Universidad Zaragoza
1542

Trabajo Fin de Grado

Optimización de la segmentación de blastocistos mediante técnicas de ensemble de modelos de deep learning.

Optimization of blastocyst segmentation using ensemble techniques of deep learning models.

Autor

Javier Ferreras Pajarín

Directores

Directora María Villota Miranda

Director Jacobo Ayensa Jiménez

Ponente Eduardo Montijano Muñoz

ESCUELA DE INGENIERÍA Y ARQUITECTURA
2025

RESUMEN

Este trabajo se centra en la segmentación automática de blastocistos a partir de imágenes de microscopio, un proceso de gran utilidad para la selección de blastocistos en los tratamientos de fecundación *in vitro*.

Se desarrollan y comparan diferentes estrategias de *ensemble learning* para combinar las predicciones de varios modelos pre-entrenados de segmentación semántica (DeepLab, HRNet, U-Net y RDU-Net), con el objetivo de mejorar la precisión y robustez de la segmentación de las tres principales estructuras del blastocisto: la Zona Pelúcida (ZP), el Trofoectodermo (TE) y la Masa Celular Interna (MCI). Las estrategias desarrolladas incluyen técnicas no supervisadas basadas en operaciones sobre las máscaras (post-procesamiento, OR, AND, voto mayoritario) y las probabilidades de salida (*softmax*, *max*, suma ponderada), así como enfoques supervisados (Regresión Logística, Perceptrón Multicapa y Random Forest).

Estas estrategias se han evaluado en dos conjuntos de datos distintos, y destaca el rendimiento de uno de los perceptrones multicapa, que alcanza el mejor equilibrio entre precisión y *recall*, con buena generalización. Estos resultados se han comparado con los del estado del arte, obteniéndose las mejores métricas para la segmentación de la ZP, el segundo puesto para el TE, y el tercero para la MCI.

Además, se ha creado un repositorio público[2] que incluye el código, las métricas y los mejores modelos entrenados, con el fin de fomentar la reproducibilidad y extensión del trabajo.

[2] `https://github.com/816410unizar/Blastocyst-Seg-Ensemble`

Índice

1. Introducción y objetivos **1**

1.1. La selección de blastocistos en la FIV 1

1.2. Objetivos y alcance 3

1.2.1. Objetivos 3

1.2.2. Alcance 4

1.3. Herramientas utilizadas 4

1.4. Desarrollo temporal 5

1.5. Estructura de la memoria 6

2. Estado del arte y punto de partida **7**

2.1. Trabajos previos 7

2.2. Resumen de resultados previos 8

2.3. Punto de partida de este trabajo 10

2.3.1. Conjuntos de datos 10

2.3.2. Modelos de deep learning 11

2.3.3. Estructura de los datos 13

3. Estrategias de ensemble no supervisado **15**

3.1. Post-procesado de máscaras individuales 15

3.2. Operadores lógicos hibridando varios modelos 19

3.3. Combinación de post-procesado y operadores lógicos 20

3.4. Operadores sobre la salida de probabilidades de varios modelos 21

4. Estrategias de ensemble supervisado **24**

4.1. Construcción de conjuntos de datos 25

4.1.1. Conjuntos de datos de entrenamiento 25

4.1.2. Conjuntos de evaluación 26

4.2. Modelos entrenados 27

5. Resultados y análisis **30**
5.1. Comparación con los modelos base 30
5.1.1. Métodos no supervisados que operan con máscaras. 32
5.1.2. Métodos no supervisados que operan con probabilidades. 33
5.1.3. Métodos supervisados. 33
5.1.4. Mejores modelos. 34
5.2. Comparación con el estado del arte 36

6. Conclusiones **38**

7. Bibliografía **40**

Lista de Figuras **46**

Lista de Tablas **47**

Capítulo 1

Introducción y objetivos

La infertilidad es un problema de salud creciente a nivel mundial que afecta aproximadamente del 10 al 17.5 % de la población adulta, tanto a hombres como a mujeres [1, 2]. En este contexto, las técnicas de reproducción asistida, especialmente la Fecundación In Vitro (FIV), se han convertido en herramientas fundamentales para ofrecer soluciones reproductivas eficaces a millones de parejas. Se prevé que el número de personas nacidas globalmente gracias a la FIV y otros tratamientos reproductivos aumente a unos cuatrocientos millones para el año 2100 [1].

La FIV es una técnica de reproducción asistida que comienza con la hiperestimulación de los ovarios para extraer múltiples ovocitos. A continuación, los ovocitos son fecundados en una placa de cultivo y los embriones resultantes se cultivan hasta el día 5 o 6, cuando alcanzan el estadio de blastocisto. En este momento se selecciona el blastocisto más viable para su posterior transferencia al útero. Si el proceso tiene éxito, dicho blastocisto se implantará correctamente y dará lugar al embarazo.

Este trabajo se centra en el momento de la selección de blastocistos, una etapa clave para maximizar las probabilidades de implantación y el éxito reproductivo.

1.1. La selección de blastocistos en la FIV

El blastocisto es el estadio que alcanza el embrión cinco o seis días tras la fecundación, cuando presenta una estructura celular compleja con varias zonas diferenciadas: la Zona Pelúcida (ZP), el Trofoectodermo (TE), la Masa Celular Interna (MCI), y la cavidad conocida como Blastocele (BC) (Figura 1.1).

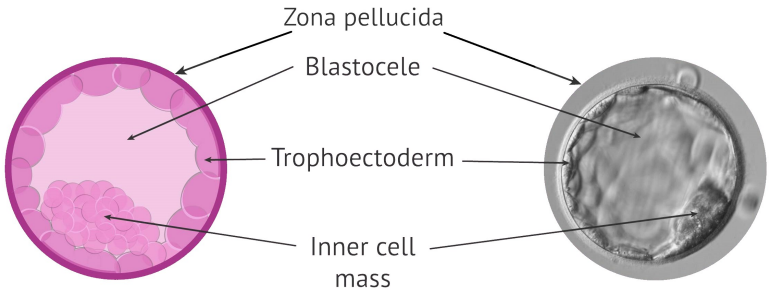

Figura 1.1: Estructuras del blastocisto (Fuente: [3]).

No todos los embriones cultivados en la FIV desarrollan el mismo potencial de implantación, y es de vital importancia seleccionar el blastocisto con mejores características para maximizar las probabilidades de éxito reproductivo, ya que, según el Consorcio Europeo de Monitorización de la FIV [4], tan solo el 34 % de transferencias embrionarias al útero logran el embarazo, y solo un 26 % culminan en un parto exitoso.

Actualmente, existen principalmente dos métodos para la selección de blastocistos: el Test Genético Preimplantacional para Aneuploidías (PGT-A, por sus siglas en inglés *Preimplantation Genetic Testing for Aneuploidy*) [5] y la evaluación morfológica [6].

El PGT-A permite detectar anomalías genéticas en los embriones antes de su transferencia al útero, diferenciando entre embriones euploides (con el número correcto de cromosomas) y aneuploides (con alteraciones cromosómicas). Cabe destacar que el PGT-A no garantiza la implantación de los embriones euploides, por lo que siempre se complementa con una evaluación morfológica visual de los blastocistos llevada a cabo por los embriólogos. Esta segunda técnica consiste en clasificar los embriones según sus características observables como el grado de expansión del blastocisto, la forma, el tamaño, o el grado de desarrollo de estructuras como el TE y la MCI.

A pesar de su utilidad clínica, ambos métodos presentan limitaciones importantes. El PGT-A, aunque eficaz para descartar embriones no viables, es un procedimiento de uso limitado por ser costoso, invasivo y conllevar riesgos, ya que implica la extracción de células del embrión en un estado muy inicial. En particular, es especialmente importante no extraer células pertenecientes a la región de la MCI, ya que las células de esta zona formarán el futuro cuerpo del feto [7]. En cuanto a la evaluación morfológica, el principal problema de este método es que conlleva una gran subjetividad, ya que depende del criterio y experiencia del embriólogo, y no se basa en estándares universales o métricas cuantitativas objetivas [8].

En este contexto, las técnicas para la segmentación de blastocistos, que incluyen desde métodos de procesamiento de imágenes hasta algoritmos de *deep learning*, ofrecen una alternativa prometedora, ya que permiten delimitar automáticamente las

estructuras del blastocisto a partir de imágenes de microscopio (Figura 1.2). De esta manera, se consigue información objetiva y cuantitativa sobre su morfología, como por ejemplo la localización exacta y tamaño de la MCI, cuyo papel es crucial en el desarrollo embrionario. Esta información puede utilizarse tanto para ayudar en la toma de decisiones clínicas, como para futuros estudios sobre la morfología y selección de blastocistos, aumentando las probabilidades de éxito de los tratamientos de FIV.

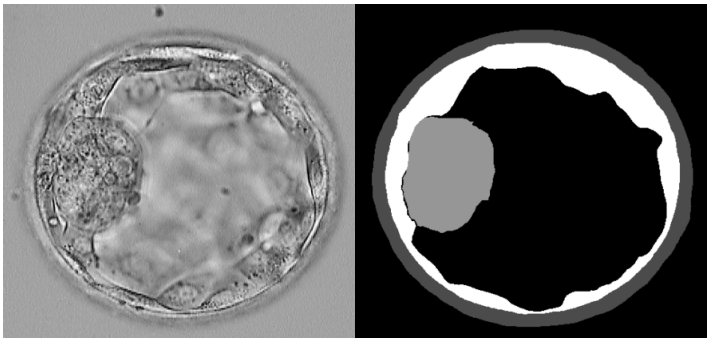

Figura 1.2: Imágen de blastocisto al microscopio (izq.) y su segmentación por un modelo de *Deep learning* (der.).

1.2. Objetivos y alcance

Este Trabajo de Fin de Grado (TFG) busca combinar los resultados de varios modelos pre-entrenados de *deep learning* utilizados para la segmentación de blastocistos, con la intención de generar máscaras más precisas y robustas que las obtenidas por modelos individuales, optimizando así los resultados actuales del estado del arte en el ámbito de la segmentación de blastocistos para la FIV.

1.2.1. Objetivos

Más concretamente, los objetivos de este TFG se pueden resumir en:

- Optimizar la segmentación automática de las principales estructuras del blastocisto (ZP, TE, y MCI).
- Implementar, evaluar y comparar diferentes estrategias de *ensemble learning* [9], tanto no supervisadas como supervisadas, aplicadas a modelos de segmentación de blastocistos.

A través de estos objetivos se pretende proporcionar información morfológica objetiva y cuantitativa del blastocisto, útil para la toma de decisiones en la FIV, como la selección de embriones o la aplicación de técnicas como el PGT-A. Además, se sentarán

las bases para futuros estudios sobre la morfología del blastocisto y para el desarrollo de sistemas automáticos de predicción del embrión óptimo a implantar.

1.2.2. Alcance

Este TFG toma como punto de partida cuatro modelos de segmentación semántica [10] (DeepLab, HRNet, U-Net y RDU-Net), pre-entrenados por Villota et al. [11] y disponibles en su repositorio público [12]. Se han obtenido las salidas de estos modelos para imágenes de blastocistos de dos conjuntos de datos distintos (descritos en la Sección 2.3.1), y a partir de estas salidas, tanto a nivel de máscaras de segmentación como a nivel de tensores de probabilidades, se han diseñado e implementado desde cero diversas estrategias de *ensemble learning*, que se resumen en:

- Técnicas de post-procesamiento de máscaras individuales.
- Uso de operadores lógicos para hibridar varios modelos (OR, AND, voto mayoritario).
- Combinación de post-procesamiento y operadores lógicos.
- Operadores sobre la salida de probabilidades de varios modelos (*softmax*, *max*, suma ponderada, reescalado de probabilidades).
- Algoritmos supervisados sobre la salida de probabilidades de varios modelos, entrenando los parámetros (Regresión Logística, Perceptrón Multicapa, *Random Forest*).

Estas técnicas se han evaluado sobre los dos conjuntos de datos utilizados, y los resultados se han comparado con los obtenidos por los principales estudios del estado del arte.

1.3. Herramientas utilizadas

Este trabajo ha sido implementado en su totalidad en *Jupyter Notebooks* utilizando el lenguaje de programación Python. Para asistir en la implementación, se han empleado diversas librerías de Python, entre las que destacan:

- **PyTorch** y **TensorFlow/Keras**: para cargar los modelos de segmentación pre-entrenados, obtener sus probabilidades de salida, y procesarlas aplicando operaciones como *softmax* o *argmax*.

– **Torchvision** y **PIL (Pillow)**: para cargar y transformar (reescalar, normalizar, cambiar formato a RGB...) las imágenes de entrada para los modelos, y para reescalar las probabilidades de salida con interpolación bilineal.

– **OpenCV**: para el procesamiento de imágenes y máscaras de segmentación (lectura y escritura de imágenes, extracción de componentes conexas, reescalado, etc.).

– **Scikit-learn**: para el entrenamiento de los modelos supervisados (Regresión Logística, Perceptrón Multicapa y *Random Forest*).

– **NumPy**: para operaciones matriciales, funciones matemáticas, uso de operadores lógicos (OR, AND), etc.

– **Pandas**: para la organización de los archivos de métricas y resultados.

– **Pathlib** y **os**: para cargar y guardar archivos.

1.4. Desarrollo temporal

La Figura 1.3 a continuación muestra un cronograma con la planificación y el desarrollo temporal del trabajo. Cada mes se divide en cuatro semanas, y para cada tarea se muestra el tiempo inicialmente planificado (azul), el tiempo real empleado (rojo), y las coincidencias entre ambos (morado).

Figura 1.3: Cronograma con la planificación y ejecución temporal del TFG.

1.5. Estructura de la memoria

El documento se ha estructurado en seis capítulos:

- **Capítulo 1: Introducción y objetivos**. En este primer capítulo se introduce el tema de la selección de blastocistos en la FIV, y se explican las técnicas actuales para realizar esta tarea y sus limitaciones. Después, se describen los objetivos y el alcance de este TFG, se detallan las herramientas utilizadas y se presenta un cronograma con el desarrollo temporal.

- **Capítulo 2: Estado del arte y punto de partida.** En este capítulo se resumen los principales estudios realizados en el área de la segmentación automática de blastocistos. A continuación, se explica el punto de partida de este TFG, enmarcado como una extensión de uno de los estudios más relevantes del estado del arte. En la explicación se incluyen los datasets y modelos empleados.

- **Capítulo 3: Estrategias de ensemble no supervisado**. En este capítulo se explican las estrategias de *ensemble* implementadas que no requieren de entrenamiento adicional. Estas técnicas se basan en post-procesado de máscaras, operadores lógicos (OR, AND, voto mayoritario) y operadores sobre las probabilidades de salida de los modelos base (*softmax*, *max*, suma ponderada, reescalado de probabilidades).

- **Capítulo 4: Estrategias de ensemble supervisado**. En este capítulo se explican las estrategias de *ensemble* implementadas que se basan en aprendizaje supervisado. Se detalla el proceso de construcción de los datasets para el aprendizaje y se explican los diferentes modelos entrenados (Regresión Logística, Perceptrón Multicapa, *Random Forest*).

- **Capítulo 5: Resultados y análisis**. En este capítulo se presentan y discuten los resultados obtenidos para cada una de las estrategias de *ensemble*, resaltando los mejores modelos. Las métricas se comparan primero con las del estudio previo en el que se basa este trabajo, y posteriormente con las del resto de estudios del estado del arte.

- **Capítulo 6: Conclusiones**. En este último capítulo se resumen las principales aportaciones del trabajo, se discuten sus limitaciones y se plantean posibles líneas de mejora y trabajo futuro.

Capítulo 2

Estado del arte y punto de partida

En los últimos años, se han seguido dos principales enfoques en el ámbito de la segmentación automática de blastocistos: los métodos de procesamiento de imágenes, y los que utilizan modelos de *deep learning*, principalmente redes neuronales convolucionales (CNNs, por sus siglas en inglés, *Convolutional Neural Networks*). Por lo general, los métodos de *deep learning* son más recientes y obtienen mejores resultados que los anteriores.

2.1. Trabajos previos

Aunque no muy numerosos, existen diversos trabajos de investigación que buscan segmentar de forma automática las principales estructuras del blastocisto: ZP, TE y MCI.

Entre las principales aportaciones en este área, cabe destacar las siguientes: en 2014, Singh et al. [13], utilizaron un algoritmo de contornos de nivel para segmentar la región del TE. Posteriormente, Kheradmand et al. [14], entrenaron en 2016 una red neuronal cuyo fundamento es aplicar Transformadas de Coseno Discretas (DCT, por sus siglas en inglés *Discrete Cosine Transform*) a imágenes para clasificar las componentes del blastocisto (ZP, TE y MCI), y en 2017 [15] utilizaron una *Fully Convolutional Network* (FCN) para segmentar la MCI. Ese mismo año, Saeedi et al. [16] desarrollaron un algoritmo que combina información de textura con características físicas para segmentar automáticamente el TE y la MCI. Entre 2017 y 2018, Rad et al. llevaron a cabo tres estudios: en el primero [17], usaron un método basado en texturas (Gabor [18] y DCT) y contornos de nivel para segmentar la MCI, en el segundo [19], utilizaron una red neuronal jerárquica (*Hierarchichal Neural Network*, HNN) para segmentar la ZP, y en el tercero [20], utilizaron un *ensemble* de redes de tipo *Dilated U-Net* para la segmentación de la MCI. En 2019, Harun et al. [21], implementaron una red profunda de tipo *Residual Dilated U-Net* para segmentar el TE y otra para la MCI. Finalmente,

en 2023, Farias et al. [22] propusieron un proceso de extracción de características a partir de imágenes de blastocistos para entrenar una red neuronal que clasifica cada píxel en ZP, TE, MCI, BC y fondo.

Más recientemente, en 2024, destaca el trabajo de Villota et al. [11], que además de implementar y comparar diversos modelos de *deep learning* para la segmentación de blastocistos, tiene como objetivo replicar y evaluar trabajos previos cuyo código no estaba disponible públicamente. En dicho estudio, se entrenan arquitecturas estándares de segmentación como DeepLab [23], HRNet [24], U-Net [25] y Residual Dilated U-Net (tratando de replicar el trabajo de Harun [21]), y se obtienen resultados competitivos para la segmentación de las tres principales estructuras del blastocisto (ZP, TE y MCI). El trabajo de Villota et al. destaca especialmente por proporcionar su código y modelos en un repositorio de GitHub público [12], facilitando la reproducibilidad y extensión de sus resultados.

La mayoría de los estudios previamente mencionados utilizan para el entrenamiento y/o evaluación el conjunto de datos propuesto por Saeedi et al. [16], cuya descripción se puede encontrar en la Sección 2.3.1.

2.2. Resumen de resultados previos

A continuación se resumen los resultados del estado del arte para la segmentación automática de las tres principales estructuras del blastocisto (ZP, TE y MCI). Las Tablas 2.1, 2.2 y 2.3 muestran los valores obtenidos en los estudios previos para métricas como *Accuracy*, *Precision*, *Recall*, *Dice Coefficient* y *Jaccard Index*. La definición cuantitativa de estas métricas así como su interpretación puede encontrarse en [26].

Cabe destacar que no es una comparación perfecta, ya que no todos los estudios utilizan el conjunto de datos propuesto por Saeedi et al. [16] para la evaluación. Algunos utilizan otros datasets privados [14, 15, 17, 13], o diferentes particiones de dicho conjunto [19]. Además, ninguno de estos trabajos, excepto el de Villota et al., ofrece acceso a sus modelos o al código, lo cual imposibilita una evaluación reproducible e imparcial. Por ello, en las tablas solo se reflejan los resultados tal y como fueron reportados en las publicaciones originales.

	Accuracy	Precision	Recall	Dice Coef.	Jaccard Idx.
Kheradmand et al. [14]	0.92	0.80	0.81	-	0.64
Rad et al. [19]	0.95	0.79	**0.91**	-	0.74
Farias et al. [22]	0.94	0.85	0.69	0.75	-
Villota et al. [11]	**0.97**	**0.92**	0.84	**0.87**	**0.78**

Tabla 2.1: Resultados del estado del arte para la segmentación de la ZP (Mejor en negrita).

	Accuracy	Precision	Recall	Dice Coef.	Jaccard Idx.
Singh et al. [13]	0.87	0.71	0.83	0.77	0.62
Kheradmand et al. [14]	0.90	0.69	0.80	0.74	0.59
Saeedi et al. [16]	0.86	0.69	0.89	0.77	-
Harun et al. [21]	**0.98**	**0.92**	**0.93**	**0.92**	**0.85**
Farias et al. [22]	0.93	0.80	0.59	0.67	-
Villota et al. [11]	0.97	0.88	0.84	0.85	0.75

Tabla 2.2: Resultados del estado del arte para la segmentación del TE (Mejor en negrita).

	Accuracy	Precision	Recall	Dice Coef.	Jaccard Idx.
Kheradmand et al. [14]	0.93	0.76	0.56	0.64	0.48
Kheradmand et al. [15]	0.96	-	-	0.87	0.77
Saeedi et al. [16]	0.91	0.77	0.84	0.79	-
Saeedi et al. (DLRS) [16]	0.93	0.84	0.78	0.83	-
Rad et al. [17]	-	0.79	0.87	0.83	0.70
Rad et al. [20]	0.98	0.89	0.92	0.90	0.82
Harun et al. [21]	**0.99**	**0.95**	**0.94**	**0.94**	**0.89**
Farias et al. [22]	0.96	0.87	0.62	0.67	-
Villota et al. [11]	0.98	0.88	0.87	0.87	0.79

Tabla 2.3: Resultados del estado del arte para la segmentación de la MCI (Mejor en negrita).

Como se puede observar, el trabajo de Villota et al. obtiene los mejores resultados para la segmentación de la ZP, mientras que el de Harun et al. reporta un mejor rendimiento en la segmentación del TE y la MCI. Sin embargo, los resultados de Harun no son fácilmente reproducibles. De hecho, el estudio de Villota et al. intentó replicar el modelo de Harun y no consiguió alcanzar el mismo rendimiento, aunque sus modelos basados en DeepLab y HRNet sí superan al resto de estudios analizados.

Por lo tanto, podemos concluir que el trabajo de Villota et al. es actualmente la referencia más sólida del estado del arte, debido a su completitud, buenos resultados y fácil reproducibilidad.

2.3. Punto de partida de este trabajo

El punto de partida de este TFG será el trabajo realizado por Villota et al. [11], debido a que es la propuesta más reciente y completa dentro de la literatura actual, y además es el único que publica su código abiertamente. Por lo tanto, este trabajo tomará como referencia sus resultados con el objetivo de mejorarlos mediante el uso de técnicas de *ensemble learning* [9].

2.3.1. Conjuntos de datos

Para este trabajo se cuenta con dos conjuntos de datos distintos.

Conjunto público. El primero es el publicado por Saeedi et al. en [16], que contiene 249 imágenes de microscopio de embriones humanos en estadio de blastocisto (Figura 2.1), anotadas manualmente por especialistas del *Pacific Centre for Reproductive Medicine* (PCRM) en Canadá. Las anotaciones incluyen la segmentación de las estructuras ZP, TE e MCI, así como información adicional sobre el grado de calidad del TE e MCI, y el resultado de la implantación. Este dataset se ha consolidado como el más utilizado en la literatura para la segmentación de blastocistos, habiendo sido empleado en la mayoría de trabajos recientes, incluyendo el de Villota et al. El conjunto de datos se obtuvo mediante solicitud directa a los autores por parte del grupo de investigación TME Lab[1] del Instituto de Investigación en Ingeniería de Aragón (I3A), Universidad de Zaragoza. En cuanto a la división del dataset, para mantener la coherencia con el estudio de Villota et al. y poder comparar los resultados obtenidos en el *ensemble*, se ha utilizado la misma división que en dicho estudio, utilizando el 85 % de las imágenes como conjunto de entrenamiento y el 15 % como conjunto de test. A lo largo de este trabajo nos referiremos a este dataset como Saeedi.

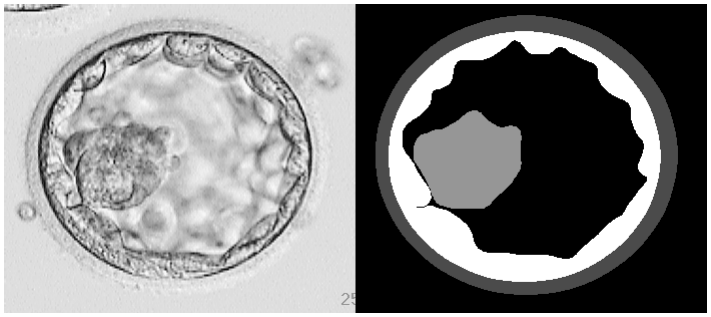

Figura 2.1: Imagen del dataset Saeedi junto a sus anotaciones de segmentación.

[1] `https://tmelab.unizar.es/`

Conjunto privado. El segundo conjunto de datos utilizado es un conjunto privado proporcionado por embriólogos del Hospital Quirónsalud de Zaragoza al grupo de investigación TME Lab. El dataset contiene 25 imágenes microscópicas de blastocistos y sus anotaciones con la segmentación de las estructuras ZP, TE e MCI (Figura 2.2). Este conjunto solamente será empleado para evaluación, con el objetivo de comprobar que las técnicas de *ensemble* implementadas generalizan bien a datos provenientes de una fuente distinta no vista durante el entrenamiento. A lo largo de este trabajo nos referiremos a este dataset como Quirón.

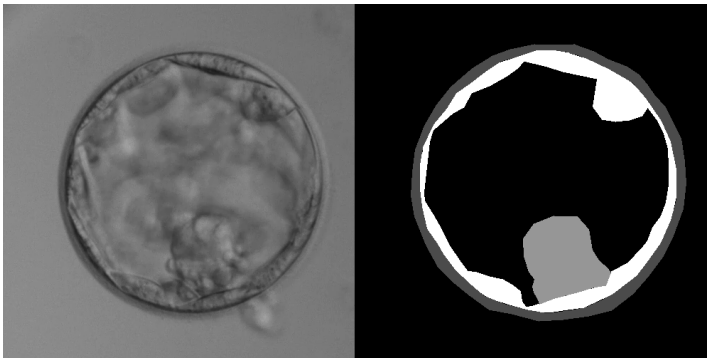

Figura 2.2: Imagen del dataset Quirón junto a sus anotaciones de segmentación.

2.3.2. Modelos de deep learning

En este trabajo se han utilizado los modelos entrenados publicados por Villota et al. [11], disponibles en su repositorio de GitHub [12]. Se han descargado los pesos de los modelos, y se han utilizado para obtener predicciones sobre los datasets Saeedi y Quirón, con el objetivo de aplicar técnicas de *ensemble learning* [9] sobre estas salidas para optimizar la precisión y robustez de la segmentación. En concreto, las arquitecturas utilizadas para el *ensemble* son DeepLab [23], High-Resolution Network (HRNet) [24], U-Net [25] y Residual Dilated U-Net (RDU-Net) [21]. Todas ellas son arquitecturas de redes neuronales convolucionales (CNNs) [27], y son comúnmente utilizadas en tareas de segmentación semántica [10]. A continuación se describe brevemente cada una de estas arquitecturas.

La arquitectura U-Net fue una de las primeras en lograr un éxito rotundo en tareas de segmentación [28]. Se trata de una arquitectura muy utilizada en el campo de la segmentación biomédica [29], diseñada para ser eficaz en tareas con conjuntos de datos limitados [30] como la nuestra. Tiene una estructura en forma de "U" (Figura 2.3) compuesta por dos partes principales: un codificador (*encoder*) que captura información contextual y reduce la resolución espacial, y un decodificador (*decoder*) simétrico que aumenta la resolución con capas de *upsampling* y genera el mapa de segmentación. Otra característica de U-Net es el uso de conexiones de salto (*skip connections*),

que conectan directamente capas del *encoder* con el *decoder*, y ayudan a preservar información perdida durante la reducción del *encoder*.

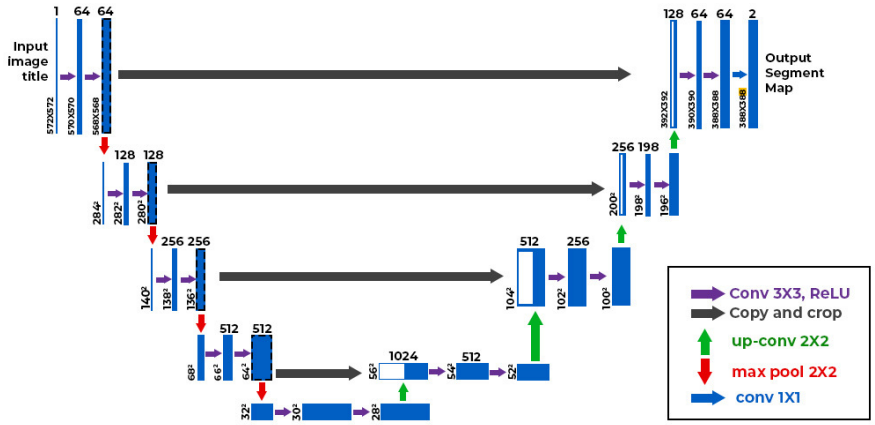

Figura 2.3: Arquitectura U-Net (Fuente: [30]).

DeepLab (Figura 2.4) es una arquitectura de CNN que destaca por usar convoluciones dilatadas (*dilated/atrous convolutions*), que permiten extraer características de las imágenes de entrada sin reducir la resolución espacial de los mapas de atributos. Además, utiliza un módulo llamado *Atrous Spatial Pyramid Pooling* (ASPP), que aplica múltiples convoluciones dilatadas con diferentes tasas de dilatación en paralelo, permitiendo extraer características a distintas escalas. Esta arquitectura se ha consolidado como una de las más robustas para tareas de segmentación en imágenes complejas.

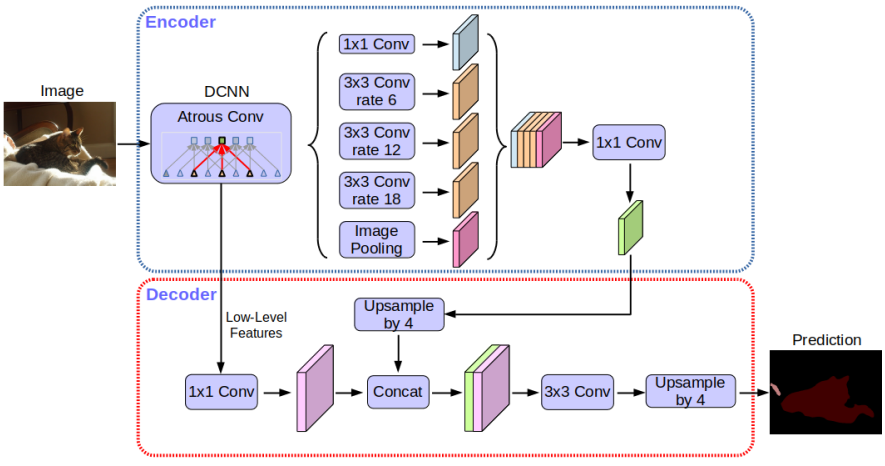

Figura 2.4: Esquema general del flujo de la arquitectura DeepLab (Fuente: [31]).

High-Resolution Network (HRNet) se llama de esta manera porque es capaz de mantener representaciones de alta resolución a lo largo de todo el *pipeline* de la red, mientras que otras arquitecturas como U-Net reducen la resolución espacial. Esto se consigue mediante la combinación de múltiples convoluciones paralelas con distintas

resoluciones, que intercambian información entre sí. Estas características hacen que HRNet logre una segmentación precisa, especialmente en imágenes que contengan detalles finos importantes, como en la segmentación de blastocistos.

Por último, RDU-Net es una variante de U-Net que incorpora bloques residuales y convoluciones dilatadas. Los bloques residuales utilizan *skip connections* y ayudan a mitigar el problema del desvanecimiento del gradiente en redes profundas [32].

2.3.3. Estructura de los datos

Para la segmentación de blastocistos, estos modelos toman como entrada imágenes de blastocistos al microscopio junto con sus anotaciones (*ground truth*). Para el conjunto de Saeedi, las anotaciones consisten en máscaras segmentadas donde cada una de las principales estructuras del blastocisto toma un valor de intensidad de píxel distinto (fondo: 0, ZP: 75, TE: 255, MCI: 150), como se puede observar en la Figura 2.5. Estos valores se convierten en etiquetas enteras (0, 1, 2, 3) para representar cada clase durante el entrenamiento.

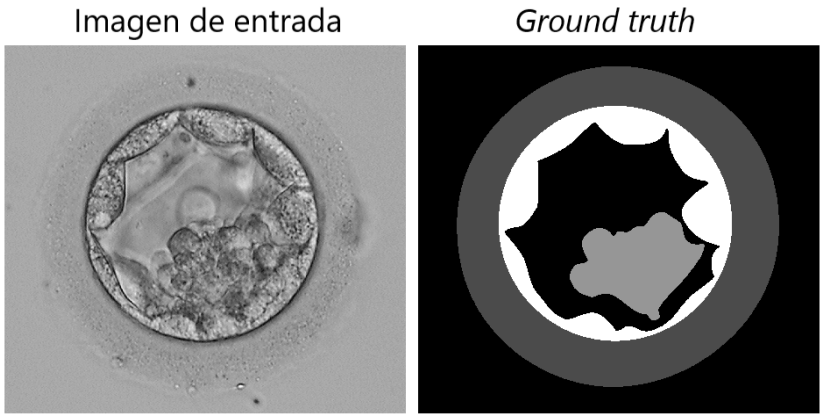

Figura 2.5: Ejemplo de entrada para los modelos, incluyendo la imagen de blastocisto al microscopio y su máscara de segmentación (*ground truth*).

Estos modelos generan como salida un tensor de probabilidades para cada imagen de entrada, que contiene para cada píxel las probabilidades de pertenecer a cada una de las 4 clases (fondo, ZP, TE y MCI). Por lo tanto, para poder comparar las predicciones de los modelos con las etiquetas del *ground truth*, el tensor de probabilidades se transforma en una máscara segmentada con el mismo formato que el usado en las anotaciones de Saeedi. Para ello, primero se aplica la función *argmax*, que selecciona para cada píxel la clase con la probabilidad más alta, y después se crea la máscara de segmentación asignando a cada píxel su valor de intensidad correspondiente (fondo: 0, ZP: 75, TE: 255, MCI: 150), como se muestra en la Figura 2.6.

A partir de las salidas de los cuatro modelos descritos, tanto a nivel de máscaras de

segmentación como a nivel de probabilidades de salida, se han implementado distintas estrategias de *ensemble learning* [9], con el objetivo de mejorar los resultados obtenidos por cada modelo individual.

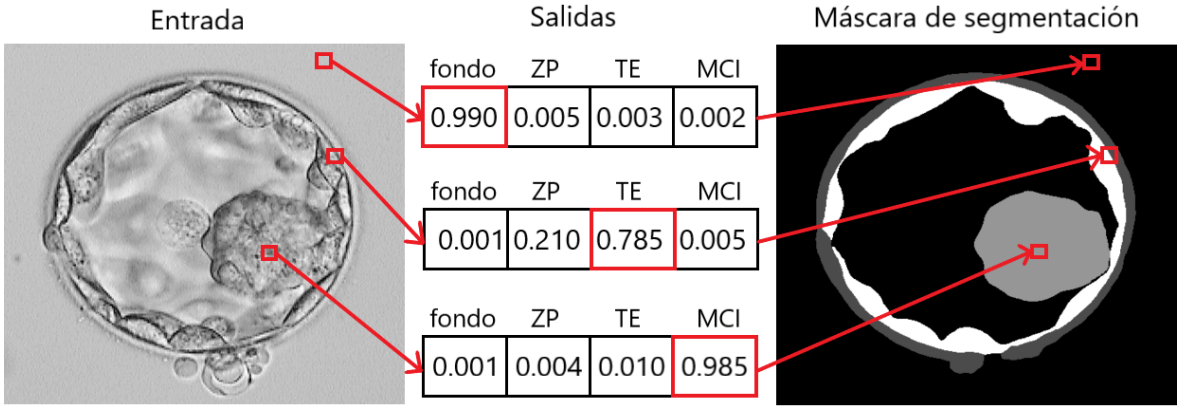

Figura 2.6: Proceso de construcción de las máscaras de segmentación a partir de las salidas de probabilidad de los modelos.

Capítulo 3

Estrategias de ensemble no supervisado

En este capítulo se describen las distintas técnicas de *ensemble* no supervisado desarrolladas para mejorar la segmentación automática de blastocistos. A diferencia de las estrategias basadas en aprendizaje supervisado descritas en el Capítulo 4, estas técnicas no requieren de entrenamiento adicional, ya que se basan en procesar y combinar las salidas individuales de los cuatro modelos entrenados y publicados en el estudio de Villota et al. [11, 12] (DeepLab, HRNet, U-Net y RDU-Net). Todas las técnicas han sido implementadas en *notebooks* de Jupyter en Python.

En total, se han planteado cuatro enfoques de *ensemble* no supervisado:

1. Post-procesado de máscaras individuales.
2. Operadores lógicos para hibridar varios modelos (OR, AND, voto mayoritario).
3. Combinación de post-procesado y operadores lógicos.
4. Operadores sobre la salida de probabilidades de varios modelos (*softmax*, *max*, suma ponderada, reescalado de probabilidades).

En las siguientes secciones se describe detalladamente en qué consiste cada estrategia, así como su lógica e implementación.

3.1. Post-procesado de máscaras individuales

Antes de comenzar con la combinación de modelos, se ha implementado un algoritmo de procesamiento de imágenes diseñado para eliminar cierto ruido presente en las máscaras de predicción generadas por los modelos base.

Como se puede observar en la Figura 3.1, algunas máscaras de predicción contienen imperfecciones, principalmente en forma de pequeños grupos de píxeles aislados que son ruido y realmente no forman parte de ninguna de las estructuras del blastocisto.

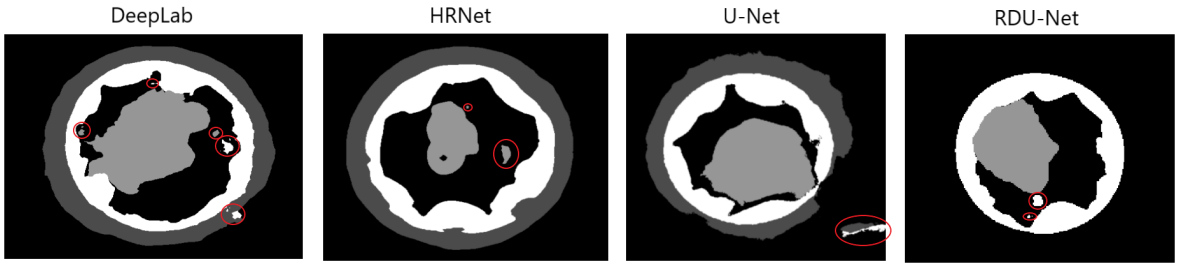

Figura 3.1: Ejemplos de predicciones con ruido de los distintos modelos utilizados.

Además, otra imperfección en las predicciones (excepto en las del modelo RDU-Net) es que contienen un suavizado o degradado de píxeles en los bordes de cada estructura, como se observa en la Figura 3.2. Este efecto no está presente en el *ground truth* (anotaciones de referencia), y afecta negativamente a la precisión de los bordes. La causa del suavizado es un *resize* (reescalado) realizado en el código original para la obtención de las máscaras, al transformar las máscaras del tamaño fijo de salida de los modelos al tamaño original de la imagen de entrada.

Figura 3.2: Ejemplo de suavizado de bordes en una predicción.

Para mitigar estos problemas, tanto los píxeles aislados con ruido como el suavizado de bordes, se han implementado tres versiones de un mismo algoritmo de post-procesado. El algoritmo base se centra en el problema de los píxeles aislados, y consiste en conservar para cada estructura (ZP, TE, MCI) las componentes conexas cuya área supere un umbral mínimo establecido, es decir, descarta los agrupamientos de píxeles aislados pequeños que suelen representar ruido. El umbral de área mínima se ha determinado de forma empírica, buscando un equilibrio entre eliminar imperfecciones y conservar las regiones relevantes. Para el desarrollo de este algoritmo se ha utilizado la librería de visión por computador *OpenCV* [33] en Python.

Las tres versiones implementadas comparten este algoritmo base de eliminación de componentes conexas pequeñas, y solo se diferencian en los pasos de preproceso aplicados para corregir el suavizado de bordes antes de aplicar el algoritmo. A continuación se describen dichas variantes.

V1: Sin preprocesado. En esta primera versión, se aplica directamente el algoritmo base de eliminación de componentes conexas pequeñas sin ningún paso previo de preprocesado. Esta versión presenta un problema; los píxeles del suavizado de bordes se eliminan ya que no coinciden exactamente con los valores definidos para las estructuras (fondo: 0, ZP: 75, TE: 255, MCI: 150). Como resultado, quedan huecos entre las estructuras, como se puede observar en la Figura 3.3.

Figura 3.3: Ejemplo de bordes de estructuras tras aplicar algoritmo V1.

V2: Eliminación previa del suavizado. Esta variante busca resolver el problema anterior eliminando primero el suavizado de bordes antes de aplicar el algoritmo base. Para eliminar el suavizado, todos los píxeles que no coinciden exactamente con los valores esperados, son reemplazados por el valor de la estructura más cercana al píxel. Para determinar dicha estructura, se busca en las cuatro direcciones cardinales (arriba, abajo, izquierda, derecha).

Aunque esta solución elimina los huecos entre estructuras (ver Figura 3.4), presenta dos inconvenientes: tiene un mayor coste computacional y sobreestima el área de las estructuras, ya que todos los píxeles del degradado se asignan a la estructura más cercana, incluso aquellos que deberían considerarse como fondo.

Figura 3.4: Ejemplo de bordes de estructuras tras aplicar algoritmo V2.

V3: Eliminación eficiente del suavizado. La tercera versión busca mantener los beneficios de la V2 pero con una implementación más eficiente y precisa. Para ello, los píxeles del suavizado de bordes se reasignan al valor más cercano en intensidad entre los cuatro valores válidos (0, 75, 150, 255), sin necesidad de inspeccionar direcciones vecinas. Este enfoque reduce el coste computacional y mejora la precisión en la separación de estructuras, como se puede ver en la Figura 3.5.

Figura 3.5: Ejemplo de bordes de estructuras tras aplicar algoritmo V3.

Para demostrar la efectividad de esta versión, en la Figura 3.6 se muestran los resultados de aplicar el algoritmo V3 a las predicciones ruidosas originales.

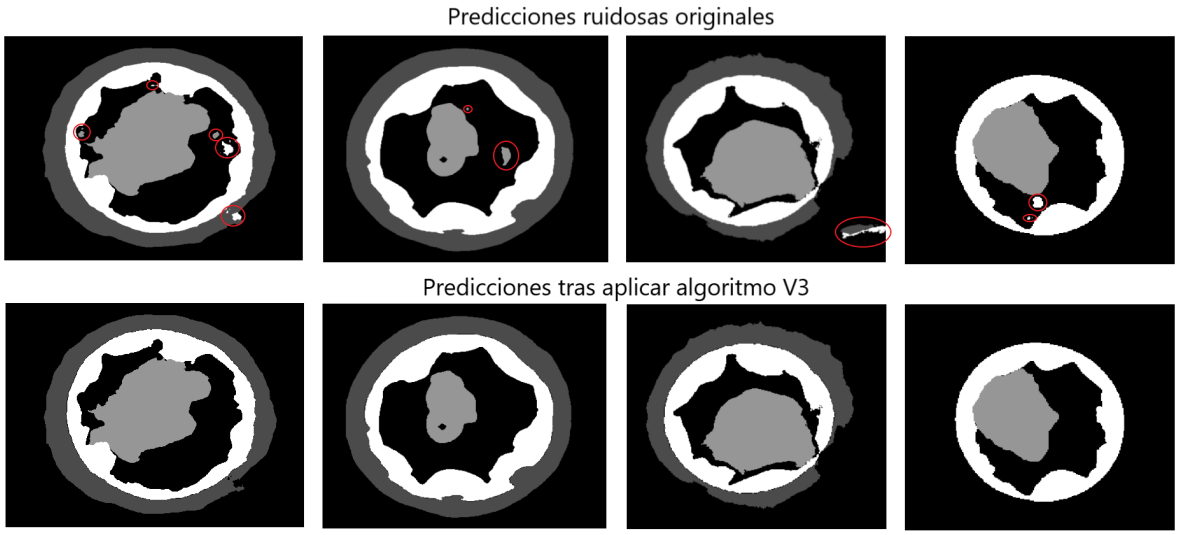

Figura 3.6: Comparación de predicciones antes y después de aplicar el algoritmo V3

Cabe destacar que esta solución tampoco es perfecta. Debido a las similitudes en los valores, algunos de los píxeles entre la ZP y el TE son asignados erróneamente al valor de la MCI, y por tanto son eliminados por el algoritmo base al no superar el umbral de área mínima, dejando pequeños huecos entre las estructuras. En la Sección 3.4 se propone una mejor solución a este problema.

3.2. Operadores lógicos hibridando varios modelos

La siguiente estrategia de *ensemble* no supervisado implementada consiste en combinar las máscaras de predicción generadas por los distintos modelos utilizando operadores lógicos clásicos (OR, AND) y la técnica de voto mayoritario. Estas técnicas son comúnmente utilizadas en tareas de *ensemble*, ya que permiten fusionar la información de varios modelos para aprovechar sus puntos fuertes y compensar sus debilidades. Dependiendo del método utilizado, se puede priorizar la exhaustividad (*recall*) o la precisión [26] de la segmentación. A continuación, se describen las diferentes técnicas aplicadas, su propósito y su implementación.

Operador lógico OR. El operador OR es útil para aumentar el *recall*, aunque puede bajar la precisión. Aplicando este operador a las salidas de varios modelos se consigue que un píxel se considere perteneciente a una estructura si al menos uno de los modelos lo ha predicho como tal. Esta estrategia es útil para asegurar que se incluyan todos los píxeles predichos de regiones relevantes. Es el caso de la MCI, esta estructura formará el futuro cuerpo del feto, y su segmentación completa es fundamental, especialmente si se desea aplicar técnicas invasivas como el PGT-A, que requieren extraer células del embrión sin dañar dicha estructura.

Se han implementado y evaluado diversas combinaciones de modelos, principalmente utilizando los dos modelos que mejores resultados obtienen en el estudio de Villota et al. (DeepLab y HRNet). En concreto, se ha implementado:

- DeepLab OR HRNet.
- DeepLab OR RDU-Net.
- DeepLab OR HRNet OR U-Net.
- DeepLab OR HRNet OR RDU-Net.

Para la implementación, se ha desarrollado un algoritmo que itera sobre las máscaras de predicción de los modelos implicados y aplica la operación OR para cada una de las estructuras del blastocisto (ZP, TE, ICM) por separado. Después, los resultados de cada estructura se recombinan en una sola imagen final, resolviendo los posibles conflictos con prioridad MCI > TE > ZP. Es decir, si un mismo píxel es asignado simultáneamente a varias estructuras, se le termina asignando la estructura de mayor prioridad. Como ya se ha comentado, la prioridad se establece teniendo en cuenta que la segmentación de la MCI es especialmente crítica.

Operador lógico AND. El operador AND tiene el efecto contrario, aumenta la precisión a costa del *recall*. Con este operador se consigue que un píxel solo se incluya en una estructura si todos los modelos implicados coinciden en su predicción. Es una estrategia más conservadora, útil para evitar falsos positivos.

Como aplicar AND con modelos de bajo rendimiento puede perjudicar el resultado, los experimentos se han centrado en combinar únicamente los mejores modelos:

- DeepLab AND HRNet.
- DeepLab AND RDU-Net (buscando una segmentación precisa de la MCI).

La implementación es similar a la anterior, aplicando el AND por separado a cada estructura, y combinando los resultados en una sola imagen final, sin necesidad de priorizar estructuras, ya que el AND no produce conflictos.

Voto mayoritario. El voto mayoritario es otra técnica común de *ensemble* que ofrece un equilibrio entre OR y AND. Con esta técnica se consigue que un píxel solo se incluya en una estructura si la mayoría de modelos participantes coinciden en su predicción. Esta estrategia permite corregir errores aislados cometidos por un modelo concreto.

Para evitar empates frecuentes si utilizáramos los cuatro modelos, se ha limitado el voto mayoritario a grupos de tres. Las combinaciones implementadas han sido:

- VotoMayoritario(DeepLab, HRNet, U-Net).
- VotoMayoritario(DeepLab, HRNet, RDU-Net).

Del mismo modo que los anteriores, el algoritmo implementado itera sobre las máscaras de los modelos implicados y aplica el criterio de mayoría para cada estructura por separado, combinando los resultados al final.

3.3. Combinación de post-procesado y operadores lógicos

La siguiente estrategia de *ensemble* no supervisado consiste en hibridar las dos estrategias previas, buscando reducir el ruido en las predicciones y al mismo tiempo combinar las salidas de varios modelos. Para ello, primero se han obtenido las máscaras "limpias" resultantes de aplicar el mejor algoritmo de post-procesado (el algoritmo denotado como V3) a las predicciones de cada modelo. Estas máscaras corregidas se han combinado entre sí utilizando operadores lógicos y voto mayoritario, del mismo modo que en la Sección 3.2. Además, también se ha experimentado con algunas variantes en

las que no se aplica el post-procesado a todos los modelos, y otras en las que se combinan operaciones de OR con AND. Más concretamente, las combinaciones implementadas son las siguientes:

- V3-DeepLab OR V3-HRNet (denominado como OR(V3-DL, V3-HR) en el Capítulo 5).
- V3-DeepLab OR V3-RDU-Net.
- DeepLab OR V3-RDU-Net.
- (V3-HRNet AND RDU-Net) OR V3-DeepLab.
- (V3-HRNet AND Deeplab) OR V3-RDU-Net.
- VotoMayoritario(V3-DeepLab, V3-HRNet, V3-RDU-Net).
- VotoMayoritario(DeepLab, V3-HRNet, V3-RDU-Net).

Estas combinaciones se han elegido en función de los resultados previos, tratando de optimizar aún más las mejores estrategias.

3.4. Operadores sobre la salida de probabilidades de varios modelos

La última estrategia de *ensemble* no supervisado implementada se basa en operar sobre las probabilidades de salida de los modelos. Los modelos utilizados no generan máscaras segmentadas directamente, sino que devuelven un tensor tridimensional de probabilidades con la forma (`n_clases`, `alto_img`, `ancho_img`), que contiene para cada píxel de la imagen de entrada, la probabilidad de pertenecer a cada una de las cuatro posibles clases (fondo, ZP, TE y MCI). Para obtener la máscara final de la segmentación, a este tensor, se aplica al tensor la función *argmax*, que asigna a cada píxel la clase con la probabilidad más alta. El modelo RDU-Net es un caso especial que utiliza dos variantes, una para predecir la probabilidad de TE y otra para la de MCI. En esta sección, se exploran diferentes técnicas de *ensemble* aplicadas a estas probabilidades de salida, antes de ser convertidas en máscaras de segmentación.

Todas las técnicas desarrolladas tienen una base común: se obtiene el tensor de probabilidades de cada modelo y se aplica la función *softmax* [34]. Esta función normaliza las probabilidades de modo que sumen 1 en cada píxel. A continuación, en lugar de aplicar la función *argmax* inmediatamente, se realiza primero el reescalado con interpolación bilineal [35] del propio tensor de probabilidades, así al convertirlo a máscaras segmentadas no hace falta reescalar las máscaras, y por lo tanto no se

genera el suavizado de bordes de estructuras descrito en la Sección 3.1, logrando así una segmentación más precisa. Esta operación de reescalado de probabilidades se ha aplicado a todos los métodos de *ensemble* descritos a continuación.

Combinación de probabilidades reescaladas. Como la operación reescalado de probabilidades logra una mejor segmentación, se han vuelto a combinar las máscaras de los mejores modelos tras aplicar este reescalado, de manera similar a la sección 3.3. En particular, se han probado las siguientes combinaciones, buscando maximizar el *recall* de las estructuras (ZP, TE y especialmente MCI):

- RP-DeepLab OR RP-HRNet (denominado como OR(RP-DL, RP-HR) en el Capítulo 5)
- RP-DeepLab OR RP-HRNet OR RP-RDU-Net
- RP-DeepLab OR RP-HRNet OR RP-RDU-Net OR RP-U-Net

Donde el prefijo RP- indica que se ha aplicado previamente el reescalado probabilístico.

Operación max. Esta estrategia de *ensemble* consiste en combinar las probabilidades de salida de varios modelos, eligiendo para cada píxel, la clase que indica el modelo que más seguro de su predicción esté, es decir, la clase asociada a la probabilidad máxima de entre todos los modelos. Esta técnica es útil para resolver ambigüedades en píxeles conflictivos, ya que tiende a descartar predicciones erróneas de baja confianza, aunque es menos eficaz cuando alguno de los modelos realiza predicciones incorrectas con seguridad. Por lo tanto, las combinaciones implementadas aplicando esta estrategia se centran en los mejores modelos (DeepLab y HRNet), y son las siguientes:

- max(Deeplab, HRNet) (denominado como Max(DL, HR) en el Capítulo 5)
- max(DeepLab, RDU-Net)
- max(HRNet, RDU-Net)
- max(DeepLab, HRNet, RDU-Net)
- max(DeepLab, HRNet, U-Net)

Para la implementación, se ha desarrollado un algoritmo que itera sobre las imágenes de un directorio de entrada, obtiene los tensores de probabilidad de los modelos implicados, y los procesa aplicando la operación max. En concreto, para cada píxel

de una imagen, primero se mantiene la probabilidad máxima por clase de entre todos los modelos, y luego se asigna al píxel la clase con la probabilidad más alta.

Suma ponderada. Esta estrategia de *ensemble* consiste en combinar las probabilidades de salida de varios modelos aplicando una suma ponderada, es decir, se multiplican las probabilidades de cada modelo por un peso específico asignado previamente, y se suman clase por clase. Posteriormente, se asigna a cada píxel la clase con la probabilidad total más alta. Esta técnica es útil para combinar varios modelos pudiendo ajustar la influencia de cada uno según su rendimiento. En este caso, se han implementado múltiples combinaciones de 2, 3 y 4 modelos, probando diversas configuraciones de pesos que suman 1 siguiendo la siguiente fórmula:

$$Y = \alpha \cdot Y_{\text{DeepLab}} + \beta \cdot Y_{\text{HRNet}} + \gamma \cdot Y_{\text{RDU-Net}} + \lambda \cdot Y_{\text{U-Net}}$$

con $\alpha+\beta+\gamma+\lambda = 1$. En general, se ha dado más peso a las predicciones de los modelos DeepLab y HRNet por obtener mejores resultados. Un ejemplo de caso probado es $(\alpha, \beta, \gamma, \lambda) = (0.4, 0.4, 0.1, 0.1)$.

Para la implementación, se ha desarrollado un algoritmo análogo al anterior, pero que en vez de la función max, aplica la suma ponderada, multiplicando las predicciones de cada modelo por su peso, sumándolas por clases, y asignando a cada píxel la clase con la probabilidad total más alta.

Capítulo 4

Estrategias de ensemble supervisado

En este capítulo se describen las técnicas de *ensemble* basadas en aprendizaje supervisado que han sido desarrolladas para mejorar la segmentación automática de blastocistos. A diferencia de las estrategias no supervisadas del Capítulo 3, estas técnicas se basan en el entrenamiento de modelos de aprendizaje automático capaces de aprender cómo combinar de forma óptima las predicciones de los modelos base (DeepLab, HRNet, U-Net y RDU-Net).

En total, se han explorado tres algoritmos supervisados:

1. Regresión Logística.

2. Perceptrón Multicapa (MLP, por sus siglas en inglés *Multilayer Perceptron*).

3. *Random Forest*.

Estos algoritmos requieren como entrada un conjunto de datos en forma de una matriz de dimensiones `(n_muestras, n_atributos)`. En este caso, cada muestra representa un píxel y sus atributos son la combinación de las probabilidades de los modelos base para ese píxel. Además se requiere un vector de etiquetas (*ground truth*) asociadas a cada muestra para el entrenamiento, de tamaño `(n_muestras)`. Por tanto, ha sido necesario construir conjuntos de datos específicos a partir de las salidas de probabilidad de los modelos base, para poder representar el problema en el formato requerido por los algoritmos supervisados. A continuación, se describe el proceso de construcción de conjuntos de datos y los experimentos realizados con los distintos algoritmos.

4.1. Construcción de conjuntos de datos

Para construir el conjunto de datos que se utilizará en las técnicas de *ensemble* supervisado, se emplearán los dos conjuntos de datos de partida (SAEEDI y QUIRÓN). El conjunto de entrenamiento se formará a partir de las imágenes del conjunto de entrenamiento de SAEEDI, y el conjunto de evaluación se formará a partir del conjunto de test de SAEEDI y el conjunto íntegro de QUIRÓN (los mismos que se han utilizado para evaluar el resto de estrategias no supervisadas).

El proceso de construcción de los conjuntos de datos comienza con la obtención de las predicciones de los modelos base (DeepLab, HRNet, U-Net y RDU-Net) para cada imagen. Después, las predicciones de cada modelo se concatenan para cada píxel, y todos los píxeles de todas las imágenes se concatenan para formar el conjunto de datos de partida para los modelos de *ensemble* supervisados. De esta manera, cada píxel se representa como un vector de 14 atributos donde:

- 12 atributos (4 clases x 3 modelos) se corresponden a las probabilidades de pertenecer a cada una de las cuatro clases (fondo, ZP, TE, MCI) según los modelos DeepLab, HRNet y U-Net.
- 2 atributos se corresponden a las probabilidades de TE y MCI del modelo RDU-Net.

Por tanto, los conjuntos de datos implementados tienen la estructura `(n_píxeles_dataset, 14)`. A continuación, se describen las distintas variantes de conjuntos de datos construidos y su propósito.

4.1.1. Conjuntos de datos de entrenamiento

Para el entrenamiento de los modelos, se ha construido un conjunto completo utilizando todos los datos disponibles, y varios conjuntos de tamaño reducido para agilizar el entrenamiento y la selección de hiperparámetros, sin comprometer la calidad del aprendizaje.

Para cada uno de estos conjuntos, se ha generado también un vector de etiquetas de *ground truth* (denominado `y_train`), que asocia a cada píxel un valor entero entre 0 y 3, correspondiente a su clase real (fondo: 0, ZP: 1, TE: 2, MCI: 3) según las anotaciones de SAEEDI. En concreto, los conjuntos de entrenamiento construidos son:

Conjunto completo. Para construir este conjunto, se incluyen todos los píxeles de todas las imágenes del conjunto de entrenamiento de SAEEDI. El conjunto resultante tiene `n_muestras` = 38 663 787, con 14 atributos por muestra.

Conjunto reducido de píxeles con incertidumbre. Este conjunto incluye únicamente los píxeles en los que al menos uno de los modelos base presenta incertidumbre, es decir, cuando la probabilidad máxima asignada por ese modelo es inferior a un umbral predefinido de 0.7. En este caso, el conjunto resultante tiene `n_muestras` = 3 506 331.

Esta estrategia permite reducir en un orden de magnitud el tamaño del conjunto completo a la hora de realizar el entrenamiento, descartando los píxeles en los que todos los modelos están seguros de su predicción. La hipótesis subyacente es que, si el algoritmo supervisado aprende a clasificar correctamente los píxeles con incertidumbre, generalizará bien a aquellos en los que hay alta confianza.

Conjunto reducido de píxeles conflictivos. Incluye únicamente los píxeles conflictivos, es decir, aquellos en los que al menos dos modelos base no están de acuerdo en su predicción de clase. El conjunto resultante tiene `n_muestras` = 2 460 378.

Este conjunto representa una versión aún más reducida que el anterior, centrada solo en los casos en los que hay desacuerdo entre modelos, que podrían aportar información más discriminatoria para el entrenamiento.

Conjunto reducido combinado. Este conjunto incluye tanto los píxeles con incertidumbre como los conflictivos, permitiendo incluir casos donde puede haber alta certeza pero desacuerdo, o incertidumbre sin conflicto. El conjunto resultante tiene `n_muestras` = 4 207 767.

De esta manera se obtiene una representación más completa incluyendo ambos casos problemáticos.

Conjunto reducido ampliado. Este conjunto es una extensión del conjunto combinado que incluye también una muestra aleatoria del 5 % de los píxeles en los que no hay ni incertidumbre ni conflicto, con el objetivo de aportar cierta representación del conjunto general. Para este caso `n_muestras` = 5 930 468.

La hipótesis es que esta combinación mejorará la capacidad de los algoritmos entrenados para generalizar tanto en casos ambiguos como en situaciones más claras.

4.1.2. Conjuntos de evaluación

Se han construido los dos siguientes conjuntos para evaluar el rendimiento de los modelos supervisados entrenados y compararlos con el resto de estrategias de *ensemble* implementadas.

Conjunto de evaluación público (Saeedi). Construido a partir de todos los píxeles de las imágenes del conjunto de test de SAEEDI. El conjunto resultante tiene `n_muestras` = 6 725 389.

Conjunto de evaluación privado (Quirón). Construido a partir de todos los píxeles de las imágenes del conjunto íntegro de QUIRÓN. En este caso `n_muestras` = 6 247 000.

Para evaluar los modelos entrenados, se obtienen sus predicciones sobre estos conjuntos de datos. Estas predicciones tienen la forma de un vector unidimensional de tamaño `(n_píxeles_dataset)`, que contiene para cada píxel de entrada, la etiqueta de clase predicha como un número entero entre 0 y 3 (fondo: 0, ZP: 1, TE: 2, MCI: 3). Con el fin de comparar estas predicciones con el resto de técnicas de *ensemble*, se ha implementado una función que transforma el vector de predicciones en imágenes segmentadas, manteniendo el tamaño original de cada imagen. Esta conversión permite evaluar los resultados usando las métricas habituales y facilita la visualización del rendimiento del modelo.

4.2. Modelos entrenados

Utilizando los conjuntos de datos previamente descritos, se han entrenado y evaluado varios modelos de clasificación supervisada con el objetivo de predecir la clase de cada píxel a partir de las salidas combinadas de los modelos base. Todos los modelos se han implementado utilizando la librería de Python *Scikit-learn* [36].

Para cada modelo, se han realizado varios experimentos con los distintos conjuntos de entrenamiento anteriormente presentados. En general, se han utilizado los conjuntos de tamaño reducido para explorar combinaciones de hiperparámetros de forma eficiente, y posteriormente se han realizado pruebas finales sobre el conjunto completo.

A continuación, se describen los modelos entrenados y los hiperparámetros utilizados:

Regresión Logística. La Regresión Logística [37] es un modelo de clasificación que ajusta una función lineal a los datos de entrada y luego transforma la salida usando una función *sigmoide* [38] (para clasificación binaria) o *softmax* [34] (para clasificación multiclase), obteniendo las probabilidades de pertenencia a cada clase. Es un modelo sencillo y eficiente, aunque solo modela relaciones lineales en los datos, por lo que puede no ser suficiente para capturar patrones complejos. En este trabajo se han entrenado modelos de Regresión Logística multiclase con diferentes configuraciones metodológicas

entre las que destacan:

- Técnicas de regularización: se ha evaluado el modelo sin regularización, con regularización L_2 (Ridge) [39] y combinando regularización L_1 (Lasso) [40] y L_2. La regularización L_2 añade a la función de pérdida la suma de los cuadrados de los coeficientes, penalizando que los pesos crezcan demasiado, lo que ayuda a reducir el sobreajuste. L_1 penaliza la suma de los valores absolutos de los coeficientes, intentando anular los coeficientes menos relevantes. Esto puede actuar como una selección automática de atributos y reducir la colinealidad. La librería de *Scikit-learn* permite controlar la fuerza de la regularización a través del hiperparámetro `C`. Se han hecho pruebas con los siguientes valores `C=1.0, 0.1, 0.01`.

- Algoritmos de optimización: se han probado los algoritmos de optimización L-BFGS y SAGA [41]. Ambos son adecuados para clasificación multiclase y conjuntos de datos grandes, y `saga` permite combinar regularización L_1 y L_2.

Perceptrón Multicapa (MLP). El MLP [42] (del inglés, *MultiLayer Perceptron*) es un modelo de red neuronal compuesto por varias capas de neuronas interconectadas, una capa de entrada, una o varias capas ocultas intermedias, y una capa de salida. A diferencia de la Regresión Logística, el MLP puede modelar relaciones no lineales complejas mediante el uso de funciones de activación no lineales como la unidad lineal rectificada (ReLU, por sus siglas en inglés *Regularized Linear Unit*) [43]. Sin embargo, tiene un mayor coste computacional y requiere un ajuste cuidadoso de los hiperparámetros. Las principales configuraciones metodológicas probadas son:

- Arquitectura de la red: se han probado varias configuraciones de capas ocultas, como `(128, 64)`, `(256, 128)` y `(128, 64, 32)`. Cuantas más capas y neuronas, mayor capacidad para modelar patrones complejos, aunque también hay mayor riesgo de sobreajuste.

- Algoritmo de optimización: se ha utilizado Adam [44], un algoritmo basado en descenso de gradiente estocástico que ajusta automáticamente el *learning rate* durante el entrenamiento.

- Regularización: se ha aplicado regularización L_2 para evitar el sobreajuste con diferentes coeficientes de penalización (`alpha=0.0001, 0.01, 0.1, 0.5, 0.7, 1.0`).

- Escalado de atributos: como paso previo al entrenamiento, se han normalizado los datos de entrada (media 0 y varianza 1) para mejorar la convergencia.

– *Early stopping*: se ha utilizado el parámetro de `early_stopping` para detener automáticamente el entrenamiento cuando no haya mejora en el conjunto de validación durante 10 iteraciones seguidas, para evitar sobreajuste. También se ha establecido una tolerancia, que indica la mínima mejora que tiene que haber en cada época para que se considere que el modelo está aprendiendo. Se han probado los siguientes valores de tolerancia, `tol=0.0001, 0.00001`.

Random Forest. *Random Forest* [45] (bosque aleatorio) es un modelo basado en entrenar múltiples árboles de decisión independientes (ver Figura 4.1). Cada árbol se entrena con diferentes subconjuntos aleatorios del conjunto original, y en cada decisión, se considera solo un subconjunto aleatorio de atributos. La clasificación final se obtiene combinando las salidas de todos los árboles por voto mayoritario.

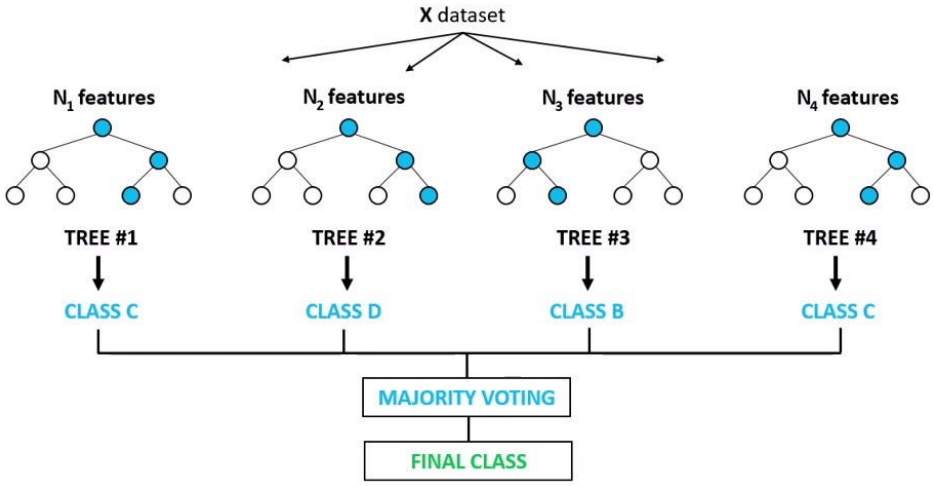

Figura 4.1: Esquema del modelo Random Forest (Fuente: [46]).

Esto resulta en un modelo eficiente y robusto al sobreajuste. Entre los hiperparámetros probados destacan:

– `n_estimators=50, 100`: número de árboles en el bosque.

– `max_depth=5, 10, 15, 20, 30`: profundidad máxima de los árboles. Menos profundidad ayuda a evitar el sobreajuste.

– `min_samples_split=2, 5`: número mínimo de muestras para dividir un nodo.

– `min_samples_leaf=1, 2`: número mínimo de muestras para considerarse hoja.

Capítulo 5

Resultados y análisis

5.1. Comparación con los modelos base

Las Tablas 5.1, 5.2, 5.3, 5.4, 5.5 y 5.6 a continuación resumen los resultados más relevantes obtenidos en los conjuntos de test Saeedi y Quirón para cada una de las estructuras del blastocisto (ZP, TE, MCI), utilizando tanto las técnicas de *ensemble* no supervisado como las supervisadas. Los resultados se comparan con el mejor modelo base del estudio de Villota et al. [11], que puede considerarse el mejor modelo público del estado del arte.

	Accuracy	Precision	Recall	Dice Coef.	Jaccard Idx.
Villota et al. [11]	0.967	**0.922**	0.837	0.872	0.783
OR(V3-DL, V3-HR)	0.969	0.896	0.894	0.891	0.809
OR(RP-DL, RP-HR)	0.969	0.897	0.892	0.890	0.808
Max(DL, HR)	**0.970**	0.915	0.877	**0.891**	**0.811**
Sum(0.4DL, 0.6HR)	0.970	0.909	0.878	0.889	0.808
RegLog_comb	0.970	0.905	0.886	0.891	0.811
MLP_conf_256	0.969	0.873	**0.918**	0.891	0.808
MLP_comb	0.968	0.878	0.911	0.890	0.807
MLP_completo	0.969	0.899	0.889	0.890	0.809
RandFor_comb	0.969	0.900	0.889	0.890	0.809

Tabla 5.1: Resultados en el conjunto de test de Saeedi para la segmentación de la ZP (Mejor en negrita).

	Accuracy	Precision	Recall	Dice Coef.	Jaccard Idx.
Villota et al. [11]	0.968	0.834	0.739	0.780	0.648
OR(V3-DL, V3-HR)	0.970	0.814	0.808	0.809	0.685
OR(RP-DL, RP-HR)	0.970	0.816	0.804	0.807	0.684
Max(DL, HR)	0.970	**0.858**	0.741	0.793	0.665
Sum(0.4DL, 0.6HR)	0.970	0.843	0.767	0.800	0.675
RegLog_comb	0.971	0.839	0.783	0.807	0.684
MLP_conf_256	0.967	0.747	**0.903**	0.815	0.693
MLP_comb	0.968	0.759	0.887	0.815	0.694
MLP_completo	**0.971**	0.808	0.830	**0.816**	**0.696**
RandFor_comb	0.970	0.822	0.805	0.811	0.688

Tabla 5.2: Resultados en el conjunto de Quirón para la segmentación de la ZP (Mejor en negrita).

	Accuracy	Precision	Recall	Dice Coef.	Jaccard Idx.
Villota et al. [11]	0.970	**0.876**	0.838	0.854	0.748
OR(V3-DL, V3-HR)	0.971	0.838	0.908	0.868	0.770
OR(RP-DL, RP-HR)	0.971	0.827	0.922	0.868	0.770
Max(DL, HR)	**0.973**	0.864	0.890	**0.873**	**0.780**
Sum(0.4DL, 0.6HR)	0.973	0.861	0.890	0.872	0.777
RegLog_comb	0.973	0.865	0.888	0.873	0.779
MLP_conf_256	0.970	0.823	0.921	0.866	0.767
MLP_comb	0.970	0.815	**0.931**	0.866	0.767
MLP_completo	0.972	0.866	0.875	0.868	0.771
RandFor_comb	0.973	0.861	0.884	0.869	0.774

Tabla 5.3: Resultados en el conjunto de test de Saeedi para la segmentación del TE (Mejor en negrita).

	Accuracy	Precision	Recall	Dice Coef.	Jaccard Idx.
Villota et al. [11]	0.964	0.855	0.727	0.781	0.647
OR(V3-DL, V3-HR)	0.967	0.820	0.815	0.812	0.690
OR(RP-DL, RP-HR)	0.967	0.812	0.828	0.815	0.694
Max(DL, HR)	0.967	0.857	0.763	0.802	0.676
Sum(0.4DL, 0.6HR)	0.967	0.851	0.768	0.803	0.677
RegLog_comb	0.967	0.855	0.764	0.803	0.677
MLP_conf_256	0.966	0.805	0.828	0.811	0.690
MLP_comb	0.966	0.794	**0.849**	0.816	0.696
MLP_completo	**0.968**	0.820	0.827	**0.819**	**0.700**
RandFor_comb	0.967	**0.859**	0.758	0.801	0.673

Tabla 5.4: Resultados en el conjunto de Quirón para la segmentación del TE (Mejor en negrita).

	Accuracy	Precision	Recall	Dice Coef.	Jaccard Idx.
Villota et al. [11]	0.983	0.885	0.873	0.872	0.795
OR(V3-DL, V3-HR)	0.983	0.873	0.920	0.889	0.808
OR(RP-DL, RP-HR)	0.983	0.868	**0.924**	0.889	0.807
Max(DL, HR)	0.983	0.902	0.894	0.890	0.810
Sum(0.4DL, 0.6HR)	0.983	0.900	0.889	0.886	0.806
RegLog_comb	0.984	**0.911**	0.883	0.887	0.811
MLP_conf_256	0.983	0.892	0.895	0.881	0.806
MLP_comb	0.984	0.899	0.892	0.883	0.809
MLP_completo	**0.984**	0.907	0.890	**0.890**	**0.813**
RandFor_comb	0.983	0.902	0.880	0.880	0.805

Tabla 5.5: Resultados en el conjunto de test de Saeedi para la segmentación de la MCI (Mejor en negrita).

	Accuracy	Precision	Recall	Dice Coef.	Jaccard Idx.
Villota et al. [11]	0.981	**0.828**	0.721	0.732	0.649
OR(V3-DL, V3-HR)	0.982	0.765	0.759	0.745	0.662
OR(RP-DL, RP-HR)	**0.982**	0.801	**0.763**	**0.747**	**0.662**
Max(DL, HR)	0.980	0.752	0.695	0.701	0.627
Sum(0.4DL, 0.6HR)	0.981	0.789	0.709	0.713	0.638
RegLog_comb	0.980	0.756	0.692	0.700	0.626
MLP_conf_256	0.982	0.808	0.755	0.746	0.662
MLP_comb	0.982	0.812	0.750	0.743	0.661
MLP_completo	0.982	0.815	0.743	0.736	0.656
RandFor_comb	0.980	0.757	0.690	0.698	0.623

Tabla 5.6: Resultados en el conjunto de Quirón para la segmentación de la MCI (Mejor en negrita).

A continuación, se describen los resultados según el tipo de *ensemble* utilizado.

5.1.1. Métodos no supervisados que operan con máscaras.

En primer lugar, respecto a los métodos de *ensemble* no supervisados que operan directamente sobre las máscaras de segmentación, el modelo con mejor rendimiento es OR(V3-DL, V3-HR), que aplica la operación OR entre las máscaras de DeepLab y HRNet tras un preprocesado con el algoritmo V3. Este modelo mejora ligeramente la *accuracy* respecto al mejor modelo de Villota et al., pero destaca especialmente por el aumento en *recall*: de 0.83 a 0.89 en ZP, de 0.83 a 0.90 en TE y de 0.87 a 0.92 en MCI, para el conjunto de Saeedi. Se obtienen mejoras similares en el conjunto de Quirón, sugiriendo que el modelo generaliza bien a conjuntos de datos con distinta procedencia. La mejora en la *accuracy* se debe principalmente al preprocesado, que

elimina imperfecciones como el suavizado de bordes y agrupaciones pequeñas de píxeles erróneos. Por otro lado, el aumento del *recall* se debe a la operación OR, que permite incluir píxeles predichos por cualquiera de los dos modelos base. Los otros métodos de combinación de máscaras no han obtenido tan buenos resultados; el operador AND por ser más restrictivo a la hora de incluir píxeles y el voto mayoritario por incorporar la información de modelos de peor rendimiento como U-Net o RDU-Net.

5.1.2. Métodos no supervisados que operan con probabilidades.

En cuanto a los métodos no supervisados que operan sobre las probabilidades de salida de los modelos base, destacan los modelos: OR(RP-DL, RP-HR), Max(DL, HR) y Sum(0.4DL, 0.6HR). El que mejor rendimiento demuestra es OR(RP-DL, RP-HR), que es similar al anterior OR(V3-DL, V3-HR), pero en este caso, en vez de aplicar el preprocesamiento V3, se aplica un reescalado de las probabilidades al tamaño original de la imagen. Este modelo obtiene un rendimiento muy similar al anterior, pero con un aumento del *recall* aún mayor, especialmente en TE (de 0.83 a 0.92) y MCI (de 0.87 a 0.92), lo que lo convierte en uno de los modelos con mayor *recall* en general.

Respecto a las otras estrategias de combinación de probabilidades, el modelo Max(DL, HR), que selecciona el valor máximo entre las predicciones de DeepLab y HRNet, es uno de los mejores en cuanto a *accuracy* en el conjunto de Saeedi. Sin embargo, su rendimiento es peor en el conjunto de Quirón (especialmente para la MCI, ver Tabla 5.6), sugiriendo que no generaliza tan bien como otros métodos. Por otro lado, el mejor modelo de suma ponderada, Sum(0.4DL, 0.6HR), que combina las probabilidades de DeepLab y HRNet con pesos 0.4 y 0.6, obtiene un rendimiento muy similar al de Max(DL, HR), aunque ligeramente peor en el conjunto de Saeedi.

5.1.3. Métodos supervisados.

En cuanto a los métodos que utilizan aprendizaje supervisado, las variantes del MLP son las que mejores resultados obtienen. Entre ellas, destaca el modelo MLP_completo, que es un perceptrón con dos capas ocultas de 128 y 64 neuronas, entrenado con el datastet completo de entrenamiento, con una fuerte regularización $L2$ y *early stopping* para evitar el sobreajuste y mejorar la generalización. Este modelo presenta un rendimiento balanceado entre *accuracy* y *recall*, con buena generalización al conjunto de Quirón. Es el modelo con mayor *accuracy* e índice de *Jaccard* [26] en la segmentación de la ZP y TE en Quirón (Tablas 5.2 y 5.4), y en la de MCI en Saeedi (Tabla 5.5).

Las otras variantes, MLP_conf_256 (256 y 128 neuronas) y MLP_comb (128

y 64 neuronas) han sido entrenadas con el conjunto de píxeles conflictivos y el conjunto combinado, respectivamente. Ambos modelos ofrecen alternativas con un *recall* superior, aunque a costa de una ligera pérdida de *accuracy*.

Respecto a los otros dos enfoques de aprendizaje supervisado, Regresión Logística (RegLog_comb) obtiene resultados similares al modelo Max(DL, HR), con buena *accuracy* pero menor *recall*, y mal resultado en la segmentación de la MCI en QUIRÓN. Por su parte, *Random Forest* (RandFor_comb) muestra un rendimiento similar a la regresión, aunque ligeramente peor.

5.1.4. Mejores modelos.

En conclusión, podemos considerar que el mejor modelo global implementado es el MLP_completo, ya que demuestra un equilibrio sólido entre *accuracy* y *recall*, obteniendo resultados consistentes en ambos conjuntos de evaluación. No obstante, si se desea priorizar el *recall*, que es especialmente crítico en estructuras como la MCI, el modelo OR(RP-DL, RP-HR) es la mejor opción, al obtener el mayor *recall* en la segmentación de la MCI en ambos conjuntos.

A continuación, la Figura 5.1 muestra un ejemplo donde se comparan visualmente las máscaras generadas por los mejores modelos respecto al estudio base de Villota et al. [11] y al *ground truth*.

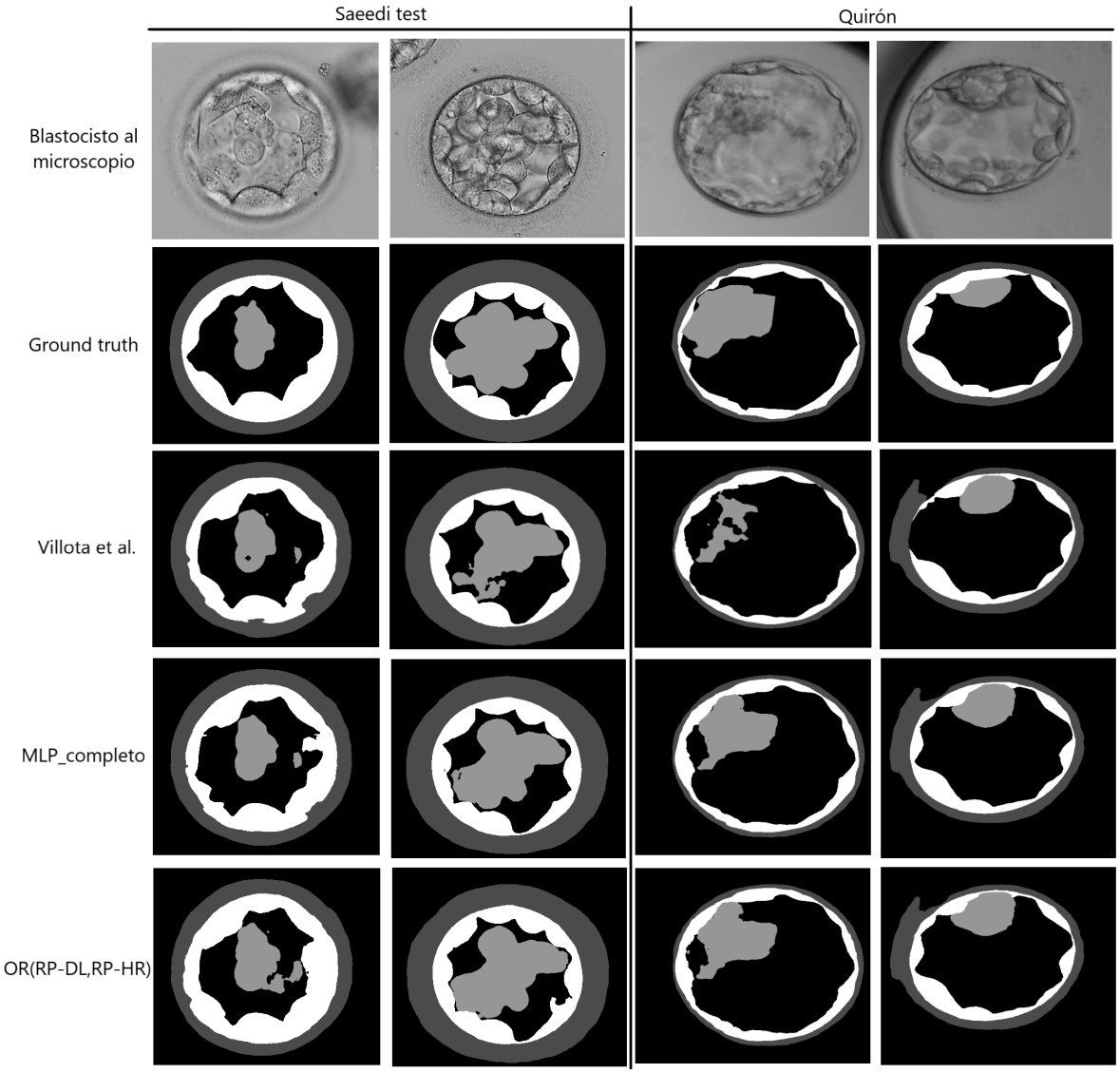

Figura 5.1: Comparación de resultados de segmentación para imágenes de ejemplo de ambos datasets de evaluación (Saeedi y Quirón).

Como podemos observar en la Figura 5.1, los mejores modelos implementados consigen mejorar notablemente los resultados del estudio base de Villota et al., especialmente en cuanto al *recall* de estructuras como el TE o la MCI, logrando una segmentación más cercana al *ground truth*, con contornos mejor definidos y menos omisiones.

El modelo MLP_completo consigue aumentar el *recall* sin comprometer la *accuracy*, logrando un buen equilibrio global. Por otro lado, el modelo OR(RP-DL, RP-HR), aunque obtiene el *recall* más alto, tiende a ser más permisivo, lo que en ciertos casos puede dar lugar a más falsos positivos, como en el caso de la primera muestra de la Figura 5.1.

5.2. Comparación con el estado del arte

A continuación, se comparan los resultados de los tres mejores modelos implementados con el resto de estudios del estado del arte.

Como se puede observar en la Tabla 5.7, el modelo MLP_completo obtiene la segmentación de la ZP más balanceada y precisa, superando al resto de modelos del estado del arte en cuanto a *accuracy*, *recall*, coeficiente de *Dice* y el índice de *Jaccard*. Por otro lado, el modelo MLP_comb obtiene el mayor *recall* para esta estructura, aunque es menos preciso.

	Accuracy	Precision	Recall	Dice Coef.	Jaccard Idx.
Kheradmand et al. [14]	0.92	0.80	0.81	-	0.64
Rad et al. [19]	0.95	0.79	0.91	-	0.74
Farias et al. [22]	0.94	0.85	0.69	0.75	-
Villota et al. [11]	0.97	**0.92**	0.84	0.87	0.78
OR(RP-DL, RP-HR)	0.97	0.90	0.89	0.89	0.81
MLP_comb	0.97	0.88	**0.91**	0.89	0.81
MLP_completo	**0.97**	0.90	0.89	**0.89**	**0.81**

Tabla 5.7: Comparación de resultados con el estado del arte para la segmentación de la ZP (Mejor en negrita).

En cuanto a la segmentación del TE (Tabla 5.8), ninguno de los modelos implementados logra superar las métricas obtenidas por Harun et al. [21], aunque el modelo MLP_comb iguala su *recall*. El modelo MLP_completo se posiciona como segundo mejor en términos generales, tras el modelo de Harun.

	Accuracy	Precision	Recall	Dice Coef.	Jaccard Idx.
Singh et al. [13]	0.87	0.71	0.83	0.77	0.62
Kheradmand et al. [14]	0.90	0.69	0.80	0.74	0.59
Saeedi et al. [16]	0.86	0.69	0.89	0.77	-
Harun et al. [21]	**0.98**	**0.92**	**0.93**	**0.92**	**0.85**
Farias et al. [22]	0.93	0.80	0.59	0.67	-
Villota et al. [11]	0.97	0.88	0.84	0.85	0.75
OR(RP-DL, RP-HR)	0.97	0.83	0.92	0.87	0.77
MLP_comb	0.97	0.82	**0.93**	0.87	0.77
MLP_completo	0.97	0.87	0.87	0.87	0.77

Tabla 5.8: Comparación de resultados con el estado del arte para la segmentación del TE (Mejor en negrita).

En la segmentación de la MCI (Tabla 5.9), las mejores métricas vuelven a ser las de Harun et al. [21]. Le sigue Rad et al. [20], cuyos resultados son comparables a los obtenidos por el modelo OR(RP-DL, RP-HR) desarrollado en este trabajo.

	Accuracy	Precision	Recall	Dice Coef.	Jaccard Idx.
Kheradmand et al. [14]	0.93	0.76	0.56	0.64	0.48
Kheradmand et al. [15]	0.96	-	-	0.87	0.77
Saeedi et al. [16]	0.91	0.77	0.84	0.79	-
Saeedi et al. (DLRS) [16]	0.93	0.84	0.78	0.83	-
Rad et al. [17]	-	0.79	0.87	0.83	0.70
Rad et al. [20]	0.98	0.89	0.92	0.90	0.82
Harun et al. [21]	**0.99**	**0.95**	**0.94**	**0.94**	**0.89**
Farias et al. [22]	0.96	0.87	0.62	0.67	-
Villota et al. [11]	0.98	0.88	0.87	0.87	0.79
OR(RP-DL, RP-HR)	0.98	0.87	0.92	0.89	0.81
MLP_comb	0.98	0.90	0.89	0.88	0.81
MLP_completo	0.98	0.91	0.89	0.89	0.81

Tabla 5.9: Comparación de resultados con el estado del arte para la segmentación de la MCI (Mejor en negrita).

Capítulo 6

Conclusiones

En este trabajo se han desarrollado diferentes técnicas de *ensemble* de modelos de *deep learning* que logran mejorar respecto al estado del arte la segmentación de las principales estructuras del blastocisto: la ZP, el TE y la MCI. Se han implementado estrategias tanto no supervisadas como supervisadas, para combinar las predicciones de cuatro modelos base (DeepLab, HRNet, U-Net y RDU-Net), entrenados y publicados por Villota et al. [11].

De todos los enfoques desarrollados, los modelos basados en aprendizaje supervisado han obtenido el mejor rendimiento, especialmente el modelo MLP_completo, un MLP con dos capas ocultas y fuerte regularización. Este modelo ha aprendido a combinar de manera óptima las probabilidades de salida de los modelos base, mejorando considerablemente la precisión global de la segmentación. Tiene sentido que el mejor modelo sea un MLP, ya que este tipo de red neuronal (con al menos una capa oculta) es un aproximador universal de funciones continuas [47], capaz de replicar funciones simples como operaciones lógicas (OR, AND, XOR, etc.), además de capturar patrones más complejos en los datos que los métodos no supervisados no pueden modelar.

Sin embargo, si el objetivo clínico es maximizar el *recall* de la segmentación para no omitir regiones relevantes (como en el caso de la MCI, que es crucial para el desarrollo del embrión), podría utilizarse el modelo OR(RP-DL, RP-HR). Este modelo combina las máscaras de DeepLab y HRNet tras reescalar sus probabilidades de salida, logrando el mayor *recall* en la segmentación de la MCI en ambos datasets evaluados.

Es importante destacar que el hecho de que se hayan empleado conjuntos de datos de evaluación no usados para el entrenamiento confirma que ambos modelos no están sobreajustados y arrojan unos resultados confiables.

En comparación con el estado del arte, las métricas obtenidas superan notablemente a las del estudio base de Villota et al. [11] en la segmentación de las tres estructuras (ZP, TE y MCI), y solo el trabajo de Harun et al. [21] presenta métricas superiores para el TE y la MCI. Cabe destacar que dicho estudio no publica su código, y aunque

su metodología ha sido replicada en trabajos posteriores, como el de Villota et al., no se han logrado reproducir sus resultados. Por lo tanto, este trabajo aporta, hasta la fecha, los mejores resultados con código abierto y reproducibles, en el ámbito de la segmentación automática de blastocistos. El código, las métricas y los mejores modelos entrenados se pueden encontrar en el siguiente repositorio público:

`https://github.com/816410unizar/Blastocyst-Seg-Ensemble`.

De esta manera, se facilita la reproducibilidad y extensión de los resultados por parte de la comunidad investigadora.

En cuanto a las limitaciones de este trabajo, la principal es que se basa en los modelos publicados por Villota et al., que fueron entrenados en un conjunto de datos relativamente pequeño (249 imágenes de blastocistos). Esto puede reducir la capacidad de generalización de los modelos a otros datasets más diversos. Además, si se hubiera contado con más capacidades computacionales, habría sido posible entrenar modelos supervisados más complejos y optimizar aún más los hiperparámetros.

Como trabajo futuro, sería interesante aplicar las estrategias de *ensemble* desarrolladas en conjuntos de datos más grandes y variados, así como reentrenar los modelos supervisados en ellos. Otra línea de trabajo podría centrarse en integrar las técnicas desarrolladas en software clínico de apoyo directo a los embriólogos. Adicionalmente, se podría desarrollar una metodología capaz de predecir el grado de calidad de cada estructura del blastocisto (presente en las anotaciones de Saeedi) a partir de características morfológicas cuantitativas extraídas de la segmentación. Esto contribuiría a determinar de forma objetiva cuál es el embrión con mayor potencial de implantación en la fecundación in vitro.

Capítulo 7

Bibliografía

[1] MM Reigstad and R Storeng. Development of in vitro fertilization, a very important part of human reproductive medicine, in the last 40 years. *Int J Womens Health Wellness*, 5(89):2474–1353, 2019.

[2] World Health Organization et al. 1 in 6 people globally affected by infertility, 2023.

[3] Reproducción Asistida ORG. Estructura del blastocisto. `https://www.reproduccionasistida.org/procedimiento-del-dgp/estructura-blastocisto/`, 2024. [Accedido: (Junio 2025)].

[4] ESHRE Add-Ons Working Group, K Lundin, JG Bentzen, G Bozdag, T Ebner, J Harper, N Le Clef, A Moffett, S Norcross, NP Polyzos, et al. Good practice recommendations on add-ons in reproductive medicine. *Human Reproduction*, 38(11):2062–2104, 2023.

[5] Hayden Anthony Homer. Preimplantation genetic testing for aneuploidy (pgt-a): The biology, the technology and the clinical outcomes. *Australian and New Zealand Journal of Obstetrics and Gynaecology*, 59(2):317–324, 2019.

[6] Basak Balaban and David K Gardner. Morphological assessment of blastocyst stage embryos: types of grading systems and their reported outcomes. In *Human gametes and preimplantation embryos: assessment and diagnosis*, pages 31–43. Springer, 2013.

[7] N Zaninovic, R Berrios, RN Clarke, R Bodine, Z Ye, and LL Veeck. Blastocyst expansion, inner cell mass (icm) formation, and trophectoderm (tm) quality: is one more important for implantation? *Fertility and Sterility*, 76(3):S8, 2001.

[8] Alison Richardson, Sophie Brearley, Saran Ahitan, Sarah Chamberlain, Tracey Davey, Lyndsey Zujovic, James Hopkisson, Bruce Campbell, and Nick

Raine-Fenning. A clinically useful simplified blastocyst grading system. *Reproductive biomedicine online*, 31(4):523–530, 2015.

[9] J. Murel and E. Kavlakoglu. What is ensemble learning? `https://www.ibm.com/think/topics/ensemble-learning#:~:text=Ensemble%20learning%20is%20a%20machine,1`, 2024. [Accedido: (Junio 2025)].

[10] Yanming Guo, Yu Liu, Theodoros Georgiou, and Michael S Lew. A review of semantic segmentation using deep neural networks. *International journal of multimedia information retrieval*, 7:87–93, 2018.

[11] María Villota, Jacobo Ayensa-Jiménez, Manuel Doblaré, and Jónathan Heras. Image processing and deep learning methods for the semantic segmentation of blastocyst structures. In *Conference of the Spanish Association for Artificial Intelligence*, pages 213–222. Springer, 2024.

[12] María Villota. Blastocyst-seg: Image processing and deep learning methods for the semantic segmentation of blastocyst structures. `https://github.com/mavillot/Blastocyst-Seg`, 2024. [Accedido: (Junio 2025)].

[13] Amarjot Singh, Jason Au, Parvaneh Saeedi, and Jon Havelock. Automatic segmentation of trophectoderm in microscopic images of human blastocysts. *IEEE Transactions on Biomedical Engineering*, 62(1):382–393, 2014.

[14] Shakiba Kheradmand, Parvaneh Saeedi, and Ivan Bajic. Human blastocyst segmentation using neural network. In *2016 IEEE Canadian conference on electrical and computer engineering (CCECE)*, pages 1–4. IEEE, 2016.

[15] Shakiba Kheradmand, Amarjot Singh, Parvaneh Saeedi, Jason Au, and Jon Havelock. Inner cell mass segmentation in human hmc embryo images using fully convolutional network. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 1752–1756. IEEE, 2017.

[16] Parvaneh Saeedi, Dianna Yee, Jason Au, and Jon Havelock. Automatic identification of human blastocyst components via texture. *IEEE Transactions on Biomedical Engineering*, 64(12):2968–2978, 2017.

[17] Reza Moradi Rad, Parvaneh Saeedi, Jason Au, and Jon Havelock. Coarse-to-fine texture analysis for inner cell mass identification in human blastocyst microscopic images. In *2017 Seventh International Conference on Image Processing Theory, Tools and Applications (IPTA)*, pages 1–5. IEEE, 2017.

[18] ScienceDirect Topics. Gabor filter - an overview — sciencedirect topics. https://www.sciencedirect.com/topics/engineering/gabor-filter, 2024. [Accedido: (Junio 2025)].

[19] Reza Moradi Rad, Parvaneh Saeedi, Jason Au, and Jon Havelock. Human blastocyst's zona pellucida segmentation via boosting ensemble of complementary learning. *Informatics in Medicine Unlocked*, 13:112–121, 2018.

[20] Reza Moradi Rad, Parvaneh Saeedi, Jason Au, and Jon Havelock. Multi-resolutional ensemble of stacked dilated u-net for inner cell mass segmentation in human embryonic images. In *2018 25th IEEE international conference on image processing (ICIP)*, pages 3518–3522. IEEE, 2018.

[21] Md Yousuf Harun, Thomas Huang, and Aaron T Ohta. Inner cell mass and trophectoderm segmentation in human blastocyst images using deep neural network. In *2019 IEEE 13th International Conference on Nano/Molecular Medicine & Engineering (NANOMED)*, pages 214–219. IEEE, 2019.

[22] Adolfo Flores-Saiffe Farias, Alejandro Chavez-Badiola, Gerardo Mendizabal-Ruiz, Roberto Valencia-Murillo, Andrew Drakeley, Jacques Cohen, and Elizabeth Cardenas-Esparza. Automated identification of blastocyst regions at different development stages. *Scientific Reports*, 13(1):15, 2023.

[23] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.

[24] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, et al. Deep high-resolution representation learning for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 43(10):3349–3364, 2020.

[25] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015.

[26] Nghi Huynh. Understanding evaluation metrics in medical image segmentation. https://medium.com/@nghihuynh_37300/

understanding-evaluation-metrics-in-medical-image-segmentation-d289a373a3f, 2023. [Accedido: (Junio 2025)].

[27] Keiron O'shea and Ryan Nash. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, 2015.

[28] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015.

[29] Reza Azad, Ehsan Khodapanah Aghdam, Amelie Rauland, Yiwei Jia, Atlas Haddadi Avval, Afshin Bozorgpour, Sanaz Karimijafarbigloo, Joseph Paul Cohen, Ehsan Adeli, and Dorit Merhof. Medical image segmentation review: The success of u-net. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[30] GeeksforGeeks. U-net architecture explained. https://www.geeksforgeeks.org/u-net-architecture-explained/, 2025. [Accedido: (Junio 2025)].

[31] Saba Hesaraki. Deeplab. https://medium.com/@saba99/deeplab-095f387f891f, 2023. [Accedido: (Junio 2025)].

[32] Lokesh Borawar and Ravinder Kaur. Resnet: Solving vanishing gradient in deep networks. In *Proceedings of International Conference on Recent Trends in Computing: ICRTC 2022*, pages 235–247. Springer, 2023.

[33] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[34] GeeksforGeeks. Softmax activation function in neural networks. https://www.geeksforgeeks.org/the-role-of-softmax-in-neural-networks-detailed-explanation-and-applications/, 2024. [Accedido: (Junio 2025)].

[35] GeeksforGeeks. What is bilinear interpolation? https://www.geeksforgeeks.org/what-is-bilinear-interpolation/, 2024. [Accedido: (Junio 2025)].

[36] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[37] Vijay Kanade. What is logistic regression? equation, assumptions, types, and best practices. https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-logistic-regression/, 2022. [Accedido: (Junio 2025)].

[38] GeeksforGeeks. Sigmoid function. https://www.geeksforgeeks.org/derivative-of-the-sigmoid-function/, 2025. [Accedido: (Junio 2025)].

[39] Alejandro Ito Aramendia. L1 and l2 regularization (part 2): A complete guide. https://medium.com/@alejandro.itoaramendia/l1-and-l2-regularization-part-2-a-complete-guide-0b16b4ab79ce, 2024. [Accedido: (Junio 2025)].

[40] Alejandro Ito Aramendia. L1 and l2 regularization (part 1): A complete guide. https://medium.com/@alejandro.itoaramendia/l1-and-l2-regularization-part-1-a-complete-guide-51cf45bb4ade, 2024. [Accedido: (Junio 2025)].

[41] Arnav R. Scikit-learn solvers explained. https://medium.com/@arnavr/scikit-learn-solvers-explained-780a17bc322d, 2022. [Accedido: (Junio 2025)].

[42] Sejal Jaiswal. Perceptrones multicapa en el aprendizaje automático: Guía completa. https://www.datacamp.com/es/tutorial/multilayer-perceptrons-in-machine-learning, 2024. [Accedido: (Junio 2025)].

[43] Bharath Krishnamurthy. An introduction to the relu activation function. https://builtin.com/machine-learning/relu-activation-function, 2024. [Accedido: (Junio 2025)].

[44] GeeksforGeeks. What is adam optimizer? https://www.geeksforgeeks.org/deep-learning/adam-optimizer/, 2025. [Accedido: (Junio 2025)].

[45] Samy Baladram. Random forest, explained: A visual guide with code examples. https://medium.com/data-science/random-forest-explained-a-visual-guide-with-code-examples-9f736a6e1b3c, 2024. [Accedido: (Junio 2025)].

[46] Ankit Chauhan. Random forest classifier and its hyperparameters. https://medium.com/analytics-vidhya/random-forest-classifier-and-its-hyperparameters-8467bec755f6, 2021. [Accedido: (Junio 2025)].

[47] GeeksforGeeks. Universal approximation theorem for neural networks. https://www.geeksforgeeks.org/deep-learning/universal-approximation-theorem-for-neural-networks/, 2024. [Accedido: (Junio 2025)].

Lista de Figuras

1.1. Estructuras del blastocisto (Fuente: [3]). 2
1.2. Imágen de blastocisto al microscopio (izq.) y su segmentación por un modelo de *Deep learning* (der.). 3
1.3. Cronograma con la planificación y ejecución temporal del TFG. 5

2.1. Imagen del dataset Saeedi junto a sus anotaciones de segmentación. . 10
2.2. Imagen del dataset Quirón junto a sus anotaciones de segmentación. . 11
2.3. Arquitectura U-Net (Fuente: [30]). 12
2.4. Esquema general del flujo de la arquitectura DeepLab (Fuente: [31]). . . 12
2.5. Ejemplo de entrada para los modelos, incluyendo la imagen de blastocisto al microscopio y su máscara de segmentación (*ground truth*). 13
2.6. Proceso de construcción de las máscaras de segmentación a partir de las salidas de probabilidad de los modelos. 14

3.1. Ejemplos de predicciones con ruido de los distintos modelos utilizados. 16
3.2. Ejemplo de suavizado de bordes en una predicción. 16
3.3. Ejemplo de bordes de estructuras tras aplicar algoritmo V1. 17
3.4. Ejemplo de bordes de estructuras tras aplicar algoritmo V2. 17
3.5. Ejemplo de bordes de estructuras tras aplicar algoritmo V3. 18
3.6. Comparación de predicciones antes y después de aplicar el algoritmo V3 18

4.1. Esquema del modelo Random Forest (Fuente: [46]). 29

5.1. Comparación de resultados de segmentación para imágenes de ejemplo de ambos datasets de evaluación (Saeedi y Quirón). 35

Lista de Tablas

2.1. Resultados del estado del arte para la segmentación de la ZP (Mejor en negrita). 9
2.2. Resultados del estado del arte para la segmentación del TE (Mejor en negrita). 9
2.3. Resultados del estado del arte para la segmentación de la MCI (Mejor en negrita). 9

5.1. Resultados en el conjunto de test de Saeedi para la segmentación de la ZP (Mejor en negrita). 30
5.2. Resultados en el conjunto de Quirón para la segmentación de la ZP (Mejor en negrita). 31
5.3. Resultados en el conjunto de test de Saeedi para la segmentación del TE (Mejor en negrita). 31
5.4. Resultados en el conjunto de Quirón para la segmentación del TE (Mejor en negrita). 31
5.5. Resultados en el conjunto de test de Saeedi para la segmentación de la MCI (Mejor en negrita). 32
5.6. Resultados en el conjunto de Quirón para la segmentación de la MCI (Mejor en negrita). 32
5.7. Comparación de resultados con el estado del arte para la segmentación de la ZP (Mejor en negrita). 36
5.8. Comparación de resultados con el estado del arte para la segmentación del TE (Mejor en negrita). 36
5.9. Comparación de resultados con el estado del arte para la segmentación de la MCI (Mejor en negrita). 37