



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



INGENIERÍA
DE LA SALUD

**TFG del Grado en Ingeniería de la
Salud**

**Segmentación automática de
células en imágenes FISH
Documentación Técnica**

Presentado por Walid Sabhi
en Universidad de Burgos

12 de febrero de 2025

Tutores: José Francisco Díez Pastor – Pedro Latorre
Carmona

Índice general

I

Apéndice E Descripción de Adquisición y Tratamiento de Datos	33
E.1. Descripción Formal de los Datos	33
E.2. Descripción Clínica de los Datos	33
Apéndice F Manual de Especificación de Diseño	35
F.1. Diseño Arquitectónico	35
Apéndice G Especificación de Requisitos	37
G.1. Diagrama de Casos de Uso	37
G.2. Explicación Casos de Uso	38
G.3. Prototipos de Interfaz o Interacción con el Proyecto	42
Apéndice H Estudio Experimental	45
H.1. Cuaderno de Trabajo	45
H.2. Configuración y Parametrización de las Técnicas	46
H.3. Detalle de Resultados	47
Apéndice I Anexo de Sostenibilización Curricular	49
I.1. Introducción	49
Bibliografía	51

Índice de figuras

A.1. Calendario de reuniones realizadas.	2
A.2. Metodología SCRUM	5
A.3. Issues 1	6
A.4. Issues 2	6
C.1. Carpeta <code>ejecucion_local</code> subida a Google Drive.	18
C.2. Configuración de GPU en Google Colab.	19
C.3. Ejecución de comandos en Google Colab para iniciar el API. . .	20
C.4. Actualización de la URL del API en el archivo <code>ejecucion.py</code> . .	20
C.5. Arranque del fichero ejecución para la obtención de predicciones.	21
C.6. Pantalla de inicio.	24
C.7. Proceso de carga de la imagen original.	24
C.8. Proceso de obtención de la imagen segmentada y del número de células detectadas.	25
C.9. Proceso de obtención de la imagen analizada, que presenta las células con contornos de distintos colores.	25
C.10. Visualización de las características de una célula seleccionada. .	26
D.1. Librerías requeridas.	31
D.2. Carpetas requeridas.	32
F.1. Diagrama de Clases.	36
F.2. Diagrama de Despliegue.	36
G.1. Diagrama de casos de uso.	37

Índice de tablas

A.1. Resumen de costes sin impuestos	8
A.2. Resumen de costes con impuestos	9
G.1. CU-1: Carga de imágenes FISH para segmentación.	39
G.2. Lista de Actores.	40
G.3. Lista de Casos de Uso.	40
G.4. Flujo de Eventos Principal.	40
G.5. Requisitos del Sistema.	41
G.6. Escenarios Alternativos del Caso de Uso.	41
G.7. Mapeo de Casos de Uso y Requisitos.	42
G.8. Mapeo de Casos de Uso y Requisitos.	42

Apéndice A

Plan de Proyecto Software

A.1. Introducción

Este anexo detalla el plan de proyecto para el desarrollo del software, diseñado para satisfacer las necesidades identificadas durante el análisis y el diseño conceptual de la solución. Ofreciendo una guía estructurada que abarca la planificación temporal, la planificación económica y la viabilidad legal, con el objetivo de coordinar eficazmente las actividades, asignar recursos y asegurar el cumplimiento de los plazos establecidos. La metodología adoptada garantiza la transparencia en cada fase del proyecto, permitiendo a todos los integrantes mantenerse informados.

A.2. Planificación Temporal

Para la gestión y desarrollo de la plataforma se adoptó la metodología Scrum [Scrum.org, 2025], lo que facilitó un trabajo colaborativo y una rápida adaptación a los desafíos propios de proyectos complejos. El proyecto se estructuró en cinco etapas, integradas en un total de 18 sprints (reuniones) realizados desde septiembre hasta mediados de febrero (Figura A.1).

El seguimiento sistemático se realizó a través de Microsoft Teams y GitHub (<https://github.com/flooki10/TFG>), lo que permitió gestionar issues, proponer mejoras e implementar cambios mediante labels y milestones. Cada sprint se planificó cuidadosamente con objetivos específicos, lo que permitió evaluar el progreso y ajustar el curso según las necesidades emergentes.

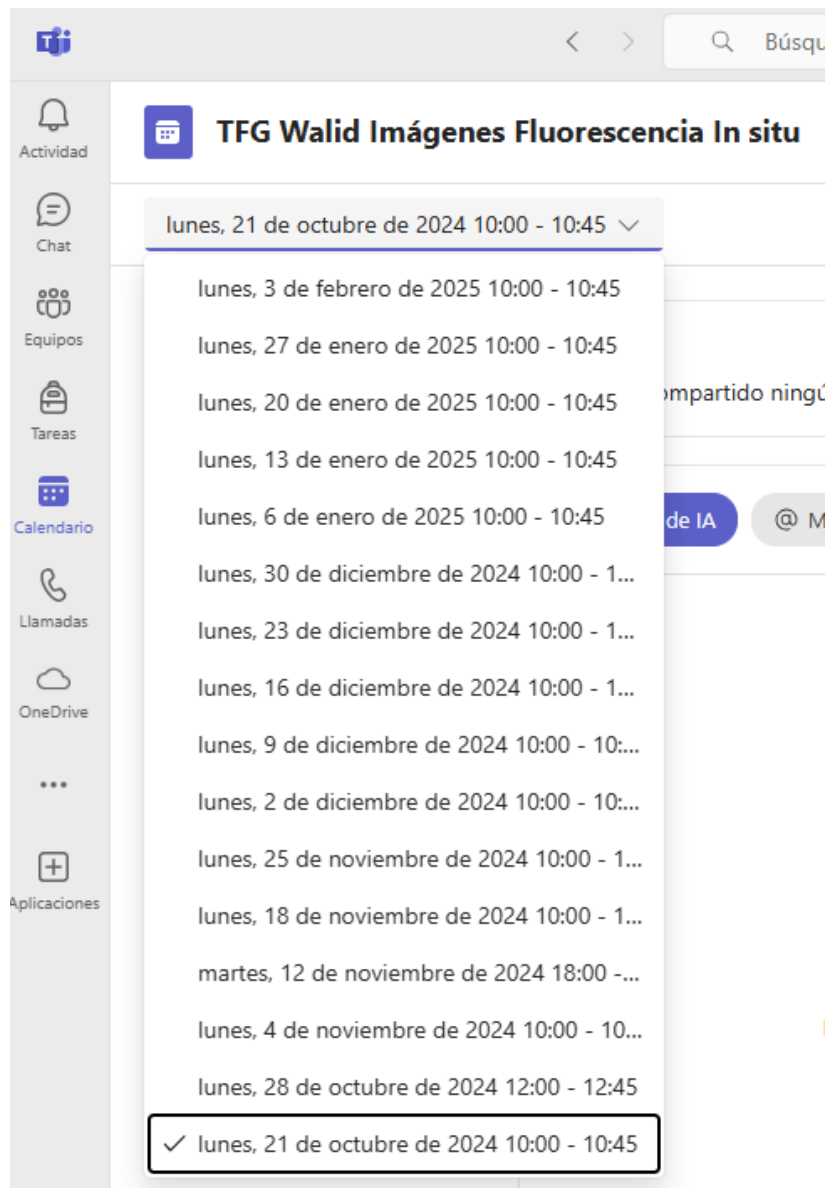


Figura A.1: Calendario de reuniones realizadas.

■ Etapa 1: Definición y Planificación Inicial

Duración: Septiembre 2024 – Octubre 2024

Objetivo: Establecer el alcance del proyecto y definir los requerimientos iniciales, explorando alternativas para abordar el problema de segmentación ante la limitada disponibilidad de imágenes [Pérez, 2025].

Sprints:

- **Sprint 1 (1–14 de septiembre 2024):** Realizar reuniones iniciales para definir el alcance del proyecto y revisar posibles modelos de segmentación.
- **Sprint 2 (15–28 de septiembre 2024):** Investigar y discutir alternativas para la segmentación [IBM, 2025], analizando algoritmos viables y recopilando información sobre procesamiento de imágenes.
- **Sprint 3 (29 de septiembre – 15 de octubre 2024):** Consolidar las decisiones tomadas, definir los criterios de evaluación y preparar las pruebas preliminares para la siguiente fase.

■ **Etapla 2: Desarrollo y Pruebas de Concepto**

Duración: Octubre 2024 – Noviembre 2024

Objetivo: Validar conceptos iniciales en torno a la segmentación de imágenes mediante pruebas preliminares con diversos algoritmos (por ejemplo, nn-UNet [MIC-DKFZ, 2025], MEDIAR [Gihun, 2025] y otros modelos disponibles) que permitan identificar y diferenciar células, incluso en casos de solapamiento.

Sprints:

- **Sprint 4 (16–22 de octubre 2024):** Implementar pruebas con algoritmos básicos de segmentación y recopilar datos preliminares para validar la factibilidad.
- **Sprint 5 (23–29 de octubre 2024):** Comparar el desempeño de diferentes modelos y evaluar su capacidad para distinguir células en situaciones de solapamiento.
- **Sprint 6 (30 de octubre – 5 de noviembre 2024):** Refinar parámetros iniciales y evaluar la asignación de valores de gris en las imágenes segmentadas.
- **Sprint 7 (6–15 de noviembre 2024):** Validar la efectividad de las técnicas evaluadas y seleccionar la metodología más prometedora.

■ **Etapla 3: Integración Avanzada y Optimización de la Segmentación**

Duración: Noviembre 2024 – Enero 2025

Objetivo: Investigar y aplicar técnicas de fine-tuning [Kumar, 2025], junto al modelo MEDIAR, para diseñar, ajustar y optimizar un modelo que ofrezca un rendimiento eficiente y preciso en la segmentación de

imágenes.

Sprints:

- **Sprint 8 (16–30 de noviembre 2024):** Revisar el desempeño de los modelos preliminares y definir criterios claros para su optimización.
- **Sprint 9 (1–15 de diciembre 2024):** Implementar técnicas de fine-tuning y realizar pruebas iniciales de integración con el modelo MEDIAR.
- **Sprint 10 (16–20 de diciembre 2024):** Optimizar el modelo mediante ajustes en sus hiperparámetros [AWS, 2025a] y evaluar mejoras en precisión y eficiencia.
- **Sprint 11 (21–31 de diciembre 2024):** Consolidar la integración del modelo optimizado y realizar pruebas comparativas de rendimiento con los datos disponibles.

■ **Etap 4: Desarrollo de la Interfaz y Funcionalidades Básicas**

Duración: Enero 2025 – Febrero 2025

Objetivo: Implementar la primera versión de la interfaz de usuario y desarrollar las funcionalidades esenciales de la plataforma, desde la carga de imágenes hasta la visualización y exportación de los resultados de segmentación.

Sprints:

- **Sprint 12 (1–6 de enero 2025):** Diseñar la interfaz de usuario, definiendo el flujo de navegación y la arquitectura básica del sistema.
- **Sprint 13 (1–3 de enero 2025):** Desarrollar funcionalidades para la carga y visualización de imágenes, realizando pruebas iniciales de usabilidad.
- **Sprint 14 (3–6 de enero 2025):** Implementar y testear las funciones de procesamiento y segmentación, asegurando la integración del modelo optimizado.
- **Sprint 15 (6–15 de enero 2025):** Desarrollar la funcionalidad de exportación de imágenes y validar el flujo completo del sistema.

■ **Etap 5: Testeo, Validación y Despliegue**

Duración: Febrero 2025

Objetivo: Realizar pruebas exhaustivas del sistema, identificar y corregir errores, y preparar la plataforma para su despliegue en producción.

Se ejecutaron sesiones de beta testing [ProductPlan, 2025] con usuarios finales para evaluar la usabilidad y efectividad, permitiendo ajustes finales tanto en la interfaz como en el algoritmo de segmentación.

Sprints:

- **Sprint 16 (20–27 de enero 2025):** Ejecutar pruebas funcionales y de usabilidad en entorno controlado, recopilando feedback inicial.
- **Sprint 17 (27–3 de febrero 2025):** Implementar correcciones basadas en el feedback obtenido y optimizar la integración de la interfaz y el algoritmo.
- **Sprint 18 (3–10 de febrero 2025):** Realizar pruebas finales de validación y coordinar el despliegue de la plataforma en producción.

SCRUM FRAMEWORK

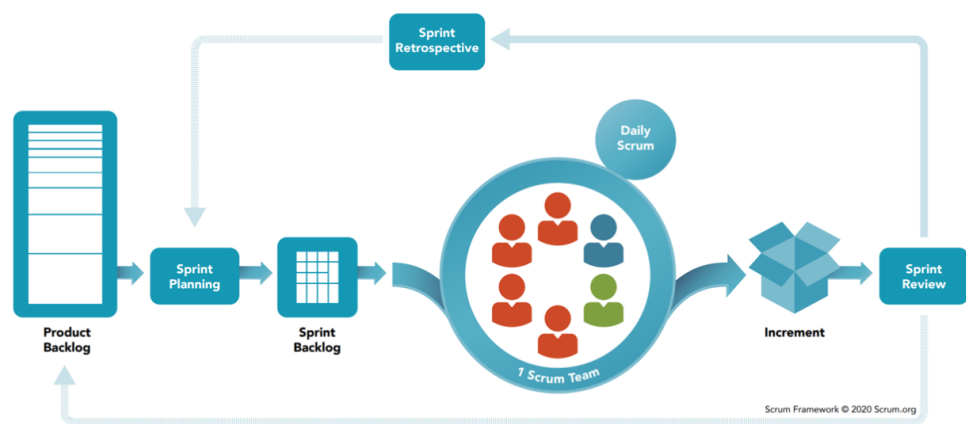


Figura A.2: Metodología SCRUM

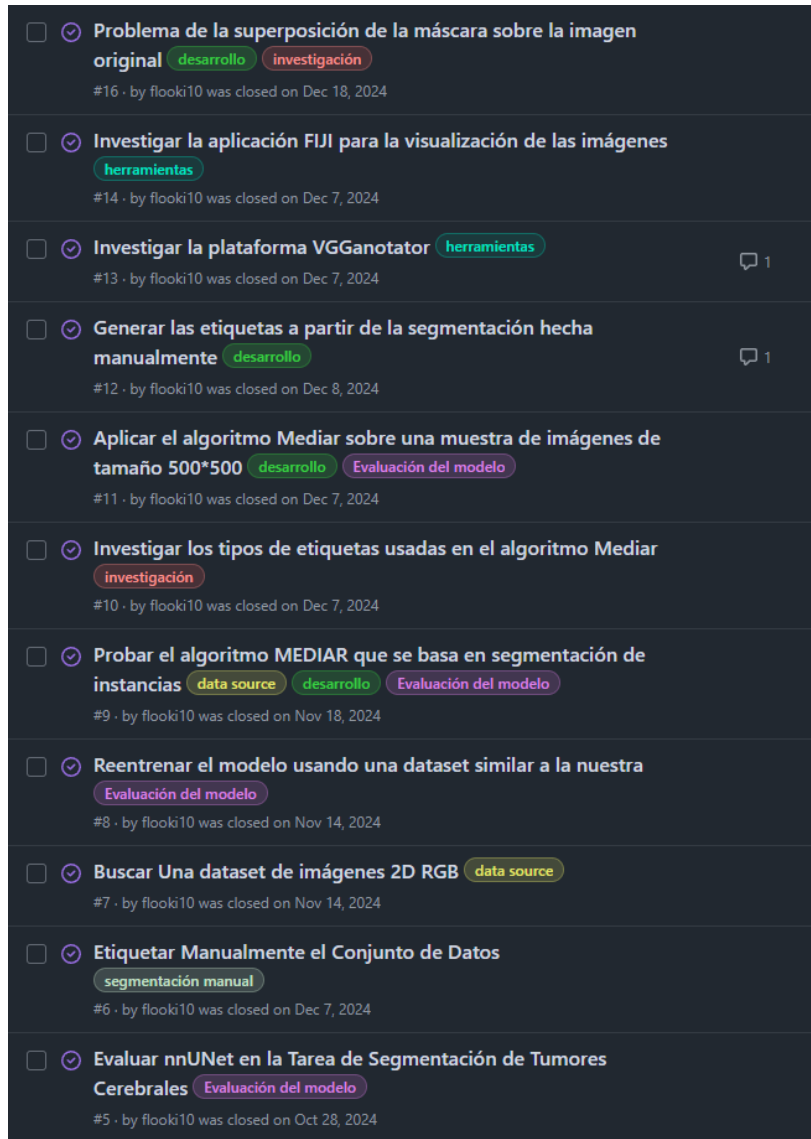


Figura A.3: Issues 1



A.3. Planificación económica

La planificación económica del proyecto contempla una estimación detallada de los costes asociados a los recursos humanos, infraestructura tecnológica, consumo energético y suscripciones a herramientas esenciales para el desarrollo del sistema.

Costes Desglosados

1. Recursos Humanos

Se estima un total de horas trabajadas calculadas a partir de una media de 4 horas diarias durante 6 meses, con un coste por hora de 20 euros:

$$4 \text{ horas/día} \times 30 \text{ días/mes} \times 6 \text{ meses} = 720 \text{ horas}$$

$$720 \text{ horas} \times 20 \text{ euros/hora} = 14,400 \text{ euros}$$

2. Infraestructura Tecnológica

Costes estimados para hardware y servicios en la nube:

- **Portátil y periféricos:** 350 euros
- **Suscripción a Google Colab:** Dos meses de suscripción al plan premium (11 euros/mes) para acceder a GPU T4 y TPU:

$$2 \times 11 \text{ euros} = 22 \text{ euros}$$

3. Costes Energéticos

El consumo estimado del portátil durante 720 horas de trabajo, con un consumo de 180 vatios, se calcula de la siguiente manera:

$$180 \text{ W} \times 720 \text{ horas} = 129,600 \text{ W-h} = 129,6 \text{ kWh}$$

Con un coste de 0,14 euros por kWh, el coste total de electricidad es:

$$129,6 \text{ kWh} \times 0,14 \text{ euros/kWh} = 18,14 \text{ euros}$$

Considerando un uso eficiente del 20 %, el coste ajustado es:

$$18,14 \text{ euros} \times 0,20 = 3,63 \text{ euros}$$

4. Costes de Internet

El coste anual del servicio de internet asciende a 375 euros. Considerando un uso compartido entre 4 miembros del hogar durante 720 horas:

$$375 \text{ euros} \div 4 \times \frac{720 \text{ horas}}{365 \times 24 \text{ horas}} \approx 7,39 \text{ euros}$$

5. Costes de Software y Licencias

Licencias y herramientas adicionales: 0 euros.

Resumen de Costes Sin Impuestos

Concepto	Cantidad (€)
Salarios	14.400,00
Portátil y periféricos	350,00
Google Colab	22,00
Electricidad (129,6 kWh a 0,14 €/kWh)	3,63
Internet	7,39
Licencias y herramientas	0,00
Subtotal Costes (sin impuestos)	14.783,02

Tabla A.1: Resumen de costes sin impuestos

Impuestos y Cotizaciones

- **IRPF (16,26 % sobre ingresos brutos):** 2.401,58 euros
- **Contingencias comunes (4,82 % sobre ingresos brutos):** 712,57 euros
- **Cotización por formación (0,10 %):** 14,78 euros
- **Cotización desempleo (1,55 %):** 229,14 euros

Resumen de Costes con Impuestos

Concepto	Cantidad (€)
Subtotal Costes (sin impuestos)	14.783,02
IRPF (16,26 %)	2.401,58
Contingencias comunes (4,82 %)	712,57
Cotización por formación (0,10 %)	14,78
Cotización desempleo (1,55 %)	229,14
Total Coste (con impuestos)	18.141,09

Tabla A.2: Resumen de costes con impuestos

A.4. Viabilidad Legal

El proyecto ha considerado cuidadosamente las implicaciones legales derivadas del uso de inteligencia artificial (IA) [AWS, 2025b] y la manipulación de imágenes médicas de pacientes reales, con el objetivo de garantizar el cumplimiento normativo, proteger los derechos de los pacientes y minimizar riesgos legales. A continuación, se detallan los aspectos clave:

Cumplimiento Normativo

El desarrollo del proyecto se ha alineado con las normativas nacionales e internacionales aplicables al tratamiento de datos de salud, que son especialmente sensibles por su naturaleza. Las disposiciones clave consideradas incluyen:

- **Reglamento General de Protección de Datos (GDPR, UE 2016/679):** [del Estado, 2025a] Asegura la legalidad del tratamiento de datos personales en la Unión Europea, imponiendo requisitos sobre el consentimiento explícito, la minimización de datos y la seguridad del tratamiento.
- **Ley Orgánica de Protección de Datos y Garantía de los Derechos Digitales (LOPD-GDD):** [del Estado, 2025b] Aplicación específica en España para proteger los datos personales de los pacientes.
- **Normas ISO 27001 e ISO 27701:** [ISO, 2025] Lineamientos internacionales sobre seguridad de la información y privacidad de datos.

Se han implementado las siguientes medidas para garantizar el cumplimiento normativo:

- Recopilación de datos limitada únicamente a la información necesaria para el análisis y segmentación de imágenes.
- Anonimización de las imágenes de pacientes para evitar la identificación directa de los individuos.
- Almacenamiento seguro de las imágenes y datos relacionados, mediante mecanismos de cifrado y control de acceso.

Uso Responsable de la Inteligencia Artificial

El proyecto ha implementado la IA para el análisis y segmentación de imágenes médicas. Conscientes de los riesgos éticos y legales asociados, se han adoptado las siguientes prácticas:

- Transparencia en el funcionamiento del algoritmo, proporcionando explicaciones comprensibles para el personal médico sobre el proceso de segmentación y clasificación de células.
- Evaluación continua del rendimiento y la precisión del modelo para garantizar resultados fiables y clínicamente válidos.
- Inclusión de controles manuales para que los profesionales de la salud puedan verificar, corregir y validar las segmentaciones automáticas generadas por el modelo.
- Exclusión explícita de decisiones clínicas automáticas, dejando siempre la responsabilidad final en manos de un profesional cualificado.

Propiedad Intelectual

El software desarrollado, incluidas las implementaciones de IA, ha sido debidamente protegido mediante:

- Registro de la propiedad intelectual del código fuente, algoritmos y diseños de la plataforma.
- Licenciamiento adecuado de las bibliotecas de software de terceros utilizadas en el desarrollo.

- Atribución correcta de derechos en casos de colaboración con otros investigadores o entidades externas.

Protección de la Información Médica

El tratamiento de imágenes de pacientes reales conlleva obligaciones adicionales para proteger la privacidad y confidencialidad de la información. Para mitigar riesgos, se han aplicado las siguientes medidas:

- Implementación de un sistema de acceso local.

Contratos y Acuerdos Legales

Para asegurar la legalidad de las relaciones con terceros, se han formalizado:

- **Contratos de Confidencialidad (NDA):** [\[Wikipedia, 2025a\]](#) Protegen la información confidencial del proyecto, incluyendo datos médicos y el funcionamiento interno de la IA.
- **Acuerdos de Colaboración:** Definen las condiciones de uso de la tecnología desarrollada, así como los derechos y obligaciones de cada parte.
- **Términos de Uso y Política de Privacidad:** Aseguran que los usuarios de la plataforma comprendan cómo se procesan sus datos y las limitaciones del software.

Revisión Legal Continua

Conscientes de la evolución constante del marco normativo en materia de IA y protección de datos, el proyecto cuenta con un plan de revisión legal periódica que incluye:

- Consultas con asesores legales especializados en salud digital y tecnología.
- Actualización de políticas internas para reflejar cambios legislativos.
- Revisión de licencias de software y documentación contractual.

Estos elementos aseguran que el desarrollo del software se lleve a cabo dentro de un marco legal robusto, minimizando riesgos jurídicos y garantizando el respeto a los derechos de los pacientes y usuarios. Además, fortalecen la credibilidad del proyecto en el ámbito sanitario y facilitan su posterior adopción y comercialización.

Apéndice B

Documentación de Usuario

En este apéndice se proporciona información detallada para la correcta instalación, configuración y uso del proyecto. Se incluyen los requisitos de hardware y software, instrucciones para la puesta en marcha y guías de usuario.

B.1. Requisitos Software y Hardware para Ejecutar el Proyecto

Ejecución en Local (Recomendado para Usuarios con GPU Compatible)

Si el usuario dispone de un ordenador con una tarjeta gráfica dedicada compatible con CUDA, se recomienda ejecutar el proyecto de forma local para obtener un mejor rendimiento y tiempos de procesamiento más cortos. Las especificaciones mínimas son:

- **Procesador:** Intel Core i5 (o superior) para tareas generales.
- **Memoria RAM:** 8 GB (recomendado 16 GB para procesamiento intensivo).
- **Almacenamiento:** Al menos 10 GB de espacio libre en disco.
- **Tarjeta Gráfica:** NVIDIA con soporte CUDA [[Wikipedia, 2025b](#)] (por ejemplo, GTX 1060 o superior). La versión del controlador CUDA debe ser compatible con TensorFlow o PyTorch.

- **Resolución de Pantalla:** 1920x1080 píxeles.

En este caso, es posible ejecutar el modelo de segmentación en el equipo local sin necesidad de recurrir a Google Colab.

Ejecución en Google Colab (Para Usuarios sin GPU [Wikipedia, 2025c] Compatible)

Para usuarios que no dispongan de una tarjeta gráfica compatible con CUDA, se recomienda utilizar Google Colab. Esta plataforma proporciona acceso a recursos computacionales con soporte para GPU de forma gratuita.

Requisitos mínimos del ordenador para usar Google Colab:

- **Procesador:** Intel Core i3 o equivalente (cualquier procesador moderno).
- **Memoria RAM:** 4 GB (mínimo).
- **Conexión a Internet:** Estable y de al menos 10 Mbps para la transferencia de archivos y acceso al entorno de Colab.

Requisitos de Software

El entorno de ejecución debe contar con:

- **Sistema Operativo:** Windows 10/11, macOS 10.15 o superior, Ubuntu 20.04 o superior.
- **Python:** Versión 3.8 o superior.
- **Librerías Python:** [NumPy, 2025], [pandas, 2025], [matplotlib, 2025]
Se recomienda instalar las dependencias mediante un archivo `requirements.txt`, que incluye las siguientes bibliotecas:
 - `numpy`, `pandas`, `matplotlib`
 - `scikit-learn`, `opencv-python`
 - `torch`, `monai`, `segmentation-models`
- **Entorno Virtual (opcional):** Uso de `virtualenv` o `conda` para aislar las dependencias del proyecto y facilitar la gestión de paquetes.

- **Servidor Web Backend:** Flask [Project, 2025] ha sido empleado para proporcionar y gestionar el API que interactúa con el frontend.
- **Interfaz de Usuario (Frontend):** Creada mediante CustomTkinter [Schimansky, 2025] para ofrecer una aplicación de escritorio con componentes gráficos personalizados y una interfaz moderna.

Apéndice C

Documentación de Usuario

C.1. Instalación / Puesta en marcha

Para poner en marcha el proyecto, siga el siguiente procedimiento. Este flujo está diseñado para facilitar el uso de la carpeta `ejecucion_local` la cual contiene todo lo necesario para obtener predicciones (sin incluir el entrenamiento del modelo) tanto para usuarios que ejecuten el proyecto localmente con GPU compatible como para aquellos que utilicen Google Colab para acceder a la GPU T4.

Paso 1: Descarga de la Carpeta de Ejecución

Descargue la carpeta `ejecucion_local` desde el repositorio. Esta carpeta ha sido preparada específicamente para simplificar el proceso de obtención de predicciones.

Paso 2: Subida de la Carpeta a Google Drive

Suba la carpeta `ejecucion_local` a su cuenta de Google Drive para poder acceder a ella desde Google Colab.

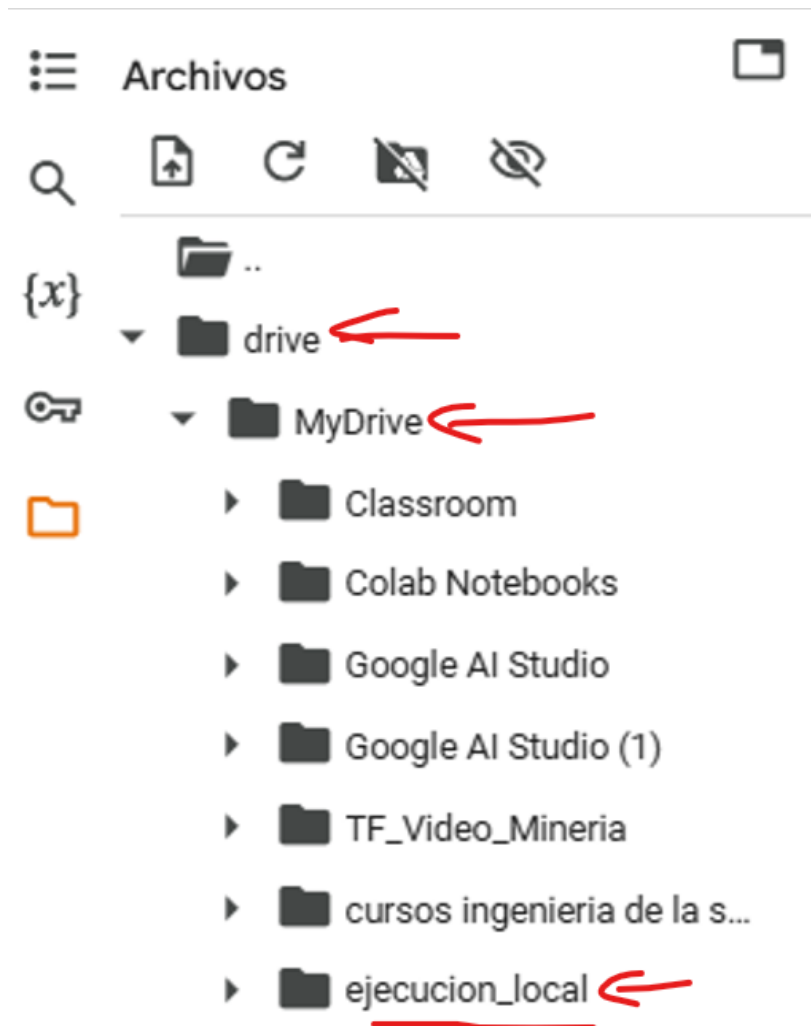


Figura C.1: Carpeta `ejecucion_local` subida a Google Drive.

Paso 3: Configuración en Google Colab

Abra una nueva notebook en Google Colab y realice los siguientes pasos:

1. Diríjase a **Entorno de ejecución** > **Cambiar tipo de entorno de ejecución**.
2. Seleccione **GPU** como acelerador de hardware (se recomienda la GPU T4).

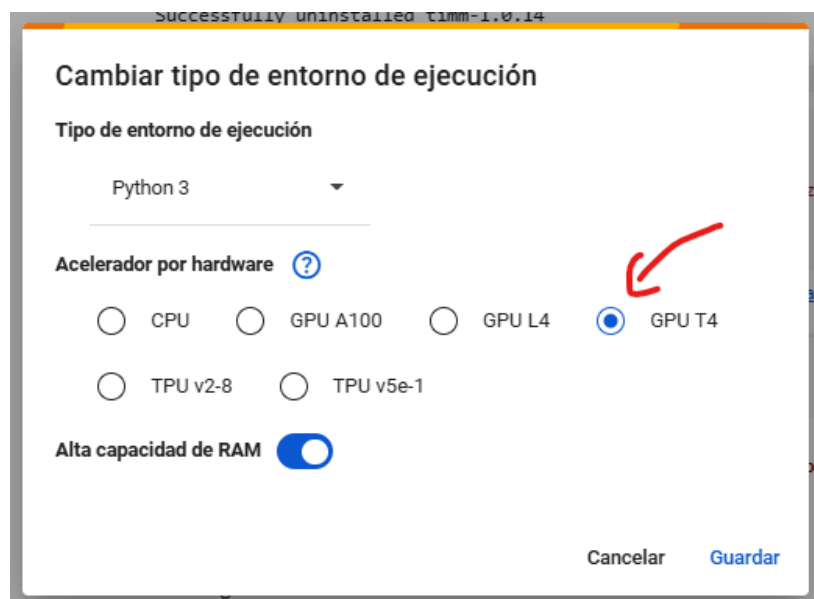


Figura C.2: Configuración de GPU en Google Colab.

Paso 4: Ejecución de Comandos en Google Colab

Con la GPU habilitada, ejecute los siguientes comandos en una celda de la notebook para montar su Google Drive, instalar las dependencias y arrancar el servidor de API:

```
%pip install -r /content/drive/MyDrive/ejecucion_local/requirements.txt

from pyngrok import ngrok
ngrok.kill()
ngrok.set_auth_token("2s32HMkIHWe1r8kggiadWbdCdw8_2nBaywQVV76JBnLUzMc4b")
public_url = ngrok.connect(5000)
print("URL pública:", public_url)
```

```
#instalamos las librerías necesarias
%pip install -r /content/drive/MyDrive/ejecucion_local/requirements.txt

[ ] #Generamos el API
from pyngrok import ngrok
ngrok.kill()
ngrok.set_auth_token("2s32H4kIHWe1r8kggiadWbdCdw8_2nBaywQVW76JBNLUzMc4b")
public_url = ngrok.connect(5000)
print("URL pública:", public_url)

[ ] #Ejecutamos el fichero que nos permite obtener las predicciones
%cd /content/drive/MyDrive/ejecucion_local
!python ejecucion.py
```

Figura C.3: Ejecución de comandos en Google Colab para iniciar el API.

Paso 5: Actualización de la API en el Código

Una vez iniciado el servidor de API, se generará una URL para acceder a él.

1. Copie la URL proporcionada por el servidor.
2. Abra el archivo `ejecucion.py` y pegue la URL en la variable correspondiente al API.
3. Actualice, de manera similar, el bloque de código de la interfaz en VS Code para que consuma esta API.

```
from pyngrok import ngrok
ngrok.kill()
ngrok.set_auth_token("2s32H4kIHWe1r8kggiadWbdCdw8_2nBaywQVW76JBNLUzMc4b")
public_url = ngrok.connect(5000)
print("URL pública:", public_url)
```

URL pública: NgrokTunnel: "https://8093-34-126-118-125.ngrok-free.app" -> "http://localhost:5000"

Figura C.4: Actualización de la URL del API en el archivo `ejecucion.py`.

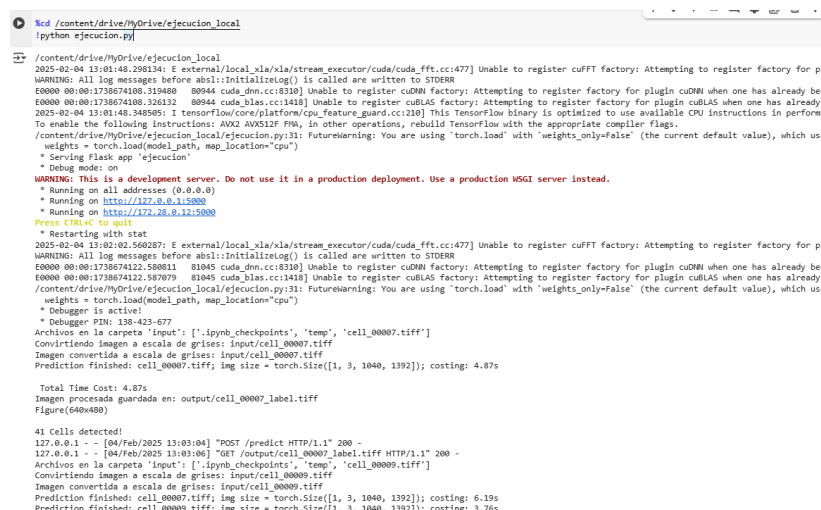
Paso 6: Ejecución Local

Finalmente, proceda de la siguiente manera:

1. Ejecute el archivo `ejecucion.py` para iniciar el servidor API de forma local.

```
%cd /content/drive/MyDrive/ejecucion_local
!python ejecucion.py
```

2. En paralelo, abra el código de la interfaz en VS Code y ejecútelo para interactuar con el servidor.



```
/content/drive/MyDrive/ejecucion_local
2025-02-04 13:01:48.298134: E external/local_xla/xla/stream_executor/cuda/cuda_fft.cc:477] Unable to register cuFFT factory: Attempting to register factory for plugin cuFFT
WARNING: all log messages before absl::InitializeLog() is called are written to STDERR
E0000 00:00:1738674108.319480 80944 cuda_dnn.cc:8310] Unable to register cuDNN factory: Attempting to register factory for plugin cuDNN when one has already been registered
E0000 00:00:1738674108.326132 80944 cuda_blas.cc:1418] Unable to register cuBLAS factory: Attempting to register factory for plugin cuBLAS when one has already been registered
2025-02-04 13:01:48.348595: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to use available CPU instructions in performance. To enable the following instructions: AVX2 AVX512F FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
/content/drive/MyDrive/ejecucion_local/ejecucion.py:31: FutureWarning: You are using 'torch.load' with 'weights_only=False' (the current default value), which uses 'weights = torch.load(model_path, map_location='cpu')'
* Serving Flask app "ejecucion"
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://172.28.0.14:5000
Press CTRL-C to quit
* Restarting with stat
2025-02-04 13:02:02.560287: E external/local_xla/xla/stream_executor/cuda/cuda_fft.cc:477] Unable to register cuFFT factory: Attempting to register factory for plugin cuFFT
WARNING: all log messages before absl::InitializeLog() is called are written to STDERR
E0000 00:00:1738674122.580811 81045 cuda_dnn.cc:8310] Unable to register cuDNN factory: Attempting to register factory for plugin cuDNN when one has already been registered
E0000 00:00:1738674122.587079 81045 cuda_blas.cc:1418] Unable to register cuBLAS factory: Attempting to register factory for plugin cuBLAS when one has already been registered
/content/drive/MyDrive/ejecucion_local/ejecucion.py:31: FutureWarning: You are using 'torch.load' with 'weights_only=False' (the current default value), which uses 'weights = torch.load(model_path, map_location='cpu')'
* Debugger is active!
* Debugger PIN: 138-423-677
Archivos en la carpeta 'input': ['.ipynb_checkpoints', 'temp', 'cell_00007.tiff']
Convirtiendo imagen a escala de grises: input/cell_00007.tiff
Imagen convertida a escala de grises: input/cell_00007.tiff
Prediction finished: cell_00007.tiff; img size = torch.Size([1, 3, 1040, 1392]); costing: 4.87s

Total Time Cost: 4.87s
Imagen procesada guardada en: output/cell_00007_label.tiff
Figure(640x480)

41 Cells detected!
127.0.0.1 - - [04/Feb/2025 13:03:04] "POST /predict HTTP/1.1" 200 -
127.0.0.1 - - [04/Feb/2025 13:03:06] "GET /output/cell_00007_label.tiff HTTP/1.1" 200 -
Archivos en la carpeta 'input': ['.ipynb_checkpoints', 'temp', 'cell_00009.tiff']
Convirtiendo imagen a escala de grises: input/cell_00009.tiff
Imagen convertida a escala de grises: input/cell_00009.tiff
Prediction finished: cell_00009.tiff; img size = torch.Size([1, 3, 1040, 1392]); costing: 6.19s
Prediction finished: cell_00009.tiff; img size = torch.Size([1, 3, 1040, 1392]); costing: 3.76s
```

Figura C.5: Arranque del fichero ejecución para la obtención de predicciones.

Con estos pasos, la plataforma estará configurada y en funcionamiento, permitiendo obtener predicciones a través de la integración del servidor API (ejecutado en Google Colab o localmente) y la interfaz de usuario en VS Code.

C.2. Manuales y/o Demostraciones Prácticas

Manual del Usuario

El proyecto cuenta con una interfaz intuitiva desarrollada en **CustomTkinter**, que permite a los usuarios interactuar con la aplicación para obtener predicciones y realizar análisis de imágenes médicas. A continuación se describen los pasos y funcionalidades principales:

1. **Ejecución del Servidor API en Google Colab:** Primero, el usuario debe ejecutar el fichero `ejecucion.py` en Google Colab. Este script

arranca el servidor API encargado de procesar las imágenes mediante el modelo de segmentación. Es esencial dejar el servidor en ejecución mientras se utiliza la aplicación.

2. **Ejecución de la Interfaz en VS Code:** Una vez que el servidor API esté en funcionamiento, el usuario debe ejecutar la interfaz desde VS Code. Al iniciarse, la aplicación muestra un menú donde se puede seleccionar el modelo de segmentación deseado.
3. **Carga de Imágenes:** La interfaz dispone de un botón para subir imágenes médicas. El usuario puede seleccionar una o varias imágenes, que se utilizarán para el procesamiento.
4. **Procesamiento de la Imagen Original:** Al pulsar el botón "**Procesar Imagen Original**", la aplicación envía la imagen al servidor API. Como resultado, se obtiene la imagen segmentada y se muestra el número de células detectadas en dicha imagen.
5. **Análisis Avanzado:** La aplicación incluye un botón "**Análisis**" que genera una imagen en la que se superponen contornos coloreados sobre la imagen original. Estos contornos identifican cada célula y muestran sus características. Para visualizar los detalles de una célula en particular, el usuario debe mover el ratón sobre la célula deseada, lo que despliega una etiqueta informativa.
6. **Descargas:** Además, se disponen de dos botones adicionales:
 - Uno para descargar la imagen segmentada en escala de grises.
 - Otro para descargar la imagen de análisis con los contornos de las células.

Este flujo de trabajo garantiza que, tras arrancar el servidor API en Google Colab y la interfaz en VS Code, el usuario pueda cargar imágenes, procesarlas, obtener los resultados de segmentación y análisis, y finalmente descargar los archivos generados para su posterior revisión o integración en otros procesos.

Demostración Práctica

Para ilustrar el funcionamiento de la plataforma, se ha preparado una demostración práctica que combina un video explicativo y capturas de pantalla de cada etapa del proceso. El video está disponible en el siguiente enlace: <https://www.dropbox.com/scl/fi/kokyd734zojg4neqkimtx/>

TFG_Grabado_Walid_Sabhi.mp4?rlkey=e0fp49b8m7gd71yif3xrgva63&st=97imng4c&dl=0.

La demostración abarca las siguientes etapas:

- **Iniciar la plataforma:** El usuario arranca el fichero interfaz.py.

- **Subida y Procesamiento de Imágenes:** El usuario carga imágenes médicas en la plataforma. A continuación, se ejecuta la segmentación de la imagen original, generando una versión segmentada en escala de grises y mostrando el número de células detectadas.

- **Análisis Avanzado y Revisión Manual:** Mediante el botón **“Análisis”**, la aplicación genera una imagen con contornos coloreados sobre la imagen original. Al mover el ratón sobre cada célula, se despliega una etiqueta con sus características. Esto permite una revisión manual y la verificación de los resultados.

- **Descarga de Resultados:** Se ofrecen dos opciones de descarga: una para la imagen segmentada en escala de grises y otra para la imagen de análisis con contornos, facilitando la integración de los resultados en otros procesos o informes.

A continuación, se presentan algunas capturas de pantalla representativas del proceso:

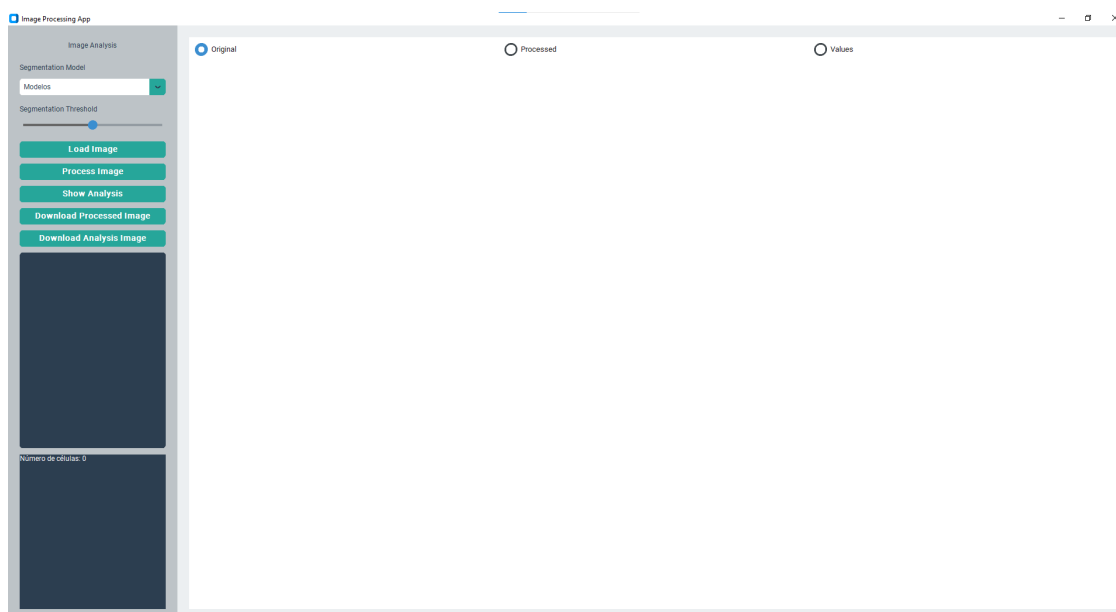


Figura C.6: Pantalla de inicio.

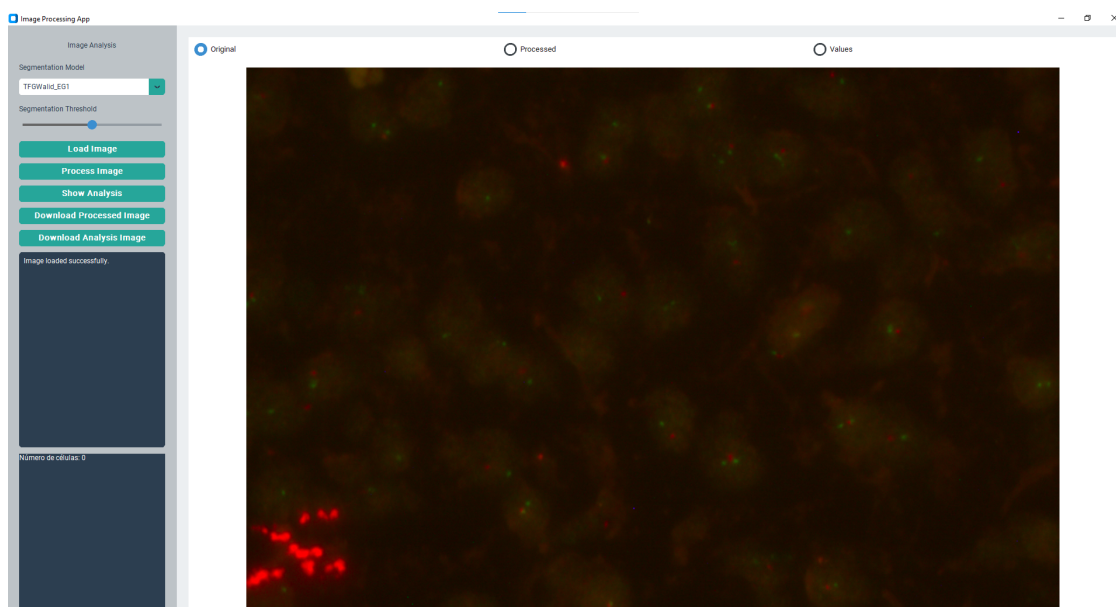


Figura C.7: Proceso de carga de la imagen original.



Figura C.8: Proceso de obtención de la imagen segmentada y del número de células detectadas.

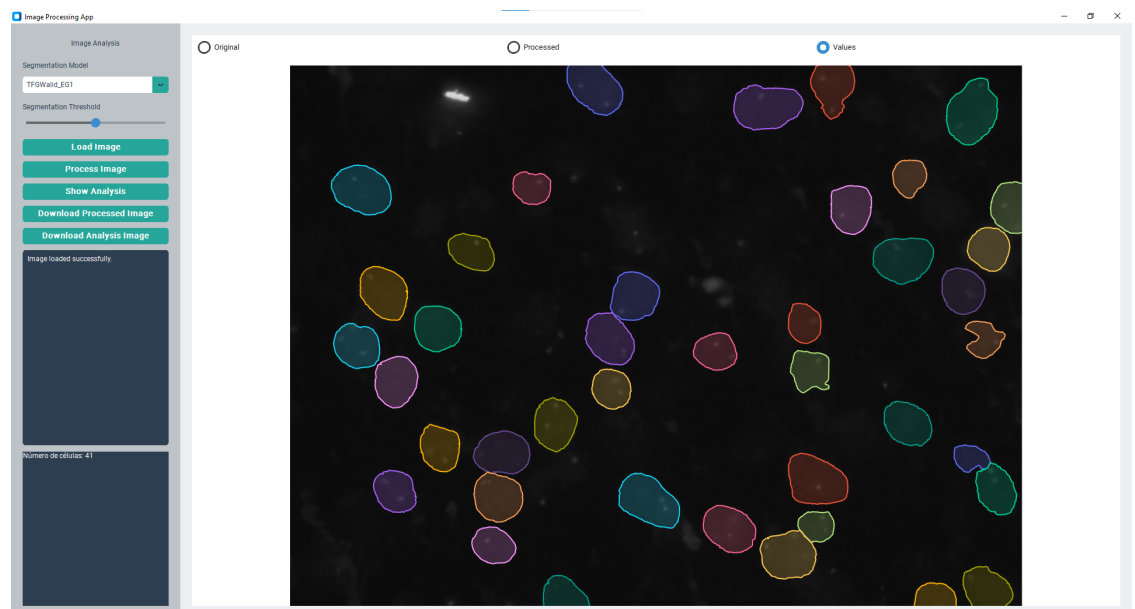


Figura C.9: Proceso de obtención de la imagen analizada, que presenta las células con contornos de distintos colores.

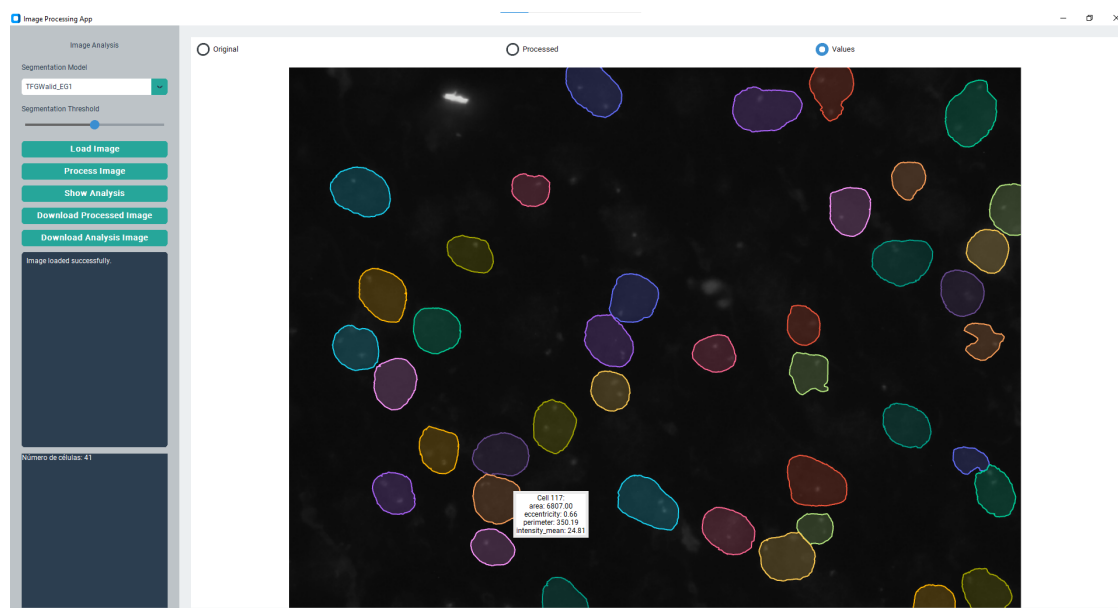


Figura C.10: Visualización de las características de una célula seleccionada.

Apéndice *D*

Manual del Desarrollador / Programador / Investigador

D.1. Estructura de Directorios

A continuación, se describe la estructura de los directorios y ficheros entregados:

- **ejecucion_local/**: Contiene el código fuente del proyecto y subcarpetas organizadas para las distintas funcionalidades.
 - *custom/*: Scripts personalizados para preprocesamiento y procesamiento de las imágenes.
 - **CellAware.py**: Implementa funciones específicas para la detección y reconocimiento de células.
 - **LoadImage.py**: Maneja la carga y formateo de las imágenes de entrada.
 - **NormalizeImage.py**: Aplica normalización a las imágenes para mejorar el rendimiento del modelo.
 - **__init__.py**: Archivo para definir el directorio como un paquete de Python.
 - **modalities.pkl**: Archivo binario para almacenar configuraciones o modalidades del proyecto.
 - *input/*: Carpeta para las imágenes de prueba y datos de entrada.
 - *output/*: Directorio donde se almacenan las imágenes procesadas y resultados de segmentación.

- **BasePredictor.py**: Clase base para los predictores del modelo de segmentación.
- **MEDIARFormer.py**: Implementación del modelo de segmentación MEDIAR.
- **TFGValid_EG1.pth**: Pesos preentrenados del modelo.
- **ejecucion.py**: Script principal para ejecutar el flujo de segmentación.
- **from_phase1.py, from_phase2.py**: Scripts para cargar y evaluar distintas fases del procesamiento.
- **pasos.txt**: Archivo de texto que describe los pasos del proceso.
- **predictor.py**: Script que implementa la lógica de predicción utilizando el modelo.
- **requirements.txt**: Lista de dependencias necesarias para ejecutar el proyecto.
- **transforms.py**: Transformaciones aplicadas a las imágenes para la entrada del modelo.
- **utils.py**: Funciones auxiliares para el manejo del flujo de procesamiento.

D.2. Integración de MEDIAR en el Proyecto

El proyecto utiliza el modelo de segmentación **MEDIAR** (Multimodal and Efficient Deep Instance Segmentation for Cells), un enfoque avanzado para la segmentación de células en imágenes de microscopía. MEDIAR se basa en una arquitectura híbrida que combina el uso de técnicas de aprendizaje profundo y un enfoque centrado tanto en los datos como en el modelo.

¿Qué es MEDIAR y cómo se integra en el proyecto?

MEDIAR se emplea como el núcleo del sistema de segmentación de células, utilizando una red neuronal profunda [AWS, 2025b] para identificar y segmentar de manera precisa las células en las imágenes de entrada. Este modelo se encuentra implementado en el archivo **MEDIARFormer.py** y está

basado en la arquitectura **SegFormer** para la extracción de características y **MA-Net** [Reddit, 2025] para la reconstrucción de la segmentación. El sistema también emplea técnicas avanzadas como **Test-Time Augmentation (TTA)** [Arxiv, 2025] y **Ensamble de Modelos** para mejorar la precisión de las predicciones.

Además, se han preentrenado los pesos del modelo en conjuntos de datos públicos, y estos se cargan en el proyecto a través del archivo `TFGValid_EG1.pth`. Durante la ejecución, el modelo realiza segmentaciones detalladas de las imágenes de células, produciendo tanto la segmentación binaria de cada célula como la estimación del número total de células.

D.3. Compilación, Instalación y Ejecución del Proyecto

Siga los siguientes pasos para la correcta ejecución del proyecto:

1. Instale las dependencias ejecutando:

```
pip install -r requirements.txt
```

Asegúrese de tener las versiones compatibles para las siguientes bibliotecas:

- `segmentation_models_pytorch==0.3.3`
 - `monai==1.3.0`
 - `scikit-image`, `pillow`, `opencv-python`
 - `pyngrok` (si requiere exposición del servidor local) [Ngrok, 2025]
2. Ejecute el fichero `ejecucion.py` en Google Colab para mantener el entorno activo si está trabajando en la nube.
 3. Si trabaja localmente, puede ejecutar el archivo desde su terminal:

```
python ejecucion.py
```

4. El sistema permitirá realizar las siguientes acciones:
 - Seleccionar el modelo de segmentación deseado.

- Subir una imagen para el procesamiento.
- Procesar la imagen original y obtener la versión segmentada junto con el número de células detectadas.
- Visualizar contornos en colores distintos para identificar las células y sus características.
- Descargar la imagen segmentada y el análisis con contornos.

D.4. Pruebas del Sistema

El sistema ha sido probado a través de una serie de evaluaciones técnicas para garantizar su funcionalidad y fiabilidad. Se realizaron pruebas unitarias para verificar el funcionamiento correcto de cada componente del sistema, así como pruebas de integración para asegurar que los diferentes módulos interactúan de manera adecuada. También se llevaron a cabo pruebas de rendimiento, evaluando la capacidad del sistema para manejar diferentes cargas de trabajo y solicitudes simultáneas, y pruebas de seguridad para identificar posibles vulnerabilidades y asegurar la protección de los datos.

D.5. Instrucciones para la Modificación o Mejora del Proyecto

Para mejorar o modificar el sistema, se sugieren las siguientes acciones:

- **Ampliación de modelos:** Integrar modelos adicionales de segmentación para mejorar la precisión.
- **Optimización del procesamiento:** Evaluar técnicas para reducir el tiempo de procesamiento de imágenes.
- **Documentación:** Ampliar la documentación del código para facilitar futuras contribuciones.
- **Mejora de la interfaz:** Incluir herramientas adicionales que faciliten la visualización y edición de los análisis.
- **Automatización de pruebas:** Implementar un conjunto robusto de pruebas automatizadas para garantizar la calidad del sistema.

requirements.txt X

```
1 segmentation_models_pytorch==0.3.3
2 monai==1.3.0
3 fastremap
4 scikit-image
5 pillow
6 opencv-python
7 pyngrok|
```

Figura D.1: Librerías requeridas.

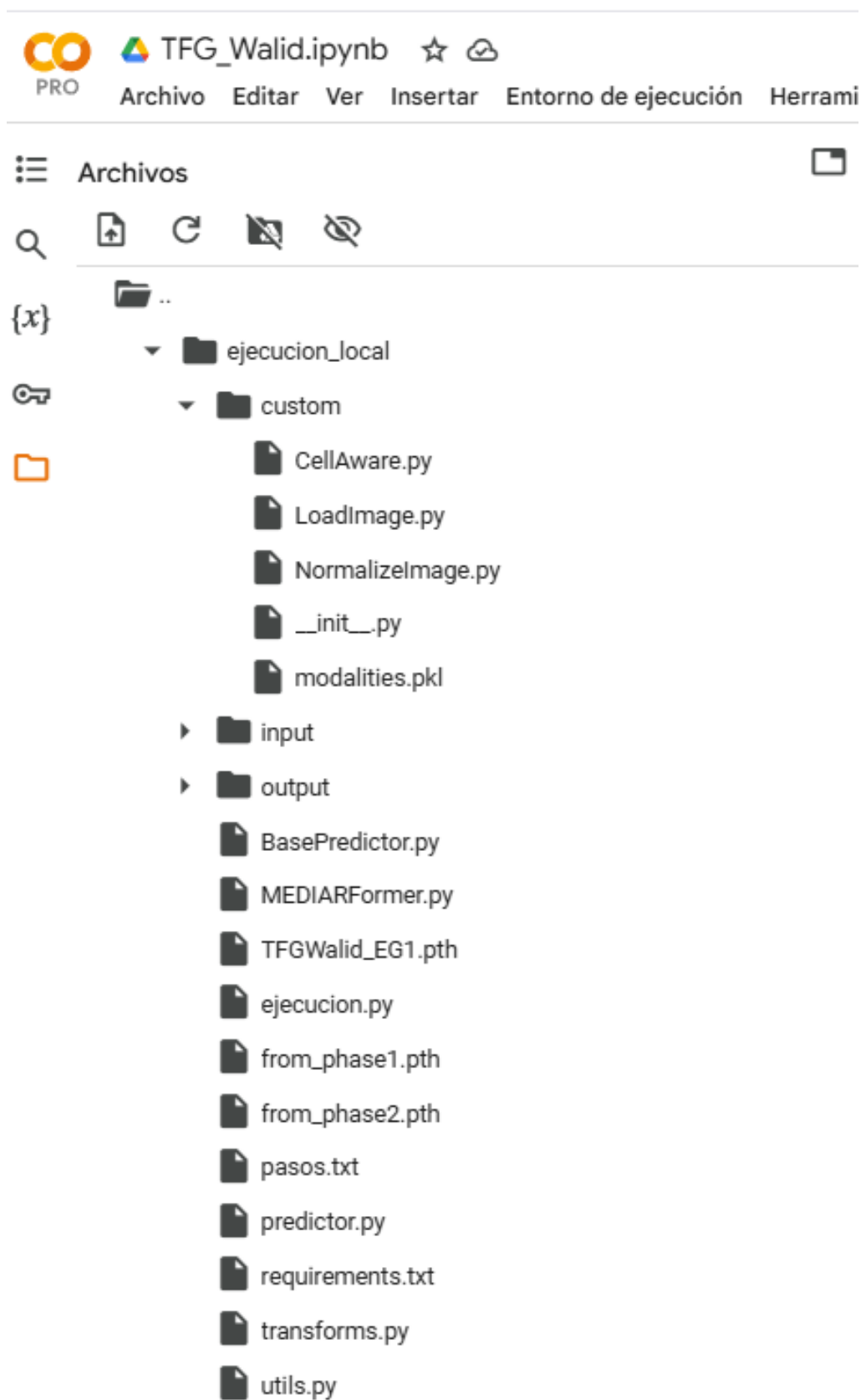


Figura D.2: Carpetas requeridas.

Apéndice E

Descripción de Adquisición y Tratamiento de Datos

E.1. Descripción Formal de los Datos

El conjunto de datos utilizado en este proyecto está compuesto exclusivamente por 25 imágenes FISH (Fluorescence In Situ Hybridization) proporcionadas por el Hospital Universitario de Burgos (HUBU). Las imágenes fueron capturadas en formato .jpg y presentan estructuras celulares teñidas mediante marcadores fluorescentes en colores rojo y verde, lo que permite visualizar las regiones específicas de interés en las células.

Cada imagen tiene una resolución fija de 1392x1040 píxeles, con tres canales de color RGB. Estas características permiten un análisis detallado de la distribución de la fluorescencia y la segmentación de las células para su posterior estudio cuantitativo. Debido al limitado tamaño del conjunto de datos, se consideraron técnicas de procesamiento y segmentación cuidadosas para optimizar el análisis de las imágenes disponibles.

E.2. Descripción Clínica de los Datos

Desde una perspectiva clínica, las imágenes FISH son una herramienta fundamental para la investigación y diagnóstico de alteraciones genéticas y la evaluación de la actividad celular. La fluorescencia roja y verde en las imágenes representa la unión específica de sondas a secuencias de ADN objetivo, permitiendo visualizar:

- **Distribución de señales fluorescentes:** La correcta identificación y segmentación de las células permite evaluar la presencia de marcadores específicos, lo cual es crucial en la detección de alteraciones genéticas.
- **Número de células:** El conteo preciso de células en las imágenes es un aspecto clave para cuantificar la presencia o ausencia de actividad celular en determinadas regiones de las muestras.
- **Análisis de fluorescencia:** La intensidad y distribución de los colores permiten inferir la expresión de genes específicos, información relevante para el diagnóstico y la evaluación de la progresión de enfermedades.
- **Evaluación morfológica:** Las imágenes FISH también brindan información indirecta sobre la morfología de las células, permitiendo identificar posibles irregularidades estructurales.

Apéndice F

Manual de Especificación de Diseño

Este apéndice recopila la documentación relativa al diseño del sistema. Se incluyen, de forma opcional, planos y la especificación arquitectónica que describen la estructura interna del software. Estos elementos son útiles para comprender y mantener el proyecto en futuras actualizaciones.

F.1. Diseño Arquitectónico

El diseño arquitectónico se detalla mediante diagramas [PlantText, 2025] que representan la estructura y las interrelaciones de los componentes del software. En este apartado se incluyen:

- **Diagrama de Clases:** Este diagrama ilustra las clases utilizadas en el sistema, sus atributos, métodos y las relaciones que existen entre ellas, facilitando la comprensión del modelo de datos y la lógica de negocio implementada.

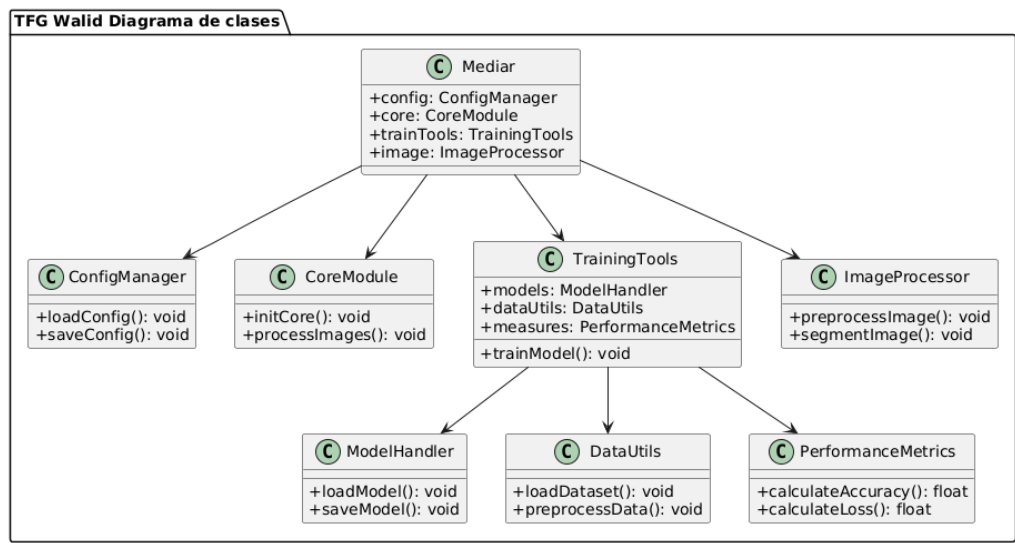


Figura F.1: Diagrama de Clases.

- **Diagrama de Despliegue:** Se muestra la distribución de los componentes del sistema en el entorno de ejecución, detallando la conexión entre servidores, dispositivos de red y otros elementos hardware cuando sea necesario.

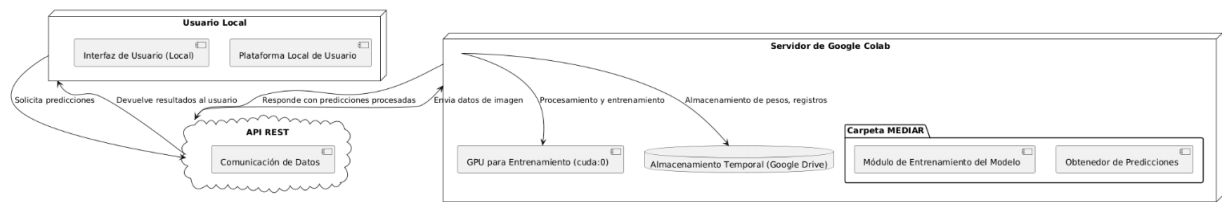


Figura F.2: Diagrama de Despliegue.

Apéndice G

Especificación de Requisitos

G.1. Diagrama de Casos de Uso

A continuación, se presenta el diagrama de casos de uso que describe las interacciones del usuario con la plataforma de segmentación de células FISH. Se detallan las principales funcionalidades del sistema, como la carga de imágenes, segmentación, análisis de resultados y exportación de predicciones.

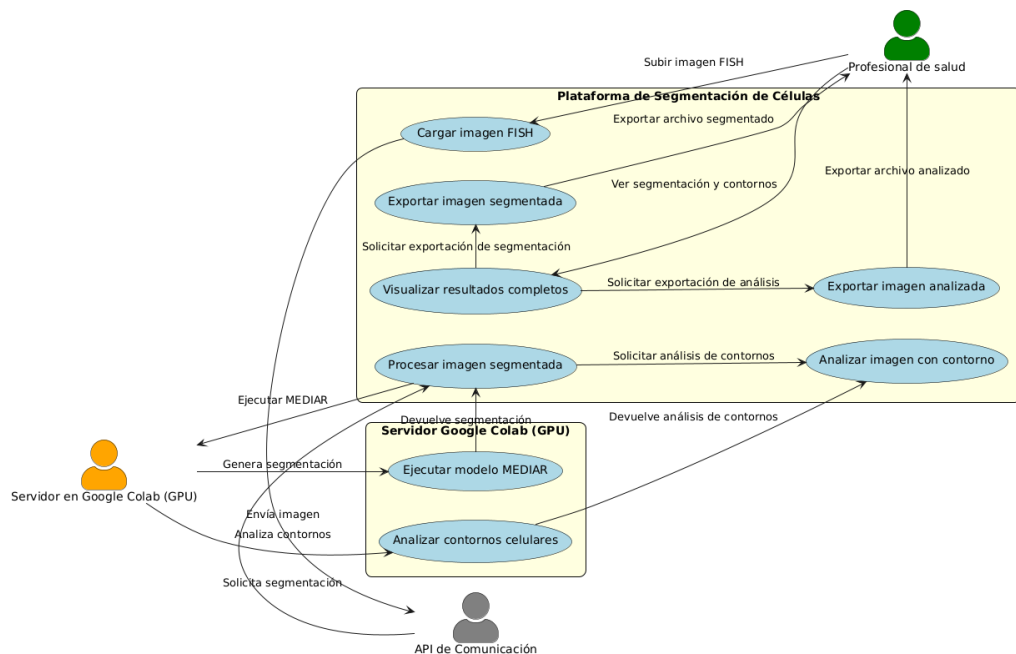


Figura G.1: Diagrama de casos de uso.

G.2. Explicación Casos de Uso

A continuación, se detallan los casos de uso principales para el proyecto.

CU-1: Carga de imágenes FISH para segmentación

CU-1	Carga de imágenes FISH para segmentación
Versión	1.1
Autor	Walid Sabhi
Requisitos asociados	R1, R2
Descripción	El profesional de salud carga imágenes FISH desde su equipo para que sean procesadas en Google Colab mediante el modelo MEDIAR, generando una imagen segmentada que podrá ser analizada posteriormente para resaltar contornos celulares.
Precondición	El usuario debe estar autenticado en la plataforma y tener acceso a Google Colab.
Acciones	<ol style="list-style-type: none"> 1. El profesional de salud accede a la sección de carga de imágenes. 2. Selecciona y carga las imágenes FISH desde su equipo. 3. La imagen se envía a través de la API de Comunicación. 4. La API transmite la imagen al servidor en Google Colab. 5. El servidor ejecuta el modelo MEDIAR para segmentar la imagen. 6. Se muestra el resultado de la segmentación en la interfaz de la plataforma.
Postcondición	Las imágenes segmentadas se almacenan temporalmente y los resultados son visibles en la interfaz, permitiendo posteriores análisis y exportación.
Excepciones	Error en la comunicación con la API o formato de imagen no compatible.
Importancia	Alta

Tabla G.1: CU-1: Carga de imágenes FISH para segmentación.

Tabla de Actores

Actor	Descripción
Profesional de salud	Usuario encargado de cargar imágenes FISH, analizar resultados y tomar decisiones clínicas.
API de Comunicación	Intermediario que transmite la imagen y los datos entre la plataforma y el servidor.
Servidor en Google Colab (GPU)	Realiza el procesamiento de imágenes ejecutando el modelo de segmentación (MEDIAR) y el análisis de contornos.

Tabla G.2: Lista de Actores.

Tabla de Casos de Uso

ID	Caso de Uso	Actor(es)	Descripción
UC1	Cargar imagen FISH	Profesional de salud	Permite al usuario subir la imagen FISH para su posterior análisis.
UC2	Procesar imagen segmentada	API, Servidor en Google Colab	Realiza la segmentación de la imagen utilizando el modelo MEDIAR.
UC3	Analizar imagen con contorno	API, Servidor en Google Colab	Identifica y resalta los contornos celulares en la imagen segmentada.
UC4	Visualizar resultados completos	Profesional de salud	Muestra la imagen segmentada junto con el análisis de contornos.
UC5	Exportar imagen segmentada	Profesional de salud	Permite la descarga de la imagen resultante de la segmentación.
UC6	Exportar imagen analizada	Profesional de salud	Permite la descarga de la imagen con el análisis de contornos aplicado.
UC7	Ejecutar modelo MEDIAR	Servidor en Google Colab	Ejecuta el modelo de segmentación sobre la imagen cargada.
UC8	Analizar contornos celulares	Servidor en Google Colab	Procesa la imagen segmentada para determinar los contornos de las células.

Tabla G.3: Lista de Casos de Uso.

Flujo de Eventos Principal

Paso	Actor	Acción	Sistema/Componente
1	Profesional de salud	Sube imagen FISH	Plataforma (UC1)
2	Plataforma	Envía imagen	API de Comunicación
3	API de Comunicación	Solicita segmentación	Plataforma (UC2)
4	Plataforma	Ejecuta modelo MEDIAR	Servidor en Google Colab (UC7)
5	Servidor en Google Colab	Devuelve segmentación	Plataforma (UC2)
6	Plataforma	Solicita análisis de contornos	Servidor en Google Colab (UC8)
7	Servidor en Google Colab	Devuelve análisis de contornos	Plataforma (UC3)
8	Profesional de salud	Visualiza resultados	Plataforma (UC4)
9	Profesional de salud	Solicita exportación	Plataforma (UC5 y UC6)
10	Plataforma	Exporta archivos	Profesional de salud

Tabla G.4: Flujo de Eventos Principal.

Tabla de Requisitos del Sistema

ID	Requisito
R1	La plataforma debe permitir la carga de imágenes FISH por parte del profesional de salud.
R2	La imagen cargada debe ser transmitida de forma segura mediante el API de Comunicación.
R3	El servidor en Google Colab debe ejecutar el modelo MEDIAR para segmentar la imagen.
R4	La imagen segmentada debe ser analizada para resaltar los contornos celulares.
R5	El profesional de salud debe poder visualizar los resultados completos (segmentación y análisis).
R6	La plataforma debe permitir la exportación de las imágenes segmentada y analizada en formato descargable.

Tabla G.5: Requisitos del Sistema.

Tabla de Escenarios Alternativos

Escenario	Descripción
Principal	El profesional de salud carga la imagen, la API envía la imagen para segmentación y análisis, y se muestran los resultados completos, permitiendo además su exportación.
Alternativo 1	Si la carga de la imagen falla, se notifica al usuario y se ofrece la opción de reintentar la carga.
Alternativo 2	Si el procesamiento o análisis falla en el servidor, se muestra un mensaje de error y se ofrece la posibilidad de reintentar el procesamiento.

Tabla G.6: Escenarios Alternativos del Caso de Uso.

Mapecto de Casos de Uso y Requisitos

Caso de Uso	Requisitos Asociados
Cargar imagen FISH (UC1)	R1
Procesar imagen segmentada (UC2)	R2, R3
Analizar imagen con contorno (UC3)	R4
Visualizar resultados completos (UC4)	R5
Exportar imagen segmentada (UC5)	R6
Exportar imagen analizada (UC6)	R6
Ejecutar modelo MEDIAR (UC7)	R3
Analizar contornos celulares (UC8)	R4

Tabla G.7: Mapecto de Casos de Uso y Requisitos.

Mapecto de Casos de Uso y Requisitos

Caso de Uso	Requisitos Asociados
Cargar imagen FISH (UC1)	R1
Procesar imagen segmentada (UC2)	R2, R3
Analizar imagen con contorno (UC3)	R4
Visualizar resultados completos (UC4)	R5
Exportar imagen segmentada (UC5)	R6
Exportar imagen analizada (UC6)	R6
Ejecutar modelo MEDIAR (UC7)	R3
Analizar contornos celulares (UC8)	R4

Tabla G.8: Mapecto de Casos de Uso y Requisitos.

G.3. Prototipos de Interfaz o Interacción con el Proyecto

A continuación, se describen los prototipos de las principales interfaces de la plataforma de segmentación de células FISH:

- **Pantalla de carga de imágenes:** Permite al usuario seleccionar y cargar imágenes desde su equipo local.
- **Pantalla de visualización de resultados:** Muestra las imágenes segmentadas junto con las estadísticas del análisis.

- **Panel de control:** Incluye funcionalidades para exportar resultados y visualizar métricas de rendimiento del sistema.

Apéndice H

Estudio Experimental

H.1. Cuaderno de Trabajo

En esta sección se documentan los métodos y experimentos realizados durante el estudio, registrando tanto los enfoques exitosos como aquellos que no alcanzaron los objetivos esperados. Cada método se describe con detalle para facilitar la replicabilidad del estudio y el análisis comparativo de los resultados obtenidos.

Método 1: Segmentación Basada en U-Net

Se implementó un modelo basado en la arquitectura U-Net para segmentar imágenes FISH de muestras celulares. El proceso incluyó un preprocesamiento que consistió en la conversión de las imágenes de RGB a escala de grises, seguido de técnicas de ajuste de contraste y umbralización para resaltar las células. Los resultados iniciales mostraron una segmentación aceptable en áreas con buen contraste; sin embargo, en zonas con baja intensidad se observaron inconsistencias. Este método permitió identificar áreas de mejora, principalmente en la optimización de los parámetros de umbralización.

Método 2: Fine-Tuning del Modelo MEDIAR

Se aplicó una estrategia de ajuste fino (fine-tuning) al modelo MEDIAR, previamente entrenado con un extenso conjunto de imágenes, para adaptarlo a la segmentación específica de imágenes FISH. El reentrenamiento se centró en las capas finales del modelo utilizando un subconjunto de imágenes

etiquetadas manualmente. Los resultados obtenidos evidenciaron una mejora en la precisión de la segmentación, especialmente en la delimitación de células adyacentes, aunque se detectaron ciertos errores en la diferenciación de instancias. Se propusieron ajustes adicionales en la estrategia de fine-tuning para superar estas limitaciones.

H.2. Configuración y Parametrización de las Técnicas

Esta sección detalla las configuraciones y parámetros empleados en cada experimento, describiendo los algoritmos, valores de parámetros, recursos y herramientas (hardware, software, versiones, etc.), así como la configuración de los experimentos (número de muestras, tipo de datos, etc.).

Técnica 1: Configuración del Modelo U-Net

La implementación del modelo basado en U-Net se realizó con la siguiente configuración:

- **Preprocesamiento:** Conversión a escala de grises, ajuste de contraste y umbralización.
- **Arquitectura:** U-Net con 4 niveles de profundidad.
- **Parámetros:** Tasa de aprendizaje de 0.001, 100 épocas y tamaño de lote de 16.
- **Recursos:** Ejecución en GPU NVIDIA T4 mediante Google Colab.

Técnica 2: Configuración del Fine-Tuning del Modelo MEDIAR

La estrategia de ajuste fino del modelo MEDIAR se configuró de la siguiente manera:

- **Modelo Base:** MEDIAR preentrenado.
- **Reentrenamiento:** Ajuste de las últimas capas utilizando 18 imágenes etiquetadas.

- **Parámetros:** Tasa de aprendizaje de 0.0001, 200 épocas y tamaño de lote de 8.
- **Recursos:** Plataforma Google Colab con GPU NVIDIA T4.

H.3. Detalle de Resultados

En esta sección se presentan los resultados obtenidos en los experimentos, organizados por técnica y método. Los resultados se analizan mediante gráficos, tablas y comparaciones cualitativas, facilitando una evaluación precisa de la eficacia de cada enfoque.

Resultados de la Técnica 1: Modelo U-Net

Los experimentos con el modelo U-Net mostraron una segmentación adecuada en regiones con alto contraste, aunque la detección de células en áreas de baja intensidad presentó variabilidad. Se generaron gráficos comparativos y se calcularon métricas como el IoU, lo que evidenció la necesidad de ajustar los parámetros de umbralización para mejorar la consistencia del modelo.

Resultados de la Técnica 2: Fine-Tuning del Modelo MEDIAR

El fine-tuning del modelo MEDIAR produjo una mejora significativa en la segmentación, especialmente en la delimitación de células individuales. Los análisis cuantitativos, basados en el F1-Score y otros índices de precisión, revelaron un incremento en la exactitud en comparación con el método basado en U-Net. Los resultados se presentan en tablas y gráficos, permitiendo una comparación directa de ambos enfoques y resaltando las ventajas del ajuste fino en este contexto.

Apéndice I

Anexo de Sostenibilización Curricular

I.1. Introducción

Este anexo reflexiona sobre la integración de criterios de sostenibilidad en el Trabajo de Fin de Grado, enfocado en la responsabilidad ambiental, social y económica. El proyecto promueve una gestión responsable de recursos y la adopción de prácticas innovadoras para el cuidado del medio ambiente, basándose en principios de economía circular [Europeo, 2025]. Se enfoca en fortalecer competencias para el análisis crítico y la planificación estratégica, guiado por directrices de organismos como la CRUE.

Cada fase del proyecto se desarrolla con rigor, incorporando aspectos técnicos, éticos y organizativos, y se destacan logros en la implementación de prácticas sostenibles. Se identifican áreas de mejora para futuros proyectos, con recomendaciones para optimizar las estrategias sostenibles. El enfoque adoptado prepara al profesional para afrontar desafíos globales con responsabilidad, consolidando una visión integral que une criterios ambientales, sociales y económicos.

La experiencia refuerza la calidad del trabajo, la capacidad de innovación y la importancia de integrar la sostenibilidad [Wikipedia, 2025d] en la formación académica. En conclusión, el proyecto reafirma el compromiso con un enfoque estratégico [Asana, 2025] e integral en sostenibilidad.

Bibliografía

- [Arxiv, 2025] Arxiv (2025). Test-time augmentation (tta). <https://arxiv.org/html/2402.06892v1>. Accessed: 2025-02-11.
- [Asana, 2025] Asana (2025). Planificación estratégica. Accessed: 2025-02-11.
- [AWS, 2025a] AWS (2025a). Hiperparámetros. <https://aws.amazon.com/es/what-is/hyperparameter-tuning/>. Accessed: 2025-02-11.
- [AWS, 2025b] AWS (2025b). Red neuronal profunda. <https://aws.amazon.com/es/what-is/neural-network/#:~:text=Las%20redes%20neuronales%20profundas%2C%20o,entre%20un%20nodo%20y%20otro>. Accessed: 2025-02-11.
- [del Estado, 2025a] del Estado, B. O. (2025a). Gdpr. <https://www.boe.es/doue/2016/119/L00001-00088.pdf>. Accessed: 2025-02-11.
- [del Estado, 2025b] del Estado, B. O. (2025b). Lopd-gdd. <https://www.boe.es/buscar/act.php?id=BOE-A-2018-16673>. Accessed: 2025-02-11.
- [Europeo, 2025] Europeo, P. (2025). Economía circular: Definición, importancia y beneficios. Accessed: 2025-02-11.
- [Gihun, 2025] Gihun, L. (2025). Mediar. <https://github.com/Lee-Gihun/MEDIAR>. Accessed: 2025-02-11.
- [IBM, 2025] IBM (2025). Segmentación de imágenes. <https://www.ibm.com/es-es/topics/image-segmentation>. Accessed: 2025-02-11.
- [ISO, 2025] ISO (2025). Iso 27001. <https://www.iso.org/standard/27001>. Accessed: 2025-02-11.

- [Kumar, 2025] Kumar, A. (2025). Fine-tuning. https://medium.com/@avikumart_/evaluation-of-fine-tuned-llm-using-monsterapi-a67a7714a65b. Accessed: 2025-02-11.
- [matplotlib, 2025] matplotlib (2025). matplotlib. <https://matplotlib.org/>. Accessed: 2025-02-11.
- [MIC-DKFZ, 2025] MIC-DKFZ (2025). nnunet. <https://github.com/MIC-DKFZ/nnUNet>. Accessed: 2025-02-11.
- [Ngrok, 2025] Ngrok (2025). Ngrok. <https://ngrok.com/>. Accessed: 2025-02-11.
- [NumPy, 2025] NumPy (2025). Numpy. <https://numpy.org/>. Accessed: 2025-02-11.
- [pandas, 2025] pandas (2025). pandas. <https://pandas.pydata.org/>. Accessed: 2025-02-11.
- [PlantText, 2025] PlantText (2025). Diagramas. <https://www.planttext.com/>. Accessed: 2025-02-11.
- [ProductPlan, 2025] ProductPlan (2025). Beta testing. <https://www.productplan.com/glossary/beta-test/>. Accessed: 2025-02-11.
- [Project, 2025] Project, F. (2025). Flask. <https://flask.palletsprojects.com/en/stable/>. Accessed: 2025-02-11.
- [Pérez, 2025] Pérez, P. (2025). Procesamiento de imágenes. <https://www.famaf.unc.edu.ar/~pperez1/manuales/cim/cap2.html>. Accessed: 2025-02-11.
- [Reddit, 2025] Reddit (2025). Ma-net. https://www.reddit.com/r/lightingdesign/comments/8yd8mh/eli5_what_is_ma_net/?rdt=52419. Accessed: 2025-02-11.
- [Schimansky, 2025] Schimansky, T. (2025). Customtkinter. <https://customtkinter.tomschimansky.com/>. Accessed: 2025-02-11.
- [Scrum.org, 2025] Scrum.org (2025). Scrum. <https://www.scrum.org/resources/blog/que-es-scrum>. Accessed: 2025-02-11.
- [Wikipedia, 2025a] Wikipedia (2025a). Acuerdo de confidencialidad (nda). https://es.wikipedia.org/wiki/Acuerdo_de_confidencialidad. Accessed: 2025-02-11.

- [Wikipedia, 2025b] Wikipedia (2025b). Cuda. <https://en.wikipedia.org/wiki/CUDA#:~:text=CUDA%20is%20a%20software%20layer,the%20execution%20of%20compute%20kernels>. Accessed: 2025-02-11.
- [Wikipedia, 2025c] Wikipedia (2025c). Graphics processing unit (gpu). https://en.wikipedia.org/wiki/Graphics_processing_unit. Accessed: 2025-02-11.
- [Wikipedia, 2025d] Wikipedia (2025d). Sostenibilidad. Accessed: 2025-02-11.