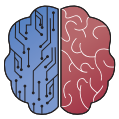
UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería de la Salud



INGENIERÍA
DE LA SALUD

TFG del Grado en Ingeniería de la Salud

Automatización diagnóstica para el análisis de FISH en muestras tumorales

Presentado por Vera García Diez
en Universidad de Burgos

8 de junio de 2023

Tutores: Pedro Latorre Carmona – José Francisco Díez Pastor

UNIVERSIDAD DE BURGOS

ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería de la Salud

D. Pedro Latorre Carmona, profesor del Departamento de Ingeniería Informática, Área de Lenguajes y Sistemas Informáticos.

D. José Francisco Díez Pastor, profesor del Departamento de Ingeniería Informática, Área de Lenguajes y Sistemas Informáticos.

Exponen:

Que la alumna D.ª Vera García Diez, con DNI 71963298R, ha realizado el Trabajo final de Grado en Ingeniería de la Salud titulado título del trabajo.

Y que dicho trabajo ha sido realizado por la alumna bajo la dirección del que suscribe, en virtud de lo cual se autoriza su presentación y defensa.

En Burgos, 8 de junio de 2023

Vº. Bº. del Tutor:

LATORRE CARMONA PEDRO - 22569294S

Firmado digitalmente por LATORRE CARMONA PEDRO - 22569294S
Fecha: 2023.06.08 17:08:08 +02'00'

D. Pedro Latorre Carmona

Vº. Bº. del Tutor:

Firmado por DIEZ PASTOR JOSE FRANCISCO - ***7838** el día 08/06/2023 con un certificado emitido por AC FNMT Usuarios

D. José Francisco Díez Pastor

Resumen

Como herramienta en la clasificación de tumores neuronales se utiliza la técnica *FISH* (*Hibridación fluorescente in situ*), mediante la cual podemos observar cromosomas o regiones de estos. El oligodendroglioma es un tipo de cáncer que se identifica mediante la co-deleción de los brazos cromosómicos *1p* y *19q*. Este cáncer es uno de los que mejor pronóstico tienen. Su tratamiento habitual es la cirugía, por lo que se van a poder establecer pautas de tratamiento y seguimiento posteriores en base al tipo de tumor.

En este proyecto se propone la automatización diagnóstica de las imágenes resultantes de la técnica *FISH*. Identificar posibles co-deleciones o deleciones mediante la detección y el recuento de sondas en función de su color de forma automática. Se va a llevar a cabo gracias al procesamiento de estas imágenes.

Descriptores

FISH, oligodendroglioma, co-deleción, automatización diagnóstica, detección, recuento, sondas.

Abstract

As a tool in the neuronal tumors classification It is used the *FISH* technique (Fluorescent In Situ Hybridization), by which we can observe chromosomes or regions of them. The Oligodendroglioma is a type of cancer which is identified by co-deletion of the *1p* and *19q* chromosomics arms. This cancer has one of the best prognosis possible. Its usual treatment is surgery, so it will be possible to establish treatment and follow-up guidelines based on the type of tumor.

In this proyect is proposed a diagnostic automation of the *FISH* technique images. Identify potential co-deletions or deletions by automatic detection and recounting of probes. It is going to be donde thanks to the processing of these images.

Keywords

FISH, Oligodendroglioma, co-delecion, diagnostic automation, detection, recounting, probes.

Índice general

Índice general iii

Índice de figuras v

Índice de tablas vi

Objetivos 1
- 1.1. Objetivos del estudio . . . 1
- 1.2. Objetivos de calidad . . . 1
- 1.3. Objetivos personales . . . 1

Introducción 3
- 2.1. Estructura de la memoria . . . 4
- 2.2. Estructura de los anexos . . . 4
- 2.3. Conceptos teóricos básicos . . . 5
- 2.4. Estado del arte y trabajos relacionados. . . . 6

Metodología 9
- 3.1. Descripción de los datos. . . . 9
- 3.2. Técnicas y herramientas. . . . 10

Conclusiones 19
- 4.1. Resumen de resultados. . . . 19
- 4.2. Discusión. . . . 22
- 4.3. Aspectos relevantes del desarrollo del proyecto. . . . 24

Lineas de trabajo futuras 29

Bibliografía **33**

Índice de figuras

3.1. (a) *ImageFISH*, fondo anaranjado, (b) imagen 1 set 1, fondo rosado 10
4.1. Tabla acortada calidad sondas rojas 20
4.2. Tabla acortada calidad sondas verdes 20
4.3. Resultados: (a) *ImageFISH*, (b) imagen 1 set 3, (c) imagen 1 set 1, (d) imagen 2 set 1, (e) imagen i set 2, (f) imagen 3 set 2 . . . 21
4.4. Representación gráfica de los resultados de calidad 24

Índice de tablas

4.1. Análisis de resultados con *t-Student*, tabla acortada 23

Objetivos

1.1. Objetivos del estudio

- Ser capaz de identificar sondas verdes/azules y rojas en las imágenes.
- Ser capaz de distinguir mediante procesado de imágenes las distintas sondas teniendo en cuenta su color.
- Ser capaz de identificar células en las imágenes.
- Realizar un filtrado de puntos detectados para poder eliminar ruido y seleccionar aquellos que sean sondas.

1.2. Objetivos de calidad

- Obtener resultados de calidad razonables, es decir, que aunque sean bajos se pueda razonar el porqué.
- Razonar los resultados en cuanto a calidades.
- Generar varios gráficos para analizar los resultados de calidad.

1.3. Objetivos personales

- Aprender a utilizar *Latex*, formatear tablas e imágenes.
- Probar técnicas de procesado de imágenes que no conocía.

- Realizar un estudio que posteriormente otras personas puedan seguir o continuar.

Introducción

La *Hibridación Fluorescente in Situ* (*Fluorescent In Situ Hybridization* o *FISH*) es una técnica biológica de laboratorio utilizada para visualizar cromosomas, detectar y localizar regiones cromosómicas o genes. Consiste en la hibridación de sondas, fragmentos cortos de ADN purificado, que contienen un marcador de fluorescencia. Generalmente, se van a utilizar dos sondas diferentes, una va a hacer de sonda control y otra va a ir dirigida a la región de interés. Esta técnica se va aplicar sobre un portaobjetos que contenga un corte FFPE (tejido fijado con formalina e incrustado en parafina).

En este caso, se va a intentar detectar una posible co-deleción de los brazos *1p* y *19q*. Esto quiere decir que el brazo corto del cromosoma uno y el brazo largo del cromosoma 19 se han perdido. Identificar esta co-deleción en células gliales nos va a servir para diagnosticar un oligodendroglioma. La importancia de diagnosticar este cáncer en concreto, es que es uno de los que mejor pronóstico tienen. Si un paciente se somete a cirugía se le van a aplicar un tratamientos y un seguimiento específico, en base a este buen pronóstico. Esta clasificación se realiza en base a la *Clasificación de tumores del sistema nervioso central* actualizada y publicada por la *OMS* en 2021[9]. Existiendo versiones anteriores, como la del 2016.

Cerca del 40 % de los tumores cerebrales son gliomas, de los cuales el 10 % son oligodendrogliomas[5]. La tasa relativa de supervivencia a 5 años en pacientes de 20 a 44 años de edad es del 90 %, de 45 a 55 es del 82 %, y de 55 a 64 años es del 69 %[10]. Convirtiéndose en el tumor del encéfalo y de la médula espinal en adultos con una mayor tasa de supervivencia.

El propósito de este proyecto es automatizar este diagnóstico. Procesar de manera automática imágenes de *FISH* de estas características, para

identificar o no esta co-deleción. El método tradicional consiste en ir analizando a mano una a una las imágenes de fluorescencia para establecer esta clasificación.

2.1. Estructura de la memoria

La memoria se divide en los siguientes apartados:

- **Objetivos del proyecto**: Se recogen los objetivos del proyecto, de calidad y personales.
- **Introducción**: Formada por una introducción sobre el tema a tratar en el proyecto, los conceptos teóricos básicos y estado del arte.
- **Metodología**: Se realiza una descripción de los datos y una enumeración de las técnicas y métodos utilizados.
- **Conclusiones**: Se exponen las conclusiones del proyecto, un resumen de los resultados obtenidos y los aspectos relevantes de este.
- **Lineas futuras**: Se resumen las posibles líneas futuras que ha de seguir el proyecto.

2.2. Estructura de los anexos

Los anexos que completan la memoria son:

- **Plan de Proyecto Software**: Se desarrollan la planificación temporal y económica, y la viabilidad legal del proyecto.
- **Documentación de usuario**: Incluye requisitos tanto de hardware como de software, el manual de usuario y las demostraciones prácticas.
- **Manual del investigador**: Donde se explica la estructura de los directorios y las instrucciones para continuar el proyecto o realizar mejoras.
- **Descripción de adquisición y tratamiento de datos**: Recoge la descripción tanto formal como clínica de los datos, extensión del apartado de metodología en cuanto a datos.

- **Manual de especificación de diseño**: Contiene los diagramas del diseño de clases, de paquetes y el diagrama de secuencia del algoritmo principal.
- **Especificación de Requisitos**: Se recoge el diagrama de casos de uso y los requisitos correspondientes.
- **Estudio experimental**: Enumeración de todos los métodos utilizados, su parametrización y los resultados obtenidos.

2.3. Conceptos teóricos básicos

- **FISH**: Hibridación Fluorescente in Situ, técnica que permite detectar secuencias específicas de ADN o ARN en preparaciones celulares o cortes de tejidos. Esto se consigue mediante sondas marcadas con un fluorocromo. Con una longitud de onda específica emitirán fluorescencia, y podrán ser detectadas de forma directa con un microscopio de fluorescencia.
- **Fluorocromo**: Compuesto químico fotorreactivo que se emplea para marcar anticuerpos u otras moléculas, por su propiedad de emitir luz de una determinada longitud de onda, cuando se le estimula con luz ultravioleta o con un láser.
- **Deleción**: Tipo de mutación que supone la pérdida de uno más nucleótidos de un segmento de ADN. Puede implicar la pérdida de solo un nucleótido o de una parte completa del cromosoma, como pueden ser un brazo.
- **Co-deleción**: Deleción combinada en dos cromosomas diferentes o incluso en el mismo cromosoma.
- **Oligodendroglioma**: Tumor primario del sistema nervioso central. Comienza en los oligodendrocitos, células que segregan una sustancia que protegen a las neuronas y ayudan con el flujo de señales eléctricas. Es más común en adultos, y sus síntomas incluyen convulsiones, dolores de cabeza o debilidad en alguna parte del cuerpo. Los síntomas van a depender de la zona cerebral en la que se encuentre el tumor. Su tratamiento es la cirugía, siempre y cuando esta sea posible. Este tipo de tumores pueden ser malignos o benignos, y la mayoría tienden a desarrollarse lentamente. Este tipo de tumor suele tener buen pronóstico.

- **Diploide**: Término que implica la presencia de dos conjuntos de cromosomas en las células del organismo, cada uno es heredado de un progenitor. Los seres humanos tenemos 23 pares de cromosomas, un total de 46 cromosomas, en células somáticas. Los gametos son células haploides, formados por 23 cromosomas.

- **Segmentación**: Campo del procesamiento de imágenes que consiste en dividir una imagen en varias regiones o grupos de píxeles que se van a llamar segmentos. Es un método de clasificación por píxel,a cada uno se le va a asignar una categoría.

- **Umbralización**: Método de segmentación cuyo objetivo es convertir una imagen en escala de grises a una nueva con solo dos niveles, blanco y negro, de manera que la imagen se divida en fondo y objetos.

- **Groundtruth**: Imágenes que representan la umbralización ideal de las imágenes a las que se les aplica esta técnica. Sirven para calcular valores en cuanto a calidades de como se ha realizado la umbralización comparándolas con los resultados obtenidos.

- **Intersección sobre unión**: Método para medir la calidad de una imagen segmentada. Consiste en comparar esta imagen segmentada con su imagen gorundtruth, y dividir el número de píxeles de su intersección entre los de su unión. Se obtendrá un valor situado entre el rango cero y uno. Cuanto mayor sea este valor, mejor será la calidad de la segmentación de la imagen. Conocido por sus siglas en inglés *IoU*, (*Intersection over Union*).

2.4. Estado del arte y trabajos relacionados.

Como uno de los primeros pasos se realizó una búsqueda de bibliografía, ya que se trata de un proyecto de investigación. Para que esta fuese más fácil se dividió en dos búsquedas diferentes, detección de sondas en imágenes de *FISH* o en técnicas parecidas como puede ser *SISH*, y detección de células de la imagen, preferiblemente de *FISH*. Por lo que esta sección va a estar dividida en esos dos apartados.

Detección de sondas

Sobre este aspecto se encontró muy poca información, y muy variada entre sí. Algunos de los métodos encontrados fueron:

- Análisis de la distribución del color en la imagen en 2D y 3D, concretamente en los canales R (rojo) y G (verde) de imágenes de *FISH* de cáncer de mama[31].
- Calculando distancias euclídeas de centro a centro, de centro a borde o de borde a borde, para detectar al distribución de los cromosomas o genes en imágenes de *FISH* en 3D[17].
- Etapas de filtrado *top-hat* seguidas de umbralización binaria, y evaluación de contraste en imágenes de *FISH*[27].
- Utilización del clasificador de *KNN*[18], basado en en la regla de los k vecinos más cercanos, muestras de tejido teñidas inmunocitoquímicamente.

Detección de células

Para detectar las células de la imagen el método más utilizado es la umbralización de las imágenes, ya sea por el método *Otsu* u otros diferentes, seguido de la aplicación del algoritmo *watershed*[31, 27]. Otra alternativa, es aplicar directamente el algoritmo *watershed* seguido de detección de bordes[18] o un algoritmo de clasificación bayesiano[26]. Otros métodos pueden ser: la identificación de leucocitos, al aplicar *K-Means*[3], y la creación de un mapa de distancias euclídeas para diagnosticar cáncer de mama mediante imágenes de *FISH*[29].

Metodología

3.1. Descripción de los datos.

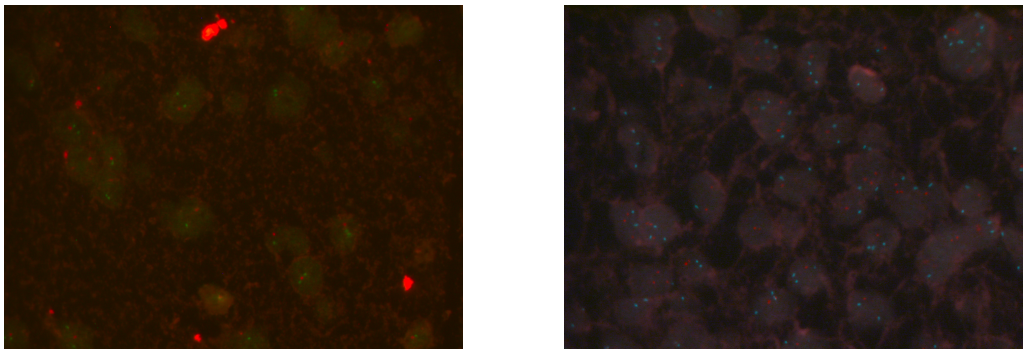
Las imágenes utilizadas han sido cedidas por el servicio de Anatomía Patológica del Hospital Universitario de Burgos. Se trata de imágenes en formato .jpg en RGB capturadas con microscopio de fluorescencia. Son imágenes de resultado de aplicar la técnica *FISH* sobre células neuronales a cortes FFPE, es decir, cortes de tejido fijado en formalina e impregnado en parafina.

Esta técnica a servir para identificar de qué tipo de tumor neuronal se trata, y poder seguir unas líneas de tratamiento.

Las imágenes proporcionadas se van a encontrar dentro de la carpeta imágenes y van a ser de dos tipos diferentes 3.1:

- *ImageFISH*, imagen independiente, primera imagen proporcionada, de fondo anaranjado.
- Set 1: Compuesto por tres imágenes de fondo rosado, imagen 1 (*image2951*), imagen 2 (*image2952*) e imagen3 (*image2954 19q*).
- Set 2: Compuesto por cinco imágenes de fondo rosado, imagen 1 (*image2945*), imagen 2 (*image2946*), imagen 3 (*image2947*), imagen 4 (*image2948*) e imagen 5 (*image2949*).
- Set 3: Compuesto por dos imágenes de fondo anaranjado, imagen 1 (*image2472*) e imagen 2 (*image2475*).

Todo ello esta detallado en el apéndice D: Descripción de adquisición y tratamiento de los datos.



(a) (b)

Figura 3.1: (a) *ImageFISH*, fondo anaranjado, (b) imagen 1 set 1, fondo rosado

3.2. Técnicas y herramientas.

Para llevar a cabo el el proyecto se han utilizado diferentes funciones de varios paquetes de *Python*. Este apartado se va a dividir según los distintos pasos realizados, en los que se recogerá información sobre qué funciones y herramientas se han utilizado, la justificación de su uso y posibles alternativas. Esta información se va a explicar con más detalle en cuanto a resultados, en el anexo *G-Estudio experimental*.

Convertir imágenes a HSV y a CIELab

El espacio de color HSV es un modelo de representación de modelos formado por tres canales, *Hue, Saturation, Value*. Es útil para seleccionar rangos de colores basados en el tono, modificar su saturación o cambiar el brillo para obtener más o menos oscuridad.

El espacio de color CIELab esta diseñado para representar los colores según son percibidos por el ojo humano. Posee tres canales, que se identifican con el brillo, la posición cromática en el eje rojo-verde y la posición cromática en el eje amarillo-azul. Puede ser útil para la comparación de colores o la gestión del color.

Para pasar de las imágenes RGB originales a HSV y CIELab se han utilizado dos funciones del módulo *skimage.color*[22], recogidas en el notebook *Obtener la imagen en HLSV y CIElab*:

- *rgb2hsv*: Convierte las imágenes en RGB a HSV.
- *rgb2lab*: Convierte las imágenes en RGB al espacio de colores CIELab.

Posibles alternativas:

- *cv2.cvtColor*[30]: Forma parte del paquete *OpenCV2*. Dependiendo del parámetro que se pase a esta función (*cv2.COLOR_RGB2HSV, cv2.COLOR_RGB2Lab*), puede pasar de RGB a HSV o a CIELab. Esta opción fue descartada, ya que es necesario instalar este paquete en el entorno de Anaconda, mientras que en la librería *skimage* esto no es necesario. Otra desventaja es la ausencia de una página con documentación formal y bien estructurada.
- Método *.convert* de la biblioteca *PIL*[6]: Mediante un parámetro se puede convertir a cualquiera de las dos imágenes (*HSV, LAB*). El principal problema es el formato en el que se ha de tener la imagen. No vale con tenerla en forma de *array* numérico de *numpy*, si no que tiene que instanciarse con la clase *Image* de este mismo paquete.

Se eligieron las dos primeras funciones, simplemente por comodidad.

Detección de sondas mediante técnicas de umbralización

Para la detección de sondas, verdes, azules o rojas, se van a probar diferentes técnicas de umbralización. Estas pruebas están recogidas en los notebooks: *Umbralización sondas rojas, Umbralización sondas verdes* y *Umbralizar sondas de las imágenes con fondo azul*. En conjunto se han utilizado como herramientas principales una serie de funciones del paquete *skimage.filters*[25] sobre los distintos canales de las imágenes RGB, HSV y CIElab. Las herramientas utilizadas son las siguientes:

- *Try_all_threshold*: Devuelve una figura comparando la salida de varios métodos: *Isodata, Li, Mean, Minimum, Otsu global, Triangle* y *Yen*.
- *Threshold_mean*: Devuelve un valor umbral en función del nivel de la escala de grises.
- *Threshold_otsu*: Devuelve un valor umbral en función del método *Otsu*.
- *rank.otsu*: Devuelve un valor umbral para cada ventana de píxeles mediante el método *Otsu*, *Otsu local*.
- *threshold_yen*: Devuelve un valor umbral en función del método *Yen*.

- *threshold_triangle*: Devuelve un valor umbral en función del algoritmo *Triangle*.
- *threshold_local*: Devuelve un valor umbral en función de los vecinos locales de píxeles, método local.
- *threshold_niblack*: Aplica umbralización *Niblack* para obtener el valor umbral por ventanas de la imágen.
- *threshold_sauvola*: Aplica umbralización *Sauvola* para obtener el valor umbral por ventanas de la imágen, variedad del método *Niblack* anterior.

Como alternativas a este paquete, se implementaron dos funciones diferentes:

- *Kmeans* del paquete *sklearn.cluster*: Realizar clustering con el algoritmo de clasificación no supervisada *KMeans*[28]. Basado en el cálculo de distancias entre un píxel (objeto) y los centroides de los grupos. Como queremos binarizar la imagen, solo se va a utilizar con dos grupos. Se complementa este método con una función de implementación propia, para generar las etiquetas correspondientes a la utilización de este algoritmo y aplicarlas como resultado. Método descartado, ya que sus resultados se alejan de lo deseado.
- Función de implementación propia *binarizar_umbral*: Recibe la imagen y un valor introducido como parámetros, este valor se utilizará como valor umbral. Alternativa descartada, no podemos saber cuál es el mejor umbral de una imagen mirando simplemente su histograma.

Como herramienta finalmente utilizada se seleccionó el método *threshold_yen*, sus resultados son los más acertados para todo tipo de sondas.

Filtrado de puntos

Para filtrar los puntos detectados en el apartado anterior se utilizan varias funciones del modulo *skimage.measure*[24] en el notebook *Prueba general de filtrado*. Una vez que se tienen los puntos detectados en una imagen binaria, para filtrarlos por tamaño hay que identificar cada punto como una región. Primero se ponen etiquetas a las zonas detectadas con *label* y luego se establecen como regiones con *regionprops*. Para realizar el filtrado por tamaño se utilizan la propiedad *area* de *regionprops*.

Como posibles alternativas se han encontrado:

- *scipy.ndimage.label*[7], para colocar las etiquetas.
- *scipy.ndimage.find_objects*, esta función solo identifica cada región con sus coordenadas, no tiene ninguna propiedad para poder filtrar por tamaño.

Detección de células

Primer paso para filtrar sondas según su posición, fuera o dentro de la célula. Pruebas realizadas en el notebook *Detección y filtrado de células*. Se utilizan como herramientas funciones ya vistas en el apartado anterior *Detección de sondas mediante técnicas de umbralización*, del paquete *skimage.filters*[25]:

- *Threshold_isodata*: Devuelve un valor umbral en función del método *Isodata*.
- *Threshold_li*: Calcula el valor umbral mediante el método iterativo de entropía cruzada mínima de *Li*.
- *Threshold_otsu*.

Solo se han probado estas funciones, ya que han sido probadas previamente en el método antes mencionado. Por lo que se podían descartar muchas funciones ya de primeras.

Como tras alternativas se ha encontrado:

- Funciones del paquete *OpenCV* que calculen el valor del umbral de forma adaptativa, como *cv.adaptiveThreshold*[16] que va a recibir distintos parámetros. Otra función sería el método *Otsu* de *OpenCv*, con *cv.threshold* pasándole el parámetro *cv.THRESH_OTSU*. Como hemos dicho antes este paquete necesita ser instalado, por lo que estas opciones han sido descartadas.

Finalmente se ha escogido el método *Li* aplicado sobre la imagen del canal verde, debido a sus buenos resultados.

Detección del borde celular

Paso realizado en el notebook *Detección y filtrado de células*. Se van a probar diferentes detectores de bordes para poder visualizar que puntos se encuentran dentro y cuales no. Para ello si han probado varios detectores del paquete también *skimage.filters*[25], utilizan el detector de su mismo nombre:

- *Sobel*.
- *Robberts*.
- *Farid*.
- *Prewitt*.
- *Scharr*.
- *Laplace*, este método se descarto desde el principio, ya ue sus resultados eran casi nulos.

Además de la función *canny* de *skimage.feature*[23]. Todos los resultados obtenidos de estos detectores son prácticamente iguales.

Alternativas:

- Los detectores de *Prewitt*, *Sobel* y *Laplace* también son proporcionados por el paquete *scipy*, *scipy.ndimage.prewitt, scipy.ndimage.sobel, scipy.ndimage.laplace*[7]. Estos métodos solo funcionan con imágenes de un solo canal. Aunque, esto no afecta a esta detección, y que se realiza sobre imágenes binarias de la binarización de células.
- En el paquete *OpenCV* también hay detectores de bordes, *cv2.Laplacian*, *cv2.Canny*, *cv2.Sobel* y *cv2.Scharr*[15]. Ya han sido explicados los problemas de este paquete en apartados anteriores.

Se han utilizado el detector de bordes *Canny*[14] de *skimage.feature*, simplemente debido a que los bordes detectados que devuelve son finos y acordes con lo que se busca.

Suavizar bordes y eliminar huecos

Para eliminar suavizar los bordes y eliminar las formas detectadas en el interior celular, se aplicó la operación morfológica de cierre y una función para rellenar huecos sobre la imagen de detección de células, la detección de bordes se aplicó después. Su primera aplicación es en el notebook *Detección y filtrado de células* y se analizan diferentes formas y tamaños de cierre, y la utilización de la función de rellenar huecos en el notebook *Estudio cierre resultados y análisis*.

Se ha utilizado la función *skimage.morphology.closing*[21], en algunos casos combinada con *scipy.ndimage.binary_fill_holes*[7], función para rellenar huecos en la imagen binaria.

Alternativas a la función de cierre:

- Como la operación morfológica de cierre esta formada por una operación de dilatación seguida de erosión, se pueden realizar estas dos operaciones en este orden de manera independiente. Si se cambia el orden de estas dos operaciones se realizaría una apertura, no un cierre, por ello el orden es muy importante. Son de la misma bilioteca y modulo que la función de cierre: *skimage.morphology.dilation* y *skimage.morphology.erosion*[21]. Al final se utilizó la función de cierre al ser solo una y no llevar a confusiones.

- *scipy.ndimage.binary_closing*[7], misma función pero de otra biblioteca. Una de sus diferencias es que esta última solo funciona en imágenes de un canal y la de la librería *sklearn* con tres canales.

Alternativa a la función de rellenar huecos:

- *Skimage.morphology.binary_fill_holes*[21] de la librería *skimage*, primera función probada, pero no generaba los resultados deseados, por lo que fue descartada.

La función *closing* tiene que recibir como parámetro la forma y tamaño deseados para realizar la operación de cierre. Se han comparado los resultados de varias formas y tamaños en el notebook *Estudio cierre resultados y análisis*, también importadas del módulo *skimage.morphology*[21], estas son:

- *disk*, con tamaño 3, 4, 5 o 6.

- *diamond*, con tamaño 3, 4, 5 o 6.
- *square*, con tamaño 3, 4, 5 o 6.
- *star*, con tamaño 3, 4, 5 o 6.

Otras formas que se podrían haber utilizado son *ball, ellipse, octagon* o *rectangle*, de mismo módulo. No se han escogido estas ya que son formas más dificiles de ejecutar, con más parámetros.

Generar imágenes groundtruth

Las imágenes groundtruth necesarias para evaluar la calidad de las imágenes se han obtenido mediante el programa de software libre *VGG Image Annotator*, requiere instalación[4]. En este programa se carga la imagen que se quiera, y una a una se seleccionan las sondas de forma manual asociándolas con una etiqueta. Esta etiqueta se corresponde con el color de sonda que se identifica. Después de este proceso se obtienen en formato JSON y COCO las imágenes groundtruth resultantes. Gracias a estas etiquetas luego se van a poder generar dos imágenes groundtruth, siguiente apartado *Leer JSON*, una para sondas rojas y otra para verdes/azules, a partir de una imagen de partida.

Algunas alternativas frente a este programa:

- Labelbox[1]: Aplicación web que permite crear anotaciones sobre imágenes propias en la nube. Estas anotaciones se pueden exportar en forma de archivos JSON, CSV, COCO o como imágenes anotadas. Su principal desventaja es que se necesita crear una cuenta para su uso.
- LabelImg[20]: Herramienta de código abierto que se puede utilizar directamente en *Jupyter notebook*. Solo permite la anotaciones de regiones mediante cuadros delimitadores. Por lo que no puede ser utilizada en este proyecto, ya que necesitamos anotar sondas con forma redondeada.

Leer JSON

Para leer los JSON correspondientes a las imágenes groundtruth utilizadas para calcular la calidad de la segmentación, en el notebook *Prueba general de filtrado*, se ha utilizado el paquete *josn*[19], concretamente la función *load*.

Para esta función no se han encontrado alternativas que realicen lo que se pide.

Generar resultados de calidad

Para calcular los resultados de calidad mediante la metodología Intersección sobre unión se ha utilizado una función para calcular la intersección y otra diferente para la unión, ambas del paquete *numpy*:

- *np.logical_and*[11]: Calcula la intersección o 'y' lógico entre dos imágenes binarias.
- *np.logical_or*[12]: Calcula la unión u 'o' lógico entre dos imágenes binarias.

Los notebooks que contienen resultados de calidad son: *Prueba general de filtrado* y *Estudio cierre resultados y análisis*.

Posibles alternativas:

- Operador lógico &.
- Operador lógico |.

Las desventajas de estos operadores es que no soportan la característica de *Broadcasting*, no se pueden utilizar con *arrays* de diferentes tamaños.

Realizar violin plot

El resto de gráficos e imágenes han sido creadas o representadas mediante *matplotlib*. En cambio, para el violin plot que representa las calidades obtenidas de las imágenes, se ha utilizado de la biblioteca *seaborn*, la funcion *violinplot*[33]. Ya que con esta biblioteca generar este gráfico era mucho más simple, y se entendía mejor que en la misma función de la biblioteca *matplotlib*[13]. Este esta recogido en el notebook *Estudio cierre resultados y análisis*.

Presentación de resultados con gráficos interactivos

Para presentar resultados de forma interactiva y que el usuario pueda cambiar valores, se han utilizado dos funciones de *ipywidgets*[2]. Por un lado, una barra para elegir diferentes números para filtrar por tamaño las sondas

detectadas, *interactive_output*, en el notebook *Prueba general de filtrado*. Un desplegable para elegir varias opciones diferentes con *Dropdown*, esta vez en el notebook *Estudio cierre resultados y análisis*.

No se han encontrado alternativas que implementen estas funciones interactivas directamente en el notebook.

Prueba t-Student pareada con datos dependientes

Para el análisis de las calidades obtenidas se ha realizado este test para identificar métodos equivalentes. Se ha utilizado la función *ttest_rel* [8] de la librería *scipy.stats*. Va a devolver como parámetros el valor t estadístico, el *p-value* y el número de grados de libertad utilizados para calcular el valor t estadístico.

Como alterativa se ha encontrado la función *ttest* [32] del paquete *pingouin*, pasándola como parámetro *paired = True* para que este test sea dependiente. Algún inconveniente es que el paquete *pingouin* hay que instalarlo en el entorno de Anaconda y es una librería poco conocida.

Resto de paquetes para los que no hay alternativas

Hay una serie de paquetes utilizados que no tienen una alternativa como tal, si no que son los más utilizados en *Python*. Estos son:

- *Matplotlib*
- *Pandas*
- *Numpy*

Conclusiones

Del desarrollo del proyecto se han obtenido varias conclusiones tanto de relacionadas con los resultados, como conclusiones teóricas.

Los resultados de calidad obtenidos de la segmentación de imágenes son más bajos de los esperados. La segmentación de imágenes es un proceso largo y difícil en el que no se suelen obtener unos resultados deseados. Esto radica en la dificultada de las técnicas propuestas a detectar o umbralizar lo que justamente se quiere.

Uno de los factores a tener en cuenta es que solo se dispone de once imágenes de dos tipos diferentes. Si estas imágenes fuesen del mismo tipo, es decir, más parecidas entre sí, quizás estos resultados eran mejores al ajustar las técnicas utilizadas y sus parámetros. De esta otra forma solo se puede llegar a una aproximación en los resultados, ya que se tienen que tener en cuenta dos tipos de imágenes a las que se van a aplicar las mismas técnicas.

4.1. Resumen de resultados.

Como resultados finales del proyecto se han obtenido los valores de calidad de segmentación para cada imagen en función del tipo de cierre realizado, forma y tamaño.

Estos resultados han sido recogidos en dos tablas, una de calidad de sondas rojas y otra de las sondas verdes. Cada una esta formada por 20 columnas y 11 filas. Estas tablas han sido acortadas, tanto la verde 4.2 como la roja 4.1, para una mejor visualización. Recogen en sus columnas las cuatro formas (disk, diamond, square, star) en cuatro tamaños diferentes (3, 4, 5 y 6), y los resultados de aplicar la función de rellenar huecos a las cuatro formas con su menor tamaño, 3. Como se puede ver en las tablas los

resultados de la segmentación de sondas verdes son iguales y los rojos no, el análisis de estos resultados se va a desarrollar en el siguiente punto.

Imagen	disk(3)	disk(6)	diamond(3)	diamond(6)	square(3)	square(6)	star(3)	star(6)	holes disk(3)
imageFISH	0.1259	0.1860	0.1224	0.1852	0.0820	0.1418	0.1709	0.1926	0.1841
imagen11	0.3163	0.3609	0.3087	0.3614	0.2385	0.3241	0.3344	0.3653	0.3601
imagen12	0.3484	0.3594	0.3482	0.3594	0.2997	0.3532	0.3594	0.3594	0.3594
imagen13	0.0288	0.0288	0.0288	0.0288	0.0251	0.0265	0.0288	0.0288	0.0288
imagen21	0.1810	0.2000	0.1810	0.2000	0.1611	0.1928	0.2000	0.2000	0.2000
imagen22	0.3257	0.3537	0.3236	0.3467	0.2761	0.3351	0.3487	0.3543	0.3290
imagen23	0.2687	0.2890	0.2690	0.2871	0.2618	0.2761	0.2770	0.2969	0.2730
imagen24	0.3621	0.3621	0.3621	0.3621	0.3331	0.3621	0.3621	0.3621	0.3621
imagen25	0.2698	0.2698	0.2698	0.2698	0.2698	0.2698	0.2698	0.2698	0.2698
imagen31	0.0206	0.0571	0.0206	0.0511	0.0140	0.0212	0.0325	0.0571	0.0571
imagen32	0.0695	0.0917	0.0628	0.0917	0.0385	0.0883	0.0890	0.0917	0.0917

Figura 4.1: Tabla acortada calidad sondas rojas

Imagen	disk(3)	disk(6)	diamond(3)	diamond(6)	square(3)	square(6)	star(3)	star(6)	holes disk(3)
imageFISH	0.5093	0.5093	0.5093	0.5093	0.5093	0.5093	0.5093	0.5093	0.5093
imagen11	0.5282	0.5282	0.5282	0.5282	0.5282	0.5282	0.5282	0.5282	0.5282
imagen12	0.5186	0.5186	0.5186	0.5186	0.5186	0.5186	0.5186	0.5186	0.5186
imagen13	0.3568	0.3568	0.3568	0.3568	0.3568	0.3568	0.3568	0.3568	0.3568
imagen21	0.5505	0.5505	0.5505	0.5505	0.5505	0.5505	0.5505	0.5505	0.5505
imagen22	0.5180	0.5180	0.5180	0.5180	0.5180	0.5180	0.5180	0.5180	0.5180
imagen23	0.5418	0.5418	0.5418	0.5418	0.5418	0.5418	0.5418	0.5418	0.5418
imagen24	0.0396	0.0396	0.0396	0.0396	0.0396	0.0396	0.0396	0.0396	0.0396
imagen25	0.4345	0.4345	0.4345	0.4345	0.4345	0.4345	0.4345	0.4345	0.4345
imagen31	0.3170	0.3170	0.3170	0.3170	0.3170	0.3170	0.3170	0.3170	0.3170
imagen32	0.4429	0.4429	0.4429	0.4429	0.4429	0.4429	0.4429	0.4429	0.4429

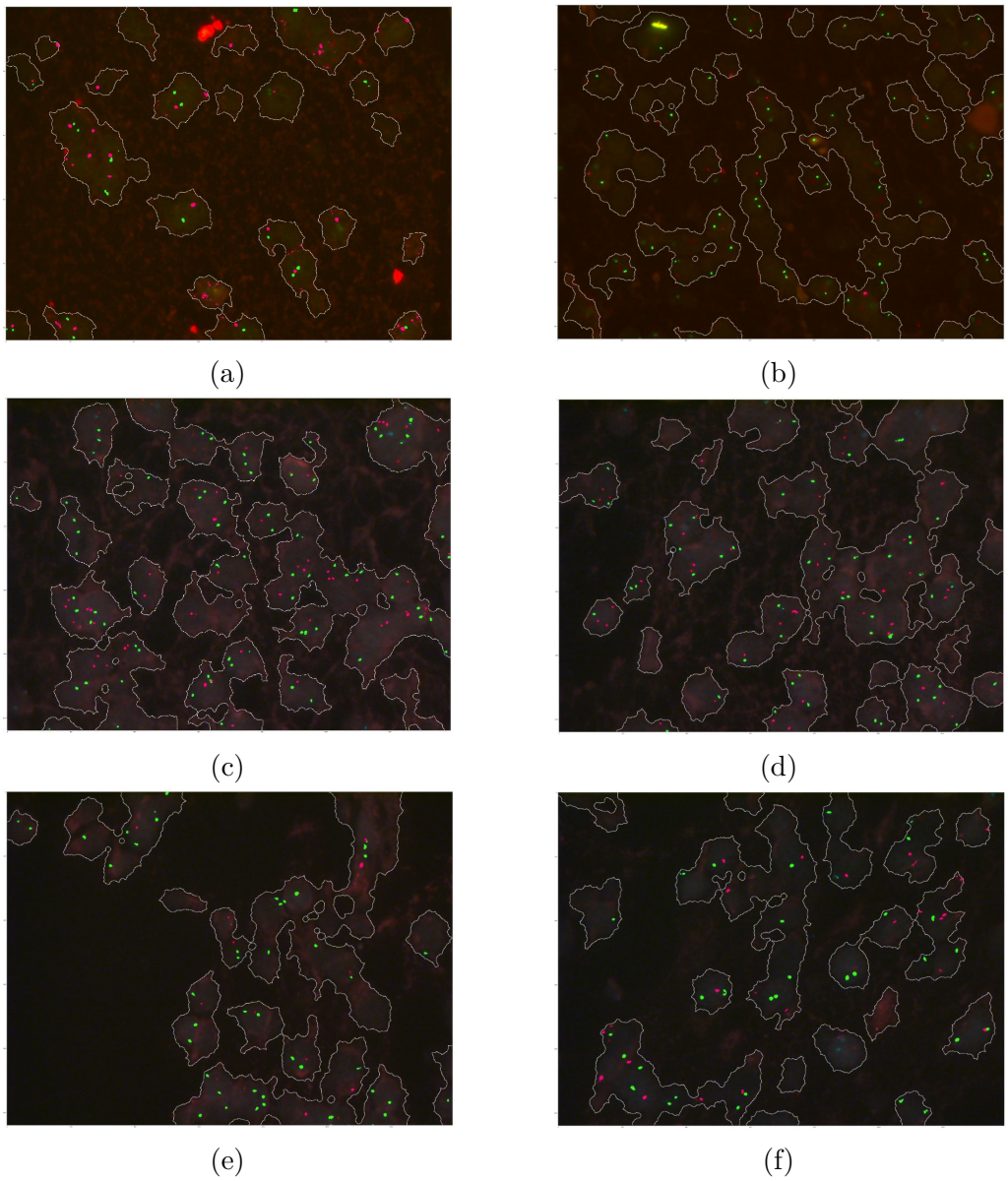
Figura 4.2: Tabla acortada calidad sondas verdes

Por otra parte, también se han recogido los resultados gráficos de la segmentación de imágenes. Estos resultados se pueden ver en las funciones *dropdown* del notebook *Estudio cierre resultados y análisis.*

Se pueden elegir distintas visualizaciones, el mejor cierre para cada imagen, ver varios tamaños de cierre sobre la misma imagen, o elegir la

imagen y el cierre concreto que se quiere visualizar. Por ellos hay tres funciones *dropdown* diferentes.

Los resultados finales obtenidos de la segmentación con el mejor cierre, *star(6)*, de varias imágenes de distinto tipo, se pueden ver en la figura 4.3.



(a) (b) (c) (d) (e) (f)

Figura 4.3: Resultados: (a) *ImageFISH*, (b) imagen 1 set 3, (c) imagen 1 set 1, (d) imagen 2 set 1, (e) imagen i set 2, (f) imagen 3 set 2

4.2. Discusión.

En cuanto al análisis numérico de los resultados, se ha llevado a cabo un test estadístico, *t-Student*, para ver cual es el mejor resultado y ver si hay métodos que se puedan considerar equivalentes.

La prueba de *t-Student* es un tipo de estadística deductiva. Se utiliza para determinar si existe una diferencia significativa entre dos grupos. Esta prueba puede ser dependiente o independiente en base a los datos.

En este proyecto se recogen datos de segmentación con diferentes cierres en varias imágenes, los datos de partida son iguales para todos los métodos. Es decir, sobre todas las imágenes se prueban los mismos cierres. Por ello se ha de realizar una *t-Student* dependiente.

Se ha utilizado la función *ttest_rel* de la librería *scipy.stats*. Esta función recibe dos parámetros, dos listas o arrays de datos, y realiza una *t-Student* pareada. La hipótesis nula es que dos muestras relacionadas o repetidas tienen valores promedio (esperados) idénticos. Es decir, que cuanto más mayor sea el *p-value* resultante, más riesgo hay de que se rechace la hipótesis nula. Dicho con otras palabras, cuanto mayor sea el *p-value* más parecidos serán los datos introducidos entre sí.

Solo se va a realizar el test sobre los datos de calidad de sondas rojas, ya que los datos de calidad de las sondas verdes son iguales, y como resultado no se obtendría un valor de *p-value* válido, si no un valor *NaN*.

Para la realización de este test se ha obtenido del promedio de calidad para todos los métodos, en cuanto a calidad de sondas rojas. A continuación, se ha realizado este test con los datos del método de mayor promedio con el resto, en orden descendente. De esta forma, se va a intentar averiguar si los métodos que no tienen el mejor promedio pero quedan cerca, son equivalentes o no. Si finalmente se consideran equivalentes es porque ambos métodos son buenos y tendrían la misma capacidad para segmentar.

Para visualizar estos resultados, se ha creado la tabla 4.1 con los métodos, sus promedios y los *p-values* correspondientes en orden descendente. En ella se puede ver que a medida que los promedios de las calidades disminuyen su p-valor es menor. Por lo que no hay riesgo de que rechaces la hipótesis nula, ya que esta no se cumple, entre los datos hay una diferencia significativa. Esta tabla ha sido acortada para ver los cuatro primeros métodos y los cuatro últimos respecto a promedio. El primer método no tiene un *p-values* en la tabla, ya que es el utilizado para realizar la prueba *t-student* pareada con el resto de métodos.

Tipo de cierre	Promedio	P-value
star(6)	0.234364	
star(5)	0.234064	0.233657
disk(6)	0.232591	0.056225
star(4)	0.231618	0.033107
diamond(3)	0.208818	0.002013
square(5)	0.207155	0.003623
square(4)	0.195418	0.001515
square(3)	0.181791	0.000559

Tabla 4.1: Análisis de resultados con *t-Student*, tabla acortada

Si fijamos por defecto el *p-value* en 0.05, vemos que los cierres *star(6), star(5)* y *disk(6)* son equivalentes y se podrían considerar los mejores métodos, ya que la probabilidad de rechazar la hipótesis nula, que sean equivalentes, es más alta que ese valor, 0.05. En contraposición, se puede identificar al método *square(3)* como peor, no hay riesgo de rechazar la hipótesis nula, es el menos equivalente en comparación con el mejor método.

Otro aspecto que podría ser interesante es la distribución de los datos de calidad. Para ello se han representado el mejor y el peor método de segmentación en forma de *box plot* y *violin plot*. 4.4

En ambos gráficos se presenta la distribución de los datos de calidad de los métodos *square(3)* y *star(6)*, peor y mejor método. Los representados en verdes se corresponden con la calidad de las sondas verdes, y los rojos con la de las rojas.

En ellos se comprueba que los resultados de calidad para sondas verdes son iguales, incluso un residuo cuya calidad es muy baja representado a modo de punto. Además se observa gran diferencia entre los métodos aplicados sobre sondas rojas.

Centrándonos en el *box plot* del peor método, se ve que su media si sitúa casi en el tercer cuartil, esto quiere decir que la mayoría de los datos son mayores que la media. Este hecho se puede comprobar en el *violin plot*, su forma se ensanchada al comenzar este tercer cuartil, los datos están concentrados en la parte superior del cuerpo del *box plot*. En cambio, en el mejor método, la media se encuentra en la mitad del cuerpo, indica que la distribución de los datos es más homogénea. Si comparamos los dos métodos entre sí, se ve que el peor método tiene valores más bajos, y el mejor valores más altos.

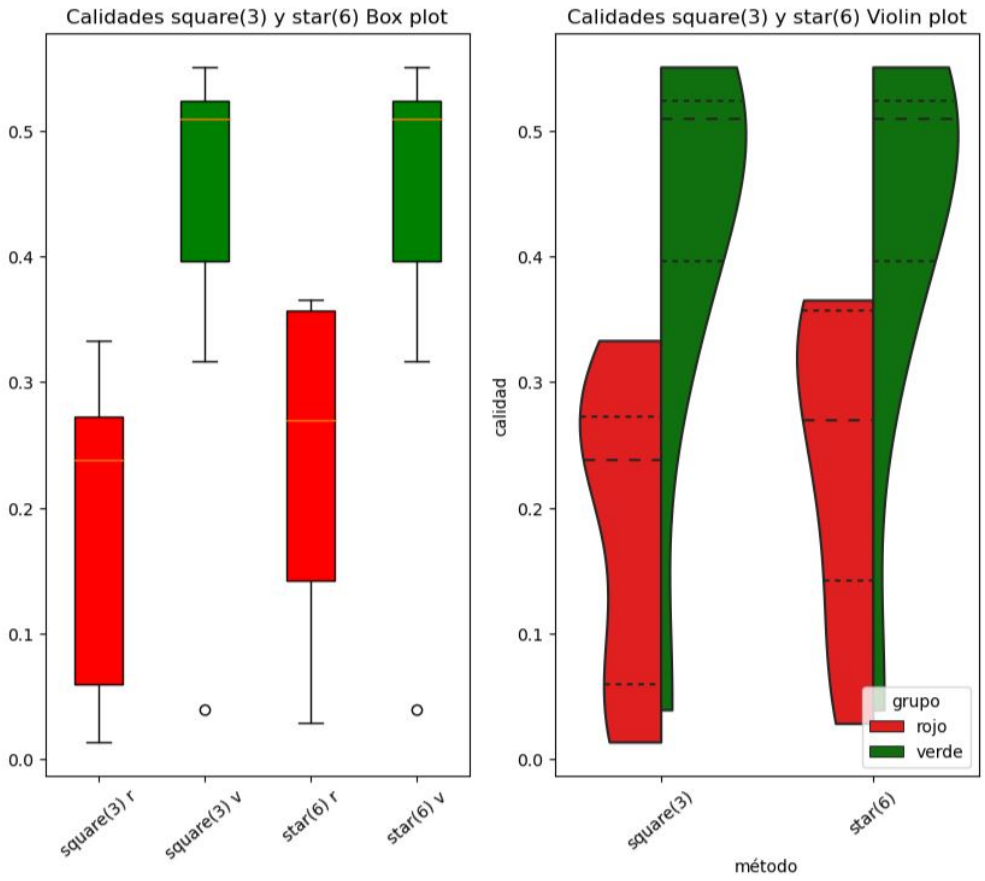


Figura 4.4: Representación gráfica de los resultados de calidad

4.3. Aspectos relevantes del desarrollo del proyecto.

Para comentar los aspectos relevantes, se va a dividir el proyecto en etapas o pasos que se han seguido.

Identificación de sondas

El primer paso en este proyecto es la identificación de sondas. La obtención de las imágenes de los tres canales de RGB, HSV y CIELab, es para que estas se puedan utilizar como base para identificar estas sondas. Es decir, que se van a probar sobre estas imágenes las diferentes técnicas de umbralización ya mencionadas.

Se comenzó a probar las técnicas de umbralización sobre la única imagen que en ese momento se tenía disponible, *imageFISH*. Y como mejores técnicas se seleccionaron: método *Yen* sobre la imagen del canal rojo de RGB para las sondas rojas y método *Yen* sobre la imagen del canal verde de RGB para las sondas verdes.

La dificultad llegó al disponer finalmente de imágenes muy dispares a la primera, por lo que se probaron las mismas técnicas y alguna otra alternativa sobre estas imágenes. Estas imágenes son las que contienen sondas azules y el fondo es rosado. Se vio que las técnicas seleccionadas funcionaban relativamente bien, pero el utilizar el canal azul para los puntos azules daba resultados mejores. En las imágenes diferentes, con sondas verdes, el canal azul es una imagen prácticamente en negro. Por lo que al final se tomó la decisión de aplicar las mismas técnicas a todas las imágenes, que resultaron ser las ya seleccionadas, ya que podían servir para los dos tipos de imágenes.

Como resultado de la umbralización se obtienen imágenes binarias, fondo negro y sondas detectadas en blanco. Estas imágenes se utilizaron como máscaras para ver los resultados sobre las imágenes reales. Cada imagen ha de tener dos imágenes binarias de umbralización, de puntos rojos y de puntos verdes/azules.

Filtrado por tamaño de sondas

Una vez se han conseguido detectar las sondas, en esta detección aparece ruido de fondo que no interesa. Por lo que se lleva a cabo un filtrado por tamaño de estas sondas, sobretodo para eliminar manchas de gran tamaño que también eran detectadas.

Este proceso se realiza sobre las imágenes binarias obtenidas de la umbralización. Este filtrado se realiza en función del área. Como las imágenes son muy diferentes entre sí, se cogieron unos valores que pudieran coger el mayor número de sondas posibles, aunque quedase algo de ruido. Estos valores son 150 y 0. El valor 0 hacía que algunas imágenes tuviesen mucho ruido, pero en otras conseguían dejar sondas importantes. Al final se priorizó la detección de sondas antes que filtrar todo el ruido.

Obtención de calidad

El siguiente paso fue obtener las calidades de la detección de sondas junto con el filtrado. Para ello se generaron las imágenes groundtruth correspondientes con las imágenes originales.

Estas imágenes groundtruth fueron generadas a mano mediante en programa *VGG Image Annotator*. Se fueron seleccionando de una a una y aplicando una etiqueta correspondiente al color de la sonda, esta era rojos o verdes. Una vez generadas las anotaciones estas se guardaron tanto en archivo JSON como en COCO. En este proyecto solo se utilizan los JSON para calcular la calidad, los COCO se generaron por si en algún momento del proyecto o en sus líneas futuras se utilizaban.

En procedimiento para obtener la calidad consistía en leer los archivos JSON, y comparar sus etiquetas con las imágenes binarias resultantes de la umbralización y el filtrado de sondas píxel a píxel. Esta comparación se realizó mediante el método intersección sobre unión, en el que los píxeles que coincidían en ambas partes, se dividían entre en total. Es decir, los píxeles identificados como sondas tanto en la imagen groundtruth como en la procesada, entre el total de píxeles identificado como sondas en ambas imágenes.

Un aspecto a destacar es la lectura de los archivos JSON, ya que convertían el fichero JSON en una imagen binaria. Al principio se vio que estos archivos generaban imágenes giradas e inversas en comparación con el resultado que debían dar. La solución consistió en cambiar los puntos de las coordenadas x con los de las coordenadas, leer los puntos de las coordenadas x como si fuesen de la y, y viceversa.

Para obtener los valores de calidad se ha utilizado el método de intersección sobre unión, entre la imagen groundtruth y la resultante de la segmentación. Estas imágenes han de ser independientes para cada color de sonda, por lo que los resultados de la segmentación se han tenido que dividir en cuanto a sondas verdes/azules y rojas. Para ello se han creado dos imágenes binarias de la segmentación, en función del tipo de sonda.

Este método de intersección sobre unión consiste en calcular los píxeles de la intersección entre ambas imágenes y dividirlo entre la unión, siempre va a ser un valor comprendido entre cero y uno.

Los primeros valores obtenidos fueron relativamente bajos, por lo que se empezaron a buscar soluciones para intentar filtrar con otras técnicas o parámetros las sondas identificadas.

Detección de células y de su contorno

Para eliminar puntos identificados como sondas, se decide seleccionar aquellas sondas que se encuentren en el interior de las células, ya que las

que están fuera no pueden ser sondas. Para ello se realiza una umbralización sobre la imagen original y sobre los canales rojos y verdes, para ver cual puede ser la umbralización con mejor resultado. En este caso la mejor técnica fue aplicar el método de umbralización de *Li* sobre el canal verde de las imágenes.

Para detectar los bordes de estas células, se probaron varios detectores de bordes. Finalmente se impuso el detector *Canny* como estándar para todas imágenes.

En las imágenes binarias resultantes de la detección celular, los bordes de las células eran muy difusos y con muchas curvas, además en el interior de las células aprecian huecos en negro no detectados como células, que generalmente eran sondas rojas. Para solucionar ambos problemas de aplico la operación morfológica de cierre sobre estas imágenes binarias.

En la operación de cierre se pueden aplicar múltiples formas y tamaños. Se decidió aplicar cuatro formas distintas (*disk, diamond, square* y *star*) y cuatro tamaños distintos (3, 4, 5 y 6) para ver que resultados se obtenían y cual podía ser el mejor. Además, para rellenar los huecos se probo una función específica junto con la operación cierre en sus cuatro formas y con el menor tamaño. Gráficamente, viendo las imágenes, era muy difícil ver cual sería mejor. Se tomo la decisión de generar una tabla con valores de calidad de cada forma con cada tamaño. Esta calidad se calcularía de la misma forma que en el apartado anterior.

Finalmente se obtuvieron dos tablas, una correspondiente a las sondas rojas y otra a las verdes/azules. En ella aparecían todos los métodos con sus diferentes tamaños.

Resultados de calidad

Al final, vemos en las tablas de calidad, que la tabla correspondiente a los sondas verdes no varía, esto es debido a que las sondas verdes no se encuentran ni en los bordes de las células ni en los huecos que se generan en ellas, por lo que todo el rato se identifican las mismas independientemente del cierre. En la tabla de calidad de sondas rojas es donde se ve esta variación, y por tanto va a ser la tabla utilizada para ver que forma y tamaño son los mejores.

Para identificar que método es el mejor o que métodos, se calcula el promedio de calidades por método. Una vez tenemos las calidades promedias ordenadas de mayor a menor. Se va comparando la mayor con el resto en

orden descendente mediante una *t-Student* pareada. Con esto conseguimos ver si las mejores técnicas son equivalentes entre sí o no, es decir, si se podría decir que todas tienen la misma eficacia o no aunque varíe levemente el valor promedio. Como mejor método se ha identificado el *star(6)* y como peor, el *square(3)*.

Para ver la distribución de los datos se van a comparar el mejor y el peor método entre sí, método con mayor promedio y con menor promedio. Esto se va a realizar mediante un *Box Plot* y un *violin Plot*.

Representación de resultados de imágenes

Para ver cual es el resultado de este procesado de imágenes se han creado tres funciones con desplegables para elegir los métodos y las imágenes a procesar para ver su resultado.

En una de ellas se tiene ya seleccionado internamente el tamaño y forma de cierre mejor, y solo se puede cambiar la imagen a la que se le aplica.

En otra función se puede elegir la imagen y la forma que se quiera ver, como resultado aparecerán los resultados de aplicar los cuatro tamaños disponibles en cuatro imágenes diferentes.

En la última, se puede elegir la imagen por un lado y por otro su forma y tamaño de manera conjunta.

Un aspecto a tener en cuenta es que tienen un tiempo de computo un tanto elevado, y una vez que una de ellas se ejecuta, para cambiar sus parámetros ha de ejecutarse de nuevo.

Lineas de trabajo futuras

Como líneas de trabajo futuras para este proyecto se proponen las siguientes:

- Uno de los aspectos que podría ser mejorado es el tiempo de ejecución. Las funciones para ver los resultados que se encuentran en el notebook *Estudio cierre resultados y análisis*, concretamente las funciones con desplegables tardan bastante en ser ejecutadas. Esto también ocurre al calcular las calidades de las sondas recogidas en las tablas de calidades.
- En estas mismas funciones de desplegables, al ser ejecutadas, se pueden volver a cambiar los valores de este desplegable. El principal problema, en las que tienen dos desplegables, es que si quieres cambiar ambos a la vez se ha de ejecutar la celda de nuevo. Puesto que si se cambia uno de los desplegables y luego el otro, se va a imprimir primero el resultado del primer desplegable cambiado con el valor antiguo del otro, y después los dos valores nuevos. Esto hace que el tiempo de ejecución se doble.
- Otro aspecto que puede ser determinante en cuanto a resultados, son las formas y tamaños que han sido analizados para la función cierre, en el notebook *Estudio cierre resultados y análisis*. Se han utilizado cuatro formas diferentes, con cuatro tamaños diferentes y con una función específica de rellenar huecos. Hay más formas y tamaños que pueden ser probados, por lo que esto ha podido generar cierto sesgo a la hora de obtener resultados, ya que puede haber mejores resultados que no se han llegado a probar.
- Un punto muy importante dentro de la programación es intentar no tener líneas de código repetidas. Esto puede verse en la carga de

imágenes de la mayoría de los notebooks, al igual que al generar los canales de esas imágenes.

- Todos los procesos de procesado de imágenes dependen mucho de las características de las propias imágenes. En este proyecto se tienen dos tipos diferentes de imágenes, fondo naranja y sondas verdes o fondo rosa y sondas azules. Se han intentado generalizar las técnicas para que sean utilizadas por igual en ambos tipos de imágenes. Sin embargo, hay ciertas imágenes cuyos resultados podrían ser mucho mejores aplicando otras técnicas que sean más específicas. Una posible mejora sería procesar imágenes que sean todavía más diferentes, para intentar conseguir una técnica aplicable a imágenes *FISH* en general, o al contrario, intentar centrarse por separado en cada tipo de imágenes, teniendo varios procesos diferentes en función de ellas.

- Una línea para seguir trabajando es la detección de células. Se consiguen detectar la gran mayoría de ellas, pero de forma aglutinada. Al superponerse unas con otras o estar muy juntas, como resultado se obtiene una masa de células. Por lo que se propone probar algún método que distinga las células, y se puedan identificar como únicas. Una posible solución es la aplicación del algoritmo *watershed*, solución muy utilizada en trabajos relacionados.

- El siguiente paso a seguir una vez se tengan las células separadas, es el conteo de sondas. Para lograr la automatización diagnostica de este tipo de cáncer neuronal, el propio programa (implementación) tiene que ser capaz de indicar si existe o no deleción o co-deleción. Esto se puede conseguir con el recuento de sondas por célula. Este recuento se realizaría mientras se encuentren dos o más sondas de control en la misma célula, si no se clasificaría como no concluyente. Los posibles resultados al realizar un conteo de sondas vienen recogidos en el anexo *Descripción de adquisición y tratamiento de datos*, concretamente en el apartado *Descripción clínica de los datos*, donde se recoge la explicación biológica de las imágenes y los posibles casos que se pueden dar.

- Otro factor que podrían ser analizado de las imágenes de *FISH*, es su disposición tridimensional. Si las imágenes son representadas en 3D, con el eje Z se podrían sacar conclusiones sobre el tamaño y forma de las sondas, pudiendo establecer otro criterio de filtrado para detectar sondas y eliminar los puntos de ruido detectados como estas. De esta forma se podría dar solución al problema de detectar sondas

que realmente no lo son, o utilizar diferentes tipos de filtrado para ellas.

Bibliografía

[1] Labelbox. https://labelbox.com/.

[2] Project Jupyter Revision 892166dd. Jupyter widgets, 2019. https://ipywidgets.readthedocs.io/en/7.x/examples/Widget%20List.html.

[3] A.S. Abdul Nasir, M.Y. Mashor, and Z. Mohamed. Segmentation based approach for detection of malaria parasites using moving k-means clustering. In *2012 IEEE-EMBS Conference on Biomedical Engineering and Sciences*, pages 653–658, 2012.

[4] Andrew Zisserman Abhishek Dutta, Ankush Gupta. Vgg image annotator (via). https://www.robots.ox.ac.uk/~vgg/software/via/.

[5] American Brain Tumor Association. Oligodendroglioma y oligoastrocitoma. 2018. https://www.abta.org/wp-content/uploads/2018/03/oligodendroglioma-oligoastrocitoma.pdf.

[6] Jeffrey A. Clark. Pillow (pil fork) documentation. 2023. https://pillow.readthedocs.io/en/stable/reference/Image.html.

[7] The SciPy community. Multidimensional image processing (scipy.ndimage), 2023. https://docs.scipy.org/doc/scipy/reference/ndimage.html.

[8] The Scipy community. scipy.stats.ttest_rel, 2023. https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.ttest_rel.html.

[9] Pieter Wesseling Daniel J Brat Ian A Cree Dominique Figarella-Branger Cynthia Hawkins H K Ng Stefan M Pfister Guido Reifenberger Riccardo Soffietti Andreas von Deimling David W Ellison David N Louis, Arie Perry. The 2021 who classification of tumors of the central nervous system: a summary. *Neuro-Oncology*, 23, 2021.

[10] Equipo de redactores y equipo de editores médicos de la Sociedad Americana Contra El Cáncer. Tasas de supervivencia de ciertos tumores de encéfalo y tumores de médula espinal en adultos, 2020. https://www.cancer.org/es/cancer/tipos/tumores-de-encefalo-o-de-medula-espinal/deteccion-diagnostico-clasificacion-por-etapas/tasas-de-supervivencia.html.

[11] NumPy Developers. numpy.logical_and, 2022. https://numpy.org/doc/stable/reference/generated/numpy.logical_and.html.

[12] NumPy Developers. numpy.logical_or, 2022. https://numpy.org/doc/stable/reference/generated/numpy.logical_or.html.

[13] The Matplotlib development team. matplotlib.pyplot.violinplot, 2023. https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.violinplot.html.

[14] Doxygen. Opencv. open source computer vision. canny edge detection, 2023. https://docs.opencv.org/3.4/da/d22/tutorial_py_canny.html.

[15] Doxygen. Opencv. open source computer vision. image filtering. image processing, 2023. https://docs.opencv.org/3.4/d4/d86/group__imgproc__filter.html#gacea54f142e81b6758cb6f375ce782c8d.

[16] Doxygen. Opencv. open source computer vision. image thresholding, 2023. https://docs.opencv.org/4.x/d7/d4d/tutorial_py_thresholding.html.

[17] J. Gellin Y. Lahbib-Mansais M. Yerle T. Boudier E. Iannuccelli, F. Mompart. Nemo: a tool for analyzing gene and chromosome territory distributions from 3d-fish experiments. *Bioinformatics*, 26(5):696–697, 2010.

[18] M.A. Padilla Castañeda S. Solano P. Tato F. Arámbula Cosío, J.A. Márquez Flores. Automatic analysis of immunocytochemically stained tissue samples. *Medical and Biological Engineering and Computing*, 43(5):672–677, 2005.

[19] Python Software Foundation. json — json encoder and decode, 2023. https://docs.python.org/3/library/json.html.

[20] Python Software Foundation. labelimg 1.8.6, 2023. https://pypi.org/project/labelImg/.

[21] Scikit image development team. Scikit-image. module: morphology, 2022. https://scikit-image.org/docs/stable/api/skimage.morphology.html.

[22] Scikit image development team. Scikit-image. module: color, 2023. https://scikit-image.org/docs/stable/api/skimage.color.html#skimage.color.rgb2lab.

[23] Scikit image development team. Scikit-image. module: feature, 2023. https://scikit-image.org/docs/stable/api/skimage.feature.html.

[24] Scikit image development team. Scikit-image. module: filters, 2023. https://scikit-image.org/docs/stable/api/skimage.measure.html.

[25] Scikit image development team. Scikit-image. module: measure, 2023. https://scikit-image.org/docs/stable/api/skimage.filters.html.

[26] Petros S. Karvelis, Alexandros T. Tzallas, Dimitrios I. Fotiadis, and Ioannis Georgiou. A multichannel watershed-based segmentation method for multispectral chromosome classification. *IEEE Transactions on Medical Imaging*, 27(5):697–708, 2008.

[27] Satoko Moriwaki Lars Martin Jakt. Extended multiplexed fluorescent in situ hybridization by combinatorial encoding of individual transcripts. *Neuromethods. In Situ Hybridization Methods*, 99:509–537, 2015.

[28] Scikit learn developers. sklearn.cluster.kmeans, 2023. https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html.

[29] Salar Razavi, Gökhan Hatipoğlu, and Hülya Yalçın. Automatically diagnosing her2 amplification status for breast cancer patients using large fish images. In *2017 25th Signal Processing and Communications Applications Conference (SIU)*, pages 1–4, 2017.

[30] Adrian Rosebrock. Opencv color spaces (cv2.cvtcolor), 2021. https://pyimagesearch.com/2021/04/28/opencv-color-spaces-cv2-cvtcolor/.

[31] S. Osowski M. Jesiotr W. Koslowski T. Les, T. Markiewicz. Localization of spots in fish images of breast cancer using 3-d shape analysis. *Journal of Microscopy*, 262(3):252–259, 2015.

[32] Raphael Vallat. pingouin.ttest, 2022. https://pingouin-stats.org/build/html/generated/pingouin.ttest.html.

[33] Michael Waskom. seaborn.violinplot, 2022. https://seaborn.pydata.org/generated/seaborn.violinplot.html.