UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIERÍA INDUSTRIAL VALENCIA

TRABAJO FIN DE GRADO EN INGENIERÍA BIOMÉDICA

DISEÑO Y DESARROLLO DE UN SISTEMA AUTOMÁTICO DE SEGMENTACIÓN MORFOLÓGICA DE BLASTOCISTOS PARA PREDECIR EL POTENCIAL DE IMPLANTACIÓN EMBRIONARIA

AUTOR: ALEJANDRO JOSÉ VERGARA RICHART

TUTORA: VALERY NARANJO ORNEDO

COTUTORA: ELENA PAYÁ BOSCH

Curso Académico: 2020-21

Resumen

La baja natalidad y el aumento de la esperanza de vida se presentan como un problema a largo plazo en nuestra sociedad. La evolución de la pirámide poblacional y su proyección futura se prevén como un problema social y económico a nivel nacional y global. La baja natalidad está relacionada, en gran medida, con el incremento de la infertilidad en la población debido a diferentes factores sociales, ambientales y económicos. En este contexto, la evaluación de embriones *in vitro* es una línea de trabajo que ha mejorado considerablemente en los últimos 20 años dejando atrás la evaluación clásica. La aparición de los incubadores con sistemas *time-lapse* han permitido la monitorización continua de los embriones aportando información morfocinética y morfológica convencional, además de posibilitar la introducción de la inteligencia artificial en el campo de la embriología.

El objetivo de este TFG es el desarrollo de modelos de *deep learning* capaces de segmentar las estructuras morfológicas que forma el blastocisto en sus últimas etapas de desarrollo embrionario. Esta información se va a emplear para evaluar la calidad de un embrión y predecir su potencial de implantación. Para ello, se hará uso de una base de datos aportada por el Instituto Valenciano de Infertilidad que consta de imágenes a 112 horas posinseminación y la información clínica asociada. Metodológicamente, se quiere hacer uso de redes neuronales convolucionales que reconstruyan la imagen en forma de máscara de etiquetas y que predigan la calidad del embrión. El trabajo incluye la segmentación manual de las imágenes, el desarrollo de modelos basados en algoritmos de *deep learning* y la validación de los mismos.

Palabras clave: *deep learning*, fecundacion *in vitro*, segmentación semántica, clasificación, red neuronal convolucional, inteligencia artificial

Resum

La baixa natalitat i l'augment de l'esperança de vida es presenten com un problema a llarg termini en la nostra societat. L'evolució de la piràmide poblacional i la seua projecció futura es preveu com un problema social i econòmic a nivell nacional i global. La baixa natalitat està relacionada, en gran manera, amb l'increment de la infertilitat en la població degut a diferents factors socials, ambientals i econòmics. En aquest context, l'avaluació d'embrions *in vitro* és una línia de treball que ha millorat considerablement en els últims 20 anys deixant arrere l'avaluació clàssica. L'aparició dels incubadors amb sistemes *time-lapse* han permés la monitorització contínua dels embrions aportant informació morfocinètica i morfològica convencional, a més de possibilitar la introducció de la intel·ligència artificial en el camp de l'embriologia.

L'objectiu d'aquest TFG és el desenvolupament de models de *deep learning* capaços de segmentar les estructures morfològiques que forma el blastocisto en les seues últimes etapes de desenvolupament embrionari. Esta informació es va a emprar per a avaluar la qualitat d'un embrió i predir el seu potencial d'implantació. Per a això, es farà ús d'una base de dades aportada per l'Institut Valencià d'Infertilitat que consta d'imatges a 112 hores postinseminació i la informació clínica associada. Metodològicament, es vol fer ús de xarxes neuronals convolucionals que reconstruïsquen la imatge en forma d'una màscara d'etiquetes i que prediguen la qualitat de l'embrió. El treball inclourà la segmentació manual de les imatges, el desenvolupament de models basats en algoritmes de *deep learning* i la seua validació.

Paraules clau: *deep learning*, fecundació *in vitro*, segmentació semàntica, classificació, xarxa neuronal convolucional, intel·ligència artificial.

Abstract

Low birth rates and increased life expectancy are a long-term problem in our society. The evolution of the population pyramid and its future projection foresee a social and economic problem at the national and global level. The low birth rate is largely related to the increase in infertility in the population due to different social, environmental and economic factors. In this context, *in vitro* embryo evaluation is a line of work that has improved considerably in the last 20 years, leaving behind the classical evaluation. The appearance of incubators with time-lapse systems has allowed the continuous monitoring of embryos, providing conventional morphokinetic and morphological information, as well as allowing the introduction of artificial intelligence in the field of embryology.

The objective of this thesis is the development of deep learning models capable of segmenting the morphological structures that the blastocyst forms in its last stages of embryonic development. This information will be used to evaluate the quality of an embryo and predict its implantation potential. For this, a database provided by the Instituto Valenciano de Infertilidad will be used, consisting of images at 112 hours post-insemination and the associated clinical information. Methodologically, we want to make use of convolutional neural networks that reconstruct the image as a labeled mask and that predict the quality of the embryo. The work will include the manual segmentation of the images, the development of models based on deep learning algorithms and their corresponding validation.

Keywords: deep learning, *in vitro* fertilization, semantic segmentation, classification, convolutional neural network, artificial intelligence.

Índice general

Parte I

Memoria

Capítulo 1

Introducción

1.1 Motivación y descripción del problema

1.1.1 Infertilidad

La infertilidad es una enfermedad del sistema reproductivo definida como el fracaso en lograr un embarazo después de 12 meses o más de relaciones sexuales regulares sin empleo de anticonceptivos debido a la deficiencia en la capacidad reproductiva como individuo o como pareja [1, 2]. La condición de enfermedad provoca que la infertilidad genere discapacidad en quien la padece [1]. De hecho, es considerada la quinta discapacidad con más prevalencia en mujeres menores de 60 años [2]. Además, afecta al 15 % de la población en edad reproductiva de los países occidentales y sigue una tendencia creciente [3, 4].

Aunque el varón es causante de entre el 25 % y el 35 % de los casos, la edad avanzada de las mujeres con deseo reproductivo puede considerarse como la principal causa actual de incremento de la infertilidad [3]. Su periodo de máxima fecundidad se sitúa entre los 20 y 30 años, donde tienen entre un 25 % y un 30 % de probabilidades de quedarse embarazadas en cada ciclo. A partir de ahí, comienza un declive fisiológico que se acentúa a partir de los 35 años y que empeora notablemente a los 40, donde las posibilidades de embarazo se reducen al 5 % [5].

Es importante mencionar la diferencia entre esterilidad e infertilidad. La esterilidad hace referencia a la imposibilidad de concebir la gestación durante un año de relaciones sexuales regulares. Mientras que la infertilidad resulta de la incapacidad de generar gestaciones que den lugar a un feto viable [3]. No obstante, esta diferenciación se hace generalmente en el medio hispanohablante, pero no en el anglosajón, por lo que los datos de prevalencia ofrecidos en los párrafos superiores se refieren a problemas de fertilidad en general, englobando así ambos términos. Del mismo modo, a lo largo del trabajo se hará uso de ambos conceptos indistintamente para hacer referencia a problemas de fertilidad.

1.1.2 Embriología y formación del blastocisto

La embriología es la rama de la biología que estudia el desarrollo y formación del embrión y del feto [6]. Debido a la temática del presente trabajo, es conveniente ser conocedor de los procesos biológicos que derivan en la formación del blastocisto a modo de contexto que permita entender de forma clara los procesos de fecundación *in vitro* (FIV) y la necesidad de incluir la tecnología en este campo.

La formación del blastocisto se inicia con la ovulación femenina, donde se libera el óvulo a la trompa de Falopio. Mientras tanto, los espermatozoides son trasportados por el tracto reproductor de la mujer hasta alcanzar el óvulo en el tercio superior de la trompa de Falopio. Estos se adhieren a la zona pelúcida, que es una membrana de glucoproteínas que envuelve el óvulo, y la atraviesan para entrar en contacto directo con este. Tras la fusión del material genético de ambos progenitores puede decirse que el proceso de fecundación ha terminado y el óvulo fecundado recibirá el nombre de cigoto [7].

El cigoto comienza su división inmediatamente después de originarse y entra en un proceso lento que dura varios días. Durante los dos primeros, el ritmo de desarrollo es de una división diaria. Tras la etapa de dos células, la división es asíncrona y las células reciben el nombre de blastómeras. Una vez generado un cúmulo de 16 blastómeras el conjunto pasa a denominarse mórula, que es un estadio intermedio entre el cigoto y el blastocisto.

Durante el proceso de formación de la mórula, cuando se alcanza el estadio de 8 células, los embriones entran en una fase de compactación, donde las blastómeras más externas se unen estrechamente entre sí constituyendo una especie de epitelio que rodea a las blastómeras internas. Entre ambos grupos de blastómeras se va constituyendo un espacio lleno de líquido denominado blastocele. Se habrá entrado en el proceso de cavitación, que generalmente comienza entorno al cuarto día posterior a la fecundación. De esta manera, se constituyen dos tipos de células: las pertenecientes a la capa epitelial externa, que forman el trofoectodermo, y las restantes aglutinadas en el interior, llamadas masa celular interna (MCI). A partir de este momento y durante el periodo de entre los 4 y 7 primeros días [8], el embrión pasa a denominarse blastocisto. Durante esta etapa, el blastocisto atraviesa diferentes fases según su grado de expansión. La primera de ellas es la de blastocisto temprano, donde el blastocele empieza a visualizarse. Seguidamente, la cavidad del blastocele termina de definirse, lo que indica el paso a la fase de cavitación, donde ya es posible distinguir el trofoectodermo y la masa celular interna. A continuación, se entra en la fase de blastocisto expandido, donde este aumenta de tamaño y se reduce el grosor de la

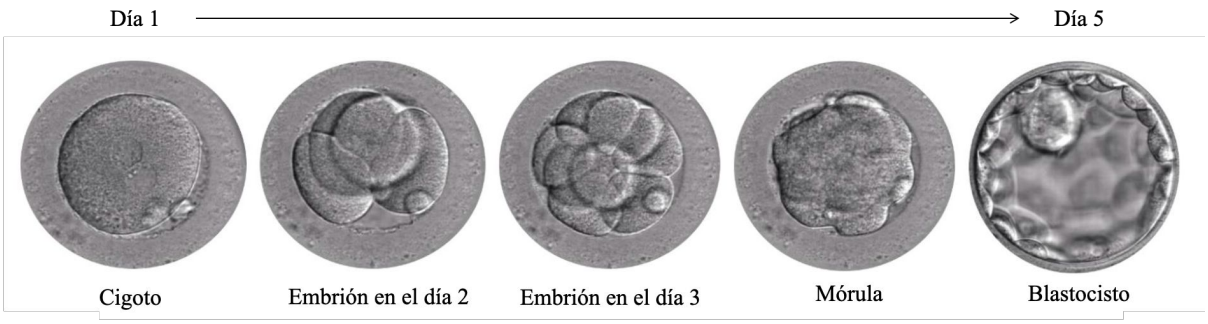

Figura 1.1: Evolución y etapas del embrión en los primeros cinco días tras la formación del cigoto.

zona pelúcida. Finalmente, se suceden dos etapas en las que el blastocisto eclosiona y consigue abandonar la zona pelúcida [7]. Todo el proceso de formación del blastocisto a partir del cigoto queda ilustrado en la Figura 1.1.

1.1.3 Técnicas de reproducción asistida

Cualquier alteración en el ciclo de formación de los blastocistos da lugar a problemas de fertilidad. Como solución a estos se pueden emplear Técnicas de Reproducción Asistida (TRA), que son un conjunto amplio de procedimientos caracterizados por la actuación directa sobre los gametos con el fin de favorecer la fecundación y la transferencia o depósito de embriones en la cavidad uterina [3].

Sin embargo, cabe destacar que, como ocurría con el embarazo natural, las tasas de éxito en estos tratamientos también se ven reducidas con un aumento en la edad de la paciente o la donante. Entre 2014 y 2016, en el Servicio Nacional de Salud de Reino Unido la probabilidad de nacimiento sano tras un tratamiento de fecundación *in vitro* fue del 29 % para mujeres menores de 35 años, del 18 % entre los 35 y 39 años y de un 6 % entre los 40 y 44 años [9].

Dentro de las TRA, un tratamiento ampliamente utilizado es la fecundación *in vitro* (FIV). En esencia, consiste en la adquisición y preparación de los gametos para realizar la fecundación del ovocito (óvulo) por el espermatozoide en condiciones de cultivo *in vitro* con una implantación posterior en la cavidad uterina [8]. Para aumentar las posibilidades de éxito se acumulan el mayor número de ovocitos posible para poder tener una mayor cantidad de embriones entre los cuales seleccionar. Por ello, se somete a la mujer a un tratamiento de Estimulación Ovárica Controlada con el objetivo de conseguir un desarrollo folicular múltiple. Tras este, se somete a la paciente a una punción ovárica para extraer los ovocitos, que serán depositados en un incubador hasta el momento de la inseminación. Estos necesitarán unas condiciones muy concretas para poder sobrevivir, todas ellas recogidas en *Revised guidelines for good practice in IVF laboratories* [10].

Para la inseminación se disponen de dos técnicas: la Fecundación *in vitro* (FIV) y la Microinyección del espermatozoide dentro del ovocito (ICSI, de sus siglas en inglés). La primera se aplica cuando hay problemas de infertilidad femenina, los parámetros seminales son normales o moderadamente alterados y tras previos intentos por Inseminación Artificial (IA). Mientras que la ICSI se recomienda cuando los problemas están asociados a la infertilidad masculina severa, con parámetros seminales muy alterados y tras fallos de IA y FIV, entre otros [8]. (*Véase Figura 1.2*)

En la FIV se pone en contacto un grupo de ovocitos con los espermatozoides, de manera que serán estos los que fecunden el óvulo por sí solos. Por el contrario, en la ICSI, se extraen espermatozoides

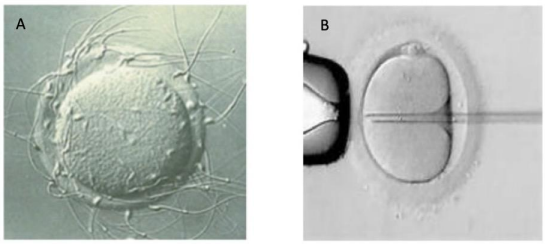

Figura 1.2: Inseminación mediante FIV (A) y mediante ICSI (B).

de una muestra de semen o mediante biopsia testicular y luego se selecciona aquel con mejor morfología y movilidad y se microinyecta en el ovocito [4].

Los ovocitos correctamente fecundados según el criterio ASEBIR [11] se disponen en un medio de cultivo en el interior del incubador. A partir de aquí empieza la etapa de evaluación de los embriones.

Actualmente se siguen los criterios de clasificación morfológica de ASEBIR (Asociación para el Estudio de la Biología de la Reproducción) cuyas recomendaciones vienen recogidas en el manual *Criterios ASEBIR de Valoración Morfológica de Oocitos,Embriones Tempranos y Blastocistos Humanos* [11]. En este se desarrollan todos los aspectos morfológicos que ayudan a identificar los embriones con mayor capacidad de desarrollo y mayor potencial implantatorio.

De esta manera, en los días 2 y 3 se evalúa el número de células, la velocidad de división, el porcentaje y tipo de fragmentación celular, el tamaño de las blastómeras y su forma, el estado de los núcleos, posibles anomalías, la zona pelúcida y el grado de compactación temprana [8]. A partir de la inspección visual del biólogo se establece una categoría para cada embrión. Del mismo modo, se realiza una evaluación morfológica en el intervalo del paso de mórula a blastocisto (del cuarto al sexto día). En el día 4 se evalúa la división celular (se espera que tenga más de 8 células), la adhesión y compactación celular y la fragmentación y vacuolización. Posteriormente, en los días 5 y 6 se evalúa el estado del blastocisto atendiendo a las características de sus distintas regiones, delimitadas en la Figura 1.3. Estas se presentan a continuación.

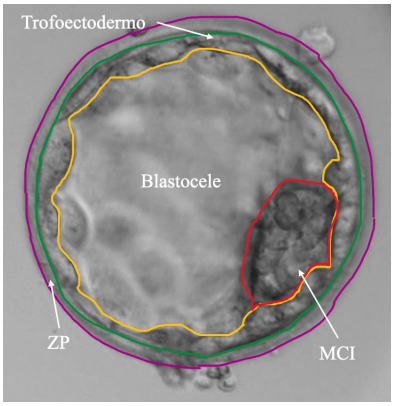

Figura 1.3: Presentación de la imagen de un blastocisto con las distintas regiones delimitadas.

- **Blastocele**. Una mayor expansión del blastocele está relacionada con tasas de implantación más elevadas.
- **Zona pelúcida (ZP)**. La expansión del blastocele y el aumento del embrión hacen que el grosor de esta estructura sea mínimo.
- **Trofoectodermo**. Esta conformado por una monocapa de células compactadas. En función del número, forma y grado de cohesión de estas el blastocisto puede tener mayores tasas de implantación; siendo preferible un epitelio homogéneo con células elípticas.
- **Masa Celular Interna (MCI)**. Los embriones de más calidad están asociados con masas de forma ovalada, con células compactas y que tengan un tamaño de entre 1900 y 3800 micrómetros cuadrados.

- **Fragmentación y vacuolización**. Asociadas a una mala tasa de implantabilidad [8].

Asimismo, es importante destacar que la evaluación no debe realizarse en una única inspección, sino que debe darse tras múltiples observaciones realizadas a lo largo del desarrollo. Y, es más, la valoración en un momento puntual debe tener en cuenta el estado del embrión en otras evaluaciones anteriores.

Por esto, una tecnología que ha marcado un antes y un después en lo que respecta a la monitorización y desarrollo de embriones es el *Time-lapse Monitoring System* (TMS). Este permite una continua evaluación del desarrollo del embrión sin necesidad de extraerlo del incubador en ningún momento y sin exponerlo a las condiciones del entorno, con lo que asegura condiciones estables para su desarrollo, similares a las de un proceso in vivo. Para ello, toma imágenes de forma automática tras periodos de tiempo de entre 5 y 20 minutos que, además, pueden ser integradas y reproducidas como un vídeo, ofreciendo así mucha más información que la generada mediante métodos convencionales [8]. Asimismo, cabe destacar que no existen problemas de seguridad con esta tecnología según [12] y que la probabilidad de embarazo aumenta cuando se hace uso de embriones monitorizados a través de TMS [13].

Debido a que el éxito en la implantación depende tanto de un desarrollo óptimo del endometrio como de la calidad del embrión [14] y que las técnicas actuales de selección de embriones implantables son poco confiables, se realizan trasferencias de múltiples embriones (METs, del inglés). Con esto se pretende aumentar las probabilidades de éxito. Sin embargo, este procedimiento supone un aumento en la tasa de embarazos múltiples, los cuales llevan asociado un riesgo significativo tanto para la madre como para el feto [15, 16].

Por este motivo, la transferencia de un solo embrión está ganando importancia y, con ello, la necesidad de mejorar en las técnicas de selección embrionaria. La tecnología TMS ha supuesto un impacto en la reproducción asistida permitiendo realizar un seguimiento exhaustivo de los diferentes estadios de forma no invasiva, contabilizar los tiempos de los ciclos celulares y, además, extraer grandes cantidades de datos relacionados con la morfología de los embriones que están siendo utilizados como entrada para algoritmos de selección de embriones a modo de posibles predictores de la implantación.

1.2 Inteligencia artificial y aprendizaje profundo

Aunque la Inteligencia Artificial (IA) es un concepto ampliamente utilizado, la realidad es que no tiene una definición clara y concisa. John McCarthy, uno de los padres de la IA, se refiere a esta como "la ciencia y la ingeniería de crear programas de computación inteligentes" [17]. No obstante, esta definición incluye el concepto de inteligencia, cuya variabilidad en su significado es el principal problema a la hora de encontrar una definición adecuada para la IA. De ahí que surjan diferentes definiciones como "la capacidad de un sistema de realizar tareas comúnmente asociadas con procesos intelectuales característicos de los humanos" [18] o "la combinación de las ciencias de la computación con robustos conjuntos de datos para permitir la resolución de problemas" [19]. En general, la mayoría de estas definiciones se forman a partir de la convergencia de conceptos como tecnología, computación, procesos inteligentes, razonamiento o resolución de tareas. Por ello, una definición apropiada podría ser "la capacidad de los sistemas computacionales

de realizar tareas mediante procesos asociados a agentes inteligentes, como el razonamiento, el aprendizaje y la toma de decisiones".

Como rama del campo de la IA surge el *Machine Learning* (ML), que es una disciplina asociada a la implementación de sistemas computacionales que puedan aprender de forma autónoma [20] y que están estrechamente relacionados con el reconocimiento de patrones, la ciencia de datos y la minería de datos [21]. Y, a su vez, es en esta disciplina donde se encuentra uno de los conceptos fundamentales para la comprensión del presente trabajo: el aprendizaje profundo (DL, del inglés - *deep learning*).

El DL abarca el conjunto de algoritmos de aprendizaje automático que intenta imitar las capacidades y el mecanismo de aprendizaje del cerebro humano [22]. Se caracteriza por hacer uso de redes neuronales artificiales. Estas se forman a partir de un conjunto de nodos que se organizan en capas y que se interconectan entre sí. El término de red neuronal deriva de su similitud con el sistema nervioso humano. El cuerpo de la neurona correspondería al nodo, mientras que las entradas a esta equivaldrían a las dendritas. Finalmente, la información generada en el nodo pasaría al siguiente de la misma manera que las neuronas trasmiten los impulsos a través de los axones [22]. Estas similitudes se presentan en la Figura 1.4.

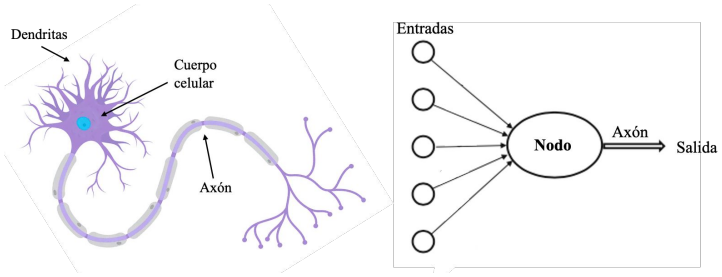

Figura 1.4: Neurona del sistema nervioso frente a un nodo de una red neuronal artificial [22].

En esencia, las redes neuronales artificiales se pueden ver como una serie de algoritmos que se basan en la continua optimización de un conjunto de parámetros que relacionan las entradas y las salidas de la red [22]. Para obtener unos parámetros óptimos es necesario trabajar con grandes cantidades de datos. Generalmente, al enseñarle más datos a la red durante el proceso de entrenamiento se consigue una mayor precisión en la predicción o clasificación, tal y como se ve en la Figura 1.5. Sin embargo, esto supone problemas asociados al tiempo de entrenamiento o las necesidades computacionales para llevarlo a cabo.

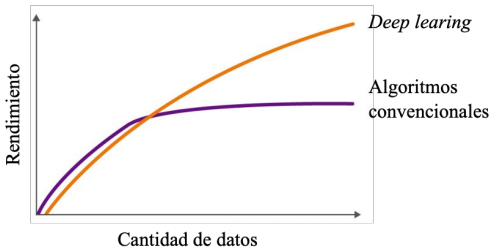

Figura 1.5: Relación entre el rendimiento y la cantidad de datos para algoritmos de *deep learning* y algoritmos convencionales. Imagen obtenida modificando [23].

Y es justamente en los datos donde radica la principal diferencia entre ML y DL. Mientras que *Machine Learning* necesita datos estructurados y con características definidas, las redes neuronales artificiales son capaces de trabajar con datos no procesados, pues son capaces de extraer de forma automática las características más determinantes del set de datos [24].

Hoy en día, estos algoritmos son cada vez más transversales a los diferentes ámbitos. Las redes neuronales artificiales actualmente se emplean tanto en la automoción como en el servicio al cliente o en servicios financieros [25, 26]. Pero, sin duda, un área donde esta tecnología tiene un gran impacto es la de salud, donde el aprendizaje profundo puede usarse para desarrollar sistemas de ayuda a la decisión, dar diagnósticos a partir del análisis de imágenes, descubrir nuevos fármacos y contribuir al desarrollo de una medicina de precisión [27].

Particularmente, en este Trabajo de Fin de Grado (TFG) se ha hecho uso de técnicas de aprendizaje profundo para la segmentación y clasificación automática de imágenes de blastocistos con el objetivo de predecir su potencial de implantación. En concreto, se ha empleado la denominada segmentación semántica, que se basa en la división de la imagen en regiones, cada una de las cuales está asociada a una categoría. Más concretamente, se puede describir como una clasificación a nivel de píxel, pues la asignación a una clase se hace en función de la categoría a la que pertenezca el objeto en el que se encuentra dicho píxel. Esta tarea trasforma la representación de la imagen, permitiendo extraer información significativa de una forma más sencilla. (*Véase la Figura 1.6*).

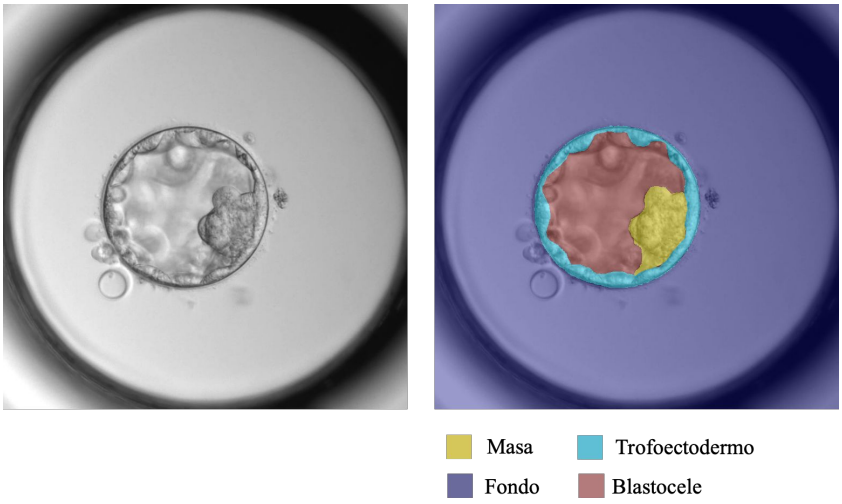

Figura 1.6: Segmentación semántica sobre una imagen embrionaria.

Por otro lado, se ha realizado una clasificación de las imágenes atendiendo a la calidad del embrión, determinada a partir del criterio ASEBIR [11]. Esta calidad depende exclusivamente de la morfología del mismo y esta relacionada con el potencial de implantación del embrión en el útero, de tal manera que una calidad mayor asegura un mayor potencial implantatorio. No obstante, la tasa de éxito en la implantación depende además de otros factores externos como el estado del útero en el momento de la trasferencia de los embriones o la edad de la gestante. Debido a la ausencia de esta información, únicamente se clasifica en cuestiones de calidad embrionaria. Para ello, se ha trabajado tanto con las imágenes originales como con las imágenes segmentadas.

1.3 Proyecto *Deep Vision*

El presente trabajo se enmarca en un proyecto de mayor envergadura llamado *Deep Vision*. Este tiene como objetivo el desarrollo de algoritmos basados en *deep learning* para la mejora de la selección embrionaria utilizando tecnologías de visión artificial. En él participan IVI-RMA Global Valencia, grupo especialista en técnicas de reproducción asistida, y el grupo de investigación CVBLab, perteneciente al Instituto de Investigación e Innovación en Bioingeniería (I3B) de la Universitat Politècnica de València.

1.4 Objetivos

El objetivo principal del presente TFG es desarrollar un sistema automático que permita predecir la calidad del embrión a partir de una única imagen del mismo, de manera que se contribuya a la mejora de las tasas de implantación de embriones en las técnicas de reproducción asistida.

Enmarcados en este propósito general, quedan una serie de objetivos secundarios, que son:

- Realizar una revisión bibliográfica para comprender las dificultades que se presentan a la hora de realizar un tratamiento de reproducción asistida con éxito y obtener conocimiento acerca del estado de las técnicas empleadas para segmentación y clasificación de embriones.
- Construcción de una base de datos de imágenes segmentadas manualmente que sirvan como referencia para el entrenamiento y la evaluación de la red neuronal de segmentación.
- Desarrollar, entrenar y evaluar una red neuronal de segmentación que permita asignar a cada píxel de la imagen original una etiqueta, separando así la imagen en regiones.
- Desarrollar, entrenar y evaluar una red neuronal de clasificación que sea capaz de predecir la calidad del embrión a partir de las imágenes originales y las imágenes segmentadas.
- Analizar los resultados y compararlos con los diferentes estudios descritos en el estado del arte.
- Identificar posibles problemas y plantear propuestas de mejora.

Capítulo 2

Marco Teórico

2.1 Introducción a las redes neuronales

2.1.1 Perceptrón

En 1958, Rossenblat [28] propone el modelo más básico de red neuronal artificial, denominado Perceptrón. Este está compuesto únicamente por una neurona, tal y como se muestra en la Figura 2.1, de manera que las entradas a dicha neurona son trasformadas en una salida empleando una variación generalizada de una función lineal [22]. Con ello, se tiene una capa de entrada y una capa de salida. Realmente, solo se contabilizan aquellas capas en las que se realicen cálculos, por lo que se dice que el Perceptrón es una arquitectura de una sola capa.

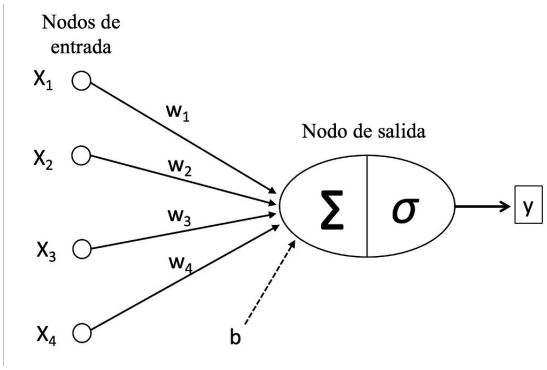

Figura 2.1: Arquitectura del Perceptrón.

En cada neurona se producen dos operaciones. En primer lugar, se computa la suma de las entradas $X = [x_1, \ldots, x_d]$ ponderada por los pesos $W = [w_1, \ldots, w_d]$, siendo d el número de entradas.

$$z = b + \sum_i w_i \cdot x_i \tag{2.1}$$

El término b representa el sesgo (en inglés *bias*) que afecta al conjunto de entradas y que predispone la neurona a una salida determinada. Es una manera de introducir la parte invariante de los datos de entrada.

En segundo lugar, se aplica una función de activación σ, que trasforma el conjunto ponderado de entradas devolviendo una salida. Estas funciones pueden ser lineales, como es el caso del Perceptrón, pero generalmente introducir no linealidad en los datos genera mejores resultados, especialmente para redes profundas. Las diferentes funciones de activación se explican en la sección 2.1.3.

2.1.2 Perceptrón Multicapa

La inclusión de nuevas capas entre las de entrada y salida supone la aparición del Perceptrón Multicapa (MLP, del inglés - *Multilayer Perceptron*). Este se compone de tres tipos de capas. La capa de entrada, que es la encargada de recibir el vector de entrada que será procesado por la red. La capa de salida, que realiza la tarea de clasificación o regresión final. Y las capas intermedias, denominadas capas ocultas, las cuales llevan el peso de la computación y se ocupan de establecer la estructura *feed forward* por la que se genera un flujo de información entre la entrada y la salida en el que se extraen las características relevantes que permiten llevar a cabo la tarea de clasificación de forma satisfactoria [29]. Asimismo, cada neurona de una capa está conectada con todas las neuronas de la siguiente capa, por lo que se puede hablar de capas *fully connected*.

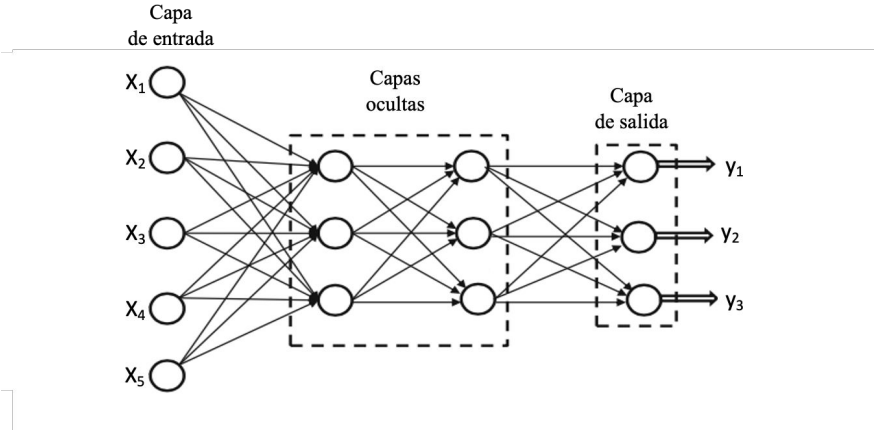

Figura 2.2: Arquitectura general del Perceptrón Multicapa.

2.1.3 Funciones de activación

Tras la suma ponderada de los pesos z se aplica una función de activación σ. El principal objetivo es definir una salida para la neurona trasformando z. En las redes neuronales profundas se emplean funciones no lineales, lo que permite hacer predicciones de mayor complejidad a partir de representaciones y ajustes más precisos a los datos de entrada. Aunque hay más, a continuación se presentan las funciones de activación más habituales.

2.1.3.1 Función escalón

Es una función de activación clásica empleada antiguamente en el Perceptrón. Se empleaba para generar salidas binarias, pero su linealidad y no diferenciabilidad suponen un problema en el proceso de optimización durante el entrenamiento. Por ello, ha sido sustituida por otras funciones no lineales. Se muestra en la Figura 2.3

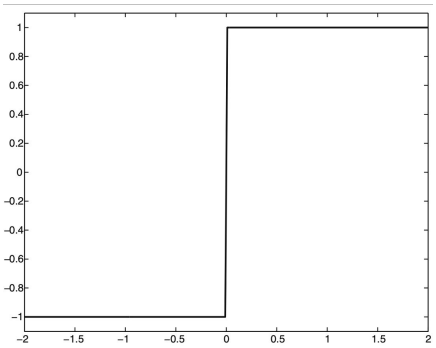

Figura 2.3: Representación de la función escalón.

2.1.3.2 Función sigmoide

Permite generar salidas entre 0 y 1, tal y como se muestra en la Figura 2.4. Asimismo, es derivable en su totalidad y permite incorporar no linealidad en la red neuronal.

$$s(z) = \frac{1}{1 + e^{-z}} \tag{2.2}$$

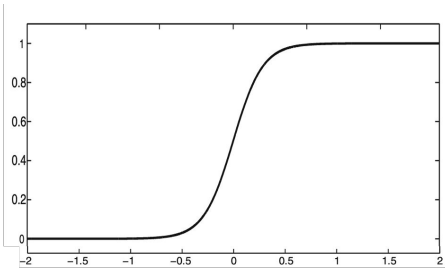

Figura 2.4: Representación de la función sigmoide.

2.1.3.3 Función tangente hiperbólica

Tiene una forma similar a la función sigmoide, pero esta está reescalada horizontalmente y trasladada verticalmente a $[-1, 1]$, tal y como se ve en la Figura 2.5. Es preferible a la sigmoide cuando se desea obtener valores positivos y negativos. Asimismo, que este centrada y que vaya a generar un mayor gradiente facilitan el entrenamiento. Permite incorporar no linealidad en la red neuronal.

$$tanh(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}} \tag{2.3}$$

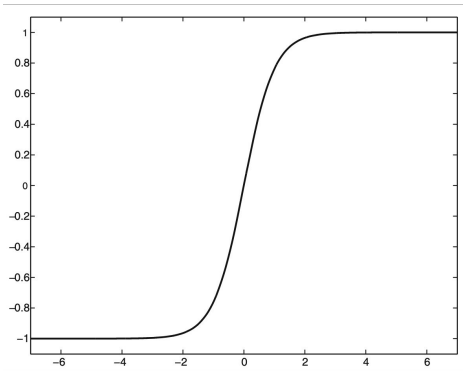

Figura 2.5: Representación de la función tangente hiperbólica.

2.1.3.4 Función ReLU

La *Rectified Linear Unit* (ReLU) trasforma los valores negativos a 0 y mantiene el mismo valor para los positivos.

$$R(z) = max(0, z) \tag{2.4}$$

Esta función ha reemplazado a la mayoría de las funciones no lineales por su facilidad a la hora de entrenar redes de múltiples capas y porque evita que todas las neuronas se activen, pues únicamente se activan aquellas con valores mayores que cero. Es ampliamente utilizada en redes convolucionales.

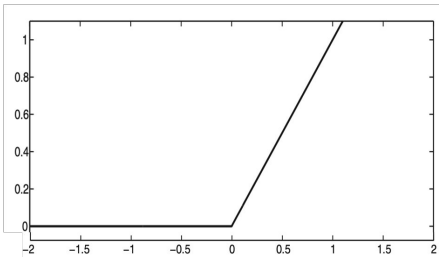

Figura 2.6: Representación de la función ReLU.

2.1.3.5 Función softmax

Es una función exponencial normalizada que permite pasar de un vector de valores reales arbitrarios a otro de valores reales entre 0 y 1. Por ello, suele utilizarse como capa final de los clasificadores, pues ofrece para la entrada la probabilidad que tiene esta de pertenecer a cada una de las clases.

$$softmax(z) = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{-z_k}} \text{para } j = 1, ..., K \tag{2.5}$$

2.1.4 Función de pérdidas

Con el objetivo de mejorar la capacidad de predicción de la red, se trabaja con una función de pérdidas o coste. Esta indica cómo de aproximada es la salida al valor objetivo que se desea. Ha de ser diferenciable para poder trabajar con el gradiente de dicha función y usarlo para calcular de qué manera se han de actualizar los pesos de la red. En DL, esta actualización se lleva a cabo de manera automática gracias al algoritmo de retropropagación, el cual utiliza programación dinámica para obtener cuánto debe variar el valor de cada peso para minimizar la función de pérdidas. Es fundamental la elección de la función de pérdidas en el sentido de que debe ser sensible a la aplicación de la red. Para problemas de regresión donde la salida sea numérica puede ser conveniente emplear el error cuadrático medio definido en la ecuación 2.6 como función de pérdidas.

$$\sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i - \hat{y_i})^2 \tag{2.6}$$

Sin embargo, en casos donde las predicciones corresponden a probabilidades de pertenencia a determinadas clases, generalmente se usan dos funciones:

- Regresión logística. Se emplea para clasificación binaria, es decir, cuando se trabaja con tan solo dos clases. Siendo $\hat{y}$ la predicción e y la etiqueta real entre $[-1, 1]$, cuando mayor sea la correlación entre ambas, menor será el resultado de la exponencial y se obtendrán menores pérdidas.

$$L = log(1 + exp(-y \cdot \hat{y})) \tag{2.7}$$

- Entropía cruzada. Se emplea para clasificación en múltiples clases. Se trabaja con la regresión logística multinomial, de manera que las pérdidas son menores conforme mayor sea la probabilidad de pertenencia a una determinada clase k. Las probabilidades van desde $\hat{y_1}, \ldots, \hat{y_k}$ y se obtienen las pérdidas asociadas a la probabilidad $\hat{y_r}$, siendo r la clase determinada en el *ground-truth* para dicha muestra.

$$L = -\sum_{i=1}^{k} y_r log(\hat{y_i}) \tag{2.8}$$

2.1.5 Algoritmo forward-backward propagation

En las redes neuronales de una sola capa, el proceso de optimización de los pesos es sencillo pues puede ser realizado de forma directa. Sin embargo, para MLP la función de pérdidas se convierte en una complicada función compuesta por los pesos de todas las capas anteriores. Por esto, en 1986 fueron Rumelhart, Hinton y Williams los que propusieron el algoritmo de retropropagación del error [30].

Dicho algoritmo calcula el gradiente de la función de pérdidas con respecto a cada uno de los pesos w_i de la red. Aunque el numero de parámetros es muy elevado, el cálculo se puede llevar a cabo con programación dinámica. Se diferencian dos fases:

1. *Forward propagation.* Se le pasan un conjunto de muestras a la red. Estas van pasando a través de las distintas capas y sobre ellas se aplican dos operaciones en cada neurona. La primera es la suma ponderada mostrada en la ecuación 2.1 y la segunda es la función de activación $H = \sigma(z)$. Finalmente, la red genera una salida y se computa la función de pérdidas L que genera un valor de error entre la salida y la referencia.

2. *Backward propagation.* En este caso el objetivo es ajustar los pesos de la red en base al error obtenido en la etapa *forward* para minimizar la función de pérdidas y, con ello, el error cometido en la predicción. Esto lo lleva a cabo un optimizador. Uno ampliamente utilizado es el *Stochastic Gradient Descend* (SGD), que realiza la siguiente actualización de pesos para cada iteración t:

$$\theta^{t+1} = \theta^t - \alpha \frac{\delta L(X, \theta^t)}{\delta \theta} \tag{2.9}$$

   siendo θ el conjunto de parámetros de la red donde se engloban tanto los pesos W como los sesgos b y siendo α el ratio de aprendizaje. De ahora en adelante, para simplificar la explicación, únicamente se hablará de parámetros o pesos de la red, puesto que el proceso de actualización de los sesgos b sigue la misma metodología.

   Como puede observarse en la ecuación 2.9, se realiza la derivada parcial de L con respecto a los parámetros, es decir, se calcula el gradiente de la función objetivo con respecto a los diferentes pesos haciendo uso de la regla de la cadena y realizando este proceso desde la capa de salida hasta la de entrada. Así, a partir de los gradientes locales se obtiene cuánto se ha de modificar cada peso para reducir el error en la capa de salida [22]. Por tanto, de lo que se trata es de ir en la dirección opuesta a la dirección del gradiente (ya que este por definición apunta en la dirección de máximo crecimiento) para encontrar un mínimo global que asegure predicciones correctas a la salida.

2.2 Redes neuronales convolucionales

Las redes neuronales convolucionales (CNN, del ingés - *Convolutional Neural Network*) están diseñadas para trabajar con estructuras con fuertes dependencias espaciales, como ocurre con las imágenes. La característica definitoria de estas redes es el empleo de una operación matemática denominada convolución. En términos de procesamiento de imagen, esta se basa en realizar el producto punto a punto entre una matriz de pesos llamada *kernel* y una matriz de valores de entrada [22]. En otras palabras, es una operación que devuelve un valor proporcional a la información compartida por ambas imágenes.

El origen de estas redes se encuentra de nuevo en la neurociencia, surgiendo gracias al estudio de Hubel y Wiesel en el córtex visual de los gatos [31]. Este estudio dio a conocer que hay áreas específicas del córtex visual que son sensibles a ciertas regiones del campo de visión, es decir, que áreas concretas del cerebro se activan frente a determinados estímulos visuales. Y lo más importante, las neuronas que se excitan dependen de la forma y orientación de los elementos del campo visual. Fue a partir de este hecho de donde surgió la idea de realizar una extracción de características jerárquica, lo que significa que la red extraiga características más generales como

bordes o formas simples en las primeras capas y figuras más complejas y detalles en las capas profundas.

Partiendo de esta idea, aparecieron varias redes convolucionales. La pionera fue LeNet-5 [32], que tenía tan solo siete capas de profundidad. Realmente, la diferencia entre esta y las redes neuronales que la sucedieron hasta la actualidad se centra únicamente en el empleo de mayor número de capas y en la inclusión de la función de activación ReLU, así como una mejoría natural en la capacidad de computo. Tomando como referencia el *challenge* anual ImageNet (conococido como *ImageNet Large Scale Visual Recognition Challenge*), este tipo de arquitectura fue capaz de reducir la tasa de error de clasificación top-5 de más del 25 % al 4 % entre los años 2011 y 2015 [33].

Las CNN funcionan con la estrategia *feed-forward*, pero tienen la característica de que las operaciones en las capas están espacialmente organizadas con conexiones entre estas. Es fundamental que las relaciones espaciales se hereden de una capa a otra para saber a qué región de la capa anterior corresponde un determinado valor de la capa actual.

Por cada capa, se tendrá una estructura tridimensional conformada por la altura, la anchura y la profundidad. Para la imagen de entrada, cada punto espacial determinado por la altura y la anchura es un píxel y la profundidad sirve para codificar su color. Para las capas ocultas, la profundidad hace referencia al número de canales de cada capa, es decir, al número de mapas de activación que genera dicha capa. Estos mapas de activación contienen información de varios tipos de formas extraídas de regiones locales de la imagen de entrada [22].

Entre los tipos de capas mayormente empleados para este tipo de arquitecturas destacan: capas convolucionales, capas de *pooling*, capas *dropout*, capas densas y capas *batch normalization*.

2.2.1 Capa convolucional

La capa convolucional es la encargada de llevar a cabo la operación de convolución. Esta se define como el producto punto a punto en todas las posiciones válidas (i, j, k) correspondientes a las posiciones de altura, anchura y profundidad de cada filtro p de tamaño $F_q \times F_q \times d_q$ y el conjunto de mapas de activación definido como $H^{(q)} = [h^{(q)}_{ijk}]$ y cuyo tamaño es $M_q \times N_q \times d_q$. Así pues, la convolución de la capa q a la capa $(q+1)$ se define tal que

$$h^{(q+1)}_{ijp} = \sum_{r=1}^{Fq}\sum_{s=1}^{Fq}\sum_{k=1}^{dq} w^{(p,q)}_{rsk} h^{(q)}_{i+r-1,j+s-1,k} \qquad \forall i \in 1, ..., L_q - F_q + 1 \tag{2.10}$$

$$\forall j \in 1, ..., B_q - F_q + 1 \tag{2.11}$$

$$\forall p \in 1, ..., d_{q+1} \tag{2.12}$$

siendo $W^{(p,q)} = [w^{(p,q)}_{ijk}]$ la matriz de parámetros para cada filtro p. Esta matriz recibe el nombre de filtro o *kernel* y es una matriz tridimensional que contiene los pesos necesarios para la extracción de características y son estos los que se optimizan con el algoritmo de retropropagación. De manera más visual, en la Figura 2.7 se representa una operación de convolución.

En la operación de convolución es necesario definir ciertos parámetros: el tamaño del *kernel*, el *stride* y el *padding*. Repecto al primero, suelen utilizarse filtros 3×3, 5×5 o 7×7. Además,

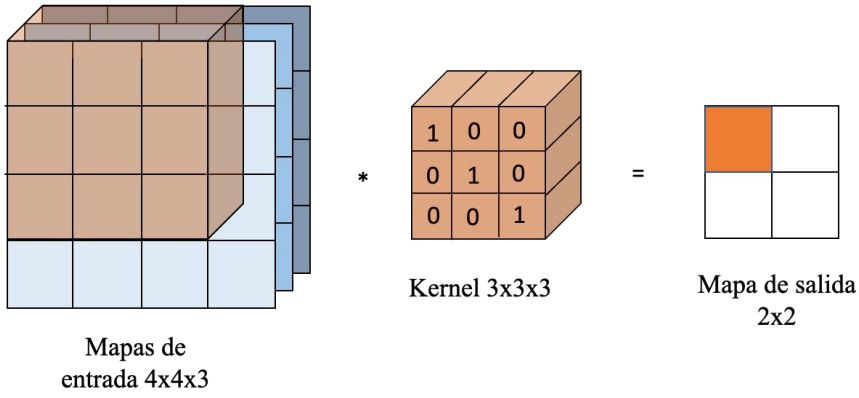

Figura 2.7: Representación de la operación de convolución.

es fundamental que la profundidad de estos filtros siempre sea igual a la de la capa donde se aplica, para poder situarlo en cada posición del mapa de activación de la capa anterior y que se superponga con este para realizar el producto punto a punto. En otras palabras, la profundidad dependerá del número de mapas de activación de la cada capa. Por otra parte, el *stride* indica cuánto se debe desplazar el filtro después de cada aplicación, ya que no es necesario realizar la convolución en cada posición de la imagen. Finalmente, el *padding* tiene que ver con una característica de la convolución por la cual se reduce el tamaño del mapa de activación resultante en comparación con el de la capa anterior después de aplicar el filtro. Esto deriva de la imposibilidad de realizar la convolución en los bordes, pues habría valores del *kernel* que no estarían solapados con valores del mapa de activación. Para solucionarlo puede usarse el *padding*, que se basa en la adición de nuevos píxeles alrededor de los bordes, de manera que cuando se aplica la convolución se puede trabajar sobre los bordes reales del mapa de activación para generar uno nuevo del mismo tamaño. (*Véase la Figura 2.8).*

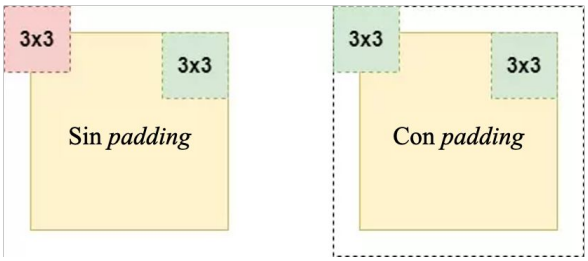

Figura 2.8: Representación del *padding*. Imagen modificada de [34]

.

Teniendo esto en cuenta, supongamos que la salida de una capa oculta es de $28 \times 28 \times 16$ y que se trabaja con un kernel $3 \times 3 \times 16$. Como la profundidad del mapa de activación y del filtro es la misma se puede realizar la operación. Entonces, por cada posición en el mapa de activación, se realiza una operación de convolución y el resultado se almacena en una posición de un nuevo mapa de activación. Finalmente, cuando el kernel ha recorrido todas las posiciones, se habrá obtenido un único mapa de características bidimensional de tamaño $28 \times 28 \times 1$ (suponiendo el uso de *padding* para mantener el tamaño original). Lo que sucede es que, por cada posición, se ha obtenido un único valor, es decir, una única característica. Por tanto, la profundidad en las capas ocultas no surge como consecuencia de la operación de convolución en sí, sino de trabajar

con varios *kernels* distintos sobre la salida de la capa anterior. De esta manera, suponiendo que se quiera mantener el tamaño $28 \times 28 \times 16$, se tendrá que hacer uso de 16 filtros, cada uno de los cuales será distinto y extraerá características o patrones diferentes a los otros. Y es debido a esta especialización de cada kernel en una característica particular de la imagen de donde deriva la necesidad de trabajar con una gran cantidad de filtros a lo largo de la red, pues una imagen encierra una gran variedad de formas y patrones.

2.2.2 Capa pooling

Esta capa trabaja sobre pequeñas regiones cuadradas del mapa de activación y reduce el tamaño espacial de los mismos, reduciendo el número de parámetros y las necesidades de computación. El más común es el denominado *Max Pooling*, que devuelve el valor máximo de cada una de las regiones. En este caso la operación se realiza de forma independiente en cada mapa de activación, por lo que no cambia la profundidad de la capa. Tiene como objetivo condensar la información y permitir a los *kernels* de las capas convolucionales sucesivas capturar regiones más grandes de la imagen, con lo que pueden especializarse en patrones más complejos.

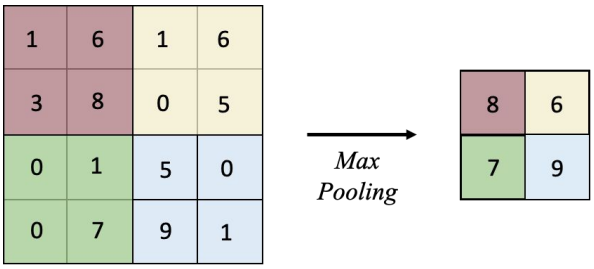

Figura 2.9: Representación de la capa *Max Pooling*.

2.2.3 Capa dropout

Para entender la necesidad de incorporar estas capas en las redes neuronales es necesario conocer el problema del sobreentrenamiento. La red debe ser capaz de clasificar otros casos que no se encuentren en el conjunto de datos de entrenamiento, es decir, de generalizar. Sin embargo, puede suceder que la red esté sobreajustando el valor de sus pesos a características muy concretas exclusivas de esos datos. Esto provoca fallos con nuevos datos porque la red es incapaz de generalizar [35]. Para abordar este problema surgieron las capas de *dropout* que, en esencia, se encargan de desactivar ciertas neuronas de forma aleatoria durante el proceso de entrenamiento, previniendo posibles sobreadaptaciones a los datos. (*Véase la Figura 2.10*).

2.2.4 Capa densa

También denominadas capas *fully connected*, se caracterizan porque cada neurona se conecta con todas las neuronas de la capa siguiente. Particularmente en las CNN, se emplean en el tramo final de la red donde, en la mayoría de los casos, se encadenan varias de estas para realizar la clasificación. Normalmente, se hace uso de las funciones de activación sigmoide o *softmax* en la última capa para obtener las probabilidades asociadas a cada clase en problemas de clasificación binaria o de clasificación múltiple, respectivamente. Además, son capas con tantas

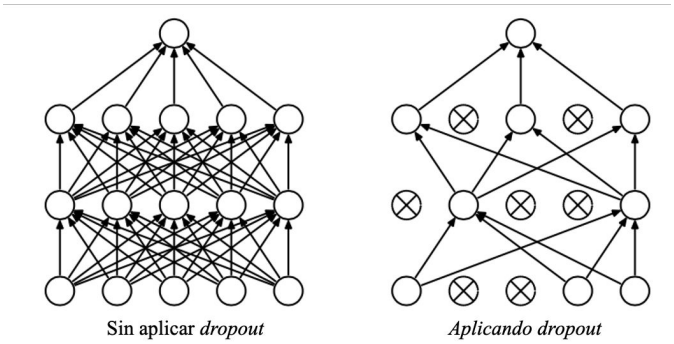

Figura 2.10: Representación de una red neuronal sin *dropout* frente a otra con capas de *dropout*. Imagen obtenida de [35].

conexiones que suelen contener una mayor cantidad de parámetros que los empleados en las capas convolucionales.

2.2.5 Capa batch normalization

Debido a que durante el entrenamiento la red va actualizando constantemente los pesos para cada *batch* de imágenes, podría ser difícil para el algoritmo ser capaz de aprender. Por tanto, lo que hacen este tipo de capas es estandarizar los valores de los pesos, de manera que tengan una media cercana a cero y una varianza aproximada de uno. Con esto se consigue estabilizar el proceso de aprendizaje y disminuir el número de épocas.

2.2.6 Arquitectura de una red convolucional

En general, este tipo de capas sigue una organización similar. En primer lugar, hay un tramo de extracción de características conformado por bloques. En cada uno de los bloques se aplican una serie de capas convolucionales, cada una con una función de activación a la salida, habitualmente la ReLU. Seguidamente, se emplea una capa de *pooling* para disminuir el tamaño de los mapas de activación y concentrar la información. Esta disminución permite aumentar el número de filtros con los que se trabaja, pues existe un compromiso entre ambos parámetros y el coste computacional. Asimismo, cabe destacar que, en líneas generales, también se incluyen capas de *batch normalization* y *dropout* para mejorar los resultados.

En segundo lugar, se entra en la fase de clasificación. Este tramo se denomina *top-model* y comienza con la conversión de los mapas de activación de la salida del último bloque convolucional en un vector unidimensional que concentra las características consideradas por la red como más importantes. A este se conectan una o varias capas *fully connected*, la última de las cuales con una función de activación *softmax* a la salida para generar un vector que recoja las probabilidades de pertenencia a cada una de las clases.

A partir de esta arquitectura base, representada en la Figura 2.11, se pueden desarrollar diferentes arquitecturas que se ajusten mejor a las necesidades del problema en cuestión. Este es el caso de los *encoder-decoder*, que son explicados a continuación en la sección 2.3.

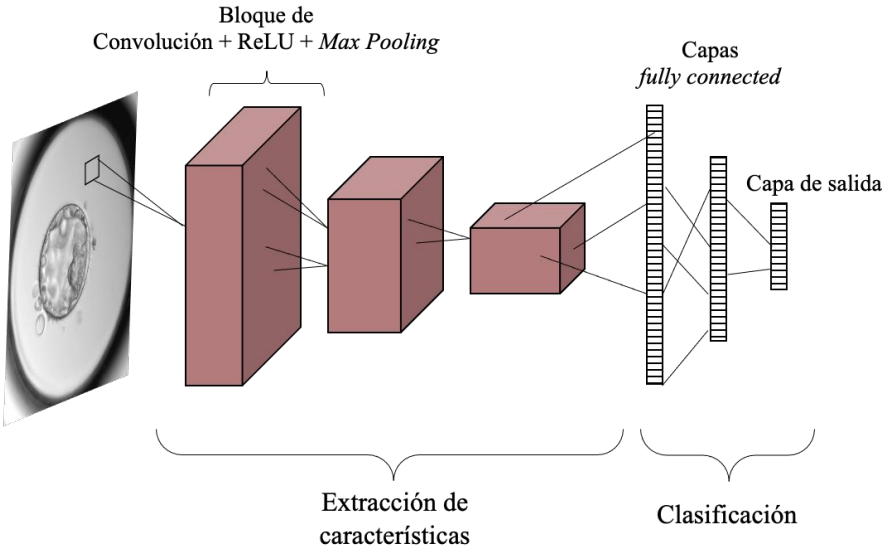

Figura 2.11: Representación de una arquitectura convolucional.

2.3 *Encoder-decoder*

Un *encoder-decoder* es un tipo de arquitectura neuronal que tiene como objetivo generar una salida con las mismas dimensiones que la entrada. La idea es introducir un vector de datos como entrada, comprimirlo para concentrar las características principales y reconstruir un nuevo vector con el mismo tamaño a la salida.

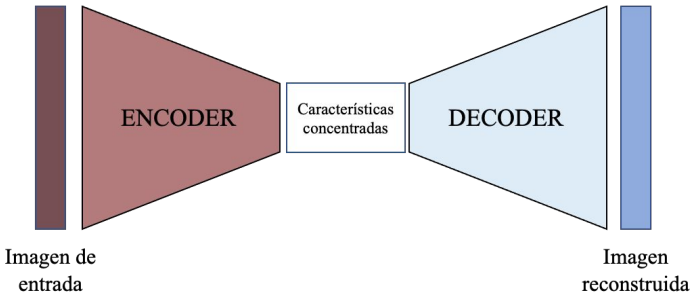

Figura 2.12: Estructura de una arquitectura *encoder-decoder*.

Para ello, la arquitectura se compone de una etapa de compresión de los datos denominada *encoder* y una fase de reconstrucción de la dimensionalidad denominada *decoder*, tal y como se muestra en la Figura 2.12. Para generar la representación condensada de los datos se emplea una reducción del número de neuronas en las capas intermedias.

La verdadera potencia de los *encoder-decoder* se encuentra en el campo de las redes neuronales profundas. Un aumento en el número de capas ocultas supone una mayor capacidad de reducción de la dimensionalidad y de extracción de características consideradas por la red como más importantes. Asimismo, es esencial combinar este aumento en la profundidad con funciones de activación no lineales, pues no se gana potencia en la representación haciendo uso únicamente de aquellas que son lineales. [22].

En los *encoder-decoder* convolucionales se hace uso de imágenes como entrada y de nuevo se busca reconstruir dicha entrada a la salida tras un proceso de compresión llevado a cabo en el *encoder*. Sin embargo, la principal diferencia con los *encoder-decoder* normales es que en este caso se pone el foco en el uso de relaciones espaciales entre los puntos para poder extraer características que puedan visualizarse en la imagen de salida. Dichas relaciones espaciales se generan gracias al uso de operaciones de convolución, tal y como sucedía en las CNN, y se consigue una reducción jerárquica de los datos.

Así como el *encoder* emplea capas convolucionales y capas de *pooling* para generar relaciones espaciales y reducir el tamaño de las capas, en el *decoder* deben producirse la inversa de estas operaciones para poder recuperar la dimensionalidad. Por ello se emplean dos tipos de capas:

- Capa de *Up Sampling*. El objetivo es el de aumentar la representación espacial de la información para finalmente recuperar el tamaño que tenía la imagen de entrada. Para ello, se realiza la interpolación de nuevos píxeles. Es necesario definir el tipo de interpolación y el tamaño que se desea conseguir.
- Capa de convolución traspuesta. También llamada deconvolución, pretende recuperar la resolución de la imagen mediante la aplicación de un filtro traspuesto. Para ello se aplica un *kernel* sobre cada una de las posiciones del mapa de entrada, así como se muestra en 2.13. Cabe destacar que hay posiciones en las que el *kernel* no solapa completamente sobre la imagen, por lo que hay una mayor contribución en las zonas centrales del mapa de salida [36]. No obstante, la red es capaz de aprender este tipo de desajustes y compensarlos de modo que no suponga un problema; o bien se puede definir un *padding* para evitar que esto ocurra.

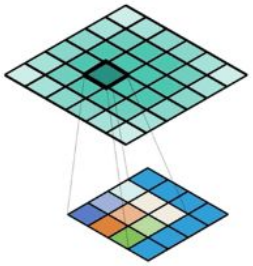

Figura 2.13: Representación de la operación de convolución traspuesta. Imagen sacada de [36].

Por tanto, la fase de *encoder* tendrá la arquitectura clásica de una CNN, es decir, capas convolucionales, seguidas de una capa de activación y otra de *pooling*. Mientras que el *decoder* trabaja con capas que recuperen la dimensionalidad y capas convolucionales seguidas de capas de activación. Generalmente, las capas se organizan de manera simétrica, en el sentido de que una capa en el *decoder* deshace el efecto de la capa correspondiente en el *encoder* para, finalmente, reconstruir las dimensiones de la entrada a la salida. Además, es fundamental saber qué función de activación tomar. Lo recomendable es trabajar con la ReLU [37], principalmente por ser sencilla de implementar y por introducir no linealidad en los datos.

Entre las aplicaciones relacionadas con la imagen se encuentra la eliminación de ruido, la creación de modelos generativos de nuevas imágenes o la segmentación semántica. Este último caso es el más interesante para el presente trabajo, pues la arquitectura de la red presentada en la

sección 4.2.1.3 está basada en una estructura *encoder-decoder*. Esta permite generar una clase para cada uno de los píxeles de la imagen de entrada, lo cual no sería posible sin una fase de decodificación que permita recuperar el tamaño original de la imagen.

Capítulo 3

Estado del arte

Las tareas que se están desarrollando en el ámbito de la embriología y la FIV son principalmente la de conteo de blastómeras, localización del embrión en la imagen, segmentación de las partes del embrión y clasificación del embrión respecto a su calidad o potencial de implantación.

En los últimos años, se esta siguiendo una tendencia basada en la combinación de la morfología y la cinética denominada "morfocinética". Esta permite estudiar las características morfológicas de forma continuada, viendo la evolución del embrión a lo largo de las diferentes fases. Surge gracias a la aparición de los sistemas TMS, que permiten obtener imágenes de gran calidad desde la fecundación hasta la implantación del embrión. Como consecuencia, se han presentado una serie de estudios que tratan de determinar la utilidad del empleo de características morfológicas en el desempeño de estas tareas.

En [38] desarrollan un sistema que combina el análisis de componentes principales (del inglés, PCA) con redes neuronales artificiales y que recibe como entradas parámetros morfocinéticos, como los tiempos en los que se pasa de un estadio al siguiente, el estado de las blastómeras o la presencia de multinucleación. El área debajo de la curva ROC (del inglés, *Receiving Operating Characteristic*) fue AUC = 0.71, lo cual estaba en sintonía con otros estudios similares, lo que apunta a que la máxima información posible que se puede extraer de la morfocinética puede terner su límite en valores marginalmente superiores a AUC=0.7, según [38]. Por otra parte, en [39] se hace una revisión de aquellos trabajos que han tratado de emplear parámetros morfocinéticos para evaluar el desarrollo embrionario, el tiempo de las divisiones, la viabilidad del embrión y las posibilidades de implantación. Por esto, muchos de los métodos presentados a continuación van a ser, en esencia, extractores de características morfológicas.

En [40] se encargaron de realizar un conteo de blastómeras. Esto es importante pues los tiempos y la sincronización en las divisiones celulares del embrión está relacionadas con la calidad del mismo. Para ello, en este artículo desarrollan una arquitectura *U-Net* residual dilatada, con la que consiguen una *accuracy* media del 88.2 % en el conteo y localización de los centroides para embriones con entre 1 y 5 blastómeras.

Por otra parte, la segmentación de las distintas regiones del embrión es ampliamente utilizada con el objetivo de extraer características de las diferentes zonas y poder realizar una clasificación. Primeramente, surgieron estrategias para segmentar las partes por separado. En [41] se hacen uso de técnicas clásicas de extracción de características como Gabor-DCT para localizar y segmentar la masa celular interna, consiguiendo un coeficiente de Jaccard de 70.3 %. Del mismo modo, en [42] se emplean técnicas de segmentación *level-set* para obtener el trofoectodermo, consiguiendo una *accuracy* del 84.3 %. Por contra, también se pueden ver trabajos donde se hace uso de redes neuronales para desempeñar esta labor. Un ejemplo es [43], que segmenta la masa haciendo uso de redes neuronales convolucionales. En concreto, se emplea la arquitectura VGG, que permite obtener un coeficiente de Jaccard del 76.5 %. De igual manera, en [44] se propone una Red Neuronal Jerárquica que permite aprender características con respecto a su posición espacial en el embrión. Con esto se consigue un índice de Jaccard del 78.1 %.

Por otro lado, también se encuentran estudios más completos donde se intenta realizar una segmentación múltiple del embrión. Este es el caso de [45], donde se emplea una CCN denominada DeepLab-V3 y un método de interpolación progresiva a modo de *decoder* para practicar la segmentación semántica de imágenes de blastocisto en masa interna, trofoectodermo, blastocele y zona pelúcida. Con ello consiguieron un índice de Jaccard medio del 0.825. En la misma línea sigue el trabajo de [46], que realiza una segmentación del ovocito para realizar una evaluación automática de este. Para ello, prueba hasta 71 arquitecturas distintas, obteniendo mejores resultados con el modelo DeepLab-v3-ResNet-18, con un coeficiente Dice de 0.897.

Hay que destacar que el fin último de todas estas técnicas es el de mejorar la tasa de éxito en los tratamientos de FIV mediante una selección de embriones de calidad y con alto potencial implantatorio. Por lo tanto, la tarea de clasificación de los embriones va a ser fundamental. En este apartado, hay que destacar el trabajo de [47] pues desarrollan un sistema llamado STORK basado en la red neuronal Inception que permite predecir la calidad del blastocisto con un AUC de 0.90 y 0.76 para dos conjuntos de datos externos. Asimismo, integran la calidad del embrión con la edad de la gestante para plantear escenarios relacionados con la probabilidad de éxito en el embarazo. También, es importante destacar la evaluación de varias arquitecturas CNN, como son Inception-V3, ResNet, Inception-Resnet y Xception, que llevan a cabo en [48]. Los resultados tanto a la hora de clasificar como para diferenciar entre blastocisto y no blastocisto son superiores con Xception, con resultados de *accuracy* del 64.95 % y 90,97 %, respectivamente. Y, por último, cabe mencionar de nuevo a [45], ya que también llevan a cabo una clasificación con un resultado del 70.9 % de *accuracy*, obtenido gracias al uso de un bloque convolucional (denominado bloque C3) que compacta, contextualiza y calibra la información, facilitando el aprendizaje.

En definitiva, en segmentación se puede ver que la tendencia es la de emplear IA para conseguir diferenciar las regiones de la imagen, pues los resultados mejoran notablemente con respecto a métodos clásicos. Y, es más, se tiende a buscar segmentaciones múltiples, lo que requiere arquitecturas cada vez más profundas y más refinadas para obtener mejores resultados que en la segmentación de regiones individuales. De forma similar pasa en el ámbito de la clasificación, donde los últimos trabajos se apoyan en DL, haciendo uso de redes complejas. Por último, hay que destacar que la mayoría de los métodos presentados hacen uso únicamente de las características morfológicas del embrión, a excepción de algunos como [47] que introducen variables externas para mejorar la clasificación.

Así pues, partiendo de los diferentes estudios presentados, se pretende desarrollar un sistema automático basado en técnicas de *deep learning* que sea capaz de llevar a cabo la segmentación múltiple del blastocisto, tomando como base una arquitectura *encoder-decoder*. El objetivo final será hacer uso de estas imágenes segmentadas para tratar de llevar a cabo una clasificación en base a la morfología del embrión para seleccionar aquellos embriones de calidad óptima. Además, cabe mencionar que las imágenes con las que se trabaja en este TFG son adquiridas con la tecnología *time-lapse* Geri®, a diferencia de la mayoría de estudios de la literatura que utilizan imágenes obtenidas con Embryoscope®, lo que supone un gran avance en el trabajo con lo último en tecnologías TMS.

Capítulo 4

Materiales y Métodos

4.1 Materiales

4.1.1 Base de datos

La base de datos de imágenes empleada en el presente TFG ha sido proporcionada por el Instituto Valenciano de Infertilidad (IVI). Contiene 375 imágenes microscópicas de embriones en el quinto día de gestación, por lo que se considera que están en su etapa de blastocisto. Estas pertenecen a una base de datos privada a la que solo acceden los miembros que participan en el proyecto *Deep Vision*. Esto se hace para garantizar la privacidad y confidencialidad del paciente.

Las imágenes se generan haciendo uso de la tecnología Geri®. Esta es una incubadora que cuenta con hasta seis compartimentos individuales, cada uno de los cuales alberga un embrión durante los 5 o 6 primeros días tras la fecundación. Además, cada uno de ellos cuenta con herramientas para asegurar unas condiciones estables y favorecer el desarrollo embrionario. Asimismo, cada compartimento incorpora un sistema de monitorización *time-lapse* de última generación que permite la adquisición de imágenes en alta resolución.

En la base de datos se tienen 129 imágenes con tamaño 928×928 *px* y 246 con tamaño 1440×1440 *px*. Asimismo, cada una de las imágenes tenía asociada una etiqueta según la calidad del embrión dictaminada por los expertos en base al criterio ASEBIR [11]. Se contaba con 113 de clase A, 233 de clase B y 27 de clase C. Las dos imágenes restantes fueron descartadas por no estar anotadas, por lo que se disponía de 373 imágenes para la clasificación. Además, es importante mencionar que todas las imágenes que componen la base de datos son correspondientes a embriones que fueron trasferidos y, es por esto, que hay muy pocas imágenes de clase C y ninguna de clase D, pues la implantación se lleva a cabo con los embriones de mejor calidad.

4.1.2 Software y harware

Para el desarrollo de los algoritmos de *deep learning* se ha hecho uso de *Python* 3.8.5 como lenguaje de programación y se ha trabajado principalmente con el *framework Keras* por ser una librería de alto nivel que ofrece código optimizado y ampliamente validado que tiene como objetivo simplificar la programación de algoritmos basados en aprendizaje profundo y que se ve sustentada por *TensorFlow* como *backend*. Como entorno de programación se ha hecho uso de *Spyder* 4.2.1.

En cuanto al *hardware* empleado, se hizo uso de un *MacBookPro 14.2* con las siguientes especificaciones:

Nombre del procesador	Intel Core i5 de doble núcleo
Velocidad del procesador	3,1 GHz
Memoria RAM	8 GB
Disco duro	500 GB
Tarjeta gráfica	Intel Iris Plus Graphics 650

Tabla 4.1: Especifiaciones para el *MacBookPro 14.2* empleado.

Además, debido a las necesidades computacionales que requiere el entrenamiento de redes neuronales profundas, se ha empleado un computador de altas prestaciones que incluía una unidad de procesamiento gráfico NVIDIA TITAN Xp.

4.2 Metodología

En este apartado se relatan el conjunto de técnicas y procedimientos empleados para llevar a cabo la segmentación y la clasificación de las imágenes. Para representar de una forma visual la línea de trabajo seguida, en la Figura 4.1 se presenta un diagrama de flujo con las diferentes fases del estudio.

4.2.1 Segmentación

4.2.1.1 Preparación de los datos

Originalmente, se contaba únicamente con las imágenes en escala de grises, pero no se tenía una base de datos de imágenes segmentadas que sirviera como referencia o *groundtruth* para evaluar la red de segmentación. Por tanto, previo al desarrollo de la red neuronal, fue necesario una segmentación manual de las imágenes en cuatro regiones: fondo, trofoectodermo, masa celular interna y blastocele. Para ello se hizo uso del *software* MATLAB®, empleando una aplicación denominada *Image Labeler*, presentada en la Figura 4.2. Esta ofrece una interfaz y herramientas específicas que facilitan la delimitación de regiones de interés en la imagen y la asociación a estas de una etiqueta. Cabe mencionar que tanto el proceso como el resultado final de la segmentación manual fue revisado y corregido por embriólogos especializados en evaluación embrionaria.

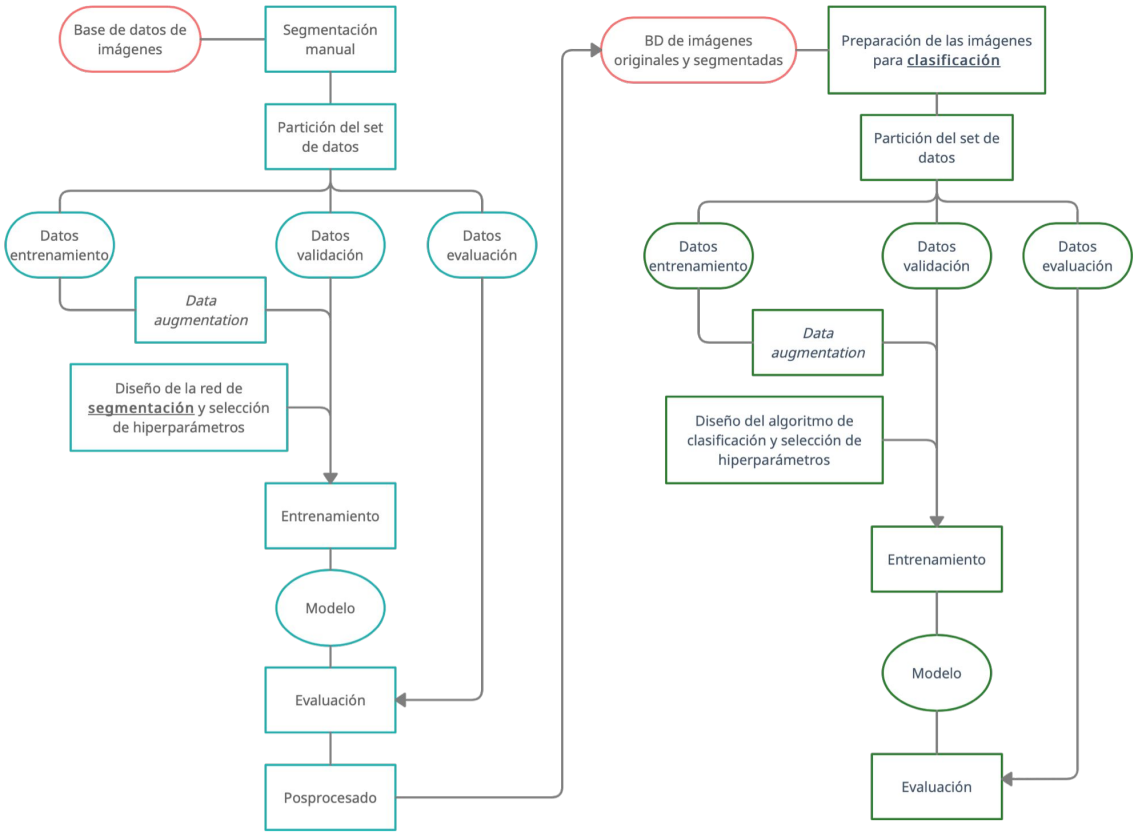

Figura 4.1: Diagrama de flujo que representa el proceso de segmentación (en azul) y el proceso de clasificación (en verde).

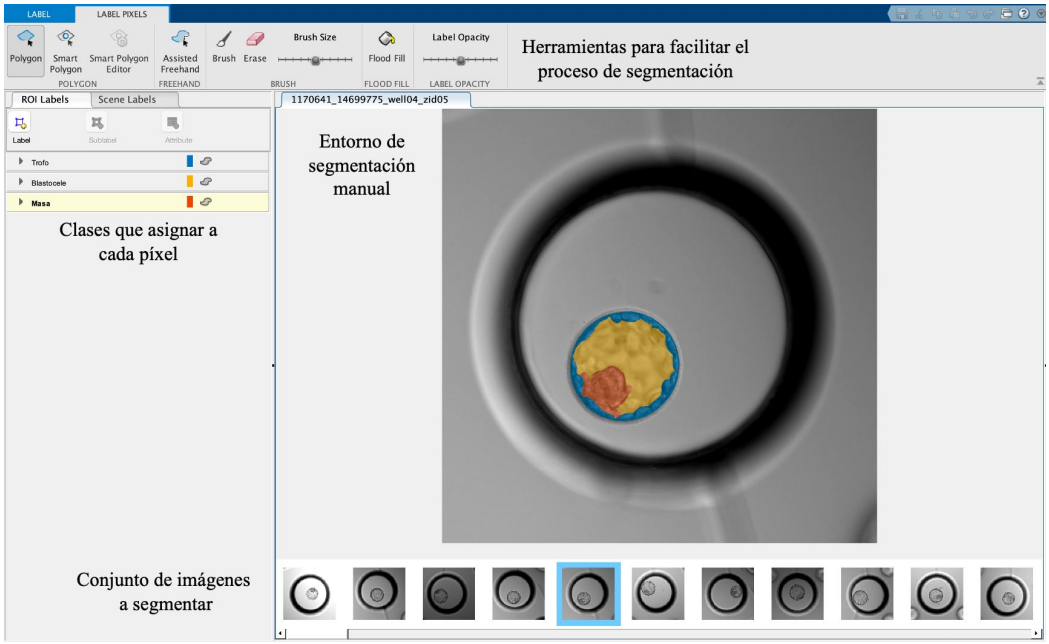

Figura 4.2: Presentación del entorno de trabajo de la aplicación *Image Labeler*.

4.2.1.2 Proceso de entrenamiento

El entrenamiento es el proceso por el cual la red neuronal va a ser capaz de aprender las características de las imágenes. En otras palabras, este se basa en alimentar a la red con el conjunto de muestras, que pasan desde la capa de entrada a la de salida en la etapa *forward*. Después, se compara la salida de la red con las imágenes del *groundtruth* y se computa el error en la predicción. Y, a continuación, se realiza la etapa *backpropagation* para la actualización de los pesos. Este proceso se repite de forma iterativa hasta alcanzar un resultado satisfactorio.

Para realizar un entrenamiento adecuado es necesario hacer una partición del set de datos en tres subconjuntos: entrenamiento, validación y test. El primero de ellos es el utilizado para alimentar a la red con imágenes y permitir el desarrollo planteado en el párrafo anterior. El conjunto de validación se emplea para monitorizar el proceso de entrenamiento, determinando qué tan bien actúa la red frente a muestras externas al conjunto de entrenamiento, indicando la capacidad de esta para generalizar. Por último, el conjunto de test se emplea cuando la red ya ha sido entrenada con el objetivo de ver su funcionamiento frente a un set de datos externo.

En este proceso intervienen tanto la arquitectura de la red como el conjunto de hiperparámetros escogido, los cuales se describen a continuación en las secciones 4.2.1.3 y 4.2.1.4, respectivamente.

Uno de los problemas principales a la hora de entrenar redes neuronales es el denominado sobreentrenamiento (del inglés, *overfitting*). Este sucede porque la red ajusta sus parámetros no solo a las características generales de las imágenes, sino también a aquellas que son particulares de ese conjunto de datos. Esto supone problemas en la clasificación de muestras externas al conjunto de datos de entrenamiento. Para evitarlo, se van a emplear capas *dropout* para desactivar neuronas de forma aleatoria y se va a practicar un procedimiento de *data augmentation*, por el que se va a aumentar el número de muestras haciendo trasformaciones sobre las originales.

4.2.1.3 Arquitectura de la red: U-Net.

Para la tarea de segmentación se ha hecho uso de una red neuronal basada en la arquitectura U-Net propuesta por Ronneberger, Fischer y Brox en [49], que presenta una arquitectura de *encoder-decoder* convolucional. Esta está conformada por una etapa *encoder* de extracción de características que permite obtener una representación abstracta de las imágenes de entrada para luego realizar una reconstrucción en la etapa *decoder* consiguiendo la clasificación individual de los píxeles de la imagen, otorgándoles a cada uno de ellos una etiqueta en función de la región de la imagen en la que se encuentran. Pero además, la U-Net cuenta con conexiones entre los bloques convolucionales del *encoder* y del *decoder* denominadas *skip connections*. Estas permiten recuperar información espacial que se pierde durante la fase de reducción de la dimensionalidad [50]. Esto junto con el gran número de filtros que se emplean en la fase de expansión permite propagar información de contexto de forma satisfactoria a capas de mayor resolución.

La etapa *encoder* sigue una estructura similar a una CNN estándar. Cuenta con cinco bloques compuestos por dos capas convolucionales, entre las que se intercala una capa *dropout*, seguidas por una capa *batch normalization* y una de *max pooling*. En cada capa convolucional se trabaja con un *kernel* de tamaño 3×3, se aplica un *padding same* y un *stride* de 1. Y la capa de *max pooling* se aplica sobre regiones 2×2 para reducir a la mitad el tamaño del mapa de entrada. Además, tras cada uno de los bloques se dobla en número de filtros empleados.

En la etapa *decoder* los bloques comienzan con una convolución traspuesta que reduce a la mitad el número de filtros y que dobla el tamaño de la imagen con un *kernel* 2×2 que deshace el efecto de la capa *max pooling* de la fase *encoder*. Seguidamente, se concatena la salida de la convolución traspuesta con el mapa de activación del bloque del *encoder* correspondiente. Después, se suceden de nuevo una convolucional, una *dropout*, otra convolucional y una *batch normalization*. En las convolucionales se emplean *kernels* de 3×3, *padding same* y un *stride* de 1.

Finalmente, a la salida, se tiene una última capa convolucional con un *kernel* de tamaño 1×1 y con cuatro filtros, con lo que se obtiene un mapa de activación por cada clase. A estos se les aplica la función *softmax* para obtener cuatro mapas de probabilidades, de manera que cada píxel tiene una probabilidad de pertenecer a cada una de las clases y se le asigna aquella de mayor probabilidad.

La arquitectura de la red descrita se presenta en la Figura 4.3 de manera visual y cuenta con 31.106.052 parámetros, de los cuales hasta 31.100.036 son parámetros entrenables, es decir, que optimizarán su valor durante el entrenamiento.

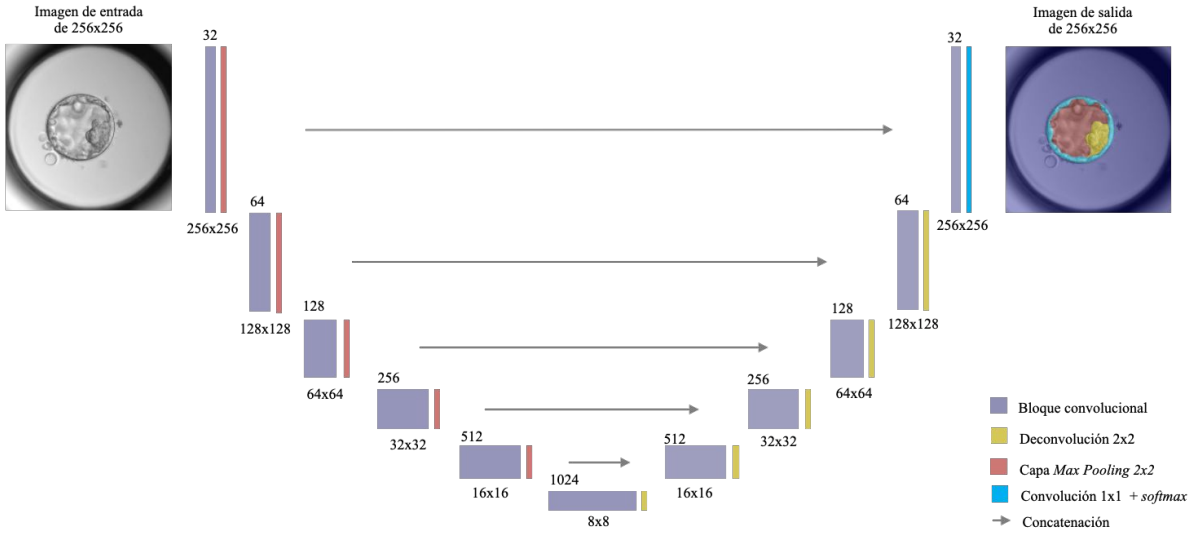

Figura 4.3: Arquitectura de la U-Net propuesta.

Para un mayor detalle en relación al bloque convolucional se muestra la Figura 4.4.

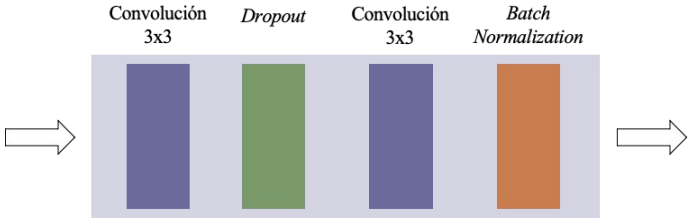

Figura 4.4: Representación del bloque convolucional, compuesto por dos capas convolucionales, una *dropout* y una *Batch Normalization*.

4.2.1.4 Hiperparámetros

Los hiperparámetros son un conjunto de valores configurables por el programador y relacionados con el diseño y el entrenamiento de la red. A diferencia de parámetros como los pesos de la red, estos no van a ser obtenidos de los datos. Asimismo, los valores óptimos para estos hiperparámetros no son conocidos en un primer momento, por lo que se tiende a emplear valores que han funcionado en otros estudios y a seguir procedimientos empíricos.

Hay hiperparámetros relativos a la arquitectura de la red como el número y tipo de capas ocultas, el tipo de función de activación utilizada, el número de los filtros por capa o el tamaño de los *kernels*. Todos ellos han sido definidos en el apartado 4.2.1.3. Por tanto, en esta sección se va a pasar a hablar de aquellos hiperparámetros relacionados con el proceso de entrenamiento y aprendizaje de la red. Estos son:

- Número de épocas. Define el número de veces que el conjunto de datos de entrenamiento pasa por la red. Es decir, cada una de las imágenes del conjunto de entrenamiento realiza, como mínimo, tantas etapas *forward-backward* como número de épocas se indique en este hiperparámetro.
- Tamaño de *batch*. Se define por *batch* a un subconjunto de imágenes del conjunto de entrenamiento que completan una iteración *forward-backward*. Por tanto, cuantos más *batches* haya por cada época, más veces se van a actualizar los pesos. Entonces, es vital el tamaño de *batch* que se seleccione. Pueden darse tres casos. Si se toma un *batch* cuyo tamaño es igual que el set de datos, únicamente se recalcularían los pesos una vez por época. Por contra, si el tamaño de *batch* es igual a uno, los pesos cambiarían con respecto a las características particulares de la imagen en cuestión. Por ello, lo habitual es emplear *batches* de entre 32 y 256 imágenes, pues es una manera de optimizar los pesos varias veces en una misma época y hacerlo en base a las características comunes de las imágenes que componen el *batch*. Este último método, denominado *Mini Batch Gradient Descend*, ha sido el empleado en este trabajo.
- Tasa de apredizaje. Es un hiperparámetro que define la velocidad de apredizaje del modelo, es decir, la velocidad con la que el modelo se acerca al mínimo global del plano definido por el gradiente. Con tasas de aprendizaje más altas la convergencia hacia el mínimo global es más rápida, pero al acercarse a saltos más grandes el mínimo global podría no ser alcanzado nunca. Por contra, con tasas más bajas se va mucho más lento y se podría caer en un mínimo local del que ya no se pudiera salir.
- Ratio asociado al *momentum*. Pondera la contribución del *momentum* en el proceso de actualización de los pesos, siendo el *momentum* una variable que almacena información acerca de los gradientes anteriores. Esta asiste al optimizador en zonas donde el gradiente es muy pronunciado y hay propensión a errores, mejorando la convergencia hacia el mínimo.
- Función de pérdidas. Es un método que indica qué tan buena es la salida de la red con respecto al *groundtruth*. El objetivo es minimizar esta función, pues eso significa optimizar los parámetros de la red para que esta consiga mejores predicciones. La función escogida es el coeficiente Dice, que se explicará en el apartado 4.2.1.5.
- Optimizador. El aprendizaje del modelo se basa en minimizar la función de pérdidas o, lo que es lo mismo, optimizar los parámetros. Por ello, es necesario un optimizador que, a

partir del valor de las pérdidas, indique cuánto se ha de modificar cada uno de los pesos de la red para converger hacia el mínimo global.

4.2.1.5 Función de pérdidas: coeficiente Dice

La función de pérdidas es un hiperparámetro que va a ser fundamental para el entrenamiento pues va a indicar en qué medida la red esta clasificando correctamente. Las más utilizadas son el Error Cuadrático Medio y la Entropía Cruzada, presentadas en la sección 2.1.4. Sin embargo, es necesario escoger la función de pérdidas según el problema a resolver, que en este caso es la segmentación semántica. Una de sus características es el desbalanceo de clases, ya que generalmente se tienen una mayor cantidad de píxeles pertenecientes a la región de fondo que a las regiones de interés. Esto hace que la red ponga el foco en aquellas áreas más representativas e ignore las otras, lo que supone problemas cuando se compara el error frente al total de los datos.

El desbalanceo es precisamente lo que sucedía con las imágenes con las que se ha trabajado en el presente TFG. Por ello, se ha calculado el coeficiente Dice para cada clase y se ha computado la media para contrarrestar el efecto del desbalanceo. En otras palabras, se ha empleado el Dice como un estadístico que permite comparar la similitud entre dos imágenes, a partir de la similitud entre sus regiones por separado.

Para una imagen binaria el coeficiente Dice puede definirse tal que:

$$Dice = \frac{2\sum_i^N p_i g_i}{\sum_i^N p_i^2 + \sum_i^N g_i^2} \tag{4.1}$$

siendo p_i el valor para cada píxel predicho por la red, g_i el valor de cada píxel en el *groundtruth* y N el número de píxeles en la imagen. En otras palabras, el Dice puede entenderse como dos veces la intersección entre la predicción y el *groundtruth* dividido por la suma de los píxeles de la predicción y del *groundtruth*.

Asimismo, para el correcto funcionamiento del algoritmo de retropropagación del error, es necesario que la función de pérdidas sea derivable respecto a la salida de la red. Por ello, a continuación se presenta dicha derivada [51].

$$\frac{\delta Dice}{\delta p_j} = 2 \cdot \frac{g_j(\sum_i^N p_i^2 + \sum_i^N g_i^2) - 2p_j(\sum_i^N p_i g_i)}{(\sum_i^N p_i^2 + \sum_i^N g_i^2)^2} \tag{4.2}$$

Por definición, el Dice trabaja con salidas binarias, pero las salidas de la red presentada son probabilísticas por emplear una función *softmax* como capa de salida. Por ese motivo, p_j representa un valor entre $[0, 1]$ mientras que g_j es 0 o 1, siendo j cada una de las clases. Así pues, para mantener la equivalencia entre numerador y denominador, no se computan los cuadrados en el denominador. Además, la función de pérdidas ha de ser minimizada, por lo que finalmente queda tal que

$$Dice_{Loss} = 1 - \frac{2\sum_i^N p_i g_i}{\sum_i^N p_i + \sum_i^N g_i} \tag{4.3}$$

de forma que su valor esta comprendido entre los valores extremos 0 y 1, para una completa disimilitud o una similitud perfecta, respectivamente.

4.2.1.6 Optimizador: Adam

Debido a la necesidad de que la red tenga el mejor conjunto de parámetros W posible, es necesario plantear la actualización de los mismos como un problema de optimización. Por ello, es necesario el empleo de un algoritmo para recalcular los pesos en base a la función de pérdidas. En este TFG, se ha empleado el optimizador Adam, que es un algoritmo de primer orden basado en la optimización del gradiente de la función de pérdidas J. Es sencillo de implementar, computacionalmente eficiente y requiere poca memoria [52].

Adam trabaja con dos *momentum* para actualizar los pesos. El primero de ellos, F,se forma a partir del suavizado exponencial del gradiente de primer orden.

$$F_i = \beta_1 F_i + (1 - \beta_1) \left(\frac{\delta L}{\delta w_i} \right) \quad \forall i \tag{4.4}$$

El segundo *momentum*, A, se computa a partir de un promediado exponencial de los gradientes al cuadrado. Este está relacionado con los valores históricos de los gradientes y va a dar más peso a aquellos calculados en iteraciones más cercanas a la actual.

$$A_i = \beta_2 A_i + (1 - \beta_2) \left(\frac{\delta L}{\delta w_i} \right)^2 \quad \forall i \tag{4.5}$$

A partir de ambos *momentum* se realiza la actualización de los pesos tal que

$$w_i = w_i - \frac{\alpha_t}{\sqrt{A_i + \epsilon}} F_i \quad \forall i \tag{4.6}$$

$$\alpha_t = \alpha \left(\frac{\sqrt{1 - \beta_2^t}}{1 - \beta_1^t} \right) \tag{4.7}$$

siendo α_t la tasa de aprendizaje dependiente de la iteración t. Este ajuste sobre la tasa de aprendizaje presentado en la ecuación 4.7 sirve principalmente para corregir la inicialización de los dos *momentum*, por lo que es importante en las primeras iteraciones [22].

Así pues, se tienen A_i y F_i como los *momentum* respecto a cada parámetro, β_1, β_2 y ε como hiperparámetros y $w_i \in W$ como los pesos a actualizar.

4.2.1.7 Métricas de evaluación

Con el objetivo de conocer en qué medida el modelo es capaz de realizar segmentaciones correctas sobre el subconjunto de datos de evaluación, se emplean dos métricas ampliamente utilizadas en el ámbito de la segmentación de imágenes médicas.

La primera de ellas es el coeficiente Dice que, como se ha comentado en la sección 4.2.1.5, permite conocer la similitud de la imagen de referencia y la imagen segmentada por la red.

En segundo lugar, se ha trabajado con el índice de Jaccard que, del mismo modo que el Dice, es una métrica que ofrece información sobre el grado de similitud entre dos imágenes. Se obtiene a partir de la relación entre la intersección y la unión de ambas.

$$Jaccard = \frac{|A \cap B|}{|A \cup B|} \tag{4.8}$$

Son métricas muy similares y, de hecho, están positivamente correladas. No obstante, mostrando otra representación de ambas métricas en la ecuación 4.9, se puede observar que en el índice Jaccard los falsos positivos (FP) y los falsos negativos (FN) tienen un mayor peso en el denominador y, por eso, el Jaccard es una métrica que penaliza más que el Dice las malas clasificaciones.

$$Dice = \frac{2\,TP}{2\,TP + FP + FN} \qquad\qquad Jaccard = \frac{TP}{TP + FP + FN} \tag{4.9}$$

4.2.1.8 Etapa de posprocesado

Una vez la red ha segmentado las imágenes, se realiza un posprocesado para pulir posibles errores. En este caso, se ha empleado un tratamiento con operadores morfológicos específico para cada una de las clases en función de sus necesidades.

Se aplica un cierre sobre el trofoectodermo para realizar un rellenado de agujeros y se emplea una apertura sobre la masa para eliminar el halo que no pertenece a esta clase, ya que en mayor medida pertenece a la clase trofoectodermo. Por ello, para poder incorporar el sobrante de la masa al trofoectodermo se realiza un *Top-Hat* sobre la masa y el resultado se le suma a la clase trofoectodermo.

Estas tres operaciones se construyen a partir de las operaciones de dilatación y erosión. En esencia, la primera de ellas hace más gruesas las figuras y la segunda más estrechas. Y combinando ambas dan lugar a las siguientes:

- Apertura. Consiste en realizar una erosión seguida de una dilatación. Con esto se consigue eliminar salientes estrechos o elementos más pequeños que el tamaño del elemento estructurante sin modificar el tamaño del resto de objetos. Este es el caso de la serie de imágenes A y la C presentada en la Figura 4.5, donde se consigue eliminar todo aquello que no pertenece a la clase deseada.
- Cierre. En este caso se aplica primero una dilatación y después una erosión. Con la dilatación se va a conseguir el rellenado de agujeros y la erosión posterior devolverá a los objetos su tamaño original. Así se observa en la serie B de la Figura 4.5.

- *Top-Hat.* Se computa como la diferencia entre el original y la apertura, con lo que se obtiene aquello que se haya eliminado con la apertura. Es interesante en el tratamiento de estas imágenes porque permite reasignar píxeles mal etiquetados a otra clase.

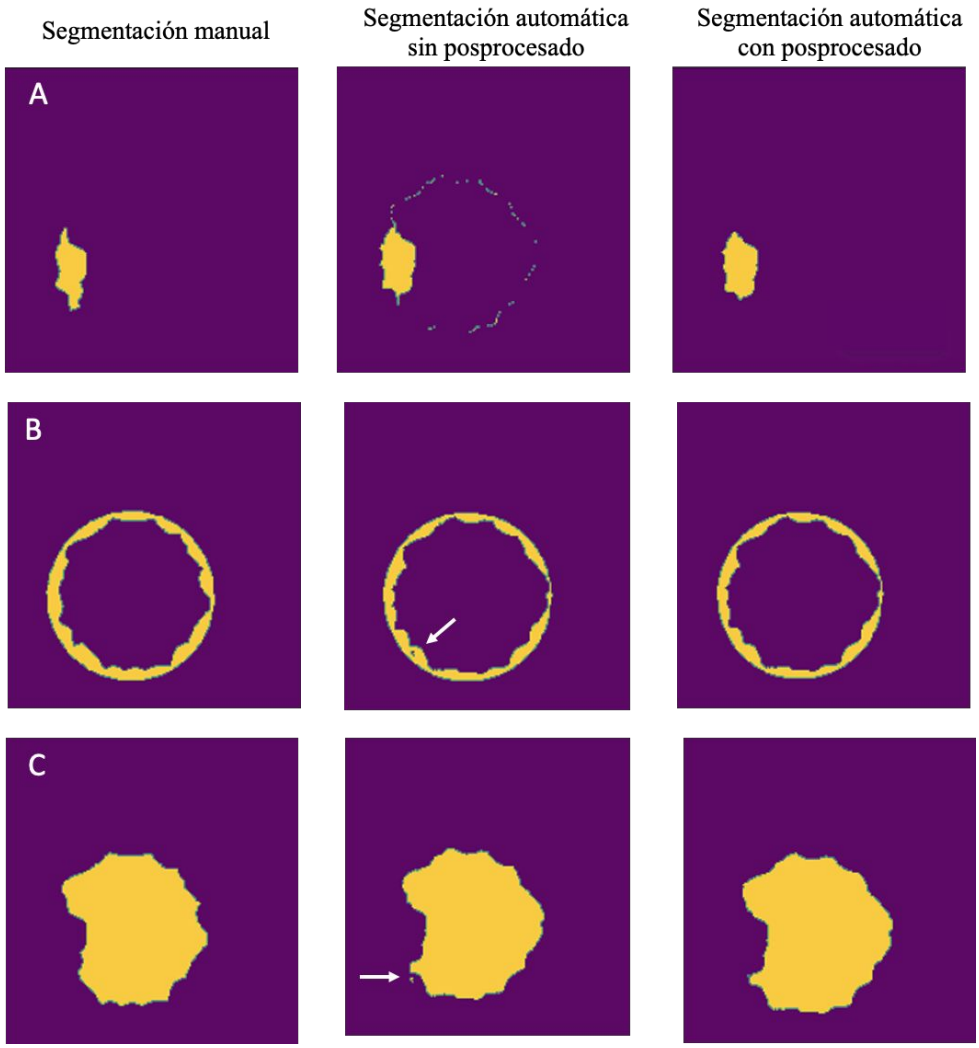

Figura 4.5: Ejemplo de el posprocesado sobre una imagen segmentada por la red. Se presentan tres series de imágenes, cada una correspondiente a una clase: masa (A), trofoectodermo (B) y blastocele(C).

4.2.2 Clasificación

Con la presente etapa se pretende clasificar a los embriones en función de su calidad, la cual es determinada por su morfología. Por ello, es de utilidad no solo emplear las imágenes de la base de datos del IVI, sino también hacer uso de las segmentaciones realizadas por la U-Net descrita en la sección anterior, de manera que se introduzca en la red información morfológica complementaria y se mejoren los resultados de clasificación.

Para llevar a cabo esta tarea se van a implementar dos métodos diferentes. El primero de ellos se basa en el concepto de trasferencia de conocimiento (en inglés, *transfer learning*), que es una metodología en la que se usa un modelo preentrenado con otros conjuntos de datos y se

adapta al problema en cuestión. Por su parte, el segundo método escogido es el denominado *AdaBoost-CNN*. Este ha sido descrito por Taherkhani, Cosma y McGinnity en [53] y consiste en el entrenamiento de varios modelos en serie y la minimización del error ponderando las muestras individualmente en función de cómo hayan sido clasificadas por el modelo anterior.

4.2.2.1 Trasferencia de conocimiento

Como consecuencia del coste computacional y temporal que supone el entrenamiento de una red neuronal profunda, surgen las técnicas de trasferencia de conocimiento. Estas se basan en la idea de trabajar con un modelo predefinido cuyo conjunto de pesos ha sido optimizado con una base de datos determinada y, a partir de estos, reentrenar la red con un set de datos propio para adaptarla al problema correspondiente. De esta manera, se evita tanto diseñar la arquitectura de la red como entrenarla desde el principio, no siendo necesario entonces contar con un conjunto de imágenes tan grande como lo sería para entrenar una red desde cero. En concreto, la técnica *transfer learning* consiste en conservar los pesos correspondientes a las capas que forman la etapa de extracción de características y reentrenar los pesos correspondientes a la etapa de clasificación.

Para aplicar este procedimiento, se ha escogido la red denominada Xception [54], que ha sido desarrollada por François Chollet. Esta utiliza bloques convolucionales con filtros de diferentes tamaños para extraer características a diferentes escalas y cuyas salidas se concatenan para formar un único mapa de activación. Además, se caracteriza por hacer uso de convoluciones separables, que es una forma alternativa y más rápida de hacer convoluciones. Esta consiste en realizar una convolución estándar en dos pasos. En el primero se aplica un filtro por cada canal del mapa de entrada (*depthwise convolution)* y en el segundo se realiza una combinación lineal de la salida del primer paso (*pointwise convolution)*. Tal y como se muestra en la Figura 4.6, si se tiene un mapa de entrada de tamaño $M \times M \times d$, se emplearán d *kernels* (uno para cada canal) y se generará una salida de tamaño $M \times M \times d$. Seguidamente, se aplicará una segunda convolución con filtros $1 \times 1 \times d$ para realizar una combinación lineal de los canales para cada posición del conjunto de mapas.

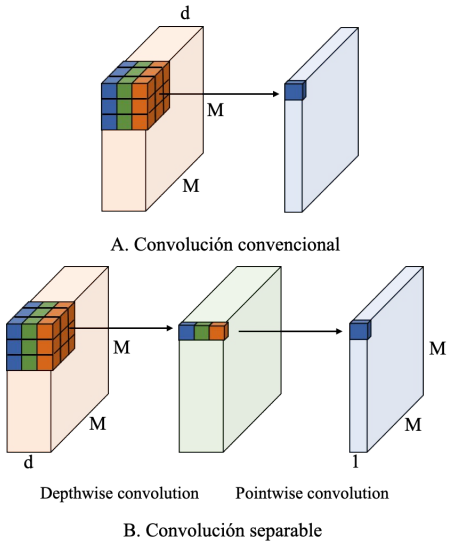

Figura 4.6: Convolución convencional (A) frente a convolución separable (B).

Por otra parte, esta red fue entrenada sobre el conjunto de imágenes conocido como Imagenet [33], que cuenta con más de 14 millones imágenes pertenecientes a 22.000 clases distintas. Y son los pesos optimizados durante el entrenamiento sobre este conjunto de datos los que se emplean en la técnica de *transfer learning* empleada en este TFG.

Finalmente, a la etapa extractora de características presentada en la Figura 4.7 se conecta una etapa de clasificación compuesta por dos capas *fully connected* entre las que se sitúa una capa *dropout* y que se conectan a una capa de *Batch Normalization*. Finalmente, se tiene a la salida otra capa *fully connected* con una *softmax* para realizar las predicciones.

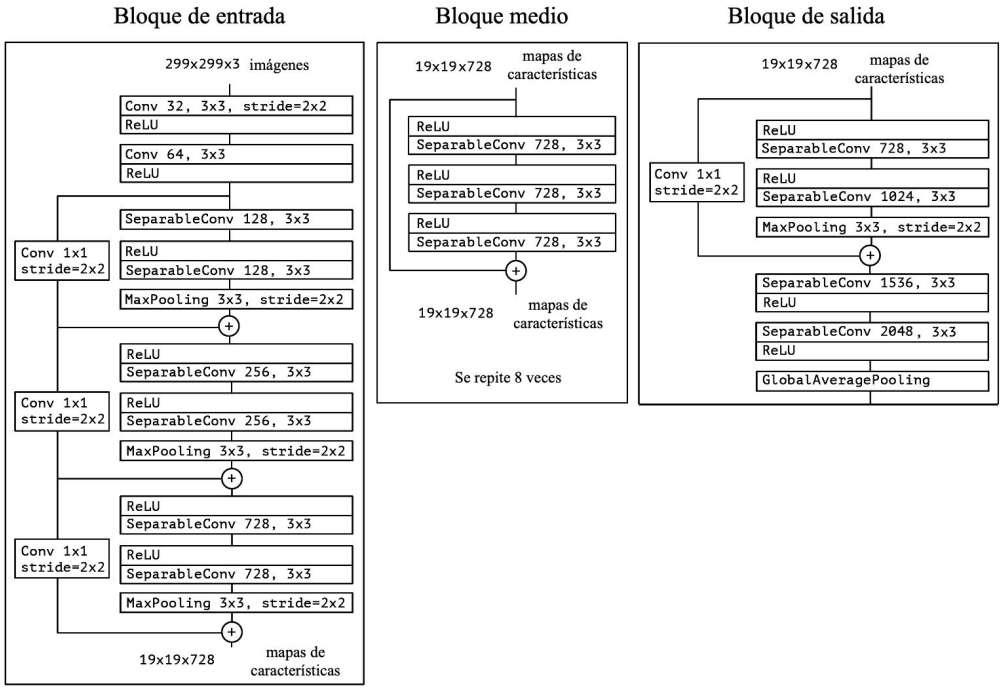

Figura 4.7: Arquitectura de Xception correspondiente a la etapa de extracción de características. Imagen modificada de [54].

Por último, mencionar que el entrenamiento sigue un proceso habitual donde se hace pasar el set de imágenes por la red en la etapa *forward* y se optimizan os pesos en la etapa *backward*. Sin embargo, hay que recalcar que solo se reajustan los pesos de la etapa de clasificación del modelo, mientras que el resto se mantienen fijos y su valor no cambia.

4.2.2.2 AdaBoost-CNN

Adaboost [55] es una técnica de clasificación que consiste en usar varios clasificadores en serie e intentar minimizar el error dando mas peso a las muestras incorrectamente clasificadas en los pasos anteriores. Se basa en el uso de *weak classifiers* (clasificador con al menos un 51 % de éxito) para construir un *strong classifier* y la predicción final se obtiene como una media ponderada de las clasificaciones de cada modelo.

En el método propuesto en [53] y empleado en este TFG se utiliza como clasificador una CNN, de manera que se aplica una trasferencia aprendizaje de un modelo al siguiente, mientras que se actualizan los pesos asociados a cada muestra en función de cómo han sido clasificadas por el

modelo anterior. Por tanto, cada clasificador se entrena en base a los errores del anterior. Así, se forma una secuencia de *weak classifiers* que dará lugar al clasificador final.

En primer lugar, se inicializan los pesos $D = [d_i]$ como $d_i = 1/n$, siendo $i = 1, 2, ..., n$ y siendo n el número de muestras para entrenamiento. Estos pesos se utilizan para entrenar la primera CNN de los M modelos, cuyos parámetros son inicializados aleatoriamente. La salida de la red para cada imagen de entrada x_i es un vector $P(x_i) = [p_k(x_i)]$ que contiene la probabilidad de pertenencia de esa imagen a cada una de las clases k. Y es con la salida de cada CNN con lo que se estiman los nuevos pesos d_i^{m+1} que serán empleados en la siguiente CNN.

$$d_i^{m+1} = d_i^m exp\left(-\alpha \frac{K-1}{K} Y_i^T log(P^m(x_i))\right) \tag{4.10}$$

siendo α el ratio de aprendizaje, $P^m(x_i)$ el vector de salida para el clasificador m e Y_i el vector de etiquetas *groundtruth* correspondiente a la muestra i. De esta manera, si Y_i^T y $P^m(x_i)$ están correlacionados, el valor de la exponencial será menor y el peso d_i^{m+1} se verá reducido. Es decir, se da menos peso a aquellas muestras bien clasificadas y viceversa, para que el siguiente modelo se fije más en las imágenes mal clasificadas.

Seguidamente, el valor de los pesos se normaliza, se guarda la CNN actual y se pasa al entrenamiento de la siguiente. Para esto último, se realiza una trasferencia de la arquitectura y de los parámetros de la CNN^m a la CNN^{m+1}, tal y como se representa en la Figura 4.8. De esta manera, la nueva red ya tiene un conocimiento sobre los datos y no necesita ser entrenada durante un gran número de épocas, lo que reduce el coste computacional del entrenamiento.

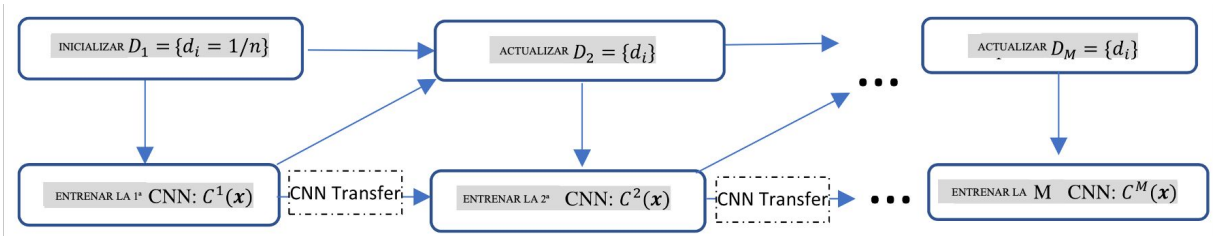

Figura 4.8: Representación del flujo de trabajo para el método Adaboost-CNN. Imagen modificada de [53].

Una vez se han entrenado todos los clasificadores es momento de realizar predicciones. Para ello, se pasa la muestra x a través de todos los clasificadores y a partir de las salidas de todos ellos se otorga una clase $C(x)$ a dicha imagen. Para ello se emplean las ecuaciones 4.11 y 4.12.

$$C(x) = argmax \sum_{m=1}^{M} h_k^m(x) \tag{4.11}$$

siendo $h_k^m(x)$ calculada tal que

$$h_k^m(x) = (K-1)\left(log(p_k^m(x)) - \frac{1}{K}\sum_{k}^{K} log(p_k^m(x))\right) \tag{4.12}$$

donde $p_k^m(x)$ es el elemento k del vector de salida P para la CNN^m. Por tanto, $h^m(x)$ es un vector que contiene las probabilidades de pertenencia a una clase determinadas por cada uno de

los clasificadores. A partir de esto, se realiza una suma de probabilidades para cada clase k y aquella clase que tenga un mayor valor de probabilidad será la que se le asigne a x.

Debido a que cada uno de los clasificadores va a ser considerado como un *weak classifier*, no es necesario el uso de una red muy profunda. Así pues, la CNN propuesta consta de tres bloques compuestos por dos capas convolucionales, una capa *max pooling* y una capa *dropout*. La salida del último bloque se conecta a una capa densa, que se conecta a otra *dropout* y esta, a su vez, a la capa de salida. Esta cuenta con una función *softmax* que genera una probabilidad por cada clase. Asimismo, las capas convolucionales cuentan con *kernels* 3×3, *padding same*, función de activación ReLU y una secuencia de 16, 32 y 64 filtros.

4.2.2.3 Función de pérdidas y optimizador

Como función de pérdidas para ambos métodos se escoge la entropía cruzada, ya descrita en la sección 2.1.4. Sin embargo, para el método Adaboost-CNN es en el cálculo de las pérdidas donde se incorpora el vector de pesos $D = [d_i]$ tal que

$$E_i = -\sum_{c=1}^{L} t_i^c log(z_i^c) d_i \tag{4.13}$$

donde E_i es el valor de pérdida asociado a la muestra i, t_i^c es el c-ésimo elemento del vector *groundtruth* Y_i correspondiente a la muestra i y z_i^c es el c-ésimo elemento del vector de salida P_i asociado a la muestra i.

Asimismo, para el ajuste de los parámetros de los modelos se ha empleado el optimizador Adam, siguiendo el procedimiento explicado en el subapartado 4.2.1.6.

4.2.2.4 Métricas de evaluación

Para determinar qué tan buenos son los resultados que generan los dos métodos se necesitan escoger métricas apropiadas y que permitan compararse con la literatura. Las escogidas son:

- Precisión. Evalúa el acierto del algoritmo en relación a aquello que este predijo como positivo.

$$Precision = \frac{TP}{TP + FP} \tag{4.14}$$

- *Recall*. Evalúa el acierto del algoritmo en relación a todo lo que realmente es positivo.

$$Recall = \frac{TP}{TP + FN} \tag{4.15}$$

- *Accuracy*. Evalúa el total de aciertos del algoritmo sobre todo el set de datos. Puede ser una métrica que dé lugar a errores, en especial cuando se trabaja con conjuntos de datos desbalanceados. Por ello, si se diera el caso, es interesante ponderar el valor que ofrezca la métrica para una clase en función del número de muestras de dicha clase con respecto al total de datos.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.16}$$

Capítulo 5

Resultados

5.1 Segmentación

El objetivo del proceso de segmentación semántica es que el sistema sea capaz de diferenciar las distintas partes de la imagen clasificando cada píxel como fondo, trofoectodermo, masa interna o blastocele. Para ello, se ha trabajado con la red presentada en la sección 4.2.1.3 y se ha entrenado con un conjunto determinado de hiperparámetros. Por una parte, se ha empleado un tamaño de *batch* de 32 imágenes reescaladas a 256×256 *px* y un total de 50 épocas. No obstante, se ha aplicado la condición *early stopping* para detener el entrenamiento si el conjunto de validación no reducía las pérdidas tras 8 épocas. Por otra parte, como optimizador se ha escogido Adam con una tasa de aprendizaje de 0.005, un *momentum* β_1 de 0.9 y β_2 de 0.99 y un ε de 10^{-8}, valores recomendados por [52]. Además, se ha reducido la tasa de aprendizaje por un factor de 0.1 cada vez que tras 5 épocas no se reducían las pérdidas en el grupo de validación.

Por otro lado, es necesario dividir el conjunto de 375 imágenes de la base de datos con el objetivo de tener muestras para entrenamiento, validación y evaluación de la red. Por tanto, se han tomado 304 imágenes para el entrenamiento, 34 para validación y 37 para evaluación.

Debido a que el subconjunto de entrenamiento no tiene un número elevado de muestras, se ha aplicado *data augmentation*. Esta idea se basa en generar nuevas imágenes a partir de trasformaciones de las originales. Únicamente se aplica sobre aquellas destinadas al entrenamiento con el fin de tener una mayor cantidad de imágenes para el proceso de aprendizaje y evitar problemas de sobreentrenamiento. De esta manara, la red tiene una mayor facilidad para generalizar con muestras externas.

Con el objetivo de no cambiar las propiedades fundamentales de los objetos de la imagen, se han aplicado únicamente dos trasformaciones. La primera de ellas ha sido la rotación aleatoria con ángulos de rotación de entre 0^{o} y 180^{o}. En segundo lugar, se aplicado un zoom con un factor de aumento con un valor entre 1 y 2.

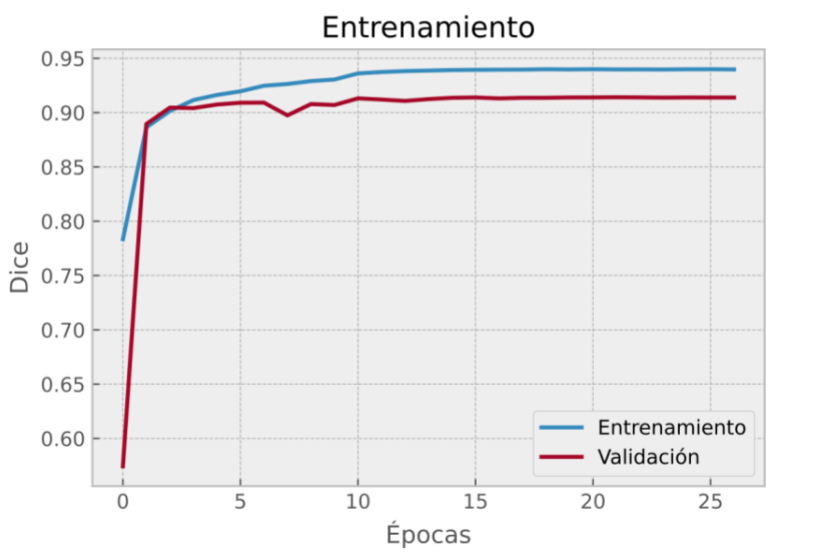

Figura 5.1: Gráfica que representa la evolución del Dice con respecto a las épocas.

Así pues, haciendo uso de este procedimiento se ha aumentado el conjunto de imágenes por un factor de 10 con lo que se trabaja con un set de entrenamiento de 3030 imágenes.

Haciendo uso de el conjunto de hiperparámetros descrito y la técnica *data augmentation* se ha obtenido la gráfica de la Figura 5.1, donde se presenta la evaluación del coeficiente Dice durante el entrenamiento para los subconjuntos de entrenamiento y validación.

Asimismo, cabe recordar que tras la segmentación automática de las imágenes, estas se han sometido a un posprocesado con operadores morfológicos, presentado en la sección 4.2.1.8. Este ayuda a rellenar agujeros y eliminar posibles errores de clasificación de píxeles.

A continuación, se presentan los resultados cualitativos. En primer lugar, en la Figura 5.2 se observa que la red es capaz de segmentar de forma muy precisa las distintas regiones y se ajusta de forma muy correcta a los bordes de estas. Esto significa que la red ha aprendido a determinar la clase a la que pertenece cada píxel a partir de las texturas y la morfología. Además, ha aprendido a discernir qué bordes son de interés y cuáles no. Esto se ve de forma clara en el trofoectodermo, pues la red es capaz de perfilar su contorno de forma muy similar a lo presentado en la segmentación manual. No obstante, se ve en la región correspondiente a la masa que la red tiende a extenderse ligeramente más que lo presente en la segmentación manual. La realidad es que el entrenamiento ha permitido obtener una red realmente precisa, por lo que llega un punto en el que es difícil diferenciar entre qué es un error o qué es una segmentación mejor que la realizada manualmente, pues la red puede haber determinado patrones intrínsecos de las imágenes que le lleven a definir el borde de la masa de esa forma concreta. En cualquier caso, esa ligera diferencia no va a suponer errores en el proceso de clasificación, ya que el cambio en la morfología del embrión es mínimo.

Por otro lado, en la Figura 5.3 se presenta una segmentación con un error en la zona del trofoectodermo. Este es un fallo claro ya que en la imagen A el borde estaba muy bien delimitado

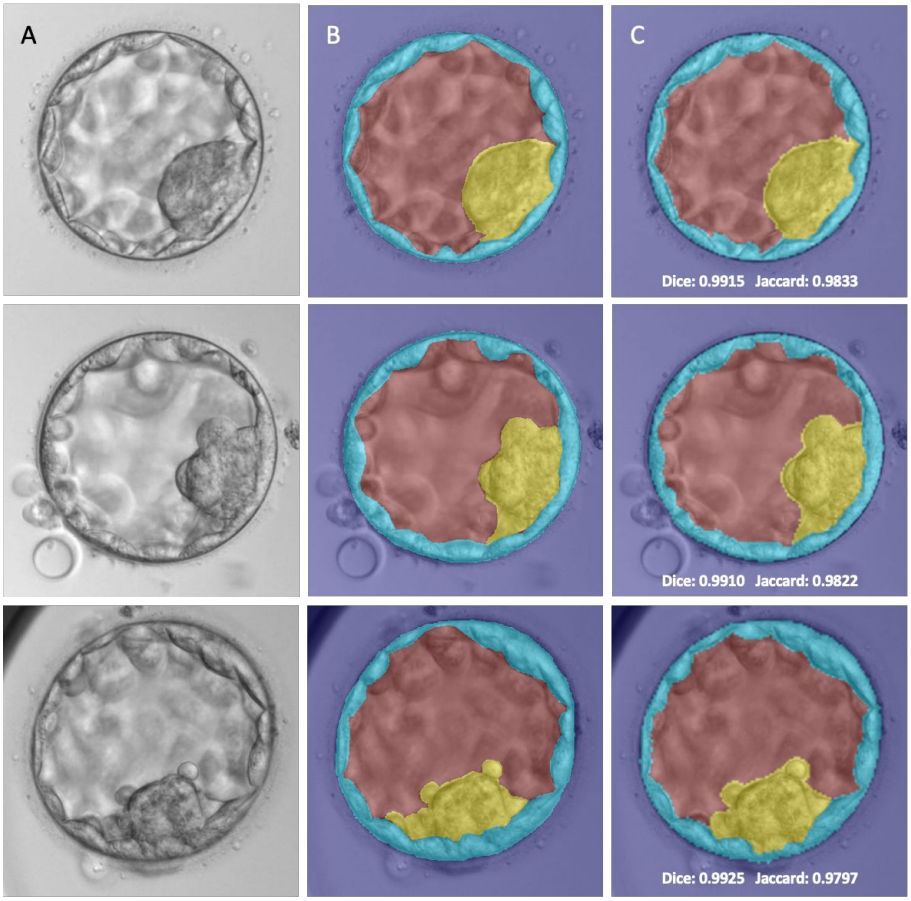

Figura 5.2: Se muestran tres series de imágenes compuestas por la imagen original (columna A), la imagen segmentada manualmente (columna B) y la imagen segmentada automáticamente (columna C). En las imágenes C se incluyen las métricas Dice y Jaccard obtenidas a partir de la imágenes C y las segmentaciones manuales B. Se presentan el fondo en morado, el trofoectodermo en azul, la masa celular interna en amarillo y el blastocele en rojo.

y la red debería haberlo segmentado correctamente. Asimismo, en la Figura 5.4 se observa que aún realizando el posprocesado hay conjuntos de píxeles que no pertenecen a la clase que corresponde. En la imagen A hay píxeles del trofoectodermo en el blastocele y, en la imagen B, ocurre a la inversa. A pesar de esto, los coeficientes Dice y Jaccard muestran valores muy elevados, pues la región que se ve afectada es de corta extensión. Debido a esto y a que el error se puede compensar con la información del contexto, los fallos en la clasificación no van a depender de estos errores en la segmentación ya que, aunque una mala segmentación suponga una mala clasificación, la segmentación presentada es suficientemente buena como para que estos errores no sean significativos.

A continuación, se presentan los resultados cuantitativos. Para estos se han empleado el coeficiente Dice y el índice Jaccard, ambas métricas descritas en el apartado 4.2.1.7. Para la evaluación del conjunto de datos de test, se han aplicado ambas métricas sobre las imágenes y se ha obtenido la media y la desviación típica para cada una de ellas. Estos resultados se presentan a continuación en la tabla 5.1.

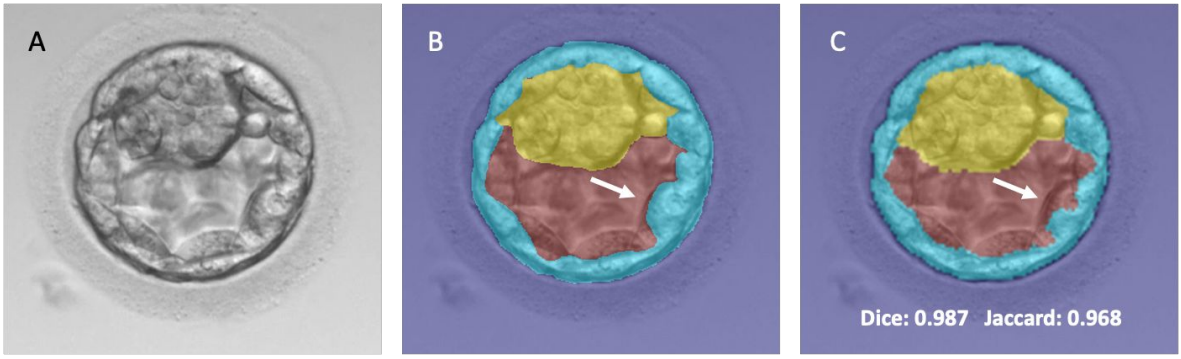

Figura 5.3: Serie compuesta por la imagen del embrión original (A), la segmentada a mano (B) y la segmentada de forma automática (C). Se señala con una flecha la zona donde se comete el fallo en la segmentación automática.

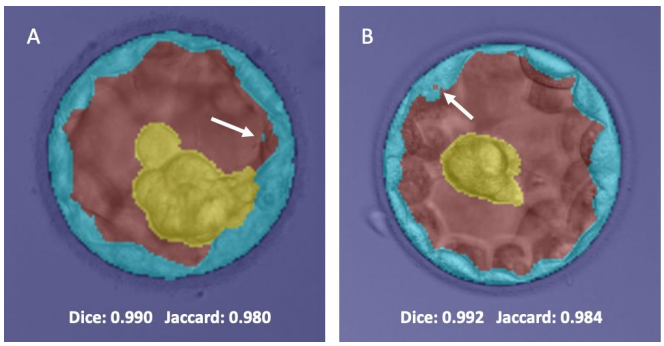

Figura 5.4: Ejemplo visual de la segmentación de embriones con errores no solucionados por el posprocesado.

Métrica	$\mu \pm \sigma$
Dice	0.992 ± 0.0025
Jaccard	0.974 ± 0.0065

Tabla 5.1: Coeficientes Dice y Jaccard medios del conjunto de valuación.

Como puede verse en la Tabla 5.1, los resultados para ambas métricas son muy elevados. Esto concuerda con los resultados cualitativos vistos anteriormente, ya que las imágenes segmentadas a mano y las segmentadas automáticamente compartían un alto grado de similitud visual. Asimismo, la desviación estándar en ambas métricas es muy baja, lo cual indica que los errores en la segmentación son pocos y muy poco significativos. Por último, es necesario destacar que el posprocesado realizado sobre las imágenes aumenta muy poco el Dice y el Jaccard, obteniendo unos valores para los mismos de 0,993 y 0,976, respectivamente. No obstante, tal y como se muestra en la Figura 4.5, es fundamental para pulir errores y dar un mayor sentido a la segmentación.

Para evaluar la relevancia de estos resultados, es necesario compararlos con el estado del arte. Para ello, se va realizar la comparativa con aquellos estudios que realicen una segmentación múltiple del embrión. Este es el caso de Rad y col. en [45] y Targosz y col. en [46], cuyos resultados se muestran en la Tabla 5.4.

Así pues, el método propuesto en el presente TFG supera por hasta casi ocho décimas el estado del arte en segmentación múltiple con IA. Asimismo, es importante destacar que en el caso de

Estudio	Jaccard
Targosz y col. [46]	0.897
Rad y col. [45]	0.8248
Método propuesto	**0.974**

Tabla 5.2: Comparativa entre los diferentes estudios haciendo uso del índice Jaccard.

Targosz y col. la segmentación se aplica sobre imágenes de ovocitos, con lo que las regiones a segmentar no son exactamente iguales. Por esto, la comparación más directa debido a la similitud en el objetivo y en las imágenes empleadas es con Rad y col., pues segmentan imágenes de blastocistos. No obstante, este estudio contó con hasta 578 imágenes, frente a las 375 del presente trabajo.

La realidad es que la literatura es escasa cuando se trata de trabajar con imágenes de embriones y aplicar técnicas de IA para realizar segmentación semántica. Por tanto, el presente trabajo supone un gran avance, tanto por la profundización en un campo con poca trayectoria como por los resultados obtenidos.

Asimismo, se debe mencionar también la contribución de este estudio, pues se hace uso de una base de datos con imágenes adquiridas por la tecnología TMS Geri®, a diferencia de los artículos mencionados, que utilizan la tecnología *Embryoscope*®. Por tanto, no había referencias ni resultados de base de los que partir, ya que esta tecnología todavía ha sido poco explotada.

5.2 Clasificación

La tarea de clasificación se basa en asignar una clase a las imágenes en función de la calidad del embrión. Esta calidad se basa exclusivamente en la morfología del mismo. En clínica, los embriólogos estudian las diferentes regiones del blastocisto y se basan en el criterio ASEBIR [11] para definir la calidad del mismo. Así pues, partiendo de esta forma de trabajar, se pretende hacer uso tanto de las imágenes segmentadas extraídas de la U-Net explicada en el apartado 4.2.1 como con las imágenes originales de la base de datos. El objetivo es dar información a la red sobre las distintas regiones de la imagen para que mejore las predicciones.

Para llevar a cabo esta tarea es importante considerar la base de datos de la que se disponía. Esta únicamente contaba 27 imágenes de clase C y ninguna de clase D, es decir, era un set de datos notablemente desbalanceado con un predominio de imágenes de las clases A y B. Así pues, se ha decidido realizar la tarea de clasificación únicamente sobre las imágenes de las clases A y B, con el objetivo de distinguir las muestras de calidad óptima de aquellas con una calidad ligeramente inferior, lo cual supone un reto incluso para embriólogos especializados en evaluación embrionaria. Para ello, se han tomado 110 muestras de clase A y 110 muestras de clase B.

Respecto a los hiperparámetros del método de *transfer learning*, explicado en la sección 4.2.2.1, se trabaja un total de 50 épocas, aplicando la condición *early stopping* con 20 épocas, y un *batch* de 64 imágenes, las cuales tienen un tamaño de 256×256 px. Para el optimizador Adam se toma una tasa de aprendizaje $\alpha = 0{,}006$, un *momentum* β_1 de 0.9 y β_2 de 0.99 y un ε de 10^{-8}. Además, se emplea la técnica de *data augmentation* para aumentar el número de imágenes en un factor de 10, realizando rotaciones aleatorias de entre 0 y 180.

Por otra parte, para el método *AdaBoost-CNN*, descrito en 4.2.2.2, se han empleado 20 modelos, con 3 épocas por modelo. Asimismo, se ha trabajado con *batches* de 32 imágenes de tamaño 64×64 *px* y, como optimizador se ha escogido Adam de nuevo, con una tasa de aprendizaje de 0,01, un *momentum* β_1 de 0.9 y β_2 de 0.99 y un ε de 10^{-8}. Por último, se ha doblado el número de imágenes siguiendo el mismo procedimiento que para el método de *transfer learning*.

Además, es fundamental destacar el modo en el que se han combinado las imágenes originales con las imágenes de las segmentaciones. Para el procedimiento de Adaboost-CNN se solapa la segmentación sobre la imagen original, así como se muestra en la Figura 5.6. Mientras que para *transfer learing* se introducen tres entradas, cada una de las cuales se corresponde al solape de la imagen original con la región correspondiente a las clases masa, trofoectodermo y blastocele, tal y como se muestra en la Figura 5.6. Por tanto, es necesario no solo una red Xception sino tres de ellas, donde cada una tiene como como entrada la imagen solapada asociada a una clase. Una vez extraídas las características de cada clase se concatenan los resultados de las tres redes y se pasa a la etapa de clasificación.

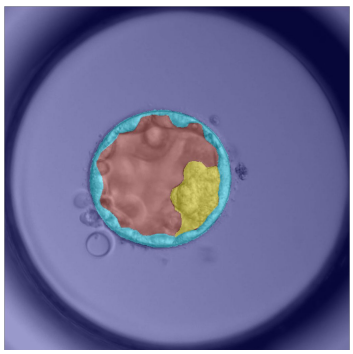

Figura 5.5: Representación de una imagen segmentada solapada sobre una imagen original.

Figura 5.6: Representación de una imagen original solapada por clases con la imagen segmentada correspondiente. De izquierda a derecha se disponen el blastocele, el trofoectodermo y la masa.

Finalmente, para el método Adaboost-CNN, debido a que el set de datos no es extenso y que la arquitectura de red es poco profunda, es necesario validar los resultados obtenidos y garantizar que son independientes tanto de la partición de los datos como de la inicialización aleatoria de los pesos que se realiza al inicio del entrenamiento. Para ello, se ha empleado la técnica de validación cruzada con cuatro grupos, cada uno compuesto por 55 muestras. Esta se basa en realizar cuatro iteraciones variando los grupos de entrenamiento, validación y evaluación como se muestra en la Figura 5.7, para finalmente realizar un promediado de los resultados obtenidos para los cuatro modelos. Esto garantiza la robustez del modelo presentado.

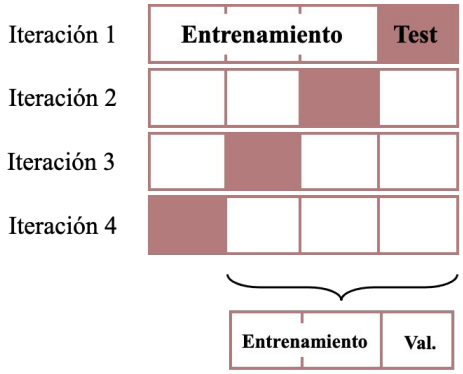

Figura 5.7: Representación de la técnica de validación cruzada con 4 grupos.

Seguidamente, se van a describir los resultados de clasificación. Estos se presentan a través de las métricas definidas en la sección 4.2.2.4, es decir, haciendo uso de la *accuracy* como medida comparativa y de la precisión y el *recall* para evaluar de forma más precisa el método propuesto. Asimismo, cabe destacar que la comparación no solo se va a realizar con la literatura, sino que también se va a realizar una comparativa entre los modelos propuestos empleando las imágenes de segmentación y sin emplearlas, con el objetivo de justificar la necesidad de realizar un procedimiento previo de segmentación semántica para mejorar las predicciones en la tarea de clasificación.

Hay que recalcar que la comparativa interna de los métodos es necesaria, pues es más robusta que una externa, ya que los resultados se obtienen para un mismo conjunto de datos. Dichos resultados se muestran en la Tabla 5.3. En ella se puede ver que el valor de la métrica *accuracy* es superior por hasta 6.9 unidades cuando se hace uso de la segmentación en el procedimiento de *transfer learning* y lo mismo sucede al emplear imágenes segmentadas en el método de Adaboost-CNN, donde se supera por 15.2 unidades al método empleando únicamente las imágenes originales. Por esto, queda ampliamente justificada la necesidad de realizar una segmentación previa de las imágenes, ya que introduce información complementaria acerca del embrión que incrementa las predicciones correctas.

Además, hay que destacar que el método Adaboost-CNN es capaz de igualar los resultados obtenidos con *transfer learning*. Esto es interesante pues es una metodología alternativa con un coste computacional y temporal bajo en comparación con los algoritmos que se usan habitualmente en las tareas de clasificación, donde predominan las redes neuronales profundas como Xception. Asimismo, a fecha de publicación de este TFG, no hay ningún estudio en la literatura que implemente este algoritmo en otro set de datos distinto a los propuestos por sus creadores en [53]. Por tanto, se está contribuyendo a la evaluación de este método al emplearlo sobre un nuevo conjunto de datos.

Tras la comparativa interna, se van a contrastar los resultados con la literatura con el objetivo de ofrecer un marco de referencia y ver qué tan buenos son estos resultados. Del mismo modo que para la tarea de segmentación, la extensión de la literatura es escasa. Sin embargo, cabe destacar el trabajo de Thirumalaraju y col. [48] y, de nuevo, el de estudio de Rad y col. [45]. En la Tabla 5.4 se muestra la comparativa entre estos y los métodos propuestos en el presente TFG.

Método	**¿Segmentación?**	***Precision***	***Recall***	***Accuracy***
Transfer learning	No	60.1	58.2	63.1
Transfer learning	Sí	70.1	70.0	**70.0**
Adaboost-CNN	No	53.2	54.0	54.6
Adaboost-CNN	Sí	71.5	69.3	**69.8**

Tabla 5.3: Comparativa interna entre los métodos propuestos empleando o no un procedimiento de segmentación previa.

Estudio	***Precision***	***Recall***	***Accuracy***
Thirumalaraju y col. [48]	-	-	64.95
Rad y col. [45]	71.1	**72.7**	**70.9**
Transfer Learning	70.1	70.0	70.0
Adaboost-CNN	**71.5**	69.3	69.8

Tabla 5.4: Comparativa entre los diferentes estudios de la literatura y los métodos propuestos.

Como se puede observar en la Tabla 5.4, la *accuracy* es mayor en [45] y supera por 0.9 unidades al método de *transfer learning* y por 1.1 al método Adaboost-CNN. No obstante, en [45] disponen de 578 imágenes frente a las 220 con las que se trabaja en este TFG, con lo que la red puede ver una mayor cantidad de muestras y mejorar en el aprendizaje. Pero, además, existe una diferencia fundamental en los objetivos de ambos trabajos: en [45] se busca determinar el resultado de la implantación, mientras que en este TFG se busca determinar la calidad del embrión. La diferencia radica en que la calidad depende de la morfología únicamente, mientras que el potencial de implantación también se ve afectado por otros factores externos. Por tanto, hay que considerar esto a la hora de comparar ambos trabajos, pues no persiguen el mismo propósito.

Por otro lado, en [48] se realiza una revisión de numerosas arquitecturas para conseguir una clasificación en 5 clases sin llevar a cabo una segmentación previa. No obstante, dos de las clases consistían en embriones que no habían alcanzado la fase de blastocisto y, de las clases restantes, solo los pertenecientes a una de ellas eran transferidos. Por tanto, la clasificación no es tan compleja como la presentada en este trabajo, pues las diferencias en la morfología son más notables entre las 5 clases que entre las 2 clases en las que se pretende clasificar en este TFG.

Capítulo 6

Conclusiones y líneas futuras

En el presente TFG se ha tratado de desarrollar un sistema automático que permita predecir la calidad del embrión a partir de una única imagen del mismo con el objetivo de conseguir una mejora en las tasas de implantación en las técnicas de reproducción asistida.

Para ello, en primer lugar, se ha llevado a cabo una revisión bibliográfica con el objetivo de conocer qué técnicas y metodologías se estaban llevando a cabo en los últimos años en relación al trabajo con imágenes de embriones, lo cual se ha descrito en el capítulo 3.

Seguidamente, se ha realizado un trabajo de segmentación manual sobre la base de datos de imágenes proporcionadas por el IVI con el objetivo de conformar un repositorio de imágenes segmentadas que sirviera como *groundtruth* para entrenar la red neuronal de segmentación.

A continuación, se ha desarrollado y entrenado un modelo *encoder-decoder* basado en un arquitectura U-Net para llevar a cabo la tarea de segmentación, de manera que se pudieran delimitar las regiones de las distintas imágenes. Posteriormente, se ha trabajado con dos métodos para llevar a cabo una clasificación de los embriones, en base a su morfología, en dos clases: calidad alta y calidad muy alta. Esto supone un reto debido a la gran similitud entre las imágenes de las dos clases, que incluso deriva en discrepancias entre los embriólogos especialistas en la materia. Por ello, es interesante generar resultados de forma automática para objetivizar y generalizar la selección de embriones, tarea que hasta el momento se lleva a cabo mediante inspección visual del experto, lo que introduce subejetividad en la elección. Por tanto, para mejorar los resultados del procedimiento de clasificación automática se ha empleado la información contenida en las imágenes segmentadas por la red *encoder-decoder*. Ambas tareas y los algoritmos empleados se han descrito con detalle en la sección 4.2.

Finalmente, en el capítulo 5 se han contrastado los resultados presentados con aquellos descritos en el estado del arte. Respecto al apartado de segmentación, se ha mostrado que el modelo propuesto consigue resultados notablemente mejores que el resto de estudios analizados. Por su parte, los métodos de clasificación igualan los resultados presentados en otros trabajos. Sin embargo, en esta última tarea todavía queda un amplio margen de mejora.

Por tanto, una de las labores de futuro es tratar de superar la tasa de acierto de los modelos de clasificación. Para ello se proponen diferentes maneras. En primer lugar, sería interesante conseguir un set de datos con un mayor número de imágenes y con suficientes muestras para todas las clases, para probar qué resultados se consiguen empleando los métodos propuestos. En segundo lugar, se debería tratar de implementar una CNN no preentrenada que incluyera nuevos tipos de bloques convolucionales, como el módulo *Squeeze and Excitation* propuesto en [56] o el bloque *Compact-Contextualize-Calibrate* definido en [45]. Del mismo modo, se debería probar a hacer más profunda la arquitectura planteada en el método de Adaboost-CNN a la par que se emplea un set de datos mayor, pues esto no se ha podido realizar con el repositorio actual porque había una tendencia al sobreentrenamiento debido al bajo número de imágenes del que se disponía. Asimismo, cuando se consiguiera un modelo con mayor capacidad de predicción, sería interesante introducir información externa como, por ejemplo, la edad de la gestante, tal y como hacen en [47]. De esta manera, no solo se estaría clasificando en términos de calidad embrionaria, sino que se pasaría a predecir el potencial de implantación del embrión.

Por otro lado, sería conveniente evaluar la red de segmentación sobre bases de datos de otras clínicas de embriología, para ver qué tan bien generaliza. Y, es más, sería interesante ver cómo se desarrolla este algoritmo en clínica una vez estuviera integrado en un sistema automático de segmentación y clasificación a partir de imágenes embrionarias.

Para terminar, recordar que el presente TFG se enmarca en un proyecto de mayor envergadura denominado *Deep Vision*, que tiene como objetivo el desarrollo de algoritmos basados en *deep learning* para la mejora de la selección embrionaria utilizando tecnologías de visión artificial. Por tanto, este trabajo es solo una parte de un gran proyecto en el que confluyen la inteligencia artificial y la embriología, pero supone otro pequeño avance hacia la mejora de la salud y el bienestar de las personas, lo cual es el fin último de este TFG, del proyecto *Deep Vision* y, en definitiva, de muchos grandes estudios del panorama científico actual.

Parte II

Presupuesto

Capítulo 7

Presupuesto

En el presente capítulo se recoge una valoración económica del proyecto desarrollado. En la sección 7.1 de desglosan los costes en tres apartados y en la sección 7.2 se ofrece un cálculo del presupuesto total asociado al desarrollo de un sistema de segmentación y clasificación automática de embriones.

7.1 Presupuestos parciales

Los presupuestos parciales se constituyen por tres secciones donde se definen los costes de mano de obra, los costes de maquinaria y los costes de materiales asociados al proyecto.

7.1.1 Costes de mano de obra

En esta sección se describen los recursos humanos que han sido necesarios para desarrollar el presente TFG. En concreto, se ha contado con la participación de: D.ª Valery Naranjo Ornedo, como directora del proyecto; D.ª Elena Payá Bosch, como cotutora del proyecto; y D. Alejandro José Vergara Richart como estudiante del Grado en Ingeniería Biomédica y autor del proyecto.

Denominación	**Uds.**	**Cantidad**	**Precio Unitario(€)**	**Total(€)**
Tutora (Catedrática)	h	16	42,00	672,00
Cotutora (Doctoranda)	h	36	17,20	619,20
Autor (Estudiante GIB)	h	312	13,80	4305,60
			Total	**5596,80**

Tabla 7.1: Cuadro de precios de la mano de obra.

7.1.2 Costes de maquinaria

En esta sección se incluyen los costes asociados al *software* y al *hardware* empleados durante el desarrollo del proyecto.

Como *software* se ha empleado MATLAB® y su aplicación *Image Processing Toolbox*, Spyder 4.2.1 como entorno de programación y la librería Keras sobre el *framework* TensorFlow. Asimismo, para la redacción del trabajo escrito se ha hecho uso de un editor de Latex en línea denominado Overleaf.

Denominación	**Uds.**	**Cantidad**	**Precio Unitario (€)**	**Periodo de amortización (años)**	**Intervalo amortizado (meses)**	**Total (€)**
Overleaf	u	1	0,00	1	2	0,00
Licencia MATLAB®	u	1	800,00	1	3	200,00
Image Preprocessing Toolbox	u	1	400,00	1	3	100,00
Python 3.8.5	u	1	0,00	1	5	0,00
Spyder 4.2.1	u	1	0,00	1	5	0,00
Keras y TensorFlow	u	1	0,00	1	5	0,00
					Total	**300,00**

Tabla 7.2: Cuadro de precios de maquinaria *software*.

Por otra parte, como *hardware* se ha empleado un servidor para el almacenamiento de datos y una GPU para realizar el entrenamiento de las redes propuestas, ambos pertenecientes al grupo CVBLab. Asimismo, se ha hecho uso de un ordenador personal MacBook Pro 14.2.

Denominación	**Uds.**	**Cantidad**	**Precio Unitario (€)**	**Periodo de amortización (años)**	**Intervalo amortizado (meses)**	**Total (€)**
MacBookPro 14.2.	u	1	1699,00	4	8	283,17
NVIDIA TITAN Xp	u	1	1349,00	4	5	140,52
Servidor Synology CVBLab	u	1	0,00	4	6	0,00
Procesador servidor Intel i7	u	1	1200,00	4	6	150,00
Disco servidor SSD 250GB	u	1	77,00	4	6	9,63
					Total	**583,31**

Tabla 7.3: Cuadro de precios de maquinaria *hardware*.

7.1.3 Costes de materiales

En esta sección se presentan los costes asociados a la tecnología necesaria para la adquisición de las imágenes. Por tanto, estos costes están asociados a la incubadora Geri® y a su mantenimiento.

Denominación	Uds.	Cantidad	Precio Unitario (€)	Periodo de amortización (años)	Intervalo amortizado (meses)	Total (€)
Incubador Geri®	u	1	85,000,00	10	1	708,33
Mantenimiento Incubador Geri®	u	1	4500,00	1	1	375,00
					Total	**1083,33**

Tabla 7.4: Cuadro de precios de los materiales.

7.2 Presupuesto total

Partiendo de los cuadros de precios descritos en el apartado 7.1, se va a configurar el presupuesto total. Asimismo, se va a incluir el porcentaje de gastos generales (13 %) y el asociado al beneficio industrial (6 %). Finalmente, se añade al precio total bruto el impuesto del IVA (21 %). Con todo ello se obtiene el presupuesto total que supone la realización del presente trabajo.

CAPÍTULOS	IMPORTE(€)
Coste de la mano de obra	5596,80
Coste de la maquinaria	883,31
Coste de los materiales	1083,33
PRESUPUESTO DE EJECUCIÓN DE MATERIAL	7,563,44
13 % de gastos generales	983,25
6 % de beneficio industrial	453,81
PRESUPUESTO DE EJECUCIÓN POR CONTRATA	9,454,31
21 % de IVA	1985,41
PRESUPUESTO TOTAL	**11.439,72**

Tabla 7.5: Presupuesto total.

Bibliografía

[1] Fernando Zegers-Hochschild y col. "The international glossary on infertility and fertility care, 2017". En: *Human reproduction* 32.9 (2017), págs. 1786-1801 (vid. pág. 3).

[2] WHO. *World report on disability*. Inf. téc. 2011 (vid. pág. 3).

[3] Sociedad Española de Fertilidad. *Saber más sobre fertilidad y reproducción asistida* (vid. págs. 3, 5).

[4] Instituto Valenciano de Infertilidad (IVI). https://ivi.es/preguntas-frecuentes/que-es-la-infertilidad/ (vid. págs. 3, 6).

[5] BetterHealth. https://www.betterhealth.vic.gov.au/health/ConditionsAndTreatments/age-and-fertility (vid. pág. 3).

[6] T. Editors of Encyclopaedia Britannica. *Embryology. Encyclopedia Britannica*. Feb. de 2019 (vid. pág. 4).

[7] Bruce M. Carlson. *Human Embryology and Developmental Biology*. Ed. por Elsevier Saunders. 2019 (vid. págs. 4, 5).

[8] Guadalupe Bueno Rodríguez. "El laboratorio de embriología". En: *Reproducción humana y laboratorio clínico. Ed Cont Lab Clín; 32: 54 - 71*. Sociedad Española de Medicina de Laboratorio, 2016-2017 (vid. págs. 4-7).

[9] National Health Service (NHS). *IVF*. https://www.nhs.uk/conditions/ivf/availability/. Jun. de 2018 (vid. pág. 5).

[10] ESHRE. *Revised guidelines for good practice in IVF laboratories*. Dic. de 2015 (vid. pág. 5).

[11] ASEBIR, ed. *Criterios ASEBIR de Valoración Morfológica de Oocitos,Embriones Tempranos y Blastocistos Humanos*. 2015 (vid. págs. 6, 9, 29, 47).

[12] Maria Fernanda Insua y col. "Obstetric and perinatal outcomes of pregnancies conceived with embryos cultured in a time-lapse monitoring system". En: *Fertility and sterility* 108.3 (2017), págs. 498-504 (vid. pág. 7).

[13] Csaba Pribenszky, Anna-Maria Nilselid y Markus Montag. "Time-lapse culture with morphokinetic embryo selection improves pregnancy and live birth chances and reduces early pregnancy loss: a meta-analysis". En: *Reproductive biomedicine online* 35.5 (2017), págs. 511-520 (vid. pág. 7).

[14] Charlotte HE Weimar y col. "In-vitro model systems for the study of human embryo–endometrium interactions". En: *Reproductive biomedicine online* 27.5 (2013), págs. 461-476 (vid. pág. 7).

[15] Ri-Na Su y col. "Maternal and neonatal outcomes in multiple pregnancy: A multicentre study in the Beijing population". En: *Chronic diseases and translational medicine* 1.4 (2015), págs. 197-202 (vid. pág. 7).

[16] Jun Wei y col. "Complications in multiple gestation pregnancy: a cross-sectional study of ten maternal-fetal medicine centers in China". En: *Oncotarget* 7.21 (2016), pág. 30797 (vid. pág. 7).

[17] John McCarthy. "What is artificial intelligence". En: (2007) (vid. pág. 7).

[18] Encyclopedia Britannica. *Artificial intelligence.* Ago. de 2020 (vid. pág. 7).

[19] IBM. *Artificial Intelligence (AI).* Jun. de 2020 (vid. pág. 7).

[20] William L. Encyclopedia Britannica Hosch. *Machine learning.* Jun. de 2020 (vid. pág. 8).

[21] Vanessa Isabell Jurtz y col. "An introduction to deep learning on biological sequence data: examples and solutions". En: *Bioinformatics* 33.22 (2017), págs. 3685-3690 (vid. pág. 8).

[22] Charu C Aggarwal y col. "Neural networks and deep learning". En: 10 (2018) (vid. págs. 8, 11, 16, 17, 21, 36).

[23] Jose Luis Marín. *Ciencia de datos, machine learning y deep learning.* 2018 (vid. pág. 8).

[24] IMB. *Deep Learning.* https://www.ibm.com/cloud/learn/deep-learning. Mayo de 2020 (vid. pág. 9).

[25] Mohammed Al-Qizwini y col. "Deep learning algorithm for autonomous driving using googlenet". En: *2017 IEEE Intelligent Vehicles Symposium (IV).* IEEE. 2017, págs. 89-96 (vid. pág. 9).

[26] JB Heaton, Nicholas G Polson y Jan Hendrik Witte. "Deep learning in finance". En: (2016) (vid. pág. 9).

[27] Andre Esteva y col. "A guide to deep learning in healthcare". En: *Nature medicine* 25.1 (2019), págs. 24-29 (vid. pág. 9).

[28] Frank Rosenblatt. "The perceptron: a probabilistic model for information storage and organization in the brain." En: *Psychological review* 65.6 (1958), pág. 386 (vid. pág. 11).

[29] Pethuru Raj y Preetha Evangeline David. *The Digital Twin Paradigm for Smarter Systems and Environments: The Industry Use Cases*. Academic Press, 2020 (vid. pág. 12).

[30] David E Rumelhart, Geoffrey E Hinton y Ronald J Williams. "Learning representations by back-propagating errors". En: *nature* 323.6088 (1986), págs. 533-536 (vid. pág. 15).

[31] David H Hubel y Torsten N Wiesel. "Receptive fields of single neurones in the cat's striate cortex". En: *The Journal of physiology* 148.3 (1959), págs. 574-591 (vid. pág. 16).

[32] Yann LeCun y col. "Gradient-based learning applied to document recognition". En: *Proceedings of the IEEE* 86.11 (1998), págs. 2278-2324 (vid. pág. 17).

[33] Princeton University Standford University. https://image-net.org/challenges/LSVRC/ (vid. págs. 17, 40).

[34] Christian Versloot. *What is padding in a neural network?* Feb. de 2020 (vid. pág. 18).

[35] Nitish Srivastava y col. "Dropout: a simple way to prevent neural networks from overfitting". En: *The journal of machine learning research* 15.1 (2014), págs. 1929-1958 (vid. págs. 19, 20).

[36] Thom Lane. *Las convoluciones traspuestas explicadas*. 2018 (vid. pág. 22).

[37] Jason Brownlee. *A Gentle Introduction to the Rectified Linear Unit (ReLU)*. Ene. de 2019 (vid. pág. 22).

[38] Robert Milewski y col. "How much information about embryo implantation potential is included in morphokinetic data? A prediction model based on artificial neural networks and principal component analysis". En: *Advances in medical sciences* 62.1 (2017), págs. 202-206 (vid. pág. 25).

[39] Caroline Pirkevi Çetinkaya y Semra Kahraman. "Morphokinetics of embryos—where are we now?" En: *Journal of Reproductive Biotechnology and Fertility* 5 (2016) (vid. pág. 25).

[40] Reza Moradi Rad y col. "Blastomere cell counting and centroid localization in microscopic images of human embryo". En: *2018 IEEE 20th international workshop on multimedia signal processing (MMSP)*. IEEE. 2018, págs. 1-6 (vid. pág. 25).

[41] Reza Moradi Rad y col. "Coarse-to-fine texture analysis for inner cell mass identification in human blastocyst microscopic images". En: *2017 Seventh International Conference on*

Image Processing Theory, Tools and Applications (IPTA). IEEE. 2017, págs. 1-5 (vid. pág. 26).

[42] Amarjot Singh y col. "Automatic segmentation of trophectoderm in microscopic images of human blastocysts". En: *IEEE Transactions on Biomedical Engineering* 62.1 (2014), págs. 382-393 (vid. pág. 26).

[43] Shakiba Kheradmand y col. "Inner cell mass segmentation in human hmc embryo images using fully convolutional network". En: *2017 IEEE International Conference on Image Processing (ICIP)*. IEEE. 2017, págs. 1752-1756 (vid. pág. 26).

[44] Reza Moradi Rad y col. "Human Blastocyst's Zona Pellucida segmentation via boosting ensemble of complementary learning". En: *Informatics in Medicine Unlocked* 13 (2018), págs. 112-121 (vid. pág. 26).

[45] Reza Moradi Rad y col. "Predicting Human Embryos' Implantation Outcome from a Single Blastocyst Image". En: *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE. 2019, págs. 920-924 (vid. págs. 26, 46, 47, 49, 50, 52).

[46] Anna Targosz y col. "Semantic segmentation of human oocyte images using deep neural networks". En: *BioMedical Engineering OnLine* 20.1 (2021), págs. 1-26 (vid. págs. 26, 46, 47).

[47] Pegah Khosravi y col. "Deep learning enables robust assessment and selection of human blastocysts after in vitro fertilization". En: *NPJ digital medicine* 2.1 (2019), págs. 1-9 (vid. págs. 26, 52).

[48] Prudhvi Thirumalaraju y col. "Evaluation of deep convolutional neural networks in classifying human embryo images based on their morphological quality". En: *Heliyon* 7.2 (2021), e06298 (vid. págs. 26, 49, 50).

[49] Olaf Ronneberger, Philipp Fischer y Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation". En: *International Conference on Medical image computing and computer-assisted intervention*. Springer. 2015, págs. 234-241 (vid. pág. 32).

[50] Michal Drozdzal y col. "The importance of skip connections in biomedical image segmentation". En: *Deep learning and data labeling for medical applications*. Springer, 2016, págs. 179-187 (vid. pág. 32).

[51] Germán García. "Deep Learning en segmentación de imagen médica". Tesis de mtría. Universitat Politècnica de València, 2017 (vid. pág. 35).

[52] Diederik P Kingma y Jimmy Ba. "Adam: A method for stochastic optimization". En: (2014) (vid. págs. 36, 43).

[53] Aboozar Taherkhani, Georgina Cosma y T Martin McGinnity. "AdaBoost-CNN: An adaptive boosting algorithm for convolutional neural networks to classify multi-class imbalanced datasets using transfer learning". En: *Neurocomputing* 404 (2020), págs. 351-366 (vid. págs. 39-41, 49).

[54] François Chollet. "Xception: Deep learning with depthwise separable convolutions". En: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, págs. 1251-1258 (vid. págs. 39, 40).

[55] Yoav Freund y Robert E Schapire. "A decision-theoretic generalization of on-line learning and an application to boosting". En: *Journal of computer and system sciences* 55.1 (1997), págs. 119-139 (vid. pág. 40).

[56] Abhijit Guha Roy, Nassir Navab y Christian Wachinger. "Concurrent spatial and channel 'squeeze & excitation'in fully convolutional networks". En: *International conference on medical image computing and computer-assisted intervention*. Springer. 2018, págs. 421-429 (vid. pág. 52).