.util import img_as_grey30555501651872355242963607867072
from skimage.util import img_as_grey61111003203744710485927215734144
from skimage.util import img_as_grey12222200640748540971185431146828
from skimage.util import img_as_grey24444401281497081942370862293656
from skimage.util import img_as_grey48888802562994163884741724587312
from skimage.util import img_as_grey97777605125988327769483449174624
from skimage.util import img_as_grey19555521041977665553896689834928
from skimage.util import img_as_grey39111042083955331107793379668956
from skimage.util import img_as_grey78222084167810662215586759337912
from skimage.util import img_as_grey15644416835591332443593411867584
from skimage.util import img_as_grey31288833671182664887186823735168
from skimage.util import img_as_grey62577667342365329774373647470336
from skimage.util import img_as_grey12515533484473065954746795554064
from skimage.util import img_as_grey25031066968946131909493591108128
from skimage.util import img_as_grey50062133937892263818987182216256
from skimage.util import img_as_grey10012426875578537637974364441312
from skimage.util import img_as_grey20024853751157075275948728882624
from skimage.util import img_as_grey40049707502314150551897457765248
from skimage.util import img_as_grey80099415004628301103794915530496
from skimage.util import img_as_grey16019883009255660220758983106096
from skimage.util import img_as_grey32039766008511320441517966212192
from skimage.util import img_as_grey64079532007022640883035932424384
from skimage.util import img_as_grey12815906406045281776607864848768
from skimage.util import img_as_grey25631812803090563553215729697536
from skimage.util import img_as_grey51263625606181127106431459398732
from skimage.util import img_as_grey10251325121236224401262811899564
from skimage.util import img_as_grey20502650242472448802525623799128
from skimage.util import img_as_grey41005200484944897605051247598256
from skimage.util import img_as_grey82001040969889795210010249597712
from skimage.util import img_as_grey16402080193779590420020498995424
from skimage.util import img_as_grey32804160387559180840040997990848
from skimage.util import img_as_grey65608320775118361680081995981696
from skimage.util import img_as_grey13121664155023672336016399196336
from skimage.util import img_as_grey26243328310047344672032799392672
from skimage.util import img_as_grey52486656620094689344065598785344
from skimage.util import img_as_grey10497331324018937888813119597072
from skimage.util import img_as_grey20994662648037875777626238194144
from skimage.util import img_as_grey41989325296075751555252476388288
from skimage.util import img_as_grey83978650592151503110504952776576
from skimage.util import img_as_grey16795730118430300622100904555352
from skimage.util import img_as_grey33591460236860601244201809110704
from skimage.util import img_as_grey67182920473721202488403618221408
from skimage.util import img_as_grey13436584094740604976807236444208
from skimage.util import img_as_grey26873168189481209953614472888416
from skimage.util import img_as_grey53746336378962401989611145776832
from skimage.util import img_as_grey10749267275792403979322229555366
from skimage.util import img_as_grey21498534551584807958644459111332
from skimage.util import img_as_grey42997069103169601918688919022664
from skimage.util import img_as_grey85994138206339203837377838045328
from skimage.util import img_as_grey17198827412678407837355777009056
from skimage.util import img_as_grey34397654825356801574711554008112
from skimage.util import img_as_grey6879530965071360314942310801624
from skimage.util
```

```

platelets: 0.00 0.95 0.55 20
accuracy: 0.97 0.97 0.97 100
recall: 0.97 0.97 0.97 100
weighted avg: 0.97 0.97 0.97 100

```

El modelo KNN obtiene una exactitud del 95%, reflejando una mayor sensibilidad al desbalance y a la distribución local de las muestras. Si tenemos que capturar correctamente personas en clases bien representadas, como el síndrome de monos constante, su desempeño fue menos consistente en clases con menor exposición, particularmente linfocito y neutrófilo. Este campo familiar indica que la promoción en el espacio de características no siempre produce mejores resultados efectivos cuando existen edificios entre los tipos y diferencias en densidad de muestra.

### 3. Modelo Máquina de Vectores de Soporte SVM

```

print("***** SVM *****")
start = time.time()
svm = SVC(kernel='linear')
svm.fit(X_train, y_train)
y_pred = svm.predict(X_test)
elapsed = time.time() - start
print(classification_report(y_test, y_pred, target_names=class_name))
results_dict_svm = {}
for name in class_name:
    results_dict_svm[name] = {
        "Accuracy": accuracy_score(y_test, y_pred),
        "Precision": precision_score(y_test, y_pred, average="macro"),
        "Recall": recall_score(y_test, y_pred, average="macro"),
        "F1Score": f1_score(y_test, y_pred, average="macro"),
        "Time": elapsed
    }
    print("***** SVM *****")
    print("precision\trecall\tf1-score\tsupport")
    print(classification_report(y_test, y_pred, target_names=class_name))
    results_dict_svm = {}
    for name in class_name:
        results_dict_svm[name] = {
            "Accuracy": accuracy_score(y_test, y_pred),
            "Precision": precision_score(y_test, y_pred, average="macro"),
            "Recall": recall_score(y_test, y_pred, average="macro"),
            "F1Score": f1_score(y_test, y_pred, average="macro"),
            "Time": elapsed
        }

```

El modelo SVM obtuvo una exactitud del 95% y demuestra una alta robustez frente al desbalance entre datos. Su capacidad para modelar formas de decisión (enfoque permisivo), una separación más precisa tanto en clases minoritarias como minoritarias, distinguiendo métricas sobredeservientes en eosinófilo, neutrófilo y linfocito. Sin embargo, la clase lymphocyte presentó valores ligeramente inferiores de F1score, lo que sugiere regiones de solamente persistente en el espacio de características. En términos generales, SVM ofrece un modelo liviano entre todos sus datos minoritarios y desempeño global consistente como uno de los clasificadores más consistentes en el análisis.

### 4. Modelo Árbol de Decisión

```

print("***** Decision Tree *****")
start = time.time()
dt = DecisionTreeClassifier(criterion='gini', class_weight='class_weight', max_depth=10)
dt.fit(X_train, y_train)
y_pred = dt.predict(X_test)
elapsed = time.time() - start
print(classification_report(y_test, y_pred, target_names=class_name))
results_dict_dt = {}
for name in class_name:
    results_dict_dt[name] = {
        "Accuracy": accuracy_score(y_test, y_pred),
        "Precision": precision_score(y_test, y_pred, average="macro"),
        "Recall": recall_score(y_test, y_pred, average="macro"),
        "F1Score": f1_score(y_test, y_pred, average="macro"),
        "Time": elapsed
    }
    print("***** Decision Tree *****")
    print("precision\trecall\tf1-score\tsupport")
    print(classification_report(y_test, y_pred, target_names=class_name))
    results_dict_dt = {}
    for name in class_name:
        results_dict_dt[name] = {
            "Accuracy": accuracy_score(y_test, y_pred),
            "Precision": precision_score(y_test, y_pred, average="macro"),
            "Recall": recall_score(y_test, y_pred, average="macro"),
            "F1Score": f1_score(y_test, y_pred, average="macro"),
            "Time": elapsed
        }

```

El modelo Decisión obtuvo una exactitud del 70%, evidenciando una mayor sensibilidad al desbalance en el conjunto de datos. Aunque el modelo basó sus predicciones en pocas variables distintivas, como el plasmocito, mostró desacuerdo notable en los tipos de linfocito y neutrófilo, lo que indica sobreexposición a regiones dominadas por datos minoritarios. Este comportamiento sugiere que la fragmentación jerárquica del espacio de decisión no resulta suficiente para capturar la variabilidad continua de las características celulares. A pesar de su interpretabilidad, el modelo presenta limitaciones claras en términos de generalización.

Posteriormente evaluamos un modelo con redes neuronales con arquitecturas:

### 5. Modelo Red Neuronal Convolucional (CNN)

```

print("***** CNN *****")
def preprocess_data():
    img = cv2.resize(img, (128,128))
    img = np.array(img).reshape(1,128,128,3)
    return img
X_train = np.array([preprocess_data(img) for img in X_train])
X_test = np.array([preprocess_data(img) for img in X_test])
X_train, y_train, X_test, y_test = train_test_split(X_train, y_train, test_size=0.2, stratify=y_train, random_state=42)

model = Sequential([
    layers.Conv2D(32,(3,3),activation='relu'),
    layers.Conv2D(32,(3,3),activation='relu'),
    layers.Conv2D(32,(3,3),activation='relu'),
    layers.MaxPooling2D(),
    layers.Flatten(),
    layers.Dense(128, activation='relu'),
    layers.Dense(64, activation='relu'),
    layers.Dense(32, activation='softmax')
])
model.compile(
    optimizer='Adam',
    loss='categorical_crossentropy',
    metrics=['accuracy']
)
start = time.time()
model.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=10, batch_size=32,
          verbose=1)
elapsed = time.time() - start
y_pred = np.argmax(model.predict(X_test), axis=1)
print(classification_report(y_test, y_pred, target_names=class_name))
results_dict_cnn = {}
for name in class_name:
    results_dict_cnn[name] = {
        "Accuracy": accuracy_score(y_test, y_pred),
        "Precision": precision_score(y_test, y_pred, average="macro"),
        "Recall": recall_score(y_test, y_pred, average="macro"),
        "F1Score": f1_score(y_test, y_pred, average="macro"),
        "Time": elapsed
    }
    print("***** CNN *****")
    print("loss\taccuracy")
    print(classification_report(y_test, y_pred, target_names=class_name))
    results_dict_cnn = {}
    for name in class_name:
        results_dict_cnn[name] = {
            "Accuracy": accuracy_score(y_test, y_pred),
            "Precision": precision_score(y_test, y_pred, average="macro"),
            "Recall": recall_score(y_test, y_pred, average="macro"),
            "F1Score": f1_score(y_test, y_pred, average="macro"),
            "Time": elapsed
        }

```

```

***** CNN *****
Epoch 1/10
18k/18k [100%] - loss: 2.0748 - val_loss: 0.1150 - val_accuracy: 0.9492
Epoch 2/10
18k/18k [100%] - loss: 0.1209 - val_loss: 0.1047 - val_accuracy: 0.9501
Epoch 3/10
18k/18k [100%] - loss: 0.1156 - val_loss: 0.1097 - val_accuracy: 0.9508
Epoch 4/10
18k/18k [100%] - loss: 0.1198 - val_loss: 0.1091 - val_accuracy: 0.9513
Epoch 5/10
18k/18k [100%] - loss: 0.1196 - val_loss: 0.1082 - val_accuracy: 0.9513
Epoch 6/10
18k/18k [100%] - loss: 0.1196 - val_loss: 0.1082 - val_accuracy: 0.9513
Epoch 7/10
18k/18k [100%] - loss: 0.1196 - val_loss: 0.1082 - val_accuracy: 0.9513
Epoch 8/10
18k/18k [100%] - loss: 0.1196 - val_loss: 0.1082 - val_accuracy: 0.9513
Epoch 9/10
18k/18k [100%] - loss: 0.1196 - val_loss: 0.1082 - val_accuracy: 0.9513
Epoch 10/10
18k/18k [100%] - loss: 0.1196 - val_loss: 0.1082 - val_accuracy: 0.9513

```

```

28/28
  26 770ms/step • accuracy: 0.3579 • loss: 1.0574 • val_accuracy: 0.2135 • val_loss: 1.0444
Epoch 6/20
  26 754ms/step • accuracy: 0.2054 • loss: 1.0023 • val_accuracy: 0.4662 • val_loss: 1.7944
Epoch 7/20
  26 759ms/step • accuracy: 0.2079 • loss: 1.0156 • val_accuracy: 0.4625 • val_loss: 1.0722
Epoch 8/20
  26 770ms/step • accuracy: 0.2079 • loss: 1.0097 • val_accuracy: 0.4126 • val_loss: 1.0497
Epoch 9/20
  26 770ms/step • accuracy: 0.2079 • loss: 1.0097 • val_accuracy: 0.4126 • val_loss: 1.0497
Epoch 10/20
  26 809ms/step • accuracy: 0.2042 • loss: 1.0193 • val_accuracy: 0.4375 • val_loss: 1.5329
Epoch 11/20
  26 772ms/step • accuracy: 0.2022 • loss: 1.0195 • val_accuracy: 0.4462 • val_loss: 1.4316
Epoch 12/20
  26 775ms/step • accuracy: 0.2014 • loss: 1.0195 • val_accuracy: 0.4367 • val_loss: 1.0444
Epoch 13/20
  26 807ms/step • accuracy: 0.1938 • loss: 1.0340 • val_accuracy: 0.3638 • val_loss: 1.2778
Epoch 14/20
  26 770ms/step • accuracy: 0.1906 • loss: 1.0381 • val_accuracy: 0.3983 • val_loss: 1.5249
Epoch 15/20
  26 751ms/step • accuracy: 0.1855 • loss: 1.0355 • val_accuracy: 0.4122 • val_loss: 1.2759
Epoch 16/20
  26 809ms/step • accuracy: 0.1859 • loss: 1.0358 • val_accuracy: 0.4150 • val_loss: 1.2842
Epoch 17/20
  26 808ms/step • accuracy: 0.1816 • loss: 1.0390 • val_accuracy: 0.3589 • val_loss: 1.2432
Epoch 18/20
  26 754ms/step • accuracy: 0.1658 • loss: 1.0408 • val_accuracy: 0.3580 • val_loss: 1.3031
Epoch 19/20
  26 764ms/step • accuracy: 0.1667 • loss: 1.0497 • val_accuracy: 0.4062 • val_loss: 1.4477
Epoch 20/20
  26 717ms/step • accuracy: 0.1650 • loss: 1.0222 • val_accuracy: 0.4062 • val_loss: 1.5499

```

Este resultado contradice el obtenido desde hace un año, dadas una exactitud del 90%, mostrando un desempeño considerablemente inferior.

Los tres modelos dígitos analizados, este resultado sugiere que la arquitectura implementada o el volumen de datos disponible no fueron suficientes para que la red aprendiera representaciones tan ágiles, robustas y generalizadas. Aunque la CNN logró capturar adecuadamente patrones en dígitos claramente con mayor separación, como es en el plátano, presentó una degradación significativa en los dígitos más oscuros, particularmente en el número 7 y myrmeces.

Futuras consideraciones: las características de estos dígitos pertenecen a un dataset con imágenes de dimensiones limitadas o distorsionadas, donde el modelo tiende a sobreajustar patrones dominantes y no logra establecer adecuadamente la optimización.

#### 6. Modelo con Transfer Learning notebook

```

print('***** Notebook6 (Transfer learning) *****')
def processImage(img):
    img = cv2.resize(img, (224,224))
    return img/255.

x_train = np.array([processImage(img) for img in x_train])
x_val = np.array([processImage(img) for img in x_val])
x_test = np.array([processImage(img) for img in x_test])

base = InceptionV3(
    weights='imagenet',
    include_top=False,
    input_shape=(224,224,3)
)

base.trainable = False

model1 = Sequential([
    base,
    layers.GlobalAveragePooling2D(),
    layers.Dense(128, activation='relu'),
    layers.Dense(128, activation='relu')
])

model1.compile(
    optimizer='adam',
    loss='categorical_crossentropy',
    metrics=['accuracy']
)

start = time.time()

model1.fit(
    x_train,
    y_train,
    epochs=10,
    validation_data=(x_val, y_val),
    verbose=1
)

elapsed = time.time() - start
print("Time taken: ", elapsed)

y_pred = np.argmax(model1.predict(x_val), axis=1)
print(classification_report(y_val, y_pred, target_names=la.classes))

results.append({
    "Accuracy": accuracy_score(y_val, y_pred),
    "Precision": precision_score(y_val, y_pred, average='macro'),
    "Recall": recall_score(y_val, y_pred, average='macro'),
    "F1Score": f1_score(y_val, y_pred, average='macro'),
    "Time": elapsed
})

j+=1

```

---

```

Modelo1: (Transfer learning) ---->
BatchSize: 64
Optimizer: Adam
Loss: categorical_crossentropy
Epoch 1/10
Epoch 1/10
  32 241ms/step • loss: 0.3586 • val_loss: 0.7012 • val_accuracy: 0.4006 • val_f1score: 0.4003
Epoch 2/10
  32 241ms/step • loss: 0.3555 • val_loss: 0.7008 • val_accuracy: 0.4006 • val_f1score: 0.4003
Epoch 3/10
  32 241ms/step • loss: 0.3544 • val_loss: 0.6973 • val_accuracy: 0.4005 • val_f1score: 0.3975
Epoch 4/10
  32 241ms/step • loss: 0.3534 • val_loss: 0.6958 • val_accuracy: 0.4007 • val_f1score: 0.3951
Epoch 5/10
  32 241ms/step • loss: 0.3443 • val_loss: 0.6992 • val_accuracy: 0.4001 • val_f1score: 0.3931
Epoch 6/10
  32 241ms/step • loss: 0.3179 • val_loss: 0.6949 • val_accuracy: 0.4007 • val_f1score: 0.3933
Epoch 7/10
  32 241ms/step • loss: 0.3144 • val_loss: 0.6960 • val_accuracy: 0.4007 • val_f1score: 0.3927
Epoch 8/10
  32 241ms/step • loss: 0.3135 • val_loss: 0.6958 • val_accuracy: 0.4008 • val_f1score: 0.3927
Epoch 9/10
  32 241ms/step • loss: 0.3131 • val_loss: 0.6958 • val_accuracy: 0.4008 • val_f1score: 0.3927
Epoch 10/10
  32 241ms/step • loss: 0.3127 • val_loss: 0.6958 • val_accuracy: 0.4008 • val_f1score: 0.3927

```

---

precision: 75.44% (Transfer learning) ---->

Este modelo obtuvo un 90.44% de exactitud, con un peso de 0.44, const. peso del mejor desempeño global del modelo, sin embargo, el volumen de datos disponible resultó ser insuficiente para lograr una completa clasificación.

Las imágenes observadas se mantienen algo yugaladas tanto en clase maioriales como minoritarias, evidenciando una capacidad efectiva de generalización en el entorno de muestra mediante el uso de class\_weight y características primariales robustas.

Aunque la dígito 7 permaneció representado al mejor desempeño, el impacto sobre el concepto global limitó y no convenció al modelo de clasificarlo.

A continuación se resumen los resultados de los tres modelos analizados para la clasificación de células sanguíneas:

```

# resultados = pd.DataFrame(resultados)
# resultados.set_index('Accuracy', inplace=True, ascending=False)

```



basophil	0.05	0.55	0.15	20
eosinophil	0.05	0.55	0.15	20
erythroblast	1.00	0.50	0.35	20
erythrocyte	1.00	0.50	0.35	20
lymphocyte	0.79	0.55	0.31	20
monocyte	1.00	0.50	0.35	20
neutrophil	0.85	0.50	0.38	20
platelet	1.00	0.50	0.35	20
red blood cell	0.85	0.50	0.38	20
white blood cell	0.85	0.50	0.38	20
Accuracy:	0.937			
Precision:	0.9410			
Recall:	0.9370			
F1-score:	0.9388			
Time:	0.43 s			

El CNN obtiene un mejor resultado en la clasificación de los tipos de células, con una precisión, recall y F1-score cercanos al 100%, confirmado por los demás reportes de comparación entre los modelos. El modelo de transferencia es el que obtiene una mejor precisión entre todos los demás, incluyendo aquellos con menor representación, lo que evidencia una adecuada comprensión del dominio del conjunto de datos. Clases como eosinofil, linfocito y neutrófilo muestran un menor crecimiento en la clasificación perfecta, mientras que linfocitos, aun que siendo la clase más deseable, muestra una mejor modelo en comparación con las otras no optimizadas.

## Sección de modelo

Se seleccionó el modelo basado en transfer learning como la alternativa final debido a su desempeño superior y a la comprensión de sus métricas a lo largo de todas las clases evaluadas. En comparación con los modelos clásicos y con el CNN entrenado desde cero, el enfoque con Modelos-V2 presenta la mayor precisión global, aunque los valores más altos y equilibrados de precisión, recall y F1-score registrados demuestran una capacidad más robusta para manejar el desbalance moderado del conjunto de datos. Este comportamiento indica que el uso de representaciones preentrenadas proporciona ventajas adicionales para manejar este efecto. La complejidad matemática del modelo se reduce al presentar una descripción más sencilla.

## Conclusiones

### Conclusiones finales

En este trabajo se llevó a cabo una evaluación comparativa exhaustiva de seis modelos de clasificación individuales, incluyendo enfoques basados en aprendizaje automático, una CNN entrenada desde cero y un modelo basado en aprendizaje por transferencia. Los modelos fueron comparados utilizando como métrica principal el F1-score macro, complementado con exactitud, precisión, recall y tiempo de ejecución. Lo que permitió una adición integral del rendimiento predictivo y del costo computacional. Esta comparativa evidenció indiscutiblemente que los enfoques, instrumentos tecnológicos e instrucciones de codificación tienen un conjunto de datos con desempeño modesto entre ellos.

Un componente crítico del proceso fue la etapa de preprocesamiento basada en segmentación del dataset, en la cual se agruparon las células individuales para generar instancias más uniformes y consistentes. Esta segmentación permitió que los modelos aprendieran patrones morfológicos específicos de cada tipo celular, reduciendo ruido y complejidad de fondo. Lo que fue fundamental para el