
UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA

FACULTAD DE INGENIERÍA, ARQUITECTURA Y DISEÑO

BIOINGENIERÍA



Predicción de la viabilidad de embriones a partir de imágenes microscópicas del tiempo de vida de fluorescencia utilizando redes neuronales convolucionales.

TESIS PRESENTADA POR

ARANZA JULIA RENEE DUARTE ARIZA

DIRECTORA

DRA.DORA LUZ FLORES

UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA

FACULTAD DE INGENIERÍA, ARQUITECTURA Y DISEÑO BIOINGENIERÍA

Predicción de la viabilidad de embriones a partir de imágenes microscópicas del tiempo de vida de fluorescencia utilizando redes neuronales convolucionales.

TESIS

Para cubrir los requisitos necesarios para obtener el título de

Bioingeniero

Presenta:

Aranza Julia Renee Duarte Ariza

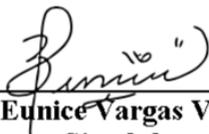
Aprobada por:



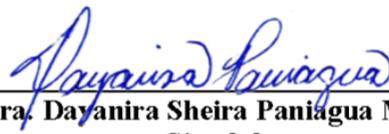
Dra. Dora Luz Flores Gutiérrez
Codirectora



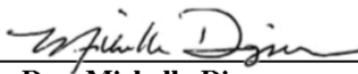
Dr Alexander Vallmitjana Lees
Codirector



Dra. Eunice Vargas Viveros
Sinodal



Dra. Dayanira Sheira Paniagua Meza
Sinodal



Dra. Michelle Digman
Sinodal

A mi familia

AGRADECIMIENTOS

Esta tesis no sería posible sin el apoyo de mis padres, Fanny Ariza Ampudia y Juan Carlos Duarte Valencia, que me han inspirado a ser la persona que soy hoy y que me han proporcionado todas y cada una de las herramientas para aprovechar las oportunidades que me ha presentado la vida. Desde el principio hasta el día de hoy me han apoyado en todos mis proyectos, especialmente durante mi carrera profesional, proporcionándome el sustento económico y emocional necesario para alcanzar mis metas. Sin ellos no habría llegado tan lejos, les debo todo. A mis hermanos, por su amor apache que me saco las sonrisas necesarias para superar los momentos difíciles. A mis abuelos y tía, Silvia Ampudia Ruedas, Alberto Diaz Mata y Verónica Ariza, quienes me inspiraron a superarme intelectualmente para conocer el mundo y sus maravillas y quienes al igual que mis padres me han apoyado en todo sentido posible. Al resto de mi familia que me han apoyado con su amor. A mi novio Maximilian, por apoyarme emocional y mentalmente para superar los aspectos más difíciles de este proyecto.

Le doy gracias a mi increíble directora de tesis, la Dra. Dora Luz Flores, quien no solamente es un ejemplo a seguir como una mujer exitosa en el campo de la ciencia e ingeniería, pero también una persona extremadamente paciente y bondadosa quien me ha apoyado en este proyecto en cada momento. Su guía y su consejo fueron clave para poder lograr esta tesis y ha sido todo un honor poder trabajar a su lado y aprender de ella. Le agradezco al Dr. Alex Vallmitjana quien me apoyo en todo momento en el desarrollo del código de mi tesis y quien sin su guía y experiencia no habría podido obtener los resultados presentados en esta tesis. Le doy las gracias también a la Dra. Michelle Digman quien me abrió un espacio en su equipo de laboratorio para desarrollar mi tesis y me puso en contacto con el Dr. Alex.

Le agradezco al resto de mi comité, la Dra., Eunice Vargas y la Dra. Dayanira Paniagua, quienes me ayudaron a pulir los detalles de este trabajo de tesis.

Le agradezco finalmente a la UABC y a la FIAD quienes me otorgaron los conocimientos y recursos necesarios para poder llegar hasta aquí, así como a la Universidad de California, Irvine quienes establecieron las bases de este trabajo.

RESUMEN

Durante las últimas décadas, la medicina ha acogido a la Inteligencia Artificial como una herramienta que le permite mejorar la calidad del cuidado de la salud. Desde el desarrollo de algoritmos que auxilian a los profesionales a identificar mutaciones genéticas hasta el correcto y temprano diagnóstico, tratamiento y predicción de enfermedades mediante el análisis de la información clínica del paciente, estas técnicas han revolucionado a la medicina al ofrecer una experiencia más precisa e individualizada que nunca. En este trabajo de tesis, se desarrolla un modelo computacional usando técnicas de aprendizaje automatizado para predecir la viabilidad y calidad de embriones pre-implantados a partir del análisis fasorial de imágenes microscópicas del tiempo de vida de fluorescencia (FLIM) con el fin de mejorar las estadísticas de éxito en la fertilización *in vitro*. El modelo fue desarrollado por medio de una red neuronal convolucional que clasifica binariamente imágenes obtenidas a partir del análisis fasorial FLIM de embriones sanos y no sanos. Trabajos anteriores sugieren que el análisis fasorial de FLIM puede detectar firmas espectroscópicas de cambios metabólicos en cada etapa del desarrollo del embrión y a partir de éstas es posible determinar una trayectoria en el espacio fasor, la cual puede ser utilizada como criterio para identificar embriones saludables. Los datos resultantes de este análisis fueron categorizados en embriones sanos y no sanos. A partir de esta clasificación previamente hecha, se pre-procesaron los datos para obtener de este análisis tres valores esenciales en la determinación de la viabilidad y calidad del embrión preimplantado antes de ser transformados en imágenes y alimentados a la red neuronal convolucional en un 70% para el entrenamiento de la misma. Se utilizó un 10% para la validación inicial del modelo y el restante 20% de los datos para evaluar la exactitud y precisión del modelo en general. . Después de haber desarrollado el modelo, este fue entrenado y validado mediante dos diferentes métodos; por medio de diez corridas y por validación cruzada. Finalmente, y mediante el detallado análisis y comparación de las métricas de evaluación resultantes de cada uno, se llegó a la conclusión que el método de validación cruzada produjo el mejor rendimiento en general. Este método dividió la base de datos en diez partes iguales para utilizar nueve de ellas, es decir 563 imágenes, para el entrenamiento y ajuste de los parámetros y la parte restante de 63 imágenes para el proceso de

validación de los datos. El modelo obtuvo una exactitud de 96.03%, una precisión del 99.70% y una especificidad del 99.64%.

ABSTRACT

During the last decades, medicine has embraced Artificial Intelligence (AI) as a tool that allows the improvement of the quality of healthcare. From the development of algorithms that help professionals identify genetic mutations to the correct and early diagnosis, treatment and prediction of diseases through the analysis of the patient's clinical information, these techniques have revolutionized medicine by offering a more accurate and personalized experience. In this thesis work, a computational model is developed using automated learning techniques to predict the viability and quality of pre-implanted embryos from the phasor analysis of fluorescence lifetime microscopic images (FLIM) in order to improve the success statistics in *in vitro* fertilization (IVF). The model was developed using a convolutional neural network that does binary classification on images obtained from the FLIM phasor analysis of healthy and non-healthy embryos. Previous work suggests that the FLIM phasor analysis can detect spectroscopic signatures of metabolic changes at each stage of embryo development. From this information it is possible to determine a trajectory which can be used as criteria to identify healthy embryos. The data resulting from this analysis were categorized into healthy and unhealthy embryos. From this previously done classification, the data were pre-processed to obtain from this analysis three essential values in determining the viability and quality of the pre-implanted embryo before being transformed into grayscale images and fed to the convolutional neural network by 70% for the training process. Of the data, 10% was used for the validation of the training model and the remaining 20% of the data was used to evaluate the accuracy and precision of the model in general. After experimentation with different network designs, it was decided to work with a model of two hidden layers, each with a convolution layer, a pooling layer and finally a fully connected layer. Once the computational model was built, experimentation with the input data shape and the hyperparameters of the network was performed to fine-tune the machine learning parameters in order to have high accuracy in the training and validation process.

ÍNDICE

AGRADECIMIENTOS	4
RESUMEN	6
ABSTRACT	8
ÍNDICE DE TABLAS	11
ÍNDICE DE FIGURAS	12
ABREVIACIONES	13
ESTRUCTURA DE LA TESIS	2
1. INTRODUCCIÓN	3
1.1 ANTECEDENTES	3
1.1.1 <i>FECUNDACIÓN IN VITRO</i>	3
1.1.2 <i>MICROSCOPÍA DE IMÁGENES DEL TIEMPO DE VIDA DE FLUORESCENCIA</i>	5
1.1.3 <i>ANÁLISIS FASORIAL</i>	8
1.1.4 <i>EVALUACIÓN DE LA CALIDAD DE EMBRIONES MEDIANTE EL ANÁLISIS FASORIAL DE FLIM.</i>	9
1.1.5 <i>REDES NEURONALES CONVOLUCIONALES</i>	11
1.1.6 <i>ESTRUCTURA DE UNA RED NEURONAL CONVOLUCIONAL</i>	12
1.1.7 <i>CASO DE COMPARACIÓN</i>	18
1.2 HIPÓTESIS	18
1.3 JUSTIFICACIÓN	19
1.4 OBJETIVOS	19
1.4.1 <i>OBJETIVO GENERAL</i>	19
1.4.2 <i>OBJETIVOS ESPECÍFICOS</i>	20
2.METODOLOGÍA	21
2.1 CONJUNTO DE DATOS	21
2.2 PRE-PROCESAMIENTO DE DATOS.	21
2.3 CONSTRUCCIÓN DEL MODELO	23
2.4 MÉTRICAS DE EVALUACIÓN	23

3.RESULTADOS	27
3.1 ESTRUCTURA DEL MODELO	27
3.2 RESULTADOS DE LAS MÉTRICAS DE EVALUACIÓN EN DIEZ CORRIDAS.	29
3.2.1 <i>MATRIZ DE CONFUSIÓN</i>	30
3.2.2 <i>EXACTITUD EN EL CONJUNTO DE ENTRENAMIENTO</i>	30
3.2.3 <i>EXACTITUD EN EL CONJUNTO DE VALIDACIÓN</i>	31
3.2.4 <i>EXACTITUD EN EL CONJUNTO DE PRUEBA</i>	32
3.2.5 <i>PRECISIÓN, SENSIBILIDAD Y ESPECIFICIDAD.</i>	34
3.2.6 <i>ROC-AUC</i>	36
3.3 RESULTADOS CON VALIDACIÓN CRUZADA <i>K-FOLD</i>.	37
3.3.1 <i>MATRIZ DE CONFUSIÓN</i>	37
3.3.2 <i>EXACTITUD</i>	38
3.3.3 <i>PRECISIÓN, SENSIBILIDAD, ESPECIFICIDAD</i>	39
3.4 COMPARACIÓN DEL MODELO CON DIEZ CORRIDAS Y VALIDACIÓN CRUZADA.	40
3.4.1 <i>EXACTITUD</i>	41
3.4.2 <i>PRECISIÓN</i>	41
3.4.3 <i>SENSIBILIDAD</i>	42
3.4.4 <i>ESPECIFICIDAD</i>	43
4.DISCUSIÓN	45
4.1 PRE-PROCESAMIENTO DE DATOS	45
4.2 DISEÑO Y CONSTRUCCIÓN DEL MODELO	45
4.3 COMPARACIÓN DE MÉTODOS DE CLASIFICACIÓN.	46
4.4 EVALUACIÓN DE EMBRIONES	46
4.5 COMPARACIÓN CON EL CASO DE ESTUDIO	47
5.CONCLUSIONES	48
6.REFERENCIAS	49
7.APENDÍCES	51
A. CÓDIGO FUENTE PARA EL MODELO DE CLASIFICACIÓN BINARIA POR UNA RED NEURONAL CONVOLUCIONAL POR DIEZ CORRIDAS.	51
B. CÓDIGO FUENTE PARA EL MODELO DE CLASIFICACIÓN BINARIA POR UNA RED NEURONAL CONVOLUCIONAL POR VALIDACIÓN CRUZADA	55
C. EXPERIMENTACIÓN CON LAS CAPAS Y NODOS DE LA CNN	58

ÍNDICE DE TABLAS

TABLA 1 ESTADÍSTICAS DE ÉXITO DE LA FECUNDACIÓN IN VITRO SEGÚN LA EDAD	5
TABLA 2 EXPERIMENTACION INICIAL EN EL DISEÑO DEL MODELO	27
TABLA 3 HIPERPARAMETROS DE LAS CAPAS OCULTAS	28
TABLA 4 RESULTADOS DE LAS METRICAS DE EVALUACION EN DIEZ CORRIDAS	29
TABLA 5 METRICAS DE VALIDACION CRUZADA 10-FOLD.....	38

ÍNDICE DE FIGURAS

FIGURA 1. 1 DIAGRAMA JABLONSKI : <i>RUTAS DE DE-EXCITACIÓN</i>	7
FIGURA 1. 2 RETARDO DE FASE Y MODULACIÓN <i>EN EL DOMINIO DE LA FRECUENCIA</i>	7
FIGURA 1. 3 EL ESPACIO VECTORIAL DE LA TRAMA FASORIAL Y EL CÍRCULO UNIVERSAL	9
FIGURA 1. 4 TRAYECTORIA D	10
FIGURA 1. 5 ESTRUCTURA BÁSICA DE UNA RED NEURONAL CONVOLUCIONAL BINARIA	12
FIGURA 1. 6 OPERACIÓN DE CONVOLUCIÓN EXPLICADA GRÁFICAMENTE.....	13
FIGURA 1. 7 RELLENO O PADDING 'SAME'	15
FIGURA 1. 8 REPRESENTACIÓN GRAFICA DE LA FUNCIÓN DE ACTIVACIÓN UNIDAD LINEAL RECTIFICADA	15
FIGURA 1. 9 REPRESENTACIÓN GRAFICA DE LA FUNCIÓN DE ACTIVACIÓN SIGMOIDE.	16
FIGURA 1. 10 MAX POOLING Y AVERAGE POOLING	17
FIGURA 2. 1 METODOLOGÍA A SEGUIR PARA EL DISEÑO Y CONSTRUCCIÓN DEL MODELO DE CLASIFICACIÓN BINARIA.	21
FIGURA 2. 2 IMAGEN 'PNG' DE UNA MUESTRA SALUDABLE Y NO SALUDABLE RESPECTIVAMENTE.	22
FIGURA 2. 3 CURVA AUC-ROC.....	26
FIGURA 3. 1 RED NEURONAL CONVOLUCIONAL BINARIA.	29
FIGURA 3. 2 MATRIZ DE CONFUSIÓN.....	30
FIGURA 3.2.1 EXACTITUD DEL CONJUNTO DE ENTRENAMIENTO.....	31
FIGURA 3.2.2 EXACTITUD DEL CONJUNTO VALIDACIÓN.	32
FIGURA 3.2.3 EXACTITUD DEL CONJUNTO DE PRUEBA.....	33
FIGURA 3.2.4 COMPARACIÓN DE EXACTITUD EN LOS CONJUNTOS DE ENTRENAMIENTO, VALIDACIÓN Y PRUEBA.....	33
FIGURA 3.2.5 PRECISIÓN PROMEDIO DE LAS 10 CORRIDAS.	34
FIGURA 3.2.6 SENSIBILIDAD EL MODELO.	35
FIGURA 3.2.7. ESPECIFICIDAD.....	35
FIGURA 3.2.8 PRECISIÓN, SENSIBILIDAD, ESPECIFICIDAD Y F1	36
FIGURA 3.2.9 PROMEDIO DE MÉTRICA ROC-AUC (IZQUIERDA) Y CURVA ROC AUC DEL MEJOR MODELO	37
FIGURA 3. 1 RED NEURONAL CONVOLUCIONAL BINARIA.	29
FIGURA 3. 2 MATRIZ DE CONFUSIÓN.....	30
FIGURA 3.3.1 MATRIZ DE CONFUSIÓN DE VALIDACIÓN CRUZADA CON 10-FOLD.....	38
FIGURA 3.3.2 EXACTITUD DE VALIDACIÓN CRUZADA 10 FOLD.....	39
FIGURA 3.3.3 MÉTRICAS DE EVALUACIÓN DE VALORACIÓN CRUZADA 10-FOLD	40
FIGURA 3.4.1 COMPARACIÓN EN LA MÉTRICA DE EXACTITUD	41
FIGURA 3.4.2 COMPARACIÓN EN LA MÉTRICA DE PRECISIÓN	42
FIGURA 3.4.3 COMPARACIÓN EN LA MÉTRICA DE SENSIBILIDAD.	43
FIGURA 3.4.4 COMPARACIÓN EN LA MÉTRICA DE ESPECIFICIDAD	43

ABREVIACIONES

H Healthy

UH Unhealthy

FIV Fertilización in vitro

FSH Hormona estimulante del folículo

hCG Gonadotropina coriónica humana

SHO Síndrome de hiper estimulación ovárica

FLIM Microscopía de imágenes del tiempo de vida de fluorescencia

TLM Time Lapse Microscopy

CNN Red Neuronal Convolutacional

TP True Positives

TN True Negatives

FP False Positives

FN False Negatives

API Interfaz de programación de aplicaciones

TPR Tasa de positivos verdaderos

FPR Tasa de falsos positivos

ESTRUCTURA DE LA TESIS

En el Capítulo 1: Introducción, se ofrece un preámbulo a los temas relacionados con este trabajo de tesis. Primeramente, se discute tanto los beneficios como las áreas de mejora en la técnica de fertilización in vitro por ende estableciendo la justificación de este trabajo. Seguido de esto se entra en materia de la microscopía de imágenes del tiempo de vida de fluorescencia (FLIM) y su enfoque fasorial. De especial importancia, se describe a grandes rasgos el estudio “Evaluación sin etiquetas de la calidad del embrión antes de la implantación mediante el análisis fasorial de FLIM” [10], el cual fue previamente hecho a este trabajo de tesis y sirve como fundamento del mismo. Después se describe el modelo de redes neuronales convolucionales, sus ventajas en el contexto de la inteligencia artificial y su estructura particular. Finalmente se establece la justificación, hipótesis y objetivos generales y específicos del presente trabajo.

En el Capítulo 2: Metodología, se describen todos los pasos y consideraciones que sigue este trabajo, desde el pretratamiento de los datos crudos hasta el diseño y construcción de un modelo de clasificación binaria que alcance los objetivos establecidos en el Capítulo 1.

En el Capítulo 3: Resultados, se pueden los valores resultantes después de haber seguido la metodología indicada en el Capítulo 2. Los resultados obtenidos están divididos en dos métodos de valoración, el primero describe una validación por medio de diferentes métricas de evaluación durante diez corridas y el segundo describe el método de validación cruzada con diez *fold*s. Los valores obtenidos son presentados en tablas y gráficamente descritos en figuras.

En el Capítulo 4: Discusión, se presenta un análisis de los resultados obtenidos en el capítulo previo y las implicaciones que sugiere cada uno de ellos. Se comparan los dos métodos utilizados en este trabajo y se establece el que tuvo un mejor desempeño.

En el Capítulo 5: Conclusión, se presentan las resoluciones finales del trabajo y se establece si se cumplieron los objetivos definidos en el primer capítulo.

1. INTRODUCCIÓN

1.1 ANTECEDENTES

1.1.1 FECUNDACIÓN *IN VITRO*

La fecundación *in vitro* (FIV) es una serie de procedimientos complejos utilizados para mejorar la fertilidad, prevenir los problemas genéticos, y para ayudar en la concepción. Durante la fecundación *in vitro*, se recolectan óvulos maduros de los ovarios y se fecundan con espermatozoides en un laboratorio. Después, el óvulo o los óvulos fecundados (embrión o embriones) se implantan en un útero. Un ciclo completo de fecundación *in vitro* lleva alrededor de tres semanas y a veces, estos pasos se dividen en diferentes partes y el proceso puede tomar más tiempo [1].

Existen cinco pasos básicos en la fecundación *in vitro* [2]:

1. Estimulación o superovulación: A la mujer se le administran fármacos con el fin de incrementar la producción de óvulos en un ciclo. Durante este paso, la mujer será sometida a ultrasonidos vaginales regulares para examinar los ovarios y a análisis de sangre para verificar los niveles hormonales.
2. Retiro del óvulo: La aspiración folicular, es una operación menor en la que se retiran los óvulos del cuerpo de la mujer. A la mujer se le administran medicamentos de tal manera que no sienta dolor durante el procedimiento y utilizando imágenes de ultrasonido como guía, el proveedor de atención médica introduce una aguja delgada en la hacia el ovario y los sacos (folículos) que contienen los óvulos. La aguja se conecta a un dispositivo de succión, que extrae los óvulos y el líquido fuera del folículo, uno a la vez. El procedimiento entonces se repite para el otro ovario.
3. Inseminación y fecundación: El espermatozoide del hombre se coloca junto con los óvulos de mejor calidad. La mezcla de espermatozoide y óvulo se denomina inseminación. Los óvulos y el espermatozoide luego se almacenan en una cámara ambientalmente controlada. Generalmente, el

espermatozoide fecunda un óvulo en unas cuantas horas después de la inseminación, si el médico piensa que la probabilidad de fecundación es baja, se puede inyectar directamente el espermatozoide dentro del óvulo, lo cual se denomina inyección intracitoplásmica de espermatozoides (ICSI, por sus siglas en inglés).

4. Cultivo del embrión: Cuando el óvulo fecundado se divide, se convierte en un embrión. El personal de laboratorio lo vigilará regularmente para asegurarse de que esté creciendo de manera apropiada. En aproximadamente 5 días, el embrión tiene varias células que se están dividiendo activamente.

5. Transferencia del embrión: Los embriones son colocados dentro del útero de la mujer de tres a cinco días después del retiro y fecundación del óvulo. El médico introduce una sonda delgada (catéter) que contiene los embriones. Si un embrión se implanta en el revestimiento del útero y empieza a crecer, se puede considerar un embarazo. Se puede colocar más de un embrión dentro de la vagina al mismo tiempo, lo cual puede llevar a embarazos múltiples. El número exacto de embriones transferidos depende de muchos factores, especialmente la edad de la mujer.

Aunque la fertilización *in vitro* es una técnica revolucionaria, es un proceso largo y difícil el cual tiene efectos secundarios. Durante la primera etapa, hay dos medicamentos principales que se utilizan durante un ciclo de FIV para inducir la ovulación: la hormona estimulante del folículo (FSH) y la gonadotropina coriónica humana (hCG).

Estas hormonas llegan a tener efectos secundarios como náusea, dolor abdominal, sensibilidad en los pechos y existe una pequeña posibilidad de causar síndrome de hiperestimulación ovárica (SHO). Este síndrome de hiperestimulación ovárica (SHO) es una respuesta excesiva a la ingesta de medicamentos (especialmente gonadotropinas inyectables) que se utilizan para hacer crecer los óvulos. El SHO provoca en una mujer una gran cantidad de folículos en crecimiento junto con niveles altos de estradiol. Esto provoca una fuga de líquido hacia el abdomen (barriga), lo que puede provocar distensión abdominal, náuseas e hinchazón del abdomen. El SHO suele ser leve y

provoca síntomas como hinchazón, náuseas y aumento de peso. En casos raros, el SHO es lo suficientemente grave como para requerir tratamiento en el hospital [3].

Las tasas de éxito de la fertilización *in vitro* dependen en distintos factores, entre ellos la edad, los antecedentes reproductivos y la causa de infertilidad. En la Tabla 1 se muestran las estadísticas de éxito según la edad [4].

TABLA 1 ESTADÍSTICAS DE ÉXITO DE LA FECUNDACIÓN IN VITRO SEGÚN LA EDAD

<i>Edad</i>	<i>Porcentaje de éxito</i>
<i>Menor de 35 años</i>	54.5%
<i>Entre 35 a 37 años</i>	41.1%
<i>Entre 38 a 40 años</i>	26.7%
<i>Entre 41 y 42 años</i>	13.8%
<i>Mayor de 43 años</i>	4.2%

En embriología clínica, la IA puede proporcionar un método objetivo para evaluar las imágenes de embriones humanos, lo que permite la identificación de las características clave del desarrollo de la viabilidad del embrión.

1.1.2 MICROSCOPÍA DE IMÁGENES DEL TIEMPO DE VIDA DE FLUORESCENCIA

La microscopía de imágenes del tiempo de vida de fluorescencia (FLIM), aprovecha la propiedad de vida útil de la fluorescencia y es una técnica de microscopía que ha ganado popularidad debido a su alta sensibilidad al entorno molecular y los cambios en la conformación molecular. La FLIM de moléculas autofluorescentes proporciona información única sobre la salud celular de una manera no destructiva y a menudo se utiliza para estudiar animales vivos y como mecanismo de

contraste para la cirugía guiada por fluorescencia. Las moléculas fluorescentes exógenas, actúan como sensores, capaces de monitorear parámetros microambientales, como la temperatura, viscosidad, pH y concentración de iones. Las interacciones proteína-proteína se pueden monitorear utilizando sensores de transferencia de energía por resonancia (FRET) de Förster que son específicos para la señalización celular, la proliferación celular, la citocinesis y otras interacciones moleculares. Por lo tanto, al aprovechar los fluoróforos endógenos y exógenos, FLIM puede monitorear procesos en células y tejidos, incluida la progresión de una enfermedad y la eficacia de los fármacos [5].

Cuando una molécula en estado fundamental (denotado como S_0) absorbe luz de energía igual o mayor que los niveles de energía superiores (S_1 , S_2 , ..., S_n), un electrón se excita a un nivel de energía más alto durante un período corto. El electrón experimentará una relajación vibratoria hasta el nivel vibratorio más bajo del estado excitado (denotado como S_1) por un proceso no radiativo llamado conversión interna. Desde el estado electrónico S_1 , las moléculas regresan al estado fundamental ya sea por un proceso radiativo o no radiativo. La Figura 1 representa los diferentes fenómenos de luminiscencia que ocurren en estos niveles [6].

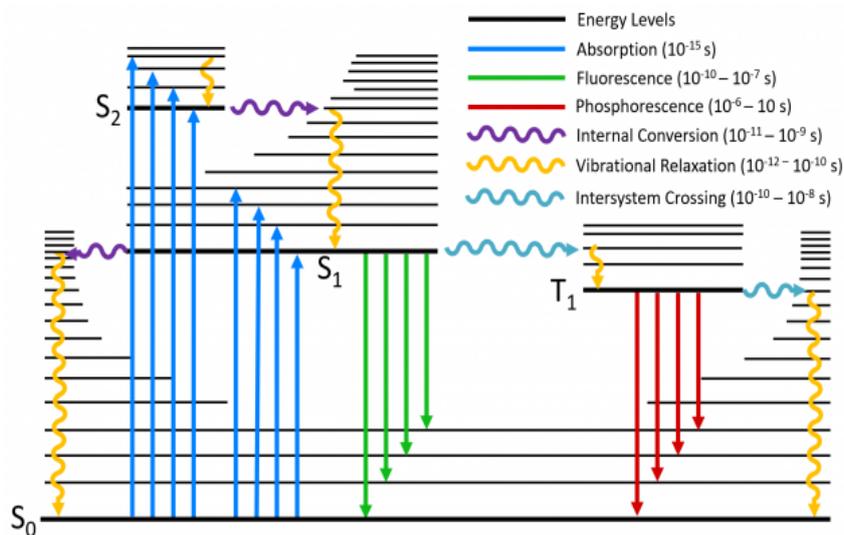


Figura 1. 1 Diagrama Jablonski : *Rutas de de-excitación, FLIM mide la tasa de caída de la fluorescencia de un fluoróforo en una escala de tiempo de subnanosegundos a cientos de nanosegundos. Como referencia, la luz viaja a una velocidad de 3×10^8 m/s [7].*

Las mediciones del tiempo de vida de fluorescencia en el dominio del tiempo utilizan un pulso corto de luz para la excitación y luego registran la desintegración exponencial de las moléculas fluorescentes, ya sea directamente (es decir, mediante detección controlada o muestreo de pulsos) o utilizando dispositivos electrónicos de resolución temporal que agrupan los fotones por sus tiempos de llegada.

Por otro lado, las técnicas de dominio de frecuencia también pueden medir el tiempo de vida de fluorescencia. Aquí, la excitación es continua con modulación de amplitud a lo largo del tiempo como una onda sinusoidal como se observa en la Figura 2 [8].

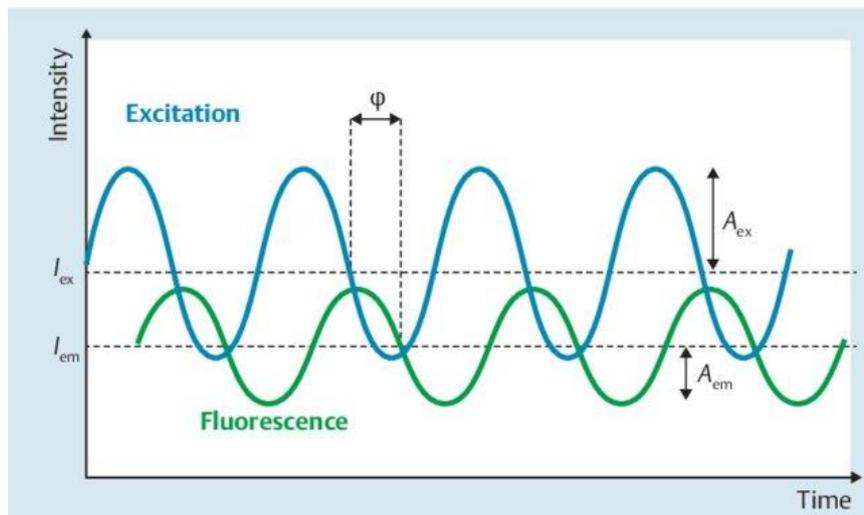


Figura 1. 2 Retardo de fase y modulación *en el dominio de la frecuencia* [9].

La señal de fluorescencia cambia de fase y amplitud con respecto a la onda de excitación. El retardo de fase (ϕ) y la modulación de amplitud (m) de un fluoróforo se visualizan trazando los

cambios de fase en un rango de frecuencias de modulación. Esta señal sinusoidal de fluorescencia resultante se puede demodular en el dominio de la frecuencia para cuantificar el retraso inducido por la disminución exponencial de la intensidad de la fluorescencia.

1.1.3 ANÁLISIS FASORIAL

El diagrama fasorial es una representación gráfica de todos los datos de FLIM en un espacio vectorial; es decir, cada pixel de una imagen FLIM se transforma en un punto del diagrama fasorial. No se hace ninguna suposición sobre el número de tasas de descomposición presentes en el medio ambiente, así como sobre el modelo específico del decaimiento (exponencial, no exponencial).

Esta transformación de los pixeles de una imagen FLIM en el diagrama fasorial se puede aplicar igualmente a los datos FLIM en el dominio del tiempo y de la frecuencia [9].

El espacio fasorial se construye utilizando dos vectores fasoriales (g , s), donde cada componente se representa en el dominio del tiempo con las ecuaciones (1) y (2) y en el dominio de la frecuencia, con en las ecuaciones (3) y (4).

$$g_{x,y}(\omega) = \int_0^{\infty} I_{x,y}(t) \cos(\omega t) dt / \int_0^{\infty} I_{x,y}(t) dt \quad (1)$$

$$s_{x,y}(\omega) = \int_0^{\infty} I_{x,y}(t) \sin(\omega t) dt / \int_0^{\infty} I_{x,y}(t) dt \quad (2)$$

$$g_{x,y}(\omega) = m_{x,y} \cos(\varphi_{x,y}) \quad (3)$$

$$s_{x,y}(\omega) = m_{x,y} \sin(\varphi_{x,y}) \quad (4)$$

donde $m_{x,y}$ y $\varphi_{x,y}$ son la relación de modulación y el retardo de fase medido dada una frecuencia de modulación particular (ω) en una ubicación de pixel (x, y) como se observa en la Figura 1.3.

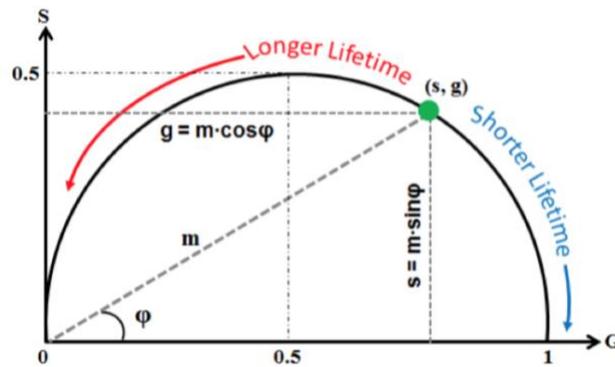


Figura 1. 3 El espacio vectorial de la trama fasorial y el círculo universal [9].

A partir de esto, la vida útil de cualquier decaimiento exponencial individual se muestra en una curva de semicírculo centrada en ($g = 0.5, s = 0$) con un radio de 0.5 en el diagrama fasorial. La trayectoria se llama Círculo Universal. En él, los tiempos de vida de fluorescencia más largos tienden a mostrarse a la izquierda y los más cortos a la derecha.

1.1.4 EVALUACIÓN DE LA CALIDAD DE EMBRIONES MEDIANTE EL ANÁLISIS FASORIAL DE FLIM.

Los métodos actuales de evaluación de embriones incluyen la observación estática de la morfología del embrión y/o la evaluación de los datos de TLM (*Time Lapse Microscopy*) morfocinéticos combinados con la morfología del blastocito. Estas evaluaciones dependen en gran medida de la experiencia y el conocimiento de los embriólogos, aunque estos análisis pueden ser muy subjetivos [10].

En la evaluación sin etiquetas de la calidad del embrión antes de la implantación mediante el análisis fasorial de FLIM (Ning Ma et al.), se aplicó el análisis fasorial de FLIM para examinar y cuantificar los cambios de los biomarcadores fluorescentes endógenos durante el desarrollo del embrión pre implantación para correlacionarlos con la viabilidad del embrión. Esto se utilizó como calibre morfológico para determinar la calidad del embrión.

Se observó que los embriones inicialmente absorben piruvato como su principal fuente de energía. A medida que los embriones se desarrollan en etapas posteriores, aumenta la necesidad de ATP para activar la transcripción. Luego los embriones pasan de la glucólisis a la fosforilación oxidativa, utilizando principalmente glucosa como fuente de energía, lo que también cambia el potencial redox relativo (relación NAD + NADH). Las firmas espectroscópicas de cada uno de estos cambios son detectadas y pueden ser usadas como criterio para identificar embriones sanos y no sanos. A esta trayectoria en el espacio fasor se le llamo Trayectoria D.

Se encontró que la trayectoria D (Figura 1.4) de los embriones pre implantados cultivados en medios carentes de nutrientes (flecha azul) se desviaba significativamente de la de los medios normales (flecha roja), lo que indica que está trayectoria puede ser exitosamente utilizada para detectar alteraciones metabólicas en los embriones.

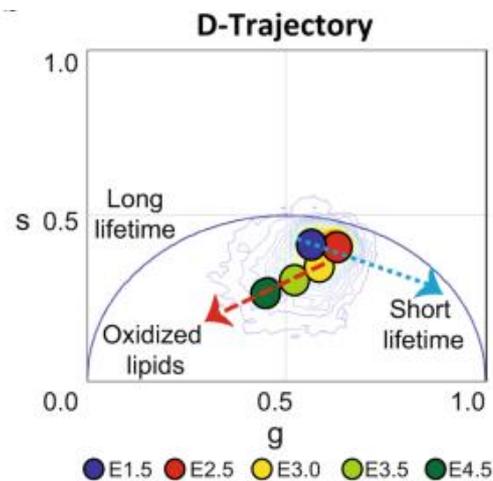


Figura 1. 4 Trayectoria D establecida en Evaluación sin etiquetas de la calidad del embrión antes de la implantación mediante el análisis fasorial de FLIM [10].

La determinación del potencial del análisis fasorial-FLIM como predictor del correcto desarrollo de embriones preimplantación se dio en varios pasos; primero realizaron imágenes fasoriales-FLIM de embriones de la etapa de 2 células durante por una lapso de tiempo de alrededor de 60 horas para identificar la etapa más deseable para predecir el potencial de desarrollo de los embriones. Al final del período de cultivo de 60 horas, se clasificaron a los embriones como sanos (H) si alcanzaron la etapa normal de blastocisto expandido completo que muestra un ICM (inner cell mas) apretado y células trofectodermo con forma de epitelio cohesivo, o no sanos (NH) si los

embriones se detuvieron antes de alcanzar la etapa de blastocisto o mostrar características morfológicas anormales como tener una cavidad de blastocele más pequeña y / o un límite celular irregular en el blastocisto.

Se aplicó un programa informático llamado *Distance Analysis*, un clasificador basado en máquinas de soporte vectorial (SVM) que utilizó parámetros espectroscópicos de histogramas fasoriales 3D de embriones y se llegó a la conclusión que su índice de evolución embriológica (EVI) es un índice cuantitativo no morfológico que puede proporcionar información útil sobre la calidad de la pre-implantación embriones.

Esto sugiere que se puede aprender a identificar exitosamente con herramientas informáticas los embriones de alta o baja calidad por medio del análisis fasorial de FLIM.

1.1.5 REDES NEURONALES CONVOLUCIONALES

Una red neuronal artificial es un modelo que emula el modo en que el cerebro humano procesa información, funciona simulando un número elevado de unidades de procesamiento interconectadas que funcionan como versiones abstractas de neuronas.

Este modelo computacional parte de grandes conjuntos de datos complejos, con los que se entrena la red y se ajustan los parámetros de aprendizaje (pesos y sesgos), hacia la obtención de un resultado en específico, siendo altamente predictivo y en gran medida preciso.

Dentro de los modelos de redes neuronales artificiales se encuentran las redes neuronales convolucionales (CNN), un tipo de modelo de aprendizaje profundo que sirve para procesar datos que tienen un patrón de cuadrícula - como lo son las imágenes- y que está diseñado para aprender de forma automática y adaptativa patrones espaciales en las imágenes, de bajo a alto nivel. Este modelo es una construcción matemática que generalmente se compone de tres tipos de capas (o bloques de construcción): convolución, agrupación y capas completamente conectadas [11]. La estructura básica de CNN se puede apreciar en la Figura 1.6.

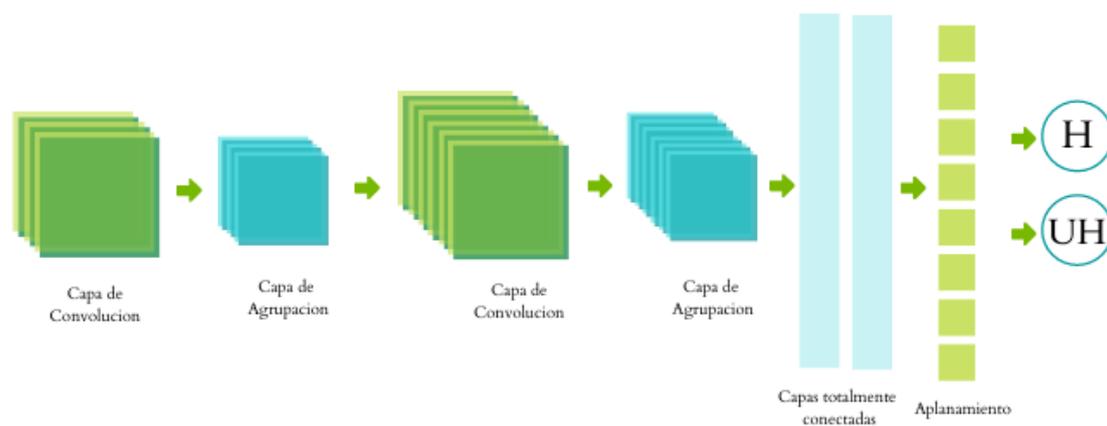


Figura 1. 5 Estructura básica de una red neuronal convolucional binaria, el resultado puede ser saludable (“H”) o no saludable (“UH”).

1.1.6 ESTRUCTURA DE UNA RED NEURONAL CONVOLUCIONAL

Las dos primeras capas --de convolución y agrupación-- realizan la extracción de características, mientras que la tercera es una capa completamente conectada y mapea las características extraídas en la salida final como una clasificación.

En las imágenes digitales, los valores de los píxeles se almacenan en una cuadrícula bidimensional (2D), es decir, una matriz de números. En la convolución, una pequeña cuadrícula de valores llamada filtro (o *kernel*), se traslapa en cada posición de la imagen multiplicando los valores y sumándolos todos para obtener el nuevo valor de ese punto. Esta operación se aplica en cada posición de la imagen para obtener la convolución (Figura 1.6). A medida que una capa alimenta su salida a la siguiente, las entidades extraídas pueden volverse más complejas de forma jerárquica y progresiva. El proceso de optimización de parámetros como los *kernels* se denomina entrenamiento, el cual se realiza a través de un algoritmo de optimización llamado retropropagación [12]. Las etapas de una red neuronal convolucional se describen a continuación [13]:

1. Capa de entrada:

La capa de entrada es un tensor con forma:

Capa de Entrada

$$= \text{No. imagenes} \times \text{Altura de las imagenes} \times \text{Ancho de las imagenes} \\ \times \text{Canales}$$

En esencia, simplemente proporciona la forma de las imágenes de entrada y no hay parámetros que puedan aprender.

2. Capa de Convolución:

La capa de la convolución es una parte fundamental de la estructura de la CNN.

En la Figura 1.6 se puede observar la operación de la convolución de manera gráfica. En esta capa se realiza una operación lineal que se utiliza para la extracción de características, donde se aplica una pequeña matriz de números, llamada filtro a través de la imagen de entrada, que es una matriz de números llamada tensor. Se calcula un producto punto por cada elemento del filtro y el tensor de entrada en cada ubicación del tensor y se suma para obtener el valor de salida en la posición correspondiente del tensor de salida, esto nos crea un mapa de características.

A este filtro se le llama parámetro, unos valores que son automáticamente optimizados durante el proceso de entrenamiento. Este procedimiento se repite aplicando múltiples filtros para formar un número arbitrario de mapas de características, que representan diferentes características de los tensores de entrada. A los filtros se les puede asignar una cantidad, una altura y un ancho. A esto se le llaman hiperparámetros, variables asignadas antes del entrenamiento.

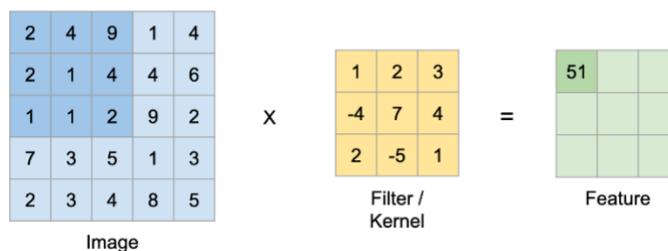


Figura 1. 6 Operación de Convolución explicada gráficamente [14].

Para calcular los parámetros que se pueden aprender aquí, se multiplica por la forma del ancho (w), la altura (h), los filtros de la capa anterior (f) y los filtros (k) en la capa actual. También es necesario agregar el término sesgo o *bias* (b), un término que se optimiza en el proceso de entrenamiento.

El número de parámetros en una capa CONV sería:

$$\begin{aligned}
 & \textit{Capa de Convolucion} \\
 & = ((\textit{Ancho} \times \textit{Altura} \times \textit{Filtros de la capa anterior}) \\
 & \quad + \textit{Bias}) \times \textit{Filtros de la capa actual}
 \end{aligned}$$

3. Padding y Stride

En la operación de convolución descrita hay dos tipos de resultados:

El primero es que el mapa de características convolucionado se reduce en dimensionalidad ya que la operación no permite que el centro de cada filtro se superponga al elemento más externo del tensor de entrada así reduciendo la altura y el ancho del mapa de características de salida en comparación con el tensor de entrada. Por otro lado, si se aplica una técnica de relleno llamada 'padding' la dimensionalidad puede permanecer igual o aumentar.

En el primer caso se usa el relleno o padding 'válido' el cual no aplica ningún tipo de relleno y en el segundo caso se aplica el relleno 'same' (Figura 1.7) en el cual se agregan filas y columnas de ceros a cada lado del tensor de entrada, para que quepa el centro de un filtro en el elemento más externo y mantenga la misma dimensión a través de la operación de convolución. Mientras que los valores no cambian, el tamaño lo hace.

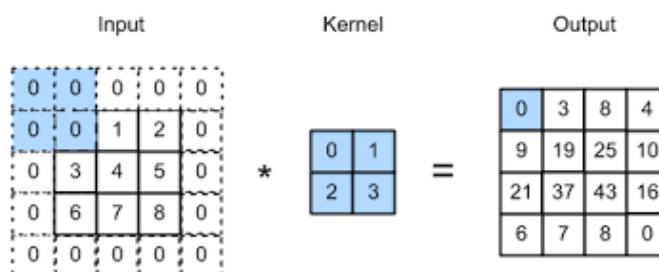


Figura 1. 7 Relleno o padding 'same' [15].

La distancia entre dos posiciones sucesivas del filtro se llama zancada o 'Stride' y la elección común de una zancada es 1.

4. Función de Activación

Después de la operación de la convolución, su salida tiene que ser pasada a través de una función de activación no lineal. Las funciones de activación no lineal más comunes son la unidad lineal rectificadora (ReLU) y la sigmoide [16].

La función de activación ReLU permite el paso de todos los valores positivos sin cambiarlos, pero asigna todos los valores negativos a cero como se puede observar en la Figura 1.8.

Se define como:

$$f(\mathbf{z}) = \max(0, z) \quad (3)$$

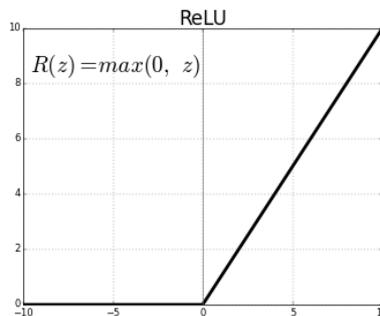


Figura 1. 8 Representación grafica de la función de activación unidad lineal rectificadora

La función de activación sigmoide actúa como una especie de función "aplastadora", comprimiendo la salida a un rango de 0 a 1. Para valores negativos grandes de z, el término e^{-z} en el denominador crece exponencialmente, y $\sigma(z)$ se aproxima a 0. Al contrario, valores positivos grandes de z reducen e^{-z} hacia 0, y $\sigma(z)$ se aproxima a 1 (Fig.1.9).

Se define como:

$$\sigma(z) = \frac{1}{1+e^{-z}} \quad (4)$$

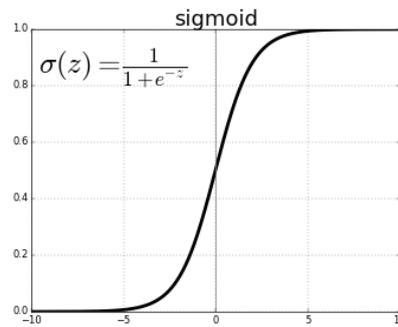


Figura 1. 9 Representación grafica de la función de activación sigmoide.

5. Capa de Agrupación

Una capa de agrupación, como se observa en la Figura 1.10, proporciona una operación de reducción de la dimensionalidad en el plano de los mapas de características para introducir una invariancia de traducción a pequeños cambios y distorsiones, y reducir el número de parámetros que se aprenden posteriormente. Cabe señalar que no hay ningún parámetro que se pueda aprender en ninguna de las capas de agrupación, mientras que el tamaño del filtro, el paso y el relleno son hiperparámetros en las operaciones de agrupación, similares a las operaciones de convolución.

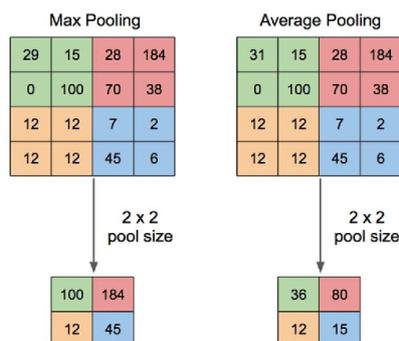


Figura 1. 10 Max pooling y Average Pooling [17].

Existen dos tipos de agrupación:

El primero y mas popular es el llamado 'Max Pooling' o agrupación máxima, que extrae parches de los mapas de características de entrada, genera el valor máximo en cada parche y descarta todos los demás valores.

El segundo llamado agrupación media mundial realiza un tipo extremo de reducción de resolución, donde un mapa de características con un tamaño de alto x ancho se reduce a una matriz de 1×1 simplemente tomando el promedio de todos los elementos en cada mapa de características, mientras que la profundidad de los mapas de características es retenida.

6. Capa totalmente conectada

Los mapas de características de salida de la convolución final o la capa de agrupación generalmente se aplanan, es decir, se transforman en una matriz unidimensional (1D) de números y se conectan a una o más capas completamente conectadas, también conocidas como capas densas en el que cada entrada está conectada a cada salida por un peso ajustado.

Una vez que se crean las características extraídas por las capas de convolución y sub-muestreadas por las capas de agrupación, un subconjunto de capas completamente conectadas las asigna a las salidas finales de la red, como las probabilidades de cada clase en las tareas de clasificación. La capa final completamente conectada normalmente tiene el mismo número de nodos de salida que el número de clases.

Esta capa, en comparación con las otras capas, tiene el mayor número de parámetros, ya que cada neurona está conectada a todas las demás.

El número de parámetros aquí es:

$((\text{Neuronas de la capa actual} \times \text{neuronas de la capa anterior}) + \text{Bias}) \times \text{Neuronas de la capa actual}$.

Capa Totalmente Conectada

$$= ((\text{Filtros de la capa actual} \times \text{Filtros de la capa anterior}) + \text{Bias}) \\ \times \text{Filtros de la capa actual}$$

1.1.7 CASO DE COMPARACIÓN

En un estudio hecho por Scripps Research Translational Institute, se utilizaron imágenes de lapso de tiempo de 10.148 embriones humanos, obtenidas del Centro de Medicina Reproductiva de WeillCornell Medicine para capacitar y validar una *deep neural network* (DNN). Los 10,148 embriones se clasificaron por calidad en tres grandes grupos: de buena calidad (n = 1345 embriones), de calidad regular (n = 4,062 embriones) y de mala calidad (n = 4,741 embriones). Las categorizaciones de clases de embriones se basaron en el estado de desarrollo de los embriones alcanzado a las 113 hpi (horas después de la infección). Se obtuvieron imágenes en distintos lapsos de tiempo de cada uno de los embriones, siete profundidades focales por punto de tiempo e imágenes en blanco y negro de 500 × 500 píxeles por profundidad focal. Tras el pre-procesamiento y la eliminación de imágenes con problemas de legibilidad y la selección aleatoria de un conjunto equilibrado de imágenes, quedaron con un total de 12,001 imágenes: 6,000 imágenes de embriones de buena calidad y 6,001 imágenes de embriones de mala calidad. Luego, se entreno un algoritmo basado en DNN de Inception-V1 utilizando los dos grupos de calidad en ambos extremos del espectro, es decir, de buena calidad y de mala calidad. La arquitectura Inception-V1 es un algoritmo de aprendizaje de transferencia, donde inicialmente se realizó un ajuste fino de los parámetros para todas las capas. Se usaron 50,000 pasos para entrenar la DNN y posteriormente se evaluó el desempeño de la DNN (llamado STORK) usando un conjunto de prueba independiente seleccionado al azar con 964 imágenes de embriones de buena calidad y 966 imágenes de embriones de mala calidad. El algoritmo de entrenamiento fue capaz de identificar imágenes de buena y mala calidad con una precisión del 96,94% (1871 predicciones correctas de 1930 imágenes) [18].

1.2 HIPÓTESIS

Usando redes neuronales convolucionales se pueden detectar patrones de morfología embrionaria asociados con su viabilidad que permiten predecir el potencial de implantación.

1.3 JUSTIFICACIÓN

La fecundación *in vitro* es una técnica de reproducción asistida que no solo proporciona una solución a problemas de infertilidad, pero también promueve la formación de familias no tradicionales. La fecundación *in vitro* tiene una larga trayectoria que ha sido perfeccionada continuamente durante los últimos años sin embargo permanece siendo un largo y costoso proceso que no exime riesgos tanto físicos como emocionales. Las tasas de éxito continúan siendo totalmente dependientes de distintos factores y incluso en los mejores casos la tasa de éxito promedio es de alrededor del 50%. Uno de los factores de mas importancia es el estado del embrión ya que se ha comprobado que la transferencia de embriones que están desarrollados saludablemente está relacionada con índices de embarazo exitosos más altos. Sin embargo, la selección correcta de embriones saludables depende enteramente en la experiencia del técnico y tiende a variar de un laboratorio a otro. La inteligencia artificial y los modelos de aprendizaje automatizado junto con el análisis fasorial de FLIM proporcionan un enfoque innovador para la selección de embriones saludables. Esto se traduce en un posible incremento en la tasa de concepción, así como en la tasa de parto de un bebe sano. Si tomamos en cuenta que la probabilidad promedio de tener un bebé es solo 30% por cada ciclo, se puede entender la magnitud y el alcance que puede suponer este enfoque.

1.4 OBJETIVOS

1.4.1 OBJETIVO GENERAL

Creación de un modelo computacional basado en redes neuronales convolucionales para la evaluación de la viabilidad embrionaria antes de la implantación a partir de datos de microscopía de imágenes del tiempo de vida de fluorescencia.

1.4.2 OBJETIVOS ESPECÍFICOS

- Pre-procesamiento de imágenes obtenidas con el enfoque fasorial FLIM utilizando: Matlab y Python.
- Crear una base de datos de entrenamiento con las imágenes previamente procesadas y clasificadas.
- Entrenar y validar un modelo de red neuronal convolucional para predecir la clasificación de las imágenes en saludables y no saludables.
- Ajustar y mejorar los parámetros del modelo para alcanzar valores de métricas de evaluación mayor al 90%.

2.METODOLOGÍA

2.1 CONJUNTO DE DATOS

El conjunto de datos, fue obtenido del Laboratorio de Dinámica de Fluorescencia (LFD) en el Departamento de Ingeniería Biomédica de la Universidad de California, Irvine de los resultados del experimento llevado a cabo por Ning Ma et al. en [10].

Consiste en un total de 146 archivos en formato R64 de análisis fasorial de FLIM de embriones de ratones. Este formato R64, fue desarrollado en el Laboratory for Fluorescence Dynamics para guardar imágenes con valores fasoriales, en donde en cada pixel contiene tres datos correspondientes a la intensidad, fase y modulación. De estos 146 archivos, 104 corresponden a embriones que fueron determinados como sanos (H) y 42 a embriones como no sanos (UH). Esta clasificación fue obtenida después de una observación de alrededor de 60 horas de embriones en estado de 2 células; los embriones fueron etiquetados como sanos (H) si alcanzaron la etapa normal de blastocisto expandido completo o no sanos (UH) si los embriones se detuvieron antes de alcanzar la etapa de blastocisto o mostrar características morfológicas anormales. En la Figura 2.1 se puede observar la metodología usada en la construcción del modelo de clasificación binaria.

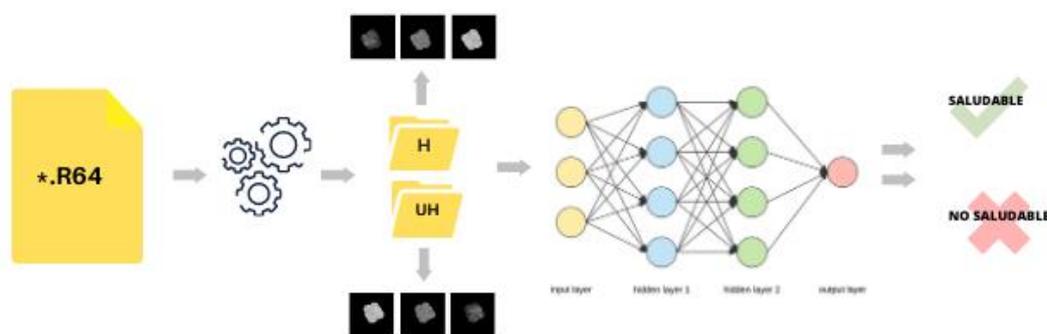


Figura 2.1 Metodología a seguir para el diseño y construcción del modelo de clasificación binaria.

2.2 PRE-PROCESAMIENTO DE DATOS.

Los archivos de análisis fasorial fueron procesados en Python y Matlab para obtener de cada archivo R64 tres matrices de 256 x 256 valores. La primera matriz representa la intensidad de la muestra (i), es decir el número de fotones contados, la segunda y tercera contienen los valores del retardo de fase (φ) y modulación de la amplitud (m) y respectivamente.

Aplicando las ecuaciones (1) y (2) del análisis fasorial FLIM usando los valores de la segunda y tercera matriz, se obtuvieron dos valores que representan la coordenada “g” y la coordenada “s” del fasor, respectivamente.

Estas matrices (intensidad, “g” y “s”) fueron transformadas utilizando la librería PIL (pillow) con el comando `Image.fromarray` que crea una imagen a partir de un objeto que exporta la interfaz de una matriz. Fue usado en modo ‘L’ el cual crea una imagen 8-bit pixeles en escala de grises y multiplicado por 300 para aumentar la calidad de las imágenes.

El resultado fueron tres imágenes ‘.png’ (Portable Network Graphics) por cada archivo R64, con dimensiones 256 x 256 x 1 ya que al ser en escala de grises cuenta con un solo canal.

Finalmente se obtuvieron en total 312 imágenes representativas de embriones saludables y 126 imágenes representativas de embriones no saludables. En la Figura 2.2 podemos observar que las dos muestras (una sana y otra no sana) son a simple vista indistinguibles la una de la otra.

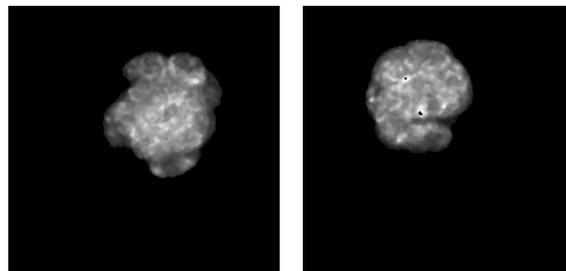


Figura 2. 2 Imagen ‘png’ de una muestra saludable y no saludable respectivamente.

Ya que la cantidad de imágenes “UH” o no saludables son considerablemente mas escasas que las saludables (312 vs 126), se balanceó la base de datos aumentando los datos representativos de

los embriones no saludables (UH) utilizando la librería ImageDataGenerator. Los primeros dos cambios realizados en la aumentación son de rango de desplazamiento de ancho y alto los cuales, como su nombre indica, recorrieron la imagen de manera horizontal y vertical respectivamente. El tercer cambio es un giro horizontal el cual refleja la imagen horizontalmente.

Finalmente obtenemos una base de datos balanceada de 628 imágenes de las cuales 312 son de embriones saludables y 316 de embriones no saludables.

2.3 CONSTRUCCIÓN DEL MODELO

Para la creación de la red neuronal convolucional se utilizó Tensorflow 2.3.0 y Keras 2.4.3.

Tensorflow es una librería de código abierto para el aprendizaje automático desarrollado por Google para construir y entrenar redes neuronales que detectan y descifran patrones y correlaciones, análogos al aprendizaje y razonamiento usados por los humanos [19]. Keras es una interfaz de programación de aplicaciones (API por sus siglas en inglés) de aprendizaje profundo escrita en Python que se ejecuta sobre la plataforma de aprendizaje automático Tensorflow. Proporciona abstracciones y bloques de construcción esenciales para desarrollar y enviar soluciones de aprendizaje automático con alta velocidad de iteración [20].

2.4 MÉTRICAS DE EVALUACIÓN

Para evaluar el desempeño de la red neuronal se utilizó el método de evaluación de k-fold cross validation, matrices de confusión, AUC-ROC, exactitud, precisión, sensibilidad y f1-score.

K-fold cross validation: La validación cruzada se utiliza principalmente para estimar la habilidad de un modelo de aprendizaje automático en datos nuevos. Es decir, se usa una muestra limitada para estimar cómo se espera que funcione el modelo en general cuando se usa para hacer predicciones sobre datos nuevos al del entrenamiento del modelo. El procedimiento tiene un solo parámetro llamado “k” que se refiere al número de grupos en los que se dividirá la base de datos. El procedimiento general es el siguiente [21]:

-
- Mezclar el conjunto de datos de forma aleatoria.
 - Dividir el conjunto de datos en 'k' grupos.
 - Tomar un grupo como conjunto de prueba.
 - Tomar los grupos restantes como conjunto de datos de entrenamiento.
 - Correr el modelo en el conjunto de entrenamiento y evaluarlo en el conjunto de prueba
 - Conservar la puntuación de la evaluación, descarta el modelo y se inicia otro grupo.
 - Repetir para múltiples combinaciones de elección del grupo de prueba.
 - Promediar la habilidad del modelo usando los puntajes de evaluación del modelo.

Matriz de Confusión: Una matriz de confusión es una herramienta que permite visualizar el desempeño de un algoritmo de aprendizaje automático. Cada columna de la matriz representa el número de predicciones de cada clase, mientras que cada fila representa las instancias en la clase real. En términos prácticos nos permite ver qué tipos de aciertos y errores está teniendo nuestro modelo a la hora de pasar por el proceso de aprendizaje con los datos [22]. Entre los valores se tiene el verdadero positivo (TP) y el verdadero negativo (TN), que representan los casos donde los embriones saludables (P) y los no saludables (N) están siendo correctamente identificados como tal. También se tienen los valores falsos positivos (FP) y falsos negativos (FN) en donde los embriones, saludables o no saludables, no están siendo correctamente identificados como su valor verdadero.

Exactitud: Es la medida de rendimiento más intuitiva y es simplemente una relación entre la observación predicha correctamente y el total de observaciones. La precisión es una gran medida, pero solo cuando tiene conjuntos de datos simétricos donde los valores de falsos positivos y falsos negativos son casi iguales. Por lo tanto, se debe observar otros parámetros para evaluar el rendimiento de su modelo. La exactitud responde a la pregunta: ¿Cuántos embriones, sean sanos o no sanos, etiquetamos correctamente?

$$\text{Exactitud} = \frac{TP+TN}{TP+FP+FN+TN} \quad (5)$$

Precisión: Es la relación entre las observaciones positivas predichas correctamente y el total de observaciones positivas predichas. Una alta precisión se relaciona con la baja tasa de falsos positivos. La precisión responde a la pregunta: ¿Cuántos de los embriones etiquetados como sanos, eran verdaderamente embriones sanos?

$$\text{Precisión} = \frac{TP}{TP+FP} \quad (6)$$

Sensibilidad: Es la proporción de observaciones positivas predichas correctamente para todas las observaciones en la clase real. La sensibilidad responde a la pregunta: ¿De todos los embriones sanos, cuantos fueron etiquetados/identificados correctamente como sanos?

$$\text{Sensibilidad} = \frac{TP}{TP+FN} \quad (7)$$

Especificidad: La especificidad es una medida que indica cuantas instancias negativas fueron etiquetadas correctamente como negativas. Responde a la pregunta: ¿De todos los embriones no sanos, cuantos fueron etiquetados/identificados correctamente?

$$\text{Especificidad} = \frac{TN}{TN+FP} \quad (8)$$

Puntuación F1: Es el promedio ponderado de precisión y recuperación. Por lo tanto, esta puntuación tiene en cuenta tanto los falsos positivos como los falsos negativos.

$$F1 = \frac{2TP}{2TP+FP+FN} \quad (9)$$

La curva AUC-ROC es una medida de rendimiento para los problemas de clasificación en varios valores de umbral. ROC es una curva de probabilidad y AUC representa el grado o medida de separabilidad. Indica cuánto es capaz el modelo de distinguir entre clases. Cuanto mayor sea el AUC, mejor será el modelo para predecir 0 como 0 y 1 como 1 [23]. Por analogía, cuanto mayor es el AUC, mejor es el modelo para distinguir entre embriones sanos y no sanos. La curva ROC se traza con la tasa de positivos verdaderos (TPR por sus siglas en ingles) contra la tasa de falsos positivos (FPR) donde TPR está en el eje “y” y FPR está en el eje “x”.

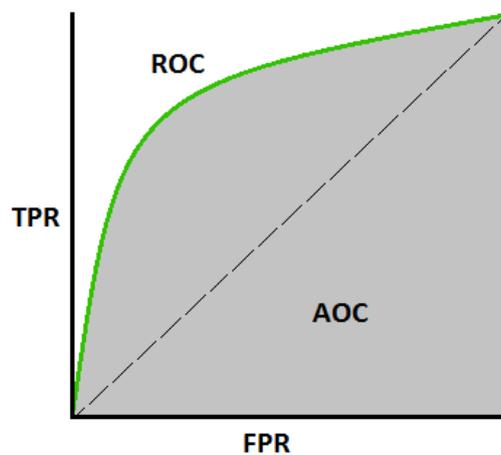


Figura 2. 3 Curva AUC-ROC

3.RESULTADOS

3.1 ESTRUCTURA DEL MODELO

Para el diseño del modelo se realizó un experimento inicial con el objetivo de encontrar la mejor estructura e hiperparámetros posibles para empezar a construirlo. El resultado del experimento, el cual consistió en variar el número de capas y nodos por cada capa fue probado en diez épocas, los resultados se pueden observar en la Tabla 2.

TABLA 2 EXPERIMENTACION INICIAL EN EL DISEÑO DEL MODELO

<i>Capa. Convolucional</i>	<i>Nodos</i>	<i>Exactitud</i>
1	32	51.999
1	64	78.399
1	128	91.2
2	32	92.399
2	64	92.5
2	128	90.399
3	32	82.400
3	64	92.399
3	128	92.499

Después de probar el código de experimentación, se obtuvieron nueve variaciones del modelo de clasificación binaria. De las nueve se seleccionaron las cuatro que tuvieron un mejor rendimiento basados en la métrica de exactitud. A partir de esto se construyeron dos modelos; el primero cuenta con dos capas de 32 y 64 nodos respectivamente y el segundo cuenta con tres capas, las primeras dos de 64 nodos y una tercera con 128 nodos. Estos modelos iniciales fueron probados en diez épocas y finalmente se encontró que lograron exactitudes similares, con el primero obteniendo una exactitud de 93.1% y el segundo de 92.3%. Sin embargo, se optó por el modelo

de dos capas ya que este corrió en un tiempo de cinco minutos con 33 segundos versus los diez minutos con 53 segundos que le tomo al segundo y necesito menos memoria computacional.

Finalmente se obtuvo una CNN que cuenta con dos capas ocultas, de forma 3- n_1 - n_2 -1, donde la entrada son los datos de entrenamiento divididos previamente en saludables y no saludables, n_1 y n_2 representan el número de neuronas en la primera y segunda capa oculta y la salida es la clasificación binaria donde 0 significa 'No Saludable' y 1 significa 'Saludable'.

Los hiperparámetros seleccionados para las capas ocultas se pueden apreciar en la Tabla 3.

TABLA 3 HIPERPARAMETROS DE LAS CAPAS OCULTAS

	<i>Neuronas (n) o filtros</i>	<i>Dimensión de los filtros</i>	<i>Relleno 'padding'</i>	<i>o Agrupación 'pooling'</i>	<i>Dimensión de la agrupación</i>	<i>Función de Activación</i>
<i>Primera capa oculta</i>	32	3,3	'same'	MaxPooling	2,2	ReLU
<i>Segunda capa oculta</i>	64	3,3	'same'	MaxPooling	2,2	ReLU

En la Figura 3.1 se observa la representación grafica del modelo creado, el modelo cuenta con dos capas ocultas, cada una con una capa de convolución, una capa de agrupación y finalmente una capa totalmente conectada, seguida de una función de aplanamiento y una capa densa para finalizar con la función de activación sigmoide.

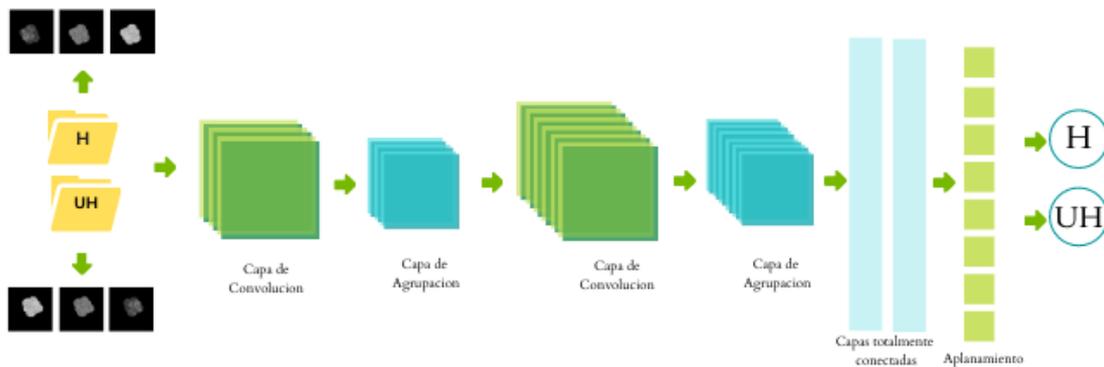


Figura 3. 1 Red neuronal convolucional binaria.

3.2 RESULTADOS DE LAS MÉTRICAS DE EVALUACIÓN EN DIEZ CORRIDAS.

Una vez construida la arquitectura del modelo, se experimentó con los hiperparámetros hasta llegar al modelo final. Este mostró el mejor desempeño al ser valorado por distintas métricas de evaluación a través del promedio de diez corridas.

A continuación, se muestran los resultados obtenidos del modelo computacional de clasificación binaria final. A partir de las diez corridas se obtuvo el promedio y la desviación estándar. Señalado en verde está la iteración en la que el modelo mostro el mejor desempeño.

TABLA 4 RESULTADOS DE LAS METRICAS DE EVALUACION EN DIEZ CORRIDAS

	<i>Exactitud de Entrenamiento</i>	<i>Exactitud de Validacion</i>	<i>Exactitud de Prueba</i>	<i>Precision</i>	<i>Sensibilidad</i>	<i>Especificidad</i>	<i>F1</i>	<i>ROC AUC</i>
1	95.57	93.40	96.18	92.94	100	93.65	96.34	0.98
2	100	92.96	92.80	95.24	90.90	94.91	93.02	0.99
3	99.77	96.83	93.60	92.93	100	86.20	94.36	0.96
4	100	90.48	93.6	91.55	97.01	89.65	94.20	0.94
5	99.7	98.41	95.2	90.90	100	90.76	95.23	0.94
6	99.09	88.89	90.40	87.3	93.22	90.76	90.16	0.9
7	99.32	93.65	95.2	94.90	100	93.10	95.24	0.96
8	100	88.89	91.2	87.14	96.82	87.87	91.72	0.96
9	99.78	95.24	96.78	94.36	100	85.48	97.10	0.98

10	100	88.89	89.69	91.37	86.88	92.18	89.07	0.96
Total	99.32	92.76	93.46	91.86	96.48	90.46	93.64	0.95
SD	1.35	3.43	2.44	2.86	4.68	3.16	2.63	0.02

3.2.1 MATRIZ DE CONFUSIÓN

En la Figura 3.2 se observa la mejor matriz de confusión obtenida después de 10 corridas. Se puede observar que la cantidad de predicciones correctas tanto positivas (H) como negativas (UH) es el 92.88% de las 125 imágenes usadas como conjunto de prueba. El error falso positivo se mantuvo alrededor de 4 embriones no sanos etiquetados como sanos en promedio. Es decir, de las 58 muestras negativas (UH) solo 4 fueron etiquetadas incorrectamente como positivas (H). Que este valor se mantenga lo mas bajo posible es de especial importancia ya que uno de los objetivos principales es identificar correctamente embriones no saludables para que estos no sean seleccionados para su implantación.

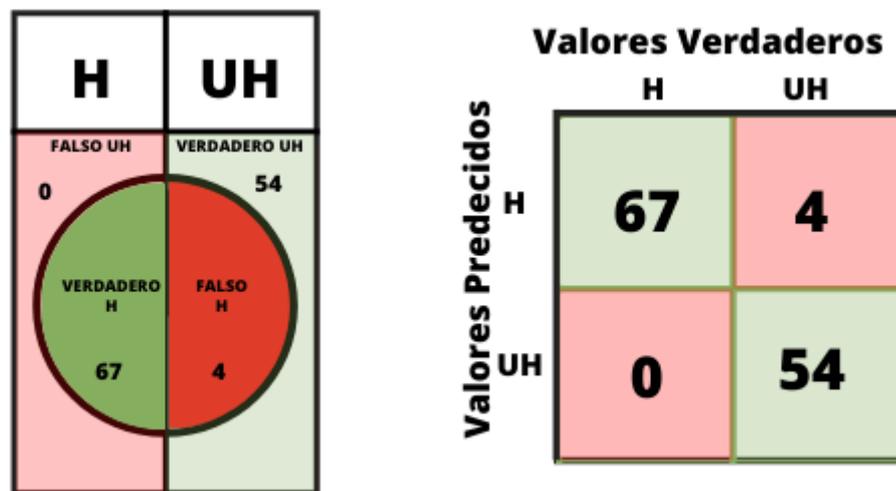


Figura 3. 2 Matriz de confusión.

3.2.2 EXACTITUD EN EL CONJUNTO DE ENTRENAMIENTO

El conjunto de entrenamiento son las muestras que fueron utilizadas para el aprendizaje del modelo, es decir, con este conjunto se ajustaron los parámetros de las capas ocultas del

clasificador binario. El algoritmo creado experimentó con distintas variables y finalmente aprendió a determinar las mejores combinaciones para un modelo predictivo óptimo. Es de gran importancia que el algoritmo no se sobreentrenara ya que el objetivo final de este es que sea adaptable a datos nuevos por lo tanto los datos fueron primero barajados. Para el entrenamiento del modelo se utilizó el 70% de la base de datos (440 imágenes) y se obtuvo un promedio de exactitud de 99.32%. En la Figura 3.21 se muestra el comportamiento de la media durante las diez corridas del entrenamiento realizadas, se puede observar que la exactitud fue mayor al 90% desde la primera corrida lo cual indica que el modelo aprendió a reconocer y clasificar los patrones existentes en las imágenes del conjunto de entrenamiento satisfactoriamente.

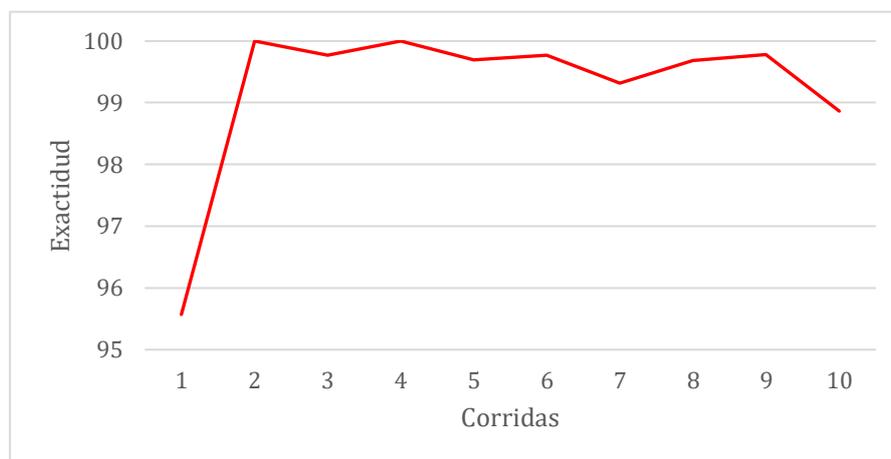


Figura 3.2.1 Exactitud del conjunto de entrenamiento.

3.2.3 EXACTITUD EN EL CONJUNTO DE VALIDACIÓN

El conjunto de validación se utilizó para ajustar los hiperparámetros, es decir, los parámetros escogidos previamente al entrenamiento. El principal objetivo es crear un modelo que funcione de manera óptima con datos nuevos por lo tanto para poder seleccionar correctamente entre las diferentes corridas que el modelo sobrelleva durante el entrenamiento es importante evaluar la función de pérdida en comparación con datos independientes o nuevos a los del conjunto de entrenamiento, es decir, los del conjunto de validación. Finalmente, la red seleccionada fue la que tuvo el mínimo error con respecto al conjunto de validación. Para este paso se utilizó un 10% de la base de datos (63 imágenes). En la Figura 3.22 se aprecia el proceso de validación durante las diez corridas. Se obtuvo una exactitud promedio de 92.76% y a partir de esto se puede determinar

que el modelo no fue sobreentrenado durante el proceso de entrenamiento y se adapta satisfactoriamente a los datos nuevos del conjunto de validación.

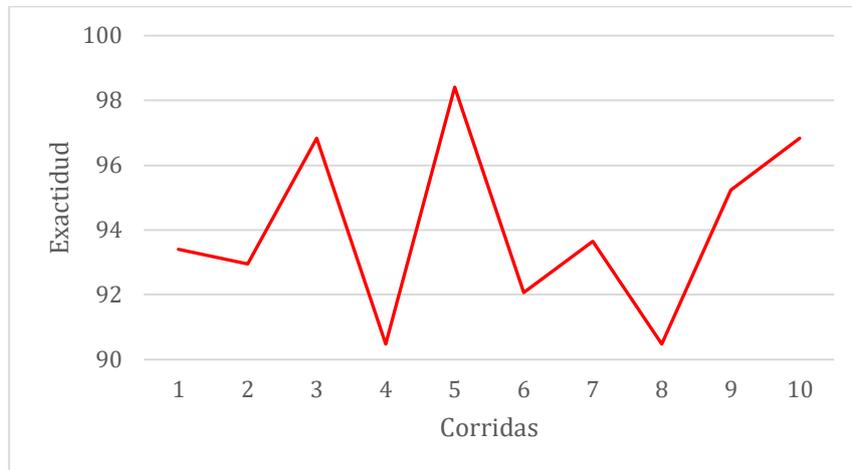


Figura 3.2.2 Exactitud del conjunto Validación.

3.2.4 EXACTITUD EN EL CONJUNTO DE PRUEBA

Este conjunto de muestras se utilizó para evaluar el rendimiento del modelo final seleccionado durante el proceso de validación. Con este conjunto se corrobora la adaptabilidad del modelo predictivo final a datos nuevos. Para las pruebas del modelo se utilizó el restante 20% de la base de datos (125 imágenes) y se obtuvo una exactitud promedio de 93.46%. En la Figura 3.23 se puede observar que el comportamiento de la exactitud de prueba durante las diez corridas es similar al de validación y el promedio final es casi idéntico. Esto demuestra que el modelo final seleccionado mediante el proceso de entrenamiento y validación, tiene un alto rendimiento prediciendo correctamente imágenes de embriones saludables de los no saludables.

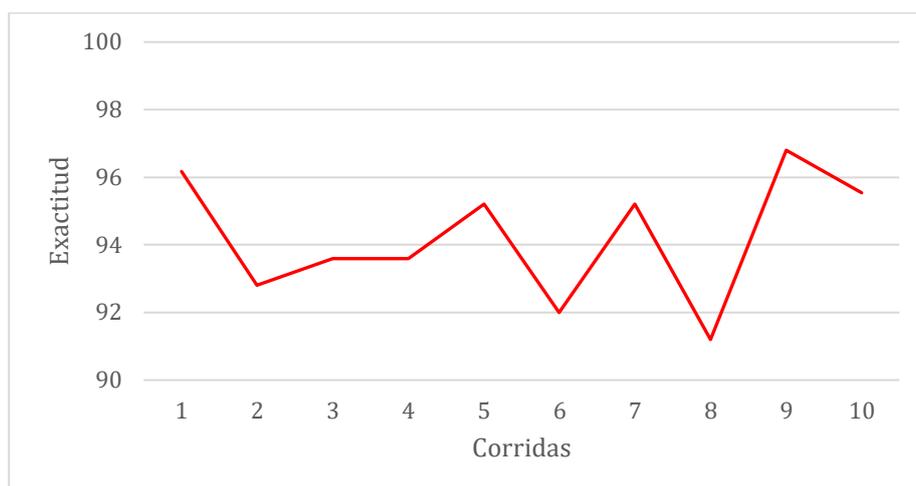


Figura 3.2.3 Exactitud del conjunto de prueba.

Comparando gráficamente las exactitudes de los conjuntos de entrenamiento, validación y prueba obtenidas mediante diez corridas en la Figura 3.24, se puede observar que la exactitud de entrenamiento está cerca del 100%, mientras que el proceso de validación como el de prueba se mantienen en el rango del 90% al 100% con un comportamiento promedio similar. Esto denota que los parámetros del modelo fueron afinados de manera satisfactoria durante el proceso de entrenamiento y para obtener una alta capacidad de adaptación y predicción de datos nuevos.

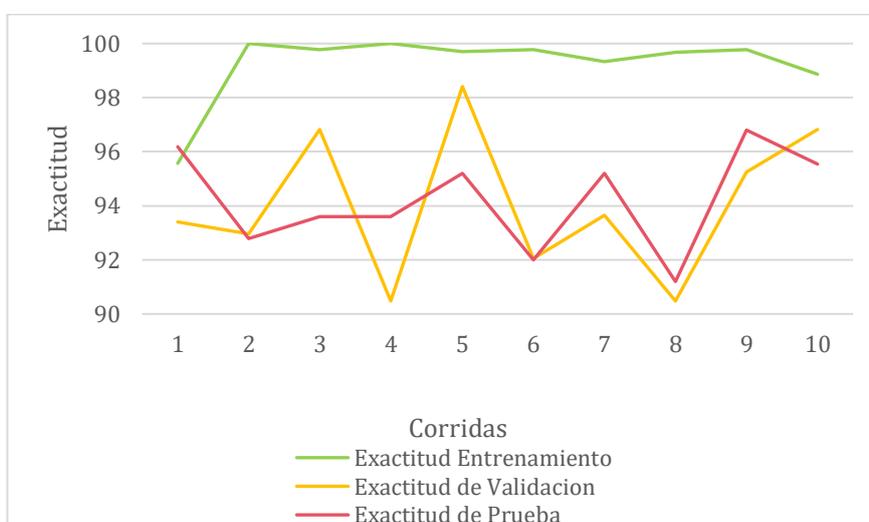


Figura 3.2.4 Comparación de exactitud en los conjuntos de entrenamiento, validación y prueba.

3.2.5 PRECISIÓN, SENSIBILIDAD Y ESPECIFICIDAD.

La precisión promedio del modelo fue de 91.46%, este porcentaje señala sobre la calidad del modelo en el proceso de clasificación. En la Figura 3.25 se puede observar el comportamiento de la precisión durante diez corridas. De las 71 muestras etiquetadas como embrión sano solo 4 en promedio fueron etiquetadas incorrectamente y de las 54 muestras etiquetadas como embrión no sano ninguna fue etiquetada incorrectamente.

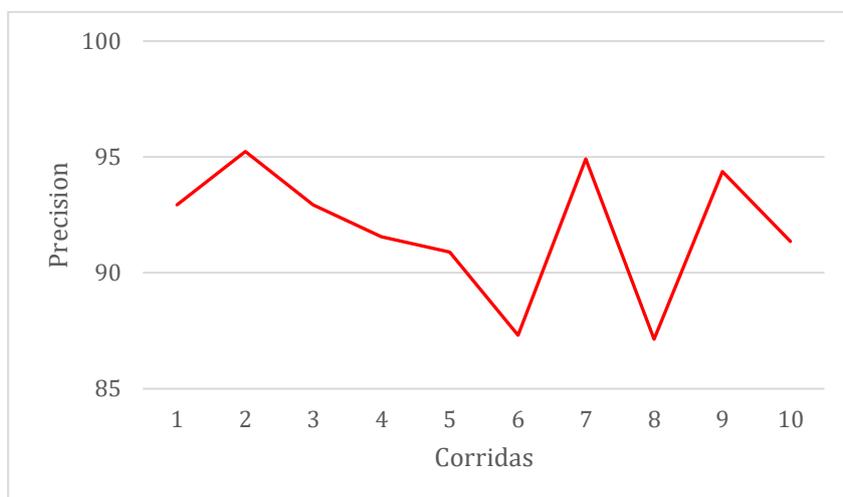


Figura 3.2.5 Precisión promedio de las 10 corridas.

La sensibilidad es de 96.48% en promedio, en la Figura 3.26 se puede apreciar claramente que este valor se mantiene en el percentil noventa. Por medio de este dato se puede determinar que los casos positivos (H) están siendo capturados casi en su totalidad. Se identificaron en su totalidad, las 67 muestras de embriones sanos.



Figura 3.2.6 Sensibilidad el modelo.

En la Figura 3.27 se observa el promedio de la métrica de especificidad durante las diez corridas. Este resulta en un porcentaje final de 90.46%, esto demuestra que el modelo está identificando satisfactoriamente las imágenes de embriones no sanos casi en su totalidad. De las 58 imágenes de embriones no sanos, 54 están siendo identificadas correctamente por el algoritmo.

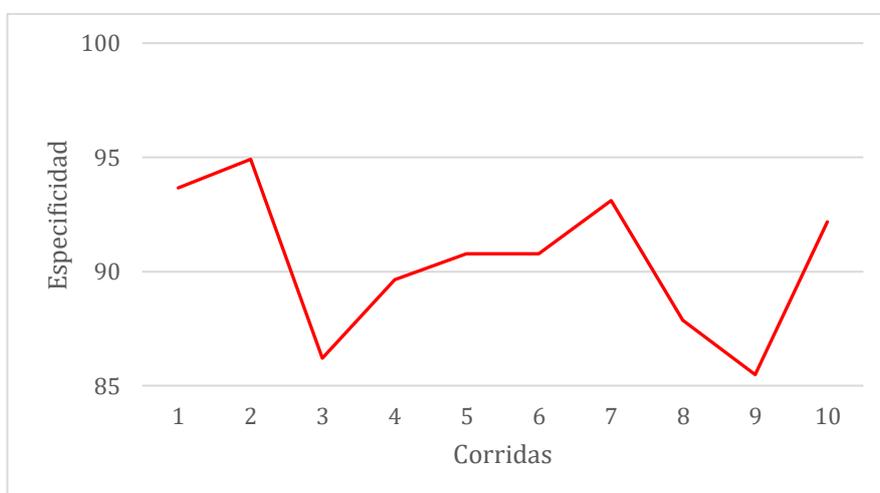


Figura 3.2.7. Especificidad

Teniendo en cuenta que el valor de precisión fue 91.5% y el valor de sensibilidad fue de 96.48%, el resultado de la métrica F1 de 94.39% en promedio, es bastante acertada al ser el armónico medio entre los dos y haber identificado en estas dos métricas correctamente y casi completamente las muestras positivas.

En la siguiente comparación grafica en la Figura 3.28 se puede apreciar que las cuatro métricas discutidas anteriormente tienen un comportamiento similar en promedio.

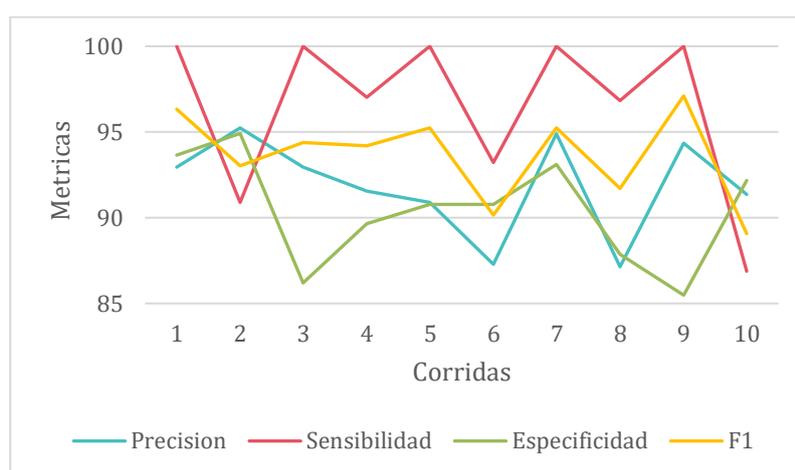


Figura 3.2.8 Precisión, Sensibilidad, Especificidad y F1.

3.2.6 ROC-AUC

En la Figura 3.29 se observa el promedio de la puntuación de ROC-AUC durante las diez corridas. En la Figura 3.29 se aprecia gráficamente la puntuación ROC-AUC de la mejor iteración. Se puede observar que la curva característica operativa del receptor se acerca a la esquina superior izquierda a lo largo del eje "y" que representa la tasa de positivos verdaderos. Esto crea un área bajo la curva extensa que concuerda con los resultados obtenidos de la sensibilidad y especificidad. Finalmente está AUC sugiere que el modelo distingue y separa satisfactoriamente entre imágenes de embriones sano y no sanos.

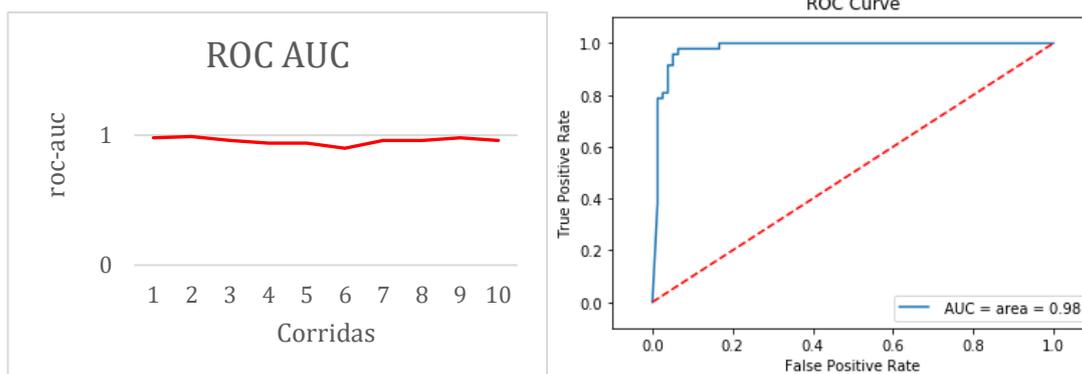


Figura 3.2.9 Promedio de métrica ROC-AUC (izquierda) y curva ROC AUC del mejor modelo (derecha).

3.3 RESULTADOS CON VALIDACIÓN CRUZADA K -FOLD.

Para la evaluación con validación cruzada K -fold se utilizó una división o “ k ” de diez-*folds*, es decir que la base de datos fue dividida en diez submuestras iguales como se muestra en la Figura 3.3. Nueve de ellas, es decir, 565 imágenes en total se utilizaron para el entrenamiento y la submuestra restante (63 imágenes) para la validación del modelo. Se realizan diez corridas donde tanto el grupo de entrenamiento como de validación cambia. Esto con el objetivo de identificar valores atípicos u otras anomalías que podrían estar solamente en un conjunto y no estén igualmente representados en otro.

3.3.1 MATRIZ DE CONFUSIÓN

En la Figura 3.31 se puede observar la mejor matriz de confusión obtenida en la Validación Cruzada 10-fold. A diferencia de la matriz obtenida por medio de 10 corridas, los falsos positivos promedio en esta evaluación son mucho menores, con solo una imagen de un embrión no sano clasificada incorrectamente como embrión sano. Adicionalmente, los falsos negativos también son escasos. A partir del conteo de los errores y los aciertos en la matriz de confusión en cada una de las clases de clasificación se puede determinar que el modelo está separando las imágenes sanas de las no sanas de manera correcta.

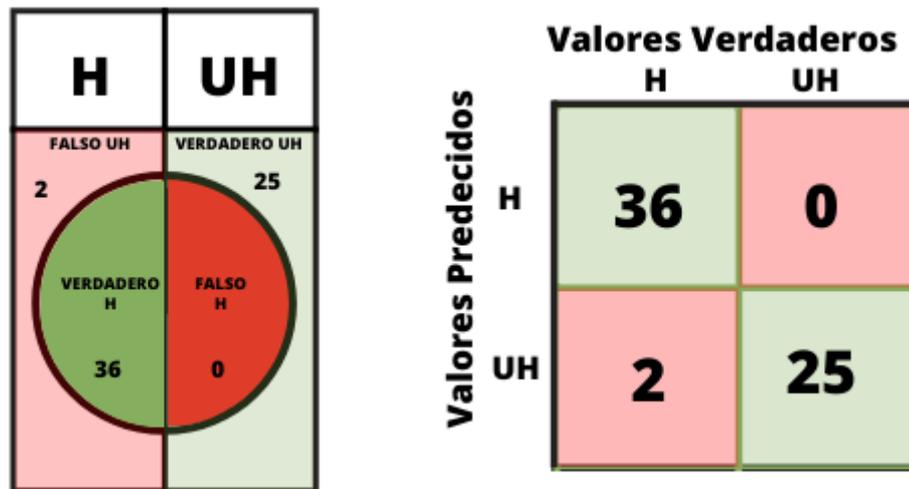


Figura 3.3.1 Matriz de Confusión de validación cruzada con 10-Fold

3.3.2 EXACTITUD

En la Tabla 5 se pueden apreciar los resultados de las métricas de evaluación obtenidas durante las 10 corridas o *fold*s de la valoración cruzada, finalmente tenemos el promedio total obtenido y la desviación estándar de cada métrica de evaluación.

TABLA 5 METRICAS DE VALIDACION CRUZADA 10-FOLD

	Exactitud	Precisión	Sensibilidad	Especificidad
Fold 1	96.82	100	94.12	100
Fold 2	93.65	100	85.71	100
Fold 3	96.82	100	93.33	100
Fold 4	98.41	100	97.37	100
Fold 5	95.24	97.06	94.28	96.43
Fold 6	95.24	100	89.65	100
Fold 7	96.82	100	94.74	100
Fold 8	96.82	100	93.75	100
Fold 9	98.41	100	96.97	100
Fold 10	92.06	100	85.30	100
Promedio	96.03	99.70	92.52	99.65
Desviación estándar	2.01	0.93	4.25	1.13

Como se observa en la Figura 3.32, se tiene una exactitud de 96.23% en promedio con una desviación estándar de 1.45, esto sugiere que el modelo está clasificando los embriones sanos y los no sanos satisfactoriamente. De las 63 muestras de prueba que se usan en cada iteración 61 imágenes tanto positivas como negativas están siendo identificadas correctamente, lo cual confirma que el algoritmo este afinado para su adaptación y predicción de datos nuevos. La desviación estándar permanece baja en la mayoría de las métricas durante el proceso, esto indica que los datos están cerca del promedio y no existe mucha varianza entre las corridas del modelo.

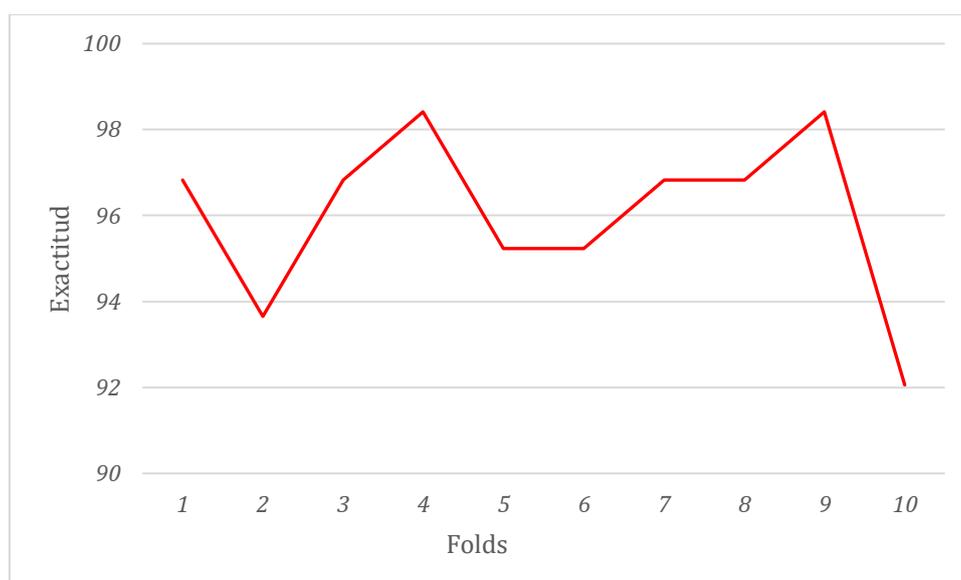


Figura 3.3.2 Exactitud de Validación Cruzada 10 Fold

3.3.3 PRECISIÓN, SENSIBILIDAD, ESPECIFICIDAD

Una precisión de 99.70% sugiere que el modelo está identificado las muestras de embriones saludables casi en su totalidad. En la matriz de confusión con el mejor rendimiento, de las 36 imágenes identificadas como embriones sanos ninguna fue identificada incorrectamente. Adicionalmente, durante los diez *folds*, solo en una iteración hubo un falso positivo. Es decir que el error falso positivo es casi inexistente en esta evaluación del modelo.

En la métrica de la sensibilidad se obtuvo un 95.52%, por medio de esto se puede determinar que las imágenes de embriones sanos están siendo etiquetadas en su mayoría. De las 38 imágenes de embriones saludables usadas en promedio para el proceso de validación solo 2 imágenes no fueron identificadas por el modelo.

La especificidad durante las diez corridas de validación cruzada indica un porcentaje de 99.64%, esto confirma que el error falso positivo es prácticamente vano, ya el modelo está identificando la mayoría de las muestras de embriones no sanos correctamente. De 25 imágenes de embriones no sanos, todas fueron correctamente etiquetadas y identificadas.

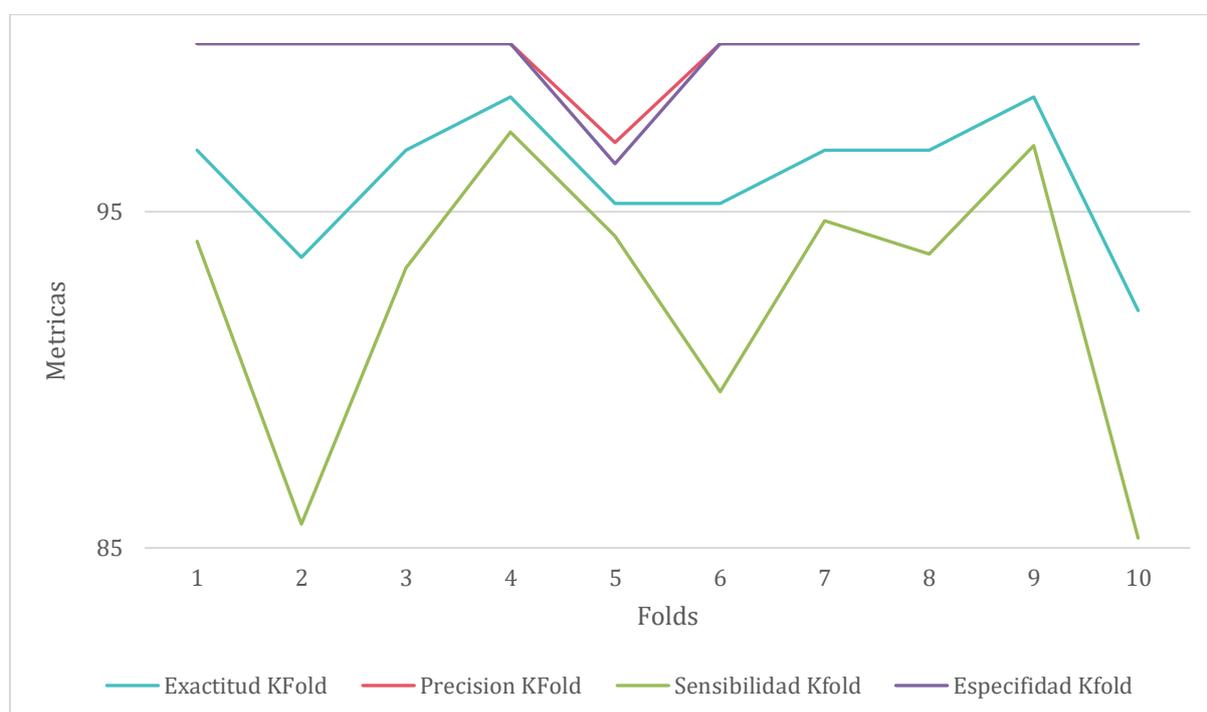


Figura 3.3.3 Métricas de Evaluación de Valoración Cruzada 10-Fold

3.4 COMPARACIÓN DEL MODELO CON DIEZ CORRIDAS Y VALIDACIÓN CRUZADA.

Con el objetivo de encontrar el mejor proceso de validación, se comparan a continuación los dos modelos previamente descritos por medio de sus métricas. El modelo por diez corridas funcionó

con un 70% de conjunto de entrenamiento, 10% de conjunto de validación y 20% de conjunto de prueba. Por otro lado, el modelo por validación cruzada con 10 *fold*s, se entrenó con el 90% de la base de datos y se validó en el 10% restante, mientras que cada *fold* cambiaba el conjunto de validación.

3.4.1 EXACTITUD

Comparando la métrica de exactitud se puede observar en la Figura 3.41 que el comportamiento de la media en la valoración cruzada 10-fold es más alta. Con un promedio de 96.93% versus 93.46%, la valoración cruzada se desempeñó mejor en la tarea de clasificación general. Lo que quiere decir que más imágenes fueron clasificadas correctamente, sean de embriones sanos o no sanos, en el modelo evaluado por validación cruzada.

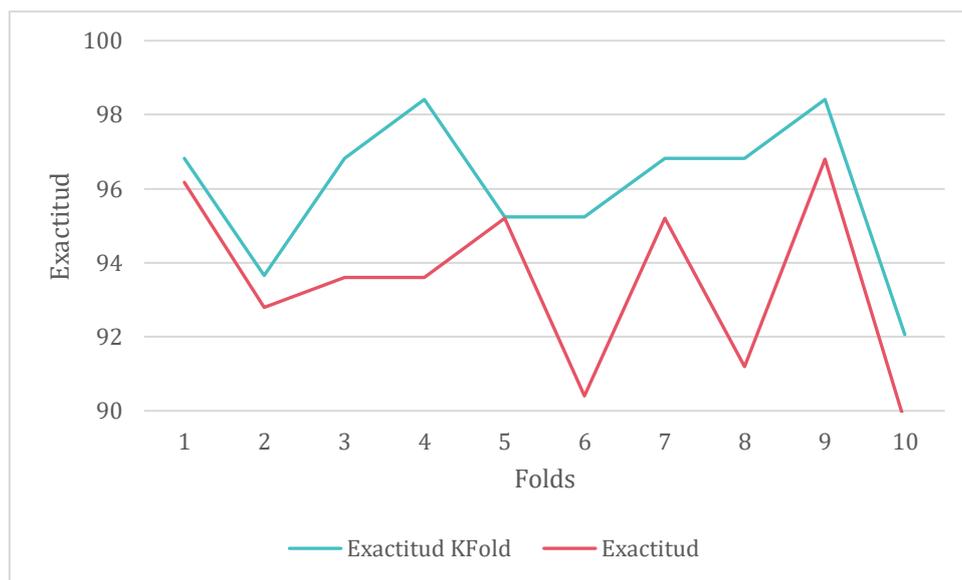


Figura 3.4.1 Comparación en la métrica de exactitud

3.4.2 PRECISIÓN

En la Figura 3.42 se puede observar que al igual que la exactitud, el modelo de validación cruzada tiene un mejor rendimiento. Con un promedio de 99.70% vs 91.86%, se puede determinar con seguridad que, en cuanto a la tarea de clasificación de imágenes de embriones saludables, el modelo entrenado por validación cruzada tuvo el mejor desempeño, con solo 1 imagen clasificada incorrectamente como saludable durante los diez *folds* realizados.

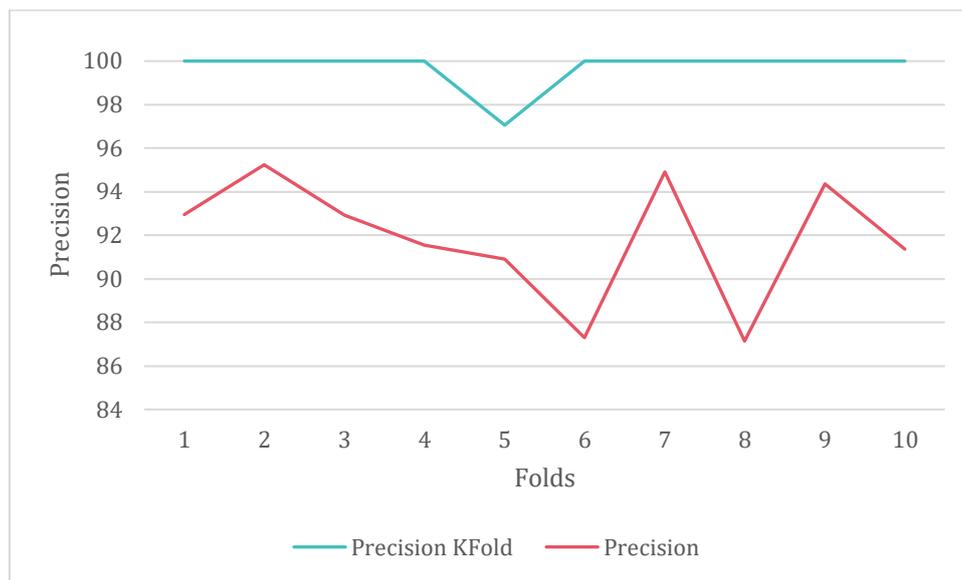


Figura 3.4.2 Comparación en la métrica de precisión

3.4.3 SENSIBILIDAD

A diferencia de las dos últimas métricas, en la Figura 3.43 se puede apreciar que el comportamiento trazado por la métrica de sensibilidad del modelo por diez corridas tuvo un mejor rendimiento al de la valoración cruzada. Esto quiere decir que el modelo entrenado por este medio, tuvo un error falso negativo relativamente menor al de la valoración cruzada, con solo dos de 38 imágenes de embriones sanos identificadas incorrectamente como embrión no sano.

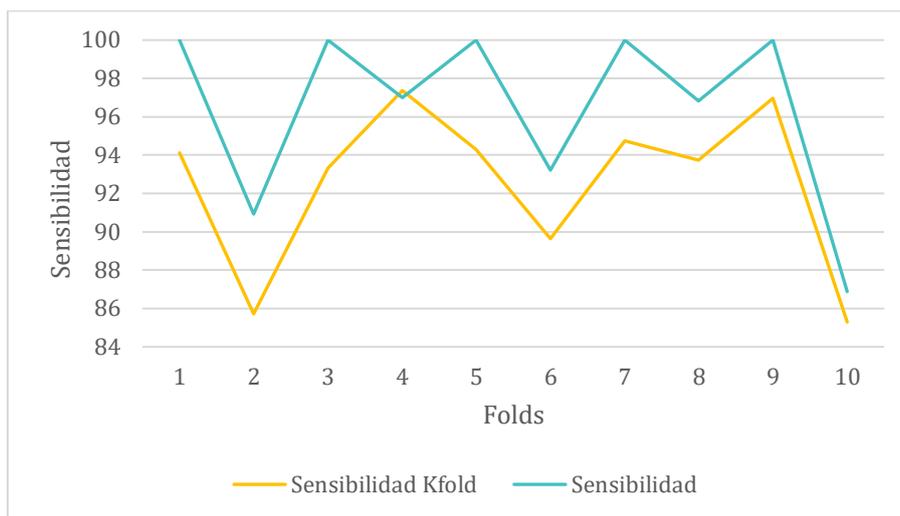


Figura 3.4.3 Comparación en la métrica de sensibilidad.

3.4.4 ESPECIFICIDAD

En cuanto a la métrica de la especificidad se puede observar en la Figura 3.44, que al igual que las primeras dos métricas discutidas, el modelo evaluado por validación cruzada tiene un mejor desempeño en cuanto a falsos positivos, con un promedio de 0.1 imágenes identificadas incorrectamente como sanas. Solo una imagen de embrión no sano usada en el conjunto de validación fue identificada incorrectamente durante los diez *folds*.

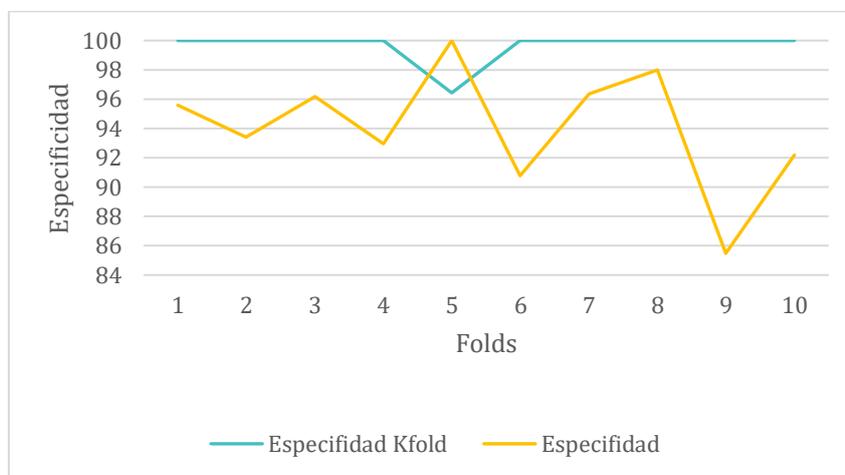


Figura 3.4.4 Comparación en la métrica de especificidad



4.DISCUSIÓN

4.1 PRE-PROCESAMIENTO DE DATOS

Durante el pre-procesamiento de datos, se transformaron las matrices de los valores “g”, “s” e “i” en tres imágenes individuales de forma 256 x 256. Se aumentaron las imágenes respectivas a embriones no saludables y se alimentó la CNN con una base de datos de 628 imágenes en total, casi el triple en comparación de la base de datos sin procesar. Este arreglo probó tener un alto rendimiento en todas las métricas de evaluación, así como en el método de validación cruzada. Cuenta con una desviación estándar mínima en todas las métricas de los dos métodos de evaluación seleccionados, lo cual estableció la confianza que se buscaba obtener en las predicciones hechas. Esto puede ser posiblemente atribuido a una base de datos mas grande y una forma de imagen mas apta para ser alimentada a las capas de la red neuronal convolucional.

4.2 DISEÑO Y CONSTRUCCIÓN DEL MODELO

Durante la experimentación para la construcción del modelo, se inició con valores pequeños para los hiperparámetros, gradualmente aumentándolos según el modelo se desempeñaba. Se encontró que usar valores grandes para los nodos de las capas sólo resultaba en la prolongación del tiempo de corrida sin producir cambios significantes en el rendimiento general del modelo. Finalmente se optó por el modelo que tuvo un mejor resultado en general, con valores de 32 y 64 para los nodos de las dos capas respectivamente.

También se experimentó con el número de capas y se encontró que dos capas era suficiente para obtener un buen desempeño en la tarea de clasificación con una exactitud promedio en el percentil 90. Con una capa el modelo obtenido tenia una desviación estándar de 16.35, lo cual no le permitió ser aceptada. Con tres capas el modelo obtenía valores en las métricas de evaluación similares al modelo con dos capas sin embargo el tiempo de corrida alargaba el proceso y en el

proceso de entrenamiento el modelo tendía a sobre ajustarse al conjunto usado para este proceso, es decir que no se adaptaba satisfactoriamente a datos nuevos y por lo tanto las predicciones no eran confiables.

4.3 COMPARACIÓN DE MÉTODOS DE CLASIFICACIÓN.

Para obtener el modelo con el mejor rendimiento en general, se utilizaron dos métodos de clasificación. El primero divide la base de datos en un conjunto de entrenamiento del 70%, de validación del 10% y de prueba del 20% para luego ser alimentado a la CNN y evaluado a lo largo de diez corridas. El segundo método de evaluación usado fue la validación cruzada, que divide la base de datos en diez *folds* o submuestras de igual cantidad para entrenar en nueve de ellas, validar en una y empezar de nuevo hasta haber validado en los diez conjuntos de submuestras creados.

Se encontró que mientras los dos métodos contaban con porcentajes altos en todas sus métricas de evaluación, el método que mejor se desempeño en la tarea de clasificación y validación de las imágenes fue el de validación cruzada. En general este metodo tuvo porcentajes mas altos en las métricas de evaluación y valores de desviación estándar mas pequeños, excluyendo en la métrica de sensibilidad donde el modelo evaluado mediante diez corridas tuvo un mejor rendimiento.

4.4 EVALUACIÓN DE EMBRIONES

En la construcción y evaluación del modelo de clasificación binaria fue importante determinar los los falsos positivo y negativo que podría cometer el modelo. En la selección de embriones se determinaron dos tipos de errores que se podrían cometer.

Falso Negativo (F-UH): El embrión es sano pero el modelo lo clasifica como no sano. En esta situación se ignoraría tal embrión y se seleccionaría uno que haya sido clasificado como sano. Este tipo de error ciertamente alargaría el proceso de selección de embriones.

Falso positivo (F-H): El embrión no es sano pero el modelo lo ha clasificado como sano. En este caso existe el riesgo de que sea seleccionado para su implantación. Sabiendo que los embriones subdesarrollados o desarrollados de manera insalubre cuentan con una posibilidad baja de implantarse y crear un embarazo, este tipo de error podría alargar el proceso en más de uno o dos ciclos FIV. En el improbable caso de que el embrión logre la implantación en el útero, siendo un embrión de baja calidad habría una posibilidad considerable de que el producto del embarazo tenga problemas de salud o que simplemente no llegue a término. En general, un falso positivo tiene una alta probabilidad de traducirse en un impacto altamente negativo.

En los dos métodos de evaluación, el objetivo de mantener el error falso positivo en lo mínimo posible fue logrado particularmente en el método de validación cruzada *10-fold* donde solo una imagen fue identificada incorrectamente como embrión sano durante las diez corridas de los *folds*.

4.5 COMPARACIÓN CON EL CASO DE ESTUDIO

En el caso de estudio [18] descrito en la introducción, se entrenó la arquitectura del algoritmo Inception-V1, utilizando 12,001 imágenes de embriones de buena calidad y de mala calidad. Posteriormente se evaluó el desempeño del DNN (llamado STORK) con un conjunto de prueba de 1930 imágenes y este fue capaz de identificar embriones de buena y mala calidad con una exactitud del 96.94%. En comparación, el modelo desarrollado en este trabajo fue diseñado desde cero (sin previo entrenamiento de la red) y entrenado y validado por dos métodos. En el método por diez corridas, el algoritmo fue entrenado con solo 440 imágenes para consecuentemente ser probado en 125 imágenes nuevas, obteniendo una exactitud promedio de 93.46%. Por otro lado, el método de validación cruzada utilizó 565 imágenes para el entrenamiento y 63 imágenes para la prueba, obteniendo una exactitud promedio de 96.03%. Se puede determinar que los resultados obtenidos en este trabajo compiten contra los obtenidos en el caso de estudio.

5.CONCLUSIONES

En este trabajo se diseñó y construyó un modelo computacional capaz de clasificar binariamente imágenes derivadas del análisis fasorial FLIM de embriones sanos y no sanos utilizando técnicas de aprendizaje automatizado. Después de haber desarrollado el modelo, este fue entrenado y validado mediante dos diferentes métodos; por medio de diez corridas y por validación cruzada. Finalmente, y mediante el detallado análisis y comparación de las métricas de evaluación resultantes de cada uno, se llegó a la conclusión que el método de validación cruzada produjo el mejor rendimiento en general. Este método dividió la base de datos en diez partes iguales para utilizar nueve de ellas, es decir 563 imágenes, para el entrenamiento y ajuste de los hiperparámetros y la parte restante de 63 imágenes para el proceso de validación de los datos. El modelo obtuvo una exactitud de 96.03%, una precisión del 99.70% y una especificidad del 99.64%.

A partir del análisis de las métricas resultantes de este método de validación se puede determinar que el modelo aprendió a reconocer y clasificar patrones pertenecientes a embriones sanos y no sanos con una exactitud promedio alta (96.03%).

Esto sugiere que el modelo fue entrenado y ajustado para su adaptación y predicción correcta de datos nuevos exitosamente. La mayoría de los embriones usados en el proceso de validación fueron correctamente identificados, particularmente los embriones no sanos. Finalmente se puede concluir que los objetivos de este trabajo de tesis fueron alcanzados exitosamente.

6.REFERENCIAS

[1] Mayo Clinic Team. Fertilización in vitro - Mayo Clinic [Internet]. Mayoclinic.org ; 2019 [citado 2021 May 25]. Disponible en: <https://www.mayoclinic.org/es-es/tests-procedures/in-vitro-fertilization/about/pac-20384716>

[2] MedlinePlus Team. Fecundación in vitro (FIV): MedlinePlus enciclopedia médica [Internet]. Medlineplus.gov. 2020 [citado 2021 May 25]. Disponible en: <https://medlineplus.gov/spanish/ency/article/007279.htm>

[3] Carolinas Fertility Institute Tea. What are the Risks and Side Effects of IVF? - Carolinas Fertility Institute [Internet]. Carolinas Fertility Institute. 2020 [citado 2021 May 25]. Disponible en: <https://www.carolinasfertilityinstitute.com/risks-side-effects-ivf/>

[4] "Does In Vitro Fertilization Really Work For Women?", *Verywell Family* [Internet]. Verywell Family 2021 [citado 2021 feb 15]. Disponible en : <https://www.verywellfamily.com/what-are-the-chances-for-ivf-success-1960213>.

[5] Datta R, Heaster TM, Sharick JT, Gillette AA, Skala MC. Fluorescence lifetime imaging microscopy: fundamentals and advances in instrumentation, analysis, and applications. *Journal of Biomedical Optics* [Internet]. 2020 May 13 [citado 2021 May 25];25(07):1. Disponible en: <https://www.spiedigitallibrary.org/journals/journal-of-biomedical-optics/volume-25/issue-07/071203/Fluorescence-lifetime-imaging-microscopy--fundamentals-and-advances-in-instrumentation/10.1117/1.JBO.25.7.071203.full?SSO=1>

[6] Omary MA, Patterson HH. Luminescence, Theory. *Encyclopedia of Spectroscopy and Spectrometry* [Internet]. 2017 [citado 2021 May 25];636-53. Disponible en: <https://www.sciencedirect.com/science/article/pii/B978012803224400193X>

[7] Edinburg Instruments. Jablonski Diagram | What is it? | Edinburg Instruments [Internet]. Edinburg Instruments. 2015 [citado 2021 May 25]. Disponible en: <https://www.edinst.com/blog/jablonski-diagram/>

[8] Gibson EA, Masihzadeh O, Lei TC, Ammar DA, Kahook MY. Multiphoton Microscopy for Ophthalmic Imaging. *Journal of Ophthalmology* [Internet]. 2011 [citado 2021 May 25];2011:1-11. Disponible en: <https://www.hindawi.com/journals/joph/2011/870879/>

[9] Coda S, Siersema P, Stamp G, Thillainayagam A. Biophotonic endoscopy: a review of clinical research techniques for optical imaging and sensing of early gastrointestinal cancer. *Endoscopy International Open* [Internet]. 2015 Sep 8 [citado 2021 May 25];03(05):E380-92. Disponible en: <https://pubmed.ncbi.nlm.nih.gov/26528489/>

[10] Ma N, Mochel NR de, Pham PD, Yoo TY, Cho K WY, Digma MA. Label-free assessment of pre-implantation embryo quality by the Fluorescence Lifetime Imaging Microscopy (FLIM)-phasor approach. *Scientific Reports* [Internet]. 2019 Sep 13 [citado 2021 May 25];9(1). Disponible en: <https://www.nature.com/articles/s41598-019-48107-2>

[11] Sumit Saha. A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way [Internet]. Medium. Towards Data Science; 2018 [citado 2021 May 25]. Disponible en: <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>

-
- [12] Yamashita R, Nishio M, Do RKG, Togashi K. Convolutional neural networks: an overview and application in radiology. *Insights into Imaging* [Internet]. 2018 Jun 22 [citado 2021 May 25];9(4):611–29. Disponible en: <https://insightsimaging.springeropen.com/articles/10.1007/s13244-018-0639-9>
- [13] Dertat A. Applied Deep Learning - Part 4: Convolutional Neural Networks [Internet]. Medium. Towards Data Science; 2017 [citado 2021 May 25]. Disponible en: <https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2>
- [14] Krut Patel. Convolutional Neural Networks — A Beginner's Guide - Towards Data Science [Internet]. Medium. Towards Data Science; 2019 [citado 2021 May 25]. Disponible en: <https://towardsdatascience.com/convolutional-neural-networks-a-beginners-guide-implementing-a-mnist-hand-written-digit-8aa60330d022>
- [15] Zhang A, Lipton ZC, Mu L, Smola AJ. Dive into Deep Learning — Dive into Deep Learning 0.16.4 documentation [Internet]. D2l.ai. 2021 [citado 2021 May 25]. Disponible en: <https://d2l.ai/>
- [16] Machine Learning for Artists Team. Neural networks [Internet]. Github.io. 2012 [citado 2021 May 25]. Disponible en: https://ml4a.github.io/ml4a/neural_networks/
- [17] Yani M, Budhi Irawan S Si, M.T., Casi Setiningsih ST M.T. Application of Transfer Learning Using Convolutional Neural Network Method for Early Detection of Terry's Nail. *Journal of Physics: Conference Series* [Internet]. 2019 May [citado 2021 May 25];1201:012052. Disponible en: <https://iopscience.iop.org/article/10.1088/1742-6596/1201/1/012052>
- [18] Thirumalaraju P, Kanakasabapathy, Manoj Kumar, Bormann CL, Gupta R, Pooniwala R, Kandula H, et al. Evaluation of deep convolutional neural networks in classifying human embryo images based on their morphological quality [Internet]. arXiv.org. 2020 [citado 2021 May 25]. Disponible en: <https://arxiv.org/abs/2005.10912>
- [19] Google Open Source Team. Projects – opensource.google [Internet]. opensource.google. 2021 [citado 2021 May 25]. Disponible en: <https://opensource.google/projects/tensorflow>
- [20] Keras Team. Keras documentation: About Keras [Internet]. Keras.io. 2021 [citado 2021 May 25]. Disponible en: <https://keras.io/about/>
- [21] Brownlee J. A Gentle Introduction to k-fold Cross-Validation [Internet]. Machine Learning Mastery. 2018 [citado 2021 May 25]. Disponible en: <https://machinelearningmastery.com/k-fold-cross-validation/>
- [22] Ignacio J. La matriz de confusión y sus métricas – Inteligencia Artificial – [Internet]. Juan Barrios. 2019 [citado 2021 May 25]. Disponible en: <https://www.juanbarrios.com/LA-MATRIZ-DE-CONFUSION-Y-SUS-METRICAS/>
- [23] Sarang Narkhede. Understanding AUC - ROC Curve - Towards Data Science [Internet]. Medium. Towards Data Science; 2018 [citado 2021 May 25]. Disponible en: <https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5>

7. APENDÍCES

A. CÓDIGO FUENTE PARA EL MODELO DE CLASIFICACIÓN BINARIA POR UNA RED NEURONAL CONVOLUCIONAL POR DIEZ CORRIDAS.

```
3 # conda install pydot
4 # brew install graphviz
5 # conda install graphviz
6 # conda install -c anaconda graphviz
7 # conda install python-graphviz
8
9 #Libraries
10 import numpy as np
11 import matplotlib.pyplot as plt
12 import matplotlib.image as mpimg
13 import os
14 import cv2
15 from tqdm import tqdm
16 from tensorflow.keras.callbacks import TensorBoard
17 import pandas as pd
18 import seaborn as sn
19 import tensorflow as tf
20 from tensorflow.keras.preprocessing.image import ImageDataGenerator
21 from tensorflow.keras.models import Sequential
22 from tensorflow.keras.layers import Dense, Dropout, Activation, Flatten, BatchNormalization
23 from tensorflow.keras.layers import Conv2D, MaxPooling2D, AveragePooling2D
24 from tensorflow.keras.losses import sparse_categorical_crossentropy, binary_crossentropy
25 from tensorflow.keras.optimizers import Adam, RMSprop
26 from sklearn.model_selection import KFold
27 from tensorflow.keras.preprocessing import image
28 from PIL import Image
29 from sklearn.metrics import confusion_matrix
30 from sklearn.metrics import confusion_matrix
31 from sklearn.metrics import plot_confusion_matrix
32 from sklearn.neighbors import KNeighborsClassifier
33
34
35 # Cargando datos
36 DATADIR = "/Users/julia/Desktop/TSG"
37 CATEGORIES = ["UH", "H"]
38
39 #Determinando tamaño de las imagenes
40 IMG_SIZE = 256
41 IMG_SIZE1 = 256
42
43
44 #Creando el conjunto de entrenamiento
45 training_data = []
46
47 def create_training_data():
48     for category in CATEGORIES:
49
50         #Creando camino al folder con las imagenes
51         path = os.path.join(DATADIR, category)
52
53         #Obtener las clasificaciones H=1 UH= 0
54         class_num = CATEGORIES.index(category)
55
56         for img in tqdm(os.listdir(path)):
57             try:
58                 img_array = cv2.imread(os.path.join(path, img), cv2.IMREAD_GRAYSCALE) # convert to array
59                 new_array = cv2.resize(img_array, (IMG_SIZE, IMG_SIZE1)) # resize to normalize data size
60                 training_data.append([new_array, class_num])
61             except Exception as e:
62                 pass
63
64
65
66 create_training_data()
67
68
69
70 # Rebarajeando los datos
71 import random
72 random.shuffle(training_data)
73
```

```

74
75 #Creando el modelo
76 X = []
77 y = []
78
79 for features,label in training_data:
80     #Asignar las features a la variable X
81     X.append(features)
82     #Asignar los labels a la variable y
83     y.append(label)
84
85
86
87 #Convirtiendo en matrices/arrays
88 X = np.array(X).reshape(-1, IMG_SIZE, IMG_SIZE, 1)
89 y = np.array(y)
90
91
92 #Guardar dataset
93 import pickle
94
95 pickle_out = open("X.pickle","wb")
96 pickle.dump(X, pickle_out)
97 pickle_out.close()
98
99 pickle_out = open("y.pickle","wb")
100 pickle.dump(y, pickle_out)
101 pickle_out.close()
102
103
104 #Configuracion del Modelo
105 batch_size =64
106 img_width, img_height, img_num_channels = 256, 256, 1
107 loss_function = binary_crossentropy
108 no_classes = 1
109 no_epochs = 10
110 optimizer = Adam()
111 verbosity = 1
112
113
114 #Separando los datos en conjuntos de entrenamiento,validacion y prueba.
115 from sklearn.model_selection import train_test_split
116 #Conjunto de Validacion
117 X_train, X_val, y_train, y_val = train_test_split(X, y, train_size=0.9, test_size=0.1, random_state=42)
118 #Conjunto de Entrenamiento y Prueba
119 X_train, X_test, y_train, y_test = train_test_split(X_train, y_train,train_size=0.78, random_state=42)
120
121
122
123
124
125 # Arquitectura de la CNN
126 model = Sequential()
127
128 model.add(Conv2D(32, (3,3),strides = 1, padding = 'same', input_shape=X.shape[1:]))
129 model.add(Activation("relu"))
130 model.add(MaxPooling2D(pool_size = (2,2)))
131
132
133 model.add(Conv2D(64, (3,3),strides=1, padding = 'same'))
134 model.add(Activation("relu"))
135 model.add(MaxPooling2D(pool_size = (2,2)))
136
137
138 model.add(Flatten())
139 model.add(Dense(512))
140 model.add(Activation('relu'))
141
142 model.add(Dense(1))
143 model.add(Activation("sigmoid"))
144
145 model.summary()
146

```

```

148
149 # Compilar el modelo
150 model.compile(loss=loss_function,
151               optimizer=optimizer,
152               metrics=['accuracy'])
153
154
155 #Graficar el modelo
156 from keras.utils.vis_utils import plot_model
157 plot_model(model, to_file='model_plot.png', show_shapes=True, show_layer_names=True)
158
159
160 #Salvar el modelo
161 model.save('TESIS.model')
162
163
164 # Ajustar el modelo a el conjunto de entrenamiento
165 print("Fit model on training data")
166 history = model.fit(X_train, y_train,
167                   batch_size=batch_size,
168                   epochs=no_epochs,
169                   verbose=verbosity,
170                   #steps_per_epoch = 3,
171                   validation_data= (X_val,y_val))
172
173
174
175 # Generar metricas de evaluacion
176 print('-----')
177 print("Evaluate on test data")
178 score = model.evaluate(X_test, y_test, batch_size=34,verbose=0)
179 print(f'Test loss: {score[0]}')
180 print(f'Test accuracy: {score[1]}')
181
182 #Graficar exactitud de entrenamiento y perdida.
183 plt.plot(history.history['accuracy'])
184 plt.plot(history.history['val_accuracy'])
185 plt.title('model accuracy')
186 plt.ylabel('accuracy')
187 plt.xlabel('epoch')
188 plt.legend(['train', 'test'], loc='upper left')
189 plt.show()
190
191
192 plt.plot(history.history['loss'])
193 plt.plot(history.history['val_loss'])
194 plt.title('model loss')
195 plt.ylabel('loss')
196 plt.xlabel('epoch')
197 plt.legend(['train', 'test'], loc='upper left')
198 plt.show()
199
200
201
202 #Predicciones
203 # y_classes predice los valores de las clases en test set
204 y_classes = model.predict_classes(X_test, verbose=0)
205 #y_test son los valores de las clases verdaderas
206
207
208 #Exactitud,Precision,Sensibilidad,F1
209
210 from sklearn.metrics import accuracy_score
211 from sklearn.metrics import precision_score
212 from sklearn.metrics import recall_score
213 from sklearn.metrics import f1_score,roc_auc_score
214
215 # accuracy: (tp + tn) / (p + n)
216 accuracy = accuracy_score(y_test, y_classes)
217 #print('Accuracy: %f' % accuracy)
218 print(accuracy)
219
220

```

```

206
207
208 #Exactitud, Precision, Sensibilidad, F1
209
210 from sklearn.metrics import accuracy_score
211 from sklearn.metrics import precision_score
212 from sklearn.metrics import recall_score
213 from sklearn.metrics import f1_score, roc_auc_score
214
215 # accuracy: (tp + tn) / (p + n)
216 accuracy = accuracy_score(y_test, y_classes)
217 #print('Accuracy: %f' % accuracy)
218 print(accuracy)
219
220
221 # precision tp / (tp + fp)
222 precision = precision_score(y_test, y_classes)
223 #print('Precision: %f' % precision)
224 print(precision)
225
226
227 # recall: tp / (tp + fn)
228 recall = recall_score(y_test, y_classes)
229 #print('Recall: %f' % recall)
230 print(recall)
231
232
233 # f1: 2 tp / (2 tp + fp + fn)
234 f1 = f1_score(y_test, y_classes)
235 #print('F1 score: %f' % f1)
236 print(f1)
237
238 #Matriz de Confusion
239 matrix = confusion_matrix(y_test, y_classes)
240 print(matrix)
241
242 # Graficar la matriz de confusion
243 import seaborn as sns
244 group_names = ['True Neg', 'False Pos', 'False Neg', 'True Pos']
245 group_counts = ["{0:0.0f}".format(value) for value in
246                 matrix.flatten()]
247 group_percentages = ["{0:.2%}".format(value) for value in
248                     matrix.flatten()/np.sum(matrix)]
249 labels = ["{v1}\n{v2}\n{v3}" for v1, v2, v3 in
250          zip(group_names, group_counts, group_percentages)]
251 labels = np.asarray(labels).reshape(2,2)
252 sns.heatmap(matrix, annot=labels, fmt='', cmap='Blues')
253
254
255 # ROC-AUC
256 y_pred = model.predict(X_test)
257 fpr, tpr, _ = metrics.roc_curve(y_test, y_pred)
258 auc_score = metrics.auc(fpr, tpr)
259 plt.clf()
260 plt.title('ROC Curve')
261 plt.plot(fpr, tpr, label=' AUC = area = {:.2f}'.format(auc_score))
262 plt.plot([0,1],[0,1], 'r--')
263 plt.xlim([-0.1,1.1])
264 plt.ylim([-0.1,1.1])
265 plt.ylabel('True Positive Rate')
266 plt.xlabel('False Positive Rate')
267 plt.legend(loc='lower right')
268 plt.show()
269 roc_auc_score(y_test, y_pred)
270
271
272
273 #Tensorboard
274 NAME = "H vs UH"
275
276 tensorboard = TensorBoard(log_dir="logs/{}".format(NAME))
277 #tensorboard --logdir=logs/.
278

```

B. CÓDIGO FUENTE PARA EL MODELO DE CLASIFICACIÓN BINARIA POR UNA RED NEURONAL CONVOLUCIONAL POR VALIDACIÓN CRUZADA

```
4
5 #Libraries
6 import numpy as np
7 import matplotlib.pyplot as plt
8 import matplotlib.image as mpimg
9 import os
10 import cv2
11 from tqdm import tqdm
12 from tensorflow.keras.callbacks import TensorBoard
13 import pandas as pd
14 import seaborn as sn
15 import tensorflow as tf
16 from tensorflow.keras.datasets import cifar10
17 from tensorflow.keras.preprocessing.image import ImageDataGenerator
18 from tensorflow.keras.models import Sequential
19 from tensorflow.keras.layers import Dense, Dropout, Activation, Flatten
20 from tensorflow.keras.layers import Conv2D, MaxPooling2D
21 from tensorflow.keras.losses import sparse_categorical_crossentropy, binary_crossentropy
22 from tensorflow.keras.optimizers import Adam
23 from sklearn.model_selection import KFold
24
25
26 #Cargando los datos
27 DATADIR = "/Users/julia/Desktop/TSG"
28
29 CATEGORIES = ["H", "UH"]
30
31
32 IMG_SIZE = 256
33 IMG_SIZE1 = 256
34
35
36 #Creando el conjunto de entrenamiento
37 training_data = []
38
39 def create_training_data():
40     for category in CATEGORIES:
41         #Creando camino al folder con las imagenes
42         path = os.path.join(DATADIR, category)
43
44         #Obtener las clasificaciones H=1 UH= 0
45         class_num = CATEGORIES.index(category)
46
47         for img in tqdm(os.listdir(path)):
48             try:
49                 img_array = cv2.imread(os.path.join(path, img), cv2.IMREAD_GRAYSCALE) # convert to array
50                 new_array = cv2.resize(img_array, (IMG_SIZE, IMG_SIZE1)) # resize to normalize data size
51                 training_data.append([new_array, class_num])
52             except Exception as e:
53                 pass
54
55
56
57 create_training_data()
58
59
60 #Rebajarear datos
61 import random
62
63 random.shuffle(training_data)
64
65
66
67 #Creando el modelo
68 X = []
69 y = []
70
71 for features, label in training_data:
72     #Asignar las features a la variable X
73     X.append(features)
74     #Asignar los labels a la variable y
75     y.append(label)
76
77
```

```

80
81 #Convertir a matrices/arrays
82 X = np.array(X).reshape(-1, IMG_SIZE, IMG_SIZE1, 1)
83
84 y = np.array(y)
85
86
87 #Guardar dataset
88 import pickle
89
90 pickle_out = open("X.pickle","wb")
91 pickle.dump(X, pickle_out)
92 pickle_out.close()
93
94 pickle_out = open("y.pickle","wb")
95 pickle.dump(y, pickle_out)
96 pickle_out.close()
97
98
99 #Configuracion del modelo
100 batch_size = 28
101 img_width, img_height, img_num_channels = 256, 256, 1
102 loss_function = binary_crossentropy
103 no_classes = 1
104 no_epochs = 10
105 optimizer = Adam()
106 verbosity = 1
107 num_folds = 10
108
109
110
111
112
113
114 #Definiendo metricas por folds
115 acc_per_fold = []
116 loss_per_fold = []
117
118
119
120 # Definir las variables de Validacion Cruzada
121 kfold = KFold(n_splits=num_folds, shuffle=True)
122
123 # K-fold Cross Validation
124 fold_no = 1
125 for train, test in kfold.split(X, y):
126
127     # Arquitectura del modelo
128     model = Sequential()
129     model.add(Conv2D(32, (3,3),padding = 'same', input_shape=X.shape[1:]))
130     model.add(Activation("relu"))
131     model.add(MaxPooling2D(pool_size = (2,2)))
132
133     model.add(Conv2D(64, (3,3),padding = 'same'))
134     model.add(Activation("relu"))
135     model.add(MaxPooling2D(pool_size = (2,2)))
136
137     model.add(Flatten())
138     model.add(Dense(512))
139
140     model.add(Dense(1))
141     model.add(Activation("sigmoid"))
142
143
144     # Compilar el modelo
145     model.compile(loss=loss_function,
146                 optimizer=optimizer,
147                 metrics=['accuracy'])
148
149     #Salvar modelo
150     model.save('embryos1.h5')
151
152

```

```

152
153
154     print('-----')
155     print(f'Training for fold {fold_no} ...')
156
157
158
159     # Ajustar al modelo de entrenamiento y validacion
160     history = model.fit(X[train], y[train],
161                       batch_size=batch_size,
162                       epochs=no_epochs,
163                       verbose=verbosity,
164                       validation_data= (X[test],y[test]))
165
166
167
168     #Generar exactitud de entrenamiento
169     print('Accuracy of Training : {history} ')
170
171
172
173
174     # Generate gmetricas de evaluacion
175     scores = model.evaluate(X[test], y[test], verbose=0)
176     print(f'Score for fold {fold_no}: {model.metrics_names[0]} of {scores[0]}; {model.metrics_names[1]} of {scores[1]*100}%')
177     acc_per_fold.append(scores[1] * 100)
178     loss_per_fold.append(scores[0])
179
180
181
182     #Predicciones
183
184     # y_classes predice los valores de las clases en test set
185     y_classes = model.predict_classes(X[test], verbose=0)
186
187     #Matriz de confusion
188     from sklearn.metrics import confusion_matrix
189     matrix = confusion_matrix(y[test], y_classes)
190     print(matrix)
191
192     #Accuracy,Precision,Recall and F1
193
194     from sklearn.metrics import accuracy_score
195     from sklearn.metrics import precision_score
196     from sklearn.metrics import recall_score
197     from sklearn.metrics import f1_score,roc_auc_score
198
199     # accuracy: (tp + tn) / (p + n)
200     accuracy = accuracy_score(y[test], y_classes)
201     #print('Accuracy: %f' % accuracy)
202     print(accuracy)
203
204
205     # precision tp / (tp + fp)
206     precision = precision_score(y[test], y_classes)
207     #print('Precision: %f' % precision)
208     print(precision)
209
210
211     # recall: tp / (tp + fn)
212     recall = recall_score(y[test], y_classes)
213     #print('Recall: %f' % recall)
214     print(recall)
215
216
217     # f1: 2 tp / (2 tp + fp + fn)
218     f1 = f1_score(y[test], y_classes)
219     #print('F1 score: %f' % f1)
220     print(f1)
221
222
223

```

```

186
187 #Matriz de confusion
188 from sklearn.metrics import confusion_matrix
189 matrix = confusion_matrix(y[test], y_classes)
190 print(matrix)
191
192 #Accuracy, Precision, Recall and F1
193
194 from sklearn.metrics import accuracy_score
195 from sklearn.metrics import precision_score
196 from sklearn.metrics import recall_score
197 from sklearn.metrics import f1_score, roc_auc_score
198
199 # accuracy: (tp + tn) / (p + n)
200 accuracy = accuracy_score(y[test], y_classes)
201 #print('Accuracy: %f' % accuracy)
202 print(accuracy)
203
204
205 # precision tp / (tp + fp)
206 precision = precision_score(y[test], y_classes)
207 #print('Precision: %f' % precision)
208 print(precision)
209
210
211 # recall: tp / (tp + fn)
212 recall = recall_score(y[test], y_classes)
213 #print('Recall: %f' % recall)
214 print(recall)
215
216
217 # f1: 2 tp / (2 tp + fp + fn)
218 f1 = f1_score(y[test], y_classes)
219 #print('F1 score: %f' % f1)
220 print(f1)
221
222
223
224 """
225 scores = model.evaluate(inputs[test], targets[test], verbose=0)
226 print('-----')
227 print(f'Score for fold {fold_no}: {model.metrics_names[0]} of {scores[0]}; {model.metrics_names[1]} of {scores[1]*100}%')
228 acc_per_fold.append(scores[1] * 100)
229 loss_per_fold.append(scores[0])
230 """
231
232
233
234
235 # Sigueinte fold
236 fold_no = fold_no + 1
237
238
239
240 # == Metricas de evaluacion ==
241 print('-----')
242 print('Score per fold')
243 for i in range(0, len(acc_per_fold)):
244     print('-----')
245     print(f'> Fold {i+1} - Loss: {loss_per_fold[i]} - Accuracy: {acc_per_fold[i]}%')
246     print('-----')
247 print('Average scores for all folds:')
248 print(f'> Accuracy: {np.mean(acc_per_fold)} (+- {np.std(acc_per_fold)})')
249 print(f'> Loss: {np.mean(loss_per_fold)}')
250 print('-----')
251
252 #Tensorboard
253 NAME = "H vs UH"
254
255 tensorboard = TensorBoard(log_dir="logs/{}".format(NAME))
256
257
258

```

C. EXPERIMENTACIÓN CON LAS CAPAS Y NODOS DE LA CNN

```

2
3 from tensorflow.keras.models import Sequential
4 from tensorflow.keras.layers import Dense, Dropout, Activation, Flatten
5 from tensorflow.keras.layers import Conv2D, MaxPooling2D
6 from tensorflow.keras.callbacks import TensorBoard
7 import pickle
8 import time
9
10 pickle_in = open("X.pickle","rb")
11 X = pickle.load(pickle_in)
12
13 pickle_in = open("y.pickle","rb")
14 y = pickle.load(pickle_in)
15
16 #Separating data
17 from sklearn.model_selection import train_test_split
18 X_train, X_val, y_train, y_val = train_test_split(X, y, train_size=0.9, test_size=0.1, random_state=42)# Create the Test and Final Training Data
19 X_train, X_test, y_train, y_test = train_test_split(X_train, y_train, train_size=0.78, random_state=42)
20
21
22 dense_layers = [1]
23 layer_sizes = [32, 64, 128]
24 conv_layers = [1, 2, 3]
25
26 for dense_layer in dense_layers:
27     for layer_size in layer_sizes:
28         for conv_layer in conv_layers:
29             NAME = "{}-conv-{}-nodes-{}-dense-{}".format(conv_layer, layer_size, dense_layer, int(time.time()))
30             print(NAME)
31
32             model = Sequential()
33
34             model.add(Conv2D(layer_size, (3, 3), input_shape=X.shape[1:]))
35             model.add(Activation('relu'))
36             model.add(MaxPooling2D(pool_size=(2, 2)))
37
38             for l in range(conv_layer-1):
39                 model.add(Conv2D(layer_size, (3, 3)))
40                 model.add(Activation('relu'))
41                 model.add(MaxPooling2D(pool_size=(2, 2)))
42
43             model.add(Flatten())
44
45             for _ in range(dense_layer):
46                 model.add(Dense(layer_size))
47                 model.add(Activation('relu'))
48
49             model.add(Dense(1))
50             model.add(Activation('sigmoid'))
51
52             tensorboard = TensorBoard(log_dir="logs/{}".format(NAME))
53
54             model.compile(loss='binary_crossentropy',
55                           optimizer='adam',
56                           metrics=['accuracy'],
57                           )
58
59             model.fit(X_train, y_train,
60                       batch_size=64,
61                       epochs=10,
62                       validation_data=(X_val,y_val),
63                       callbacks=[tensorboard])
64
65             print('-----')
66             print("Evaluate on test data")
67             score = model.evaluate(X_test, y_test, batch_size=34,verbose=0)
68             print('Test loss: {}'.format(score[0]))
69             print('Test accuracy: {}'.format(score[1]))
70
71

```