UNIVERSITAT POLITÈCNICA DE VALÈNCIA

_ TELECOM ESCUELA TÉCNICA VLC SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

SISTEMA DE AYUDA EN LA SELECCIÓN DE EMBRIONES PARA LA FERTILIZACIÓN IN VITRO MEDIANTE LA ELABORACIÓN Y EL ANÁLISIS DE UNA ROBUSTA BASE DE DATOS

Cristian Camilo Pulgarín Ospina

Tutor: Valeriana Naranjo Ornedo

Cotutor: Adrián Colomer Granero

Trabajo Fin de Grado presentado en la Escuela Técnica Superior de Ingenieros de Telecomunicación de la Universitat Politècnica de València, para la obtención del Título de Graduado en Ingeniería de Tecnologías y Servicios de Telecomunicación

Curso 2018-19

Valencia, 10 de septiembre de 2019

Escuela Técnica Superior de Ingeniería de Telecomunicación
Universitat Politècnica de València
Edificio 4D. Camino de Vera, s/n, 46022 Valencia
Tel. +34 96 387 71 90, ext. 77190
www.etsit.upv.es

VLC/ CAMPUS VALENCIA, INTERNATIONAL CAMPUS OF EXCELLENCE

EMAS Gestión ambiental verificada REG.NO. ES-CV- 000030

Agradecimientos

Quiero agradecer a mi tutora Valery por confiar en mí para este proyecto y darme la oportunidad de introducirme en el mundo de reconocimiento de imágenes mediante técnicas de *Deep Learning*, igualmente quiero agredecer al apoyo que he tenido de todo el grupo de investigación CVBlab.

También agradecer a mi familia y amigos por brindarme su apoyo en todo momento, en especial a mi tía Olga que ha sido como una madre, por último y más importante agradecer a María Consuelo y Luis Antonio, mis padres, sin ellos nunca podría haber a ver vivido, experimentado y alcanzado nada de lo que he hecho. Viniendo de una familia muy humilde en Colombia hemos pasado mil y una adversidad, pero ese espíritu de nunca rendirse y luchar en busca de un futuro mejor es el que me ha inspirado en muchas ocasiones para no tirar la toalla, habéis sido el mejor espejo en que mirarme, gracias por enseñarme lo importante que es ser humilde, por todo ello muchas gracias.

Resumen

En el marco tecnológico actual es cada vez más importante la adquisición, el tratamiento y el análisis de datos para hallar patrones de comportamiento de estos y así poder abordar la solución a un determinado problema. El problema que nos ocupa en este trabajo es la predicción de éxito de la implantación de embriones. Este trabajo fin de grado se enmarca dentro de un proyecto de colaboración entre el instituto de infertilidad de Valencia y el grupo CVBLab de la Universitat Politècnica de València.

El trabajo ha consistido en dos partes bien diferenciadas: por un lado la construcción de una base de datos amplia a partir de una gran cantidad de datos suministrados por el IVI que sirva de entrada al modelo que hemos desarrollado en la segunda parte del trabajo. Se trata de un modelo de predicción de tiempos de división celular con los que poder predecir el éxito de la implantación de embriones en técnicas de fecundación in vitro utilizando técnicas de aprendizaje automático, concretamente redes neuronales convolucionales (*deep learning*). Los resultados obtenidos son prometedores y permiten poner de manifiesto la bondad de estas técnicas en esta línea de investigación.

Resum

En el marc tecnològic actual és cada vegada més important l'adquisició, el tractament i l'anàlisi de dades per a trobar patrons de comportament d'aquests i així poder abodar la solucióa un determinat problema. El problema que ens ocupa en aquest treball és la predicció d'èxit de la implantació d'embrions. Aquest treball fi de grau s'emmarca dins d'un projecte de col·laboració entre l'institut d'infertilitat de València i el grup CVBLab de la Universitat Politècnica de València.

El treball ha cosistit en dues parts ben diferenciades: d'una banda la construcció d'una base de dades àmplia a partir d'una gran quantitat de dades subministrades pel IVI que servisca d'entrada al model que hem desenvolupat en la segona part del treball. Es tracta d'un model de predicció de temps de divisió célular amb els quals poder predir l'èxit de la implantació d'embrions en tècniques de fecundació in vitro utilitzant tècniques de aprenentatge automàtic, concretament xarxes neuronals convolucionals (*deep learning*). Els resultats obtinguts són prometedors i permeten posar de manifest la bondat d'aquestes tècniques en aquesta línia d'investigació.

Índice general

Capítulo 1

Introducción

1.1 Motivación y descripción del problema

El problema de infertilidad es algo bastante común en los españoles, alrededor de un 17 % de la población lo sufre. Las causas son tan variadas como, el exceso de ejercicio físico, sobrepeso, contaminación ambiental, exposición al calor por tiempo prolongado, diabetes, consumo de alcohol, tabaco, patologías, defectos congénitos, etc. Estos factores son cada vez más habituales en nuestra sociedad por lo que es probable que ese 17 % pueda aumentar de forma notable. Este problema que sufren muchas parejas hace que se pongan en manos de clínicas de reproducción asistida, en busca de tratamientos que les garanticen una alta efectividad y seguridad de embarazo.

Los métodos más utilizados en la actuliadad son la inseminación artificial y la fecundación in vitro porque son las técnicas de reproducción asistida qué mayor tasa de éxito proporcionan.

1.1.1 Inseminación Artificial

La inseminación artificial es un método de reproducción asistida poco complejo en el que se introduce una muestra de semen previamente tratada en el útero de la paciente, de este modo el embrión comienza a crecer desde el principio en un entorno natural. Esta técnica aumenta la eficacia de embarazo con respecto a relación sexual. Este tratamiento es especialmentete indicado para, [9]:

- Mujeres sin pareja que tienen óvulos de calidad.
- Mujeres con alteraciones en la ovulación.
- Mujeres con alguna alteración en la entrada del útero.
- Parejas heterosexuales en las que el hombre presenta anomalías leves o moderadas en la calidad de los espermatozoides.
- Parejas heterosexuales en las que el hombre sea portador de una enfermedad genética.

La inseminación artificial es el método de reproducción asistida más utilizado, su precio asequible hace que sea la primera alternativa para las mujeres que tienen dificultades para quedarse embarazadas de forma natural.

La tasa de éxito de la inseminación artificial es de aproximadamente el 15-20 % por ciclo. Tras 4 intentos de inseminación artificial, se puede conseguir una tasa acumulada del 45-50 %, generalmente estas cifras mejoran si se usa semen de donantes.

1.1.2 FIV

La Fecundación in vitro es un método de reproducción asistida de alta complejidad. Este proceso a diferencia de la inseminación artificial, no se produce en el entorno natural del útero sino que se lleva acabo en el entorno de un laboratorio. Este tratamiento consiste en seleccionar un óvulo y un espermatozoide para después en el laboratiorio -in vitro- realizar la fecundación con el fin de obtener embriones de una gran calidad que puedan ser transferidos al útero materno dando lugar a un embarazo.

La fecundación in vitro es recomendada para parejas de pacientes que cumplen los siguientes perfiles [10]:

- Parejas en las que el hombre tiene una mala calidad de espermatozoides.
- Mujeres de edad avanzada.
- Mujeres con problemas en las trompas que afectan directamente a la calidad del óvulo.
- Mujeres después de varias inseminaciones sin éxito.
- Parejas en las que es necesario hacer un estudio genético preimplantacional.

En la fecundación in vitro, la edad de la mujer es el factor determinante para el éxito del tratamiento. Así, la tasa de embarazo media acumulada (la que obtenemos utilizando todos los embriones obtenidos en un mismo ciclo) para mujeres de edad igual o menor de 34 años es del 78 %. Para mujeres de entre 35 y 37 años, la tasa se sitúa en el 63 %, para aquellas que tienen entre 38 y 40 años, es de un 54 % y, finalmente, para las que tienen igual o más de 40 años, la tasa de embarazo es de un 38 %. Estos datos son para casos de FIV en los que el óvulo es propio.

Cuando el óvulo es donadado la tasa de éxito aumenta puesto que la edad deja de influir en la fecundación in vitro, ya que la de la donante de esos ovocitos siempre tendrá menos de 35 años. En este caso la tasa de embarazo media acumulada por ciclo de recepción de ovocitos (la que obtenemos utilizando todos los embriones obtenidos en un mismo ciclo) para las mujeres es de un 79 %, sin importar su edad.

1.1.3 Determinación de la calidad del embrión den la FIV

Para determinar cuando un embrión es más o menos viable para ser tranferidos se tienen muchos elementos en cuenta: edad, tipo de infertilidad, si los gametos son donados o propios, parámetros morfocinéticos del embrión, etc.

En la actulidad el uso de sistemas como el Embryoscope nos permite tener más controlado un embrión, esto facilita el control y análisis de parámetros que afectan en la elección[24].

Durante el crecimiento embrionario incubadores como el Embryoscope van tomando imágenes para analizar el comportamiento individual de cada embrión e ir controlando los parámetros que el especialista va a tener en cuenta para la elección del embrión y su posterior implantación en el útero de la paciente. Uno de estos elementos estudiados son los tiempos de división celular

durante las primeras horas del desarrollo embrionario, ya que como se índica en varios estudios [22] [8], se ha demostrado que según el momento en el que se produzcan las divisiones celulares, desde el momento de la fecundación ,se puede determinar qué embrión es más viable para ser implantado dando lugar a la gestación del embrión.

Hasta la introducción de sistemas de toma de imágenes automáticas como el sistema Embryoscope, la labor de medir los tiempos de división era realizada por un experto que con una frecuencia basada en su experiencia extraía la muestra del incubar observaba el estado del crecimiento embrionario y basándose en ello determinaba la calidad del embrión y su posibilidad de gestar en caso de ser implantado en el útero.

Los sistemas actuales de cultivos de embriones se encargan de tomar imágenes con una frecuencia determinada, con esto eliminamos la subjetividad en la toma de datos del crecimiento embrionario con lo que podemos medir los tiempos de división celular de una forma mejor.

Para la elección de embriones se han ideado múltiples sistemas, en los que tomando los datos que proporciona el incubador se analizan con algoritmos para determinar cual es el embrión más apto. Los tiempos que se miden durante la segmentación en [20] y que vamos a usar en nuestro trabajo como referencia son:

- **t2**: Es el instante en el que el embrión se divide por primera vez desde que es fecundado el óvulo.
- **t3**: Es el momento en el que produce la segunda segmentación de una de las dos células, teniendo tres.
- **t4**: Instante en el que se produce un nuevo clivaje, pasando a tener 4 células.
- **t5**: Momento en que se produce una nueva división, teniendo un embrión formado por cinco células
- **cc2**: Se define como el periodo que tarda el embrión de pasar de dos células a tres.
$$cc2 = t3 - t2 \tag{1.1}$$
- **cc3**: Periodo de tiempo que trascurre entre la segunda y tercera segmentación.
$$cc3 = t4 - t3 \tag{1.2}$$

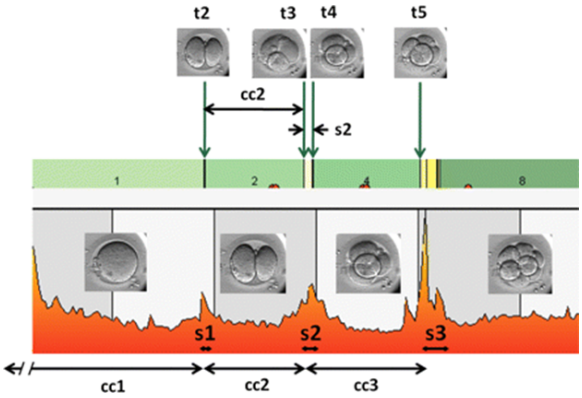


Figura 1.1: Representación gráfica de los tiempos medidos durante el crecimiento embrionario,[20].

Anteriormente se realizó un primer diseño de un sistema que estima los tiempos de división celular, esto se desarrolló en el TFM "Desarrollo de un sistema de procesamiento de imágenes para la predicción sobre el éxito en la implementación de blastocitos con parámetros morfocinéticos",[28] con una base de datos de 263 vídeos de los que se extraía los tiempos morfocinéticos representados en la Figura 1.1.

El objetivo de esto es el de encontrar una forma objetiva de detectar los tiempos de segmentación, tiempos con los que se demuestra en [20] qué se puede elegir un embrión que al ser implantado tiene altas probabilidades de gestación, para posteriormente con esos datos entrenar una red neuronal que nos prediga la tasa de éxito en la implantación, obteniendo así un sistema que nos garantice una mejor fiabilidad con respecto a otros modelos elaborados hasta el momento.

En [28] se consigue desarrollar un buen estimador de tiempos de división, pero no se logra un buen modelo de predicción de éxito en la implantación, esto puede ser debido a que los parámetros morfocinéticos proporcionan muy poca información para elegir un embrión por lo que haría falta realizar una predicción usando otros parámetros del embrión durante su crecimiento embrionario. Por otro lado estos resultados también podrían deberse a una falta de un base de datos más grande de la que extraer muchas más características de los embriones ya que 263 muestras para entrenar, validar y testear un modelo de predicción pueden ser muy escasos como para poder llegar a alcanzar un modelo de predicción robusto y fiable.

1.2 Objetivos

El grupo de investigación **Cvblab** está dirigido por la Dra. Valery Naranjo, catedrática del departamento de Comunicaciones de la Universidad Politécnica de Valencia, Cvblab lleva más de 20 años colaborando en proyectos de distintos ámbitos, uno de ellos es el campo de la sanidad, más concretamente en la ayuda al diagnóstico de imágenes, para ello se analizan imágenes de distintas patologías con técnicas de *deep learning* con el fin de entrenar redes neuronales que puedan diagnosticar y detectar enfermedades de una forma automática y más eficiente [2].

El **Instituto valencia de Infertilidad** es una de las clínicas de reproducción asistida más importantes a nivel mundial en la actualidad, su estatus en el mercado se debe a la inversión que se hace en I+D para alcanzar cada vez mejores métodos y técnicas que garanticen una mayor tasa de éxito de embarazos en sus pacientes,[11].

Por esto, en 2018 se inició un proyecto de colaboración en el que se quería aumentar la tasa de éxito de los tratamientos FIV, fruto de ello fue el TFM [28], en 2019 este acuerdo se renovó y se afianzó logrando tener un mayor acceso a la base de datos de los tratamientos realizados por el IVI.

El objetivo principal de este TFG es el tomar los datos de los tratamientos FIV de los que ahora dispone el Cvblab con la renovación del proyecto de colaboración con el fin de organizarlos, ordenarlos para analizarlos y buscar relación entre ellos encontrando datos útiles con lo que generar una robusta y fiable base datos con la que comprobar que los modelos desarrollados hasta ahora para este proyecto, son fiables con otras muestras y además realizar un modelo de clasificación celulara para después usar las imágenes clasificadas y estimar con ellas los tiempos de división.

Por tanto los pasos y objetivos específicos son los siguientes:

- Analizar, transformar y seleccionar los datos facilitados por el IVI con el fin de crear una base de datos organizada y robusta.
- Diseñar un modelo de clasificación celular en Python basado en la estructura propuesta por Aisha Khan, Stephen Gould y Mathieu Salzmann en [16].
- Usar el clasificador diseñado con la nueva base datos comprobando así la robustez y fiabilidad del modelo con un conjunto de datos más amplio.
- Adaptar el modelo de "Estimación del movimiento del embrión (SAD)", [28], para calcular los tiempos de división celular de los embriones de nuestra base de datos.

Capítulo 2

Construcción de la base de datos

2.1 Introducción

En este capítulo se explica el proceso de creación de la base de datos, para generar nuestra base datos debemos primero ver cuales van a ser nuestras fuentes de datos, en este caso contamos con bases de datos SQLite exportadas por los sistemas de adquisición de imágenes Embryoscope y unas hojas excel llamadas Sivis donde tenemos datos demográficos de los pacientes, datos del tipo de infertilidad y parámetros de los embriones.

Primero hay que ver como están organizados los datos para darles un orden y que sea más fácil acceder a ellos de forma automática mediante el uso de una aplicación en Java, después se va analizar los datos que contienen las bases de datos SQLite y las hojas Excel para determinar que datos son útiles para exportar imágenes del crecimiento embrionario y de que forma se puede relacionar los datos de ambas fuentes de información.

A continuación se describirá como se toman los distintos tipos de datos que se nos proporciona desde el IVI, como son ordenados, analizados y transformados para ser guardados en nuestra base de datos, así como el proceso de construcción de la misma.

2.2 Embryoscope

En las técnicas de reproducción asistida es muy importante recrear un entorno lo más parecido posible a las condiciones en las que se encontraría el cigoto los primeros días después de la fecundación, por lo que hay que tener en cuenta elementos como: temperatura, porcentaje de CO_2, porcentaje de Oxígeno, metabolitos en el medio de cultivo, volátiles, etc.

Los instrumentos que se utilizan para simular las condiciones óptimas son incubadores embrionarios, cuya eficacia se mide en la capacidad de que recupere lo más rápido posible las condiciones óptimas del cigoto cuando este es extraído del incubador para ser analizado. Existen distintos tipos: el **incubador convencional** con alta capacidad pero poca consistencia para mantener las condiciones del cultivo, el **incubador Benchtop** que permite recuperar el nivel de gases en un corto período de tiempo y por último tenemos el **incubador Time-lapse**, mucho más estable y

que además incorpora un sistema de monitorización continuo que sigue el desarrollo embrionario del cigoto.

El **Embryoscope** es un incubador time-lapse por lo que cuenta con un sistema de monitorización que toma constantemente imágenes de distintos planos y otros datos de interés del cigoto con una determinada frecuencia, esto facilita que se le pueda hacer un seguimiento a un embrión sin tener que extraerlo del incubador. En caso de ser totalmente necesario revisar el embrión extrayéndolo, el sistema recupera rápidamente las condiciones de tal forma que minimiza los daños por la manipulación.

Figura 2.1: Imágen de un incubador Time-lapse (Embryoscope).

La Figura 2.1 muestra el aspecto físico de un Embryoscope. Como se puede apreciar este incubador cuenta con una pantalla que muestra las últimas imágenes tomadas a los distintos embriones. Por otro lado cuenta con un teclado el cual permite al experto interactuar con el incubador, para moverse por su menú o pedir que muestre determinados datos.

En los tratamientos de infertilidad se suele fecundar más de un embrión a la vez, para así incrementar la posibilidad de encontrar un embrión apto y conseguir una mayor tasa de éxito en la implantación, por esto en los incubadores se usa una portaobjetos(*slide*) con diferentes ranuras(*slots*) a los cuales se les llama *Wells* conteniendo cada uno un embrión de tal forma que se les puede realizar un seguimiento de forma independiente.

2.2.1 Análisis de los datos del Embryoscope

El Embryoscope exporta los datos generados en un archivo con formato ".pbd". Este tipo de archivos son bases de datos SQLite que pueden abrirse con distintos tipos de programas como por ejemplo; *Quicken* , *Microsoft Visual Studio*, e incluso *Notpad*, para la realización de este trabajo utilizaremos *BD Browser for SQL* que es un programa de código abierto que trabaja con bases de datos *SQLite*.

SQLite es un programa para gestionar bases de datos relacionales como MySql pero a diferencia de estas, las BD hechas con SQLite se componen de un único archivo ASCII en el que están contenidos todas las tablas con sus correspondientes datos, por este motivo estas bases de datos se pueden abrir con un editor como *Notepad*, este tipo de archivos choca con respecto a otras bases de datos como MySQL ya que en estas últimas una base de datos se compone de una proyecto en el que hay varios archivos, uno por tabla y para poder acceder a sus datos en necesario el uso de una aplicación específica que interprete el proyecto y nos deje acceder a los datos.

Existen desventajas y ventajas de usar un tipo u otro de almacenamiento de datos,[3].

- **SQLite**
  - **Ventajas**
    - Es ligero, al comparar MySQL vs SQLite, vemos que SQLite es una base de datos muy liviana, por lo que es fácil de usar como software integrado en dispositivos: televisores, teléfonos, cámaras, etc.
    - No requiere instalación ni configuración, solamente basta con descargar las bibliotecas SQLite en el ordenador y con esto ya estaremos listos para crear la base de datos.
    - Confiable, actualiza su contenido de manera continua, por lo que se pierde poco o ningún trabajo en caso de que se produzca algún error en la ejecución.
    - Portátil, SQLite funciona en todos los sistemas operativos de 32 y 64 bits, además es posible utilizarlo e integrarlo con todos los lenguajes de programación sin problemas de compatibilidad.
    - Accesible a través de una amplia variedad de herramientas de terceros como por ejemplo DB Browser[4].
  - **Desventajas**
    - No permite concurrencia de conexiones, esto quiere decir que si un usuario está modificando datos, otro no podrá hacerlo a la vez.
    - SQLite, esta orientado para funcionar en aplicaciones que tengan un tráfico bajo o medio. Hoy en día los sitios webs manejan un nivel de tráfico medio-alto.
    - El tamaño de la base de datos está restringido a 2GB en la mayoría de los casos.
- **SQL**
  - **Ventajas**
    - MySQL es reconocido mundialmente por ser el sistema de administración de bases de datos más seguro y confiable utilizado en las aplicaciones web más populares.
    - Garantiza alto rendimiento, ya que este sistema está ideado para dar servicio cliente servidor a una gran velocidad.
    - Ofrece una escalabilidad incomparable para facilitar la administración de aplicaciones profundamente integradas.
    - Posee un tiempo de actividad constante, estando activo las 24 horas del día los 7 días de la semana.
  - **Desventajas**
    - Las bases de datos SQL tienen un gran tamaño, ya es un sistema de gestión orientado a usar y manejar gran cantidad de datos.
    - Al ser un sistema de código abierto las actualizaciones y mejoras del sistema están supeditadas la gente que quiera colaborar lo que en muchos casos se traduce en actualizaciones y mejoras lentas.

- Es un sistema que depende de los recursos que el servidor tenga para garantizar un correcto y ágil funcionamiento de nuestros sitios y aplicaciones.

La sencillez y ligereza de los archivos "*pdb*" exportados por los Embryoscope hace que sea más razonable usar sistemas de datos SQLite, el programa elegido para manejar estos archivos es *BD Browser for SQL*, se puede descargar de forma gratuita en esta página:[4].

He elegido *BD Browser for SQL* por la sencillez de la interfaz de usuario,lo que facilita la compresión de los datos y estructura de la BD por otras personas que no están tan acostumbradas a manejar una bases de datos.

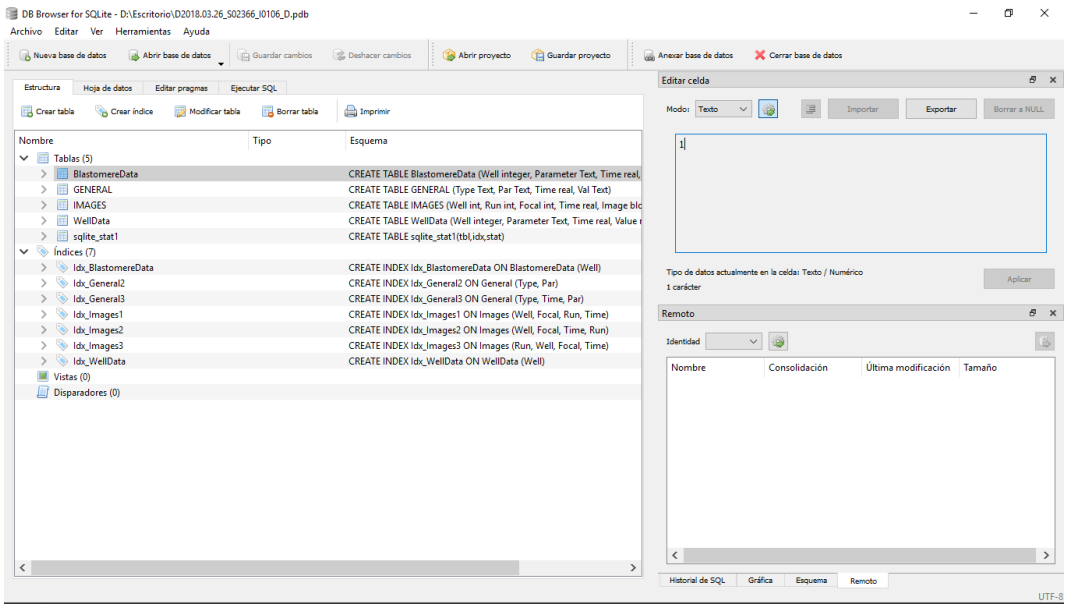


Figura 2.2: Tablas de la BD

La Figura 2.2 muestra la estructura de tablas de la base de datos donde vemos *BlastometerData* y *WellData*, estas tablas que contiene datos que se toman del proceso embrionario pero que para nuestro propósito no son útiles, además existe otra tabla llamada *sqlite_stat1* que nos da información de algún tipo de consulta que se ha realizado a la base de datos durante el proceso de adquisición de muestras, en algunos ".pdb" esta aparece y en otros no por lo que también descartamos su uso. Por tanto de todas las tablas de la imágen nos centraremos den la Tabla *General* e *Images*.

La tabla *Images* tiene cuatro campos (Figura 2.3):

- Well, es el *slot* de cada embrión.
- Run, es un valor que incrementa a medida que se toman imágenes.
- Focal, se toma imágenes a diferentes planos del cigoto, dependiendo del modelo de incubador tenemos más o menos planos.
- Time, instante en el que se toma la imágenes.
- Images, este campo tienes los datos codificados de la imágen.

En la Figura 2.3 se puede apreciar estos campos, por ejemplo en el registro 38 tenemos el *well* 1, esto nos indica que los siguientes campos de la tabla para el mismo registro hacen referencia al embrión que está puesto en el primer *slot* del porta objetos, el segundo campo es el *run* que en este caso es 1, esto significa que es la primera imágen que se ha tomado, en tercer lugar tenemos el *Focal* que es el plano desde el que se ha tomado la imágen para este registro es 0, a continuación tenemos el *Time* mostrándonos el momento en el que se ha tomado la foto, como se puede ver en los registros 37, 38, 39 el campo *Time* tienen el mismo valor, esto es debido a que las imágenes de diferentes planos del mismo *slot* se toman a la vez, por último tenemos el campo Image donde tenemos la imágen del embrión.

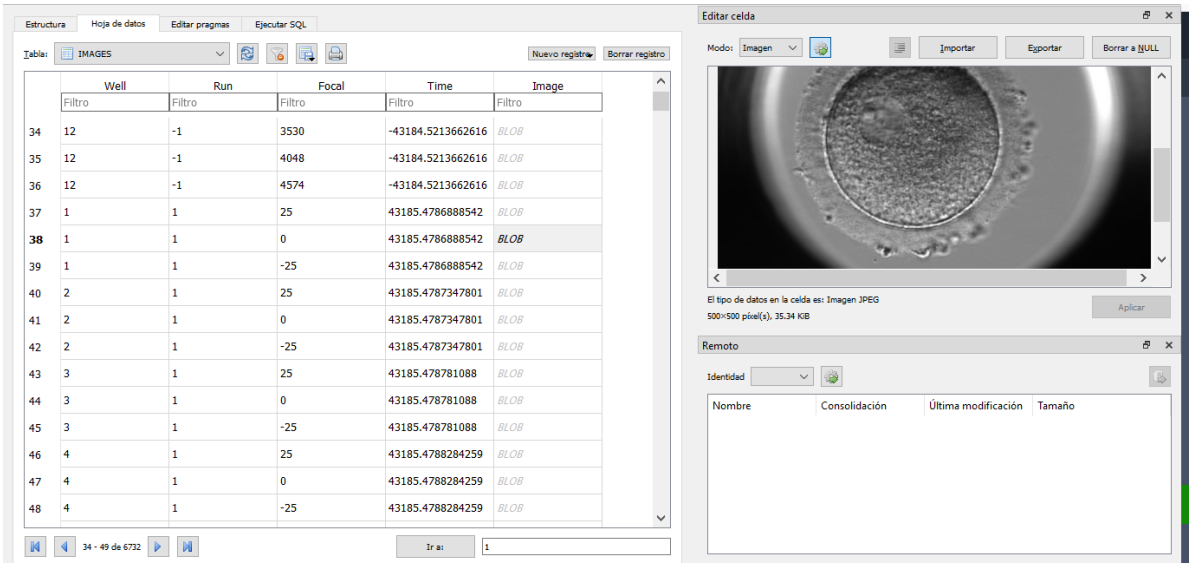


Figura 2.3: Campos de la tabla *Images*

En la tabla *General* tenemos datos como el momento de la fecundación del óvulo, el ID del embrión. el número total de embriones, el ID del *slide*, tipo de tratamiento, el ID del paciente, etc. Todos los parámetros nos son útiles a la hora de saber a qué paciente y tratamientos pertenece la base de datos que se está estudiando, además de permitirnos relacionar los datos del Embryoscope con los proporcionados por el Sivis.

En la Figura 2.4 vemos el aspecto de la tabla *General*, por ejemplo en el registro seleccionado en azul vemos tres campos: *Type*, *Par*, *Time*, *Val*, en este registro tenemos uno de los datos importantes para nosotros, el momento de la fertilización, esto lo sabemos porque tenemos en el campo *Par* el valor "Fertilization" y el tiempo en el campo *Val*, del mismo modo buscando en el campo *Par* podemos encontrar los datos del *slideID*, número de paciente, etc.

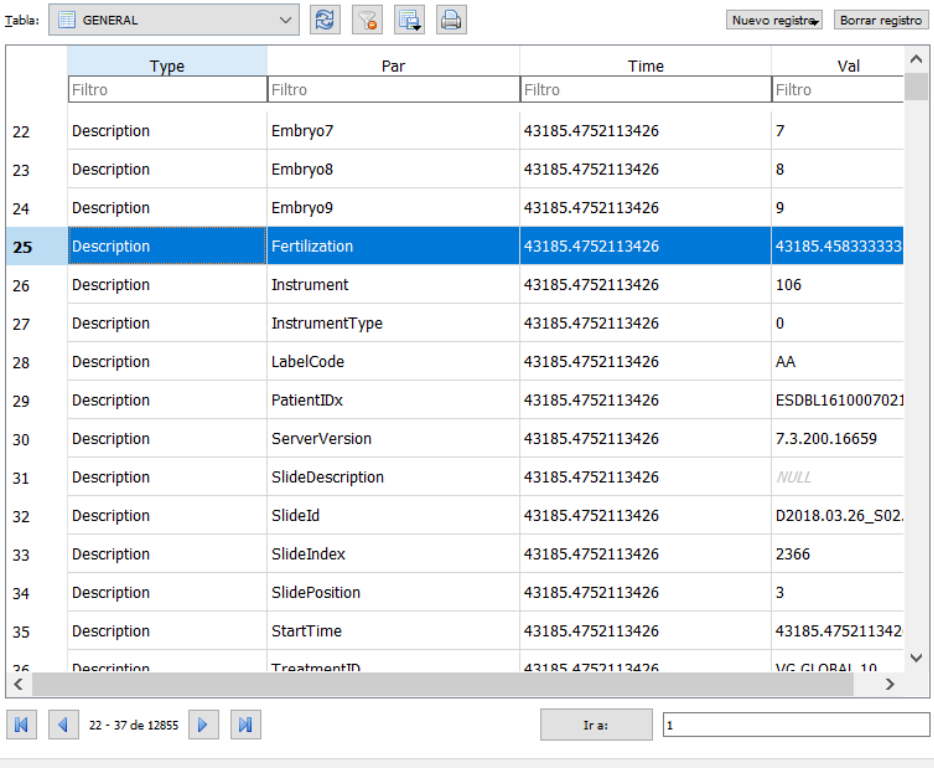


Figura 2.4: Campos de la tabla *General*

2.2.2 Organización de los datos de los Embryoscope

El primer trabajo arduo al que nos enfrentamos en este trabajo fue organizar los datos aportados por el IVI y extraidos del Embryoscope.

Desde el IVI se nos proporcionó dos discos duros con las bases de datos de distintos tipos de tratamiento y distintos modelos de Embryoscope desde 2009 a 2018.

Cada disco duro estaba organizado de forma diferente, En uno las bases de datos exportadas estaban agrupadas por carpetas según el módelo de Embryoscope y en otro lo estaba por año, además algunos datos estaban repetidos.

Así pues el primer trabajo consistió en el análisis, limpieza y organización de los datos de los Embryoscope en directorios siguiendo una estructura clara y ordenada, para ello se siguió el criterio de agrupar los ".pdb" por año y por modelo de Embryoscope como se muestra en la Figura 2.5.

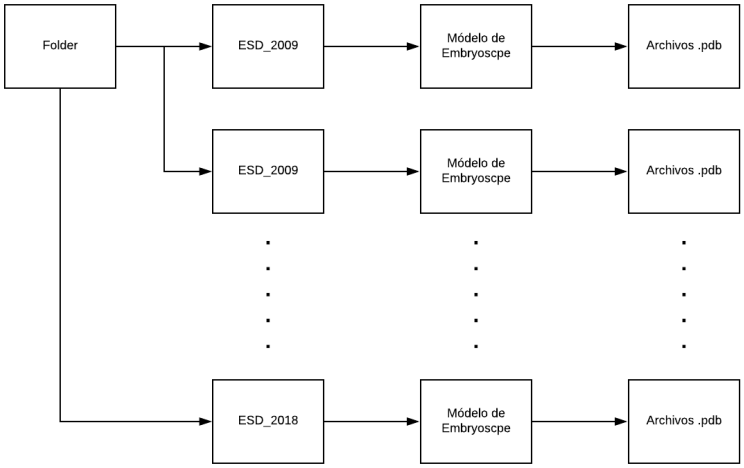


Figura 2.5: Organización de carpetas

2.3 Sivis

El Sivis es el sistema de gestión de historial de pacientes del IVI. La información almacenada en Sivis contiene datos de los Embryocope, anotaciones que se hacen durante el crecimiento embrionario, datos demográficos, parámetros morfocinéticos, tipo de infertilidad, parámetros de la calidad de los óvulos y los ovarios, datos del historial médico, etc. El sistema permite la exportación de todos estos datos en un fichero Excel.

Ya que en el sistema existe una gran cantidad de variables para este proyecto solo se ha exportado los datos que pueden ser relevantes en la creación de los modelos necesarios para relacionar dichos datos con los extraíidos de los Embryoscope.

Los datos extraídos del Sivis son:

- **ID del paciente**: número de historial del paciente.
- **Fecha de punción**: el día en el que se ha fecundado el óvulo.
- **Tasa de implementación**: es un dato que nos mide el éxito de los embriones implantados, los valores son; 0 %, 33 %, 50 % y 100 %. Este parámetro se mide teniendo en cuenta cuantos embriones de los que se han implado se han gestado.
- **Gestación**: este datos se representa de forma binaria, 1 cuando se produce gestación de alguno de los embriones y 0 cuando no es así.
- **Fin del embrión**: este valor es importante porque nos dice que se ha hecho con un embrión ya que este puede ser implantado, no viable o congelado.
- **Número de prtocolo**: valor numérico del tratamiento que se le está dando a la paciente.
- **Tratamiento**: este nos dice si los óvulos son de la misma paciente, en cuyo caso se llama *ICSI* o si son donados *OVODON*.
- **Orden**: es un ID que se le da al embrión en el Embryoscope para poder diferenciarlos cuando hay en crecimiento muchos embriones de una misma paciente.

Datos demográficos como la edad o parámetros morfocinéticos pueden ser importantes pero de momento queremos crear un modelo de predicción centrado en los tiempos de división.

2.4 Estructura de la base de datos BD_IVI_VALENCIA

Como hemos dicho, uno de los objetivos de este proyecto es la creación una base de datos SQLite en la que vamos a unificar datos del Sivis y del Embryoscope, cuya estructura se muestra en la figura 2.6, donde puede observarse las siguientes tablas:

- **ESD**: es la tabla donde están los datos extraídos de las bases de datos del Embryoscope. Tiene seis campos de los cuales los más importantes son:
  - Link: parámetro único formado concatenando ID del embrión y el tratamiento de la paciente.
  - Well: el slot que ocupa cada embrión en el Embryoscope.
  - SlideID: es el ID que se le da al conjunto de embriones estudiados en el Embryoscope para una paciente.
  - EmbryonId, es un ID único que se le da a un embrión para poder identificarlo en un slide.
  - TreatmenID: valor numérico único que tiene un conjunto de embriones de una paciente que han sido igualmente tratados.
  - Slide Description: es una breve información sobre qué tipo de estudio que se está realizando en el incubador.
- **Sivis**: datos extraídos del Sivis. Esta tabla tiene diez campos de los cuales los de mayor relevancia son:
  - Link: se forma de igual manera que el link de la tabla ESD.
  - Gestación: es el valor binario entre 1 y 0 que indica si un embrión implantado ha sido gestado o no.
  - Fecha de Punción: es el día en el que ha sido fecundado el óvulo.
  - Fecha de Trasferencia: fecha en la que se implanta el embrión en el útero de la paciente.
- **ExportImag**: esta tabla es en la que volcamos los datos necesarios para exportar las imágenes que nos son útiles.
  - Link: es el número creado con el que se une los datos de la tabla del Sivis y ESD.
  - Gestación: valor binario 1 o 0 extraído de la hoja del Sivis, este dato lo usamos como etiqueta de éxito o fracaso.
  - SlideID: valor con el que el Embryoscope nombra el ”pbd” exportado.
  - Well: slot en que se pone el embrión dentro del incubador.

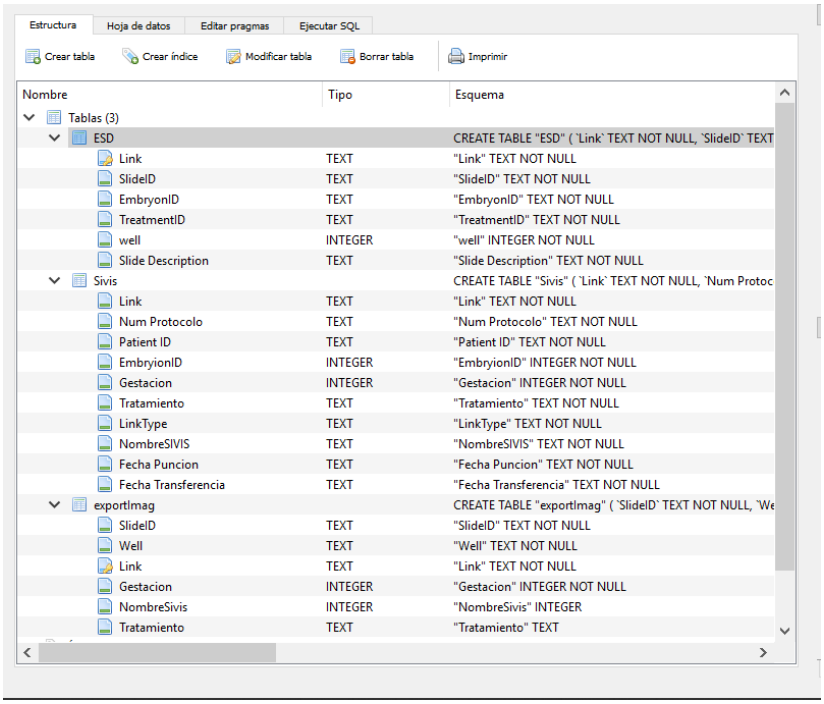


Figura 2.6: Estructuras de tablas base de datos BD_IVI_VALENCIA.

Con estas tablas podemos trazar fácilmente un embrión, por ejemplo, vamos a tomar el registro de un embrión cualquiera que haya sido gestado, Figura 2.7, en este registro vemos que en el campo *Gestación* de la tabla *ExportImag* el valor es uno, lo que significa que esos datos son de un embrión que ha sido gestado, otros datos que tenemos es el *SlideID* y el *Well*, con estos datos sabemos que en el *Well* 3 de la tabla *Imag* de la base de datos *D2018.02.28_S02341_I0106_D* están las imágenes que corresponden a este embrión, además tenemos en el campo link el valor *718080913* que une la tabla *ExportImag* con *ESD* y *Sivis*.

Filtrando por el campo link en las tablas *ESD*, Figura 2.8, y *Sivis*, Figura 2.9, obtenemos otros datos que no son necesarios para exportar las imágenes pero que si son importantes para los modelos de predicción, además nos dan trazabilidad por que en ambas tablas tenemos un campo donde nos dice que archivo ".pdb" o Excel pudiendo en todo momento ir a la fuente datos y comprobar si los datos de BD_IVI_VALENCIA son correctos o no.

Tabla: exportImag

	SlideID	Well	Link	Gestacion	NombreSivis	Tratamiento
	Filtro	Filtro	Filtro	Filtro	Filtro	Filtro
7	D2018.02.28_S02340_I0106_D	8	71808098	1	ExportCristian2014_2018_5.xls	Ovodon/ICSI
8	D2018.02.28_S02341_I0106_D	3	718080913	1	ExportCristian2014_2018_5.xls	Ovodon/ICSI

Figura 2.7: Ejemplo registro de un embrión gestado en la tabla ExportImag.

Tabla: ESD

	Link	SlideID	EmbryonID	TreatmentID	well	Slide Description
	718080913	Filtro	Filtro	Filtro	Filtro	Filtro
1	718080913	D2018.02.28_S02341_I0106_D	13	7180809	3	ESD PEDIDO // TD5/6 // CICLO NAT

Figura 2.8: Ejemplo registro de un embrión gestado en la tabla ESD.

Tabla: Sivis

	Link	Num Protocolo	Patient ID	EmbryionID	Gestacion	Tratamiento	LinkType	NombreSIVIS	Fecha Puncion	Fecha Transferencia
	718080913	Filtro	Filtro	Filtro	Filtro	Filtro	Filtro	Filtro	Filtro	Filtro
1	718080913	7180809	863142	13	1	Ovodon/ICSI	NUM_PROTO...	ExportCristia...	28-feb-2018	05-mar-2018

Figura 2.9: Ejemplo registro de un embrión gestado en la tabla Sivis.

2.4.1 Programación de BD_IVI_VALENCIA

El lenguaje de programación elegido para construir nuestra base de datos ha sido Java [26] [19], usando como entorno de programación Eclipse [6].

Nuestro proyecto en Eclipse se compone de 4 clases: la primera es *EsdManage* donde está la clase principal donde comienza a ejecutarse el código, en segundo lugar tenemos *Sivis* y *dataExcel* con las que vamos a manejar los archivos Excel del Sivis, por último tenemos *dataBase*, para realizar conexiones y manejo de las bases de datos exportadas por el Embryoscope. En los Apéndice A, Apéndice B y Apéndice C se representan más detalladamente las clases programadas , ilustradas mediante sus diagramas de flujo.

La Figura 2.10 representa la forma de proceder que vamos a tener a la hora de desarrollar la construcción de nuestra base de datos en un diagrama ETL, el cual se define como el proceso de extraer, transformar y cargar, se usa para definir aplicaciones de manejo de datos de una forma gráfica.

En primer lugar extraeremos la información de los archivos exportados por los incubadores en la tabla ESD de nuestra base de datos. En segundo lugar haremos lo mismo con las hojas del Sivis, guardando los datos de interés en la tabla Sivis. A continuación relacionando la información de ambas tablas filtraremos y obtendremos los datos que nos son útiles para exportar las imágenes de embriones que han sido gestados pero no han llegado a nacer, etiquetándolos como **fallo** y las imágenes de embriones que han nacido, evento que etiquetamos como **éxito**.

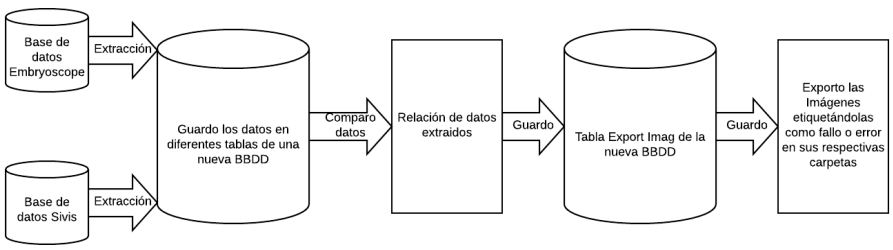


Figura 2.10: Diagrama del proceso ETL (Extract, Transform and Load).

Para poder manejar archivos SQLite y Excel desde Java tendremos que hacer uso de API's en nuestro proyecto.

Una *Application Programming Interface* (API) es una interfaz que nos permite interconectar aplicaciones que no tienen la misma estructura o están programadas en lenguajes totalmente diferentes, por ejemplo con estas herramientas podemos integrar funciones y métodos propios del lenguaje SQL para manejar bases de datos desde una aplicación programada en Java, Python, Visual, Android, etc.

Para nuestro proyecto haremos uso de una API para conectarnos y manejar bases de datos SQLite importando el paquete *java.sql.** [25] en nuestro proyecto.

Para manejar las estructuras Excel usaremos la API *Apache.POI*, para ello descargaremos los paquetes en [5] y los importaremos en nuestro proyecto.

2.4.1.1 Clases, Métodos y Funciones para el manejo de bases de datos del Embryoscope

Para extraer los datos de los Embryoscope hay que acceder a la ruta donde están los datos, subsección 2.2.2, por lo que hay que listar todas las subcarpetas y archivos de la carpeta *ScopDate* para obtener el path absoluto de cada uno de los datos (ficheros ".pdb")

Para ello en primero lugar desde la clase donde se comienza a ejecutar el programa (Apéndice A) vamos a ejecutar el método "*extraerTodosPdb*", que nos va a listar las rutas que necesitamos y se va a conectar a cada base de datos para extraer los datos que necesitamos guardándolos en la tabla *ESD* de la base de datos "*BD_IVI_VALENCIA*", Figura 2.11.

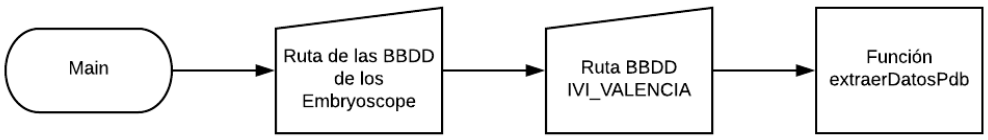


Figura 2.11: Llamada del método ExtraerDatos desde Main

El método *extraerTodosPdb*, Apéndice A, funciona de la siguiente forma:

- 1: Los argumentos del método son: la ruta donde tenemos nuestra BD (*BD_IVI_VALENCIA*) y la ruta donde tenemos el Embryoscope.
- 2: Se creó un objeto con la Api "*java.sql*" con el que poner conectar y manejar la base de datos *BD_IVI_VALENCIA*, denominado" *dBSivisEsd*".
- 3: Usamos la función *listarRutas*, (Apéndice A), para recorrer la carpeta *ScopeData* con el fin de listar en un ArrayList todas las rutas absolutas de cada una de las bases de datos.
  - 3.1: 'Para cada path almacenado en el ArrayList se estable una conexión con las base de datos de un Embryoscope mediante el objeto *getconnection* de Java.sql. A este objeto de conexión le llamamos *firstConnection*.
  - 3.2: Una vez hemos establecido la conexión extraemos los datos, Apéndice B. Para ello le realizamos una consulta sql a la base de datos *firstConnection* pidiéndole los datos tratamientoId, SlideId, EmbryonID, Well, SlideDescription y Link.
  - 3.3: Los datos extraídos los manejamos y casteamos al tipo de datos que queremos para después guardarlos en la BD.

- 3.4: Cerramos la conexión a la base de datos del objeto *firstConnection*.

- 4: Una vez el bucle a recorrido todas las rutas cerramos el objeto *dBSivisEsd*.

Clase *dataBase* se utiliza para realizar todas las funciones que son necesaria para la extraer, transformar y cargar los datos de los archivos ".pdb" exportados por el Embryoscope. Para ello hacemos uso de la API, mediante la cual podemos definir funciones y métodos que nos permitan conectarnos a una base de datos SQLite e interactuar con ella. Apéndice B.

La clase *dataBase*, Apéndice B, está compuesta por las siguientes funciones:

- ***dataBase***: es un constructor, se conecta a la base de datos pasándole como argumento de entrada la ruta donde está la base de datos.
- ***getConnection***: devuelve un objeto *connection* de la API "*java.sql*" con la conexión que ha establecido el constructor.
- ***closeConnection***: cierra la conexión establecida con la base de datos SQLite.
- ***getTreatmentId***: tiene como argumentos el nombre de la base de datos y la ruta donde está situada. Con esto ejecuta una pregunta sql para extraer el TreatmentId, SlideId, EmbryonId, SlideDescription. Estos datos además del link que es creado concatenando el TreatenentID y el EmbryonID, se añaden en una variable ArrayList y es devuelto por la función.
- ***copiDataEsdTable***: este método se encarga de guardar los datos extraídos de los archivos ".pdb" que se pasan como argumento en una matriz "*ArrayList<ArrayList>*" en la que la primera columna es el link y el resto de columnas de la misma fila son los datos relacionados con ese mismo link. Antes de guardar los datos se ejecuta una SQL buscando en la tabla ESD el link que se va a introducir, si este ya está en la tabla no introducimos los datos en la tabla, con esto evitamos tener datos duplicados.
- ***copiSivisTable***: con este método introducimos los datos que se han extraído previamente de los Excel "Sivis", estos datos se pasan como argumento en una matriz del tipo "*ArrayList<ArrayList>*" en la que cada fila está relacionado con un mismo link, por lo que a la hora de guardar en la tabla se busca si este link está ya en la base de datos *BD_ IVI_ VALENCIA*, si no lo está el resto de las columnas de la misma fila son guardadas en distintas variables e introducidas con un String SQL *INSERT INTO SIVIS Values* y los datos por orden de campos en la tabla, después se ejecuta una *executeUpdate()* con el string SQL, para guardar los datos en la tabla.
- ***readPicture*** es el método con el que leemos la tabla *exportImag* de la base de datos *IVI_ VALENCIA* para exportar las imágenes que queremos. Como argumentos se le pasa el link, well, indicador y ruta de destino donde se van a guardar las imágenes.
- ***EscribirElog***: este método ha sido creado para debugear el programa y detectar fallos a la hora de ejecutar SQL, exportar imágenes, problemas de conexiones, base de datos dañadas, etc. Su funcionamiento es sencillo, abre un archivo txt donde se escribe el error que se produzca, para esto el método pide como argumentos la ruta donde se va a guardar el archivo "txt" y lo que se va a escribir en él, esto es útil porque al tener una base de datos de casi 3 terabytes, cuando se ejecuta el código puede tardar bastantes horas con lo que los errores y warnings no son rastreables al no poder estar en frente de la pantalla durante

su ejecución por lo que guardando estos errores podemos consultarlos cuando queramos y detectar que tipo de fallo tenemos y el porque más fácilmente.

2.4.1.2 Manejo y extracción de datos del Embryoscope.

Con la ayuda de las clases, métodos y funciones anteriormente descritas, creamos a una objeto *dataBase* con el que realizamos la conexión a una base de datos de las generadas por el Embryoscope para a continuación mediante la función *getTreatemtID* de la clase *dataBase* realizar preguntas SQL y sacar los valores de TreatmentID, SlideID, SlideDescription y el EmbryonId.

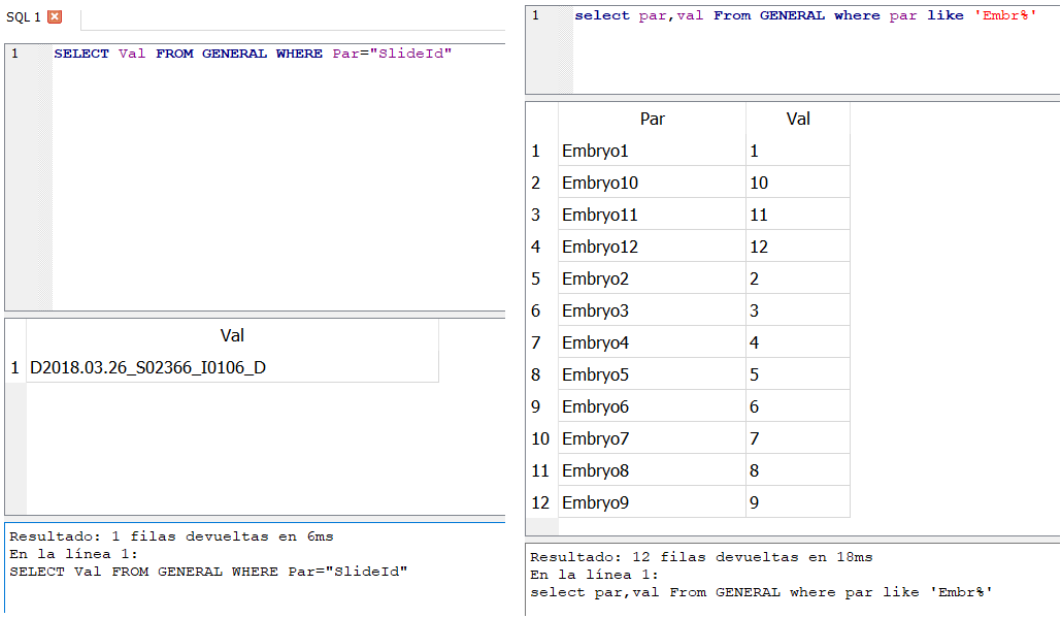


Figura 2.12: Ejemplos de consulta SQl en las tablas ESD y Sivis.

Como se aprecia en la Figura 2.12 dependiendo de los parámetros que se pidan en la consulta (SELECT), la respuesta tiene más o menos campos. Un ejemplo de una consultas en la que se nos devuelven varios campos en la imagen de la derecha de la Figura 2.12, esta ilustra como construir un consulta sql para obtener el EmbryonID y el well. Dichos campos deberían ser almacenados en variables independientes, el campo *Par* nos indica en que posición de la slide se ha puesto un embrión, y el campo *Val* nos indica cuál es el Id del embrión que se ha puesto en esa posición del slide. Estos datos son muy importantes puesto que concatenando el valor del ID del embrión y el número de tratamiento vamos a obtener el parámetro que va a ser único y con el cual vamos a poder trazar fácilmente cualquier embrión, además de ser el parámetro mediante el cual va a ser posible fusionar bases de datos del Embryoscope con los datos del Sivis, como se ve en subsección 2.4.2.

Cabe mencionar que el valor TreatmentId no está presente en los archivos ".pdb" exportados por los Embryoscope anteriores al 2014, por lo que en este momento aunque tengamos datos del 2009 a 2018 solo podremos usar los posteriores a 2014. Se ha buscado otra forma de poder relacionar los datos de los embriones del Embryoscope y el Sivis pero en estas bases de datos no hay ningún otro parámetro con el que poder conformar algún tipo de link único.

Una vez tenemos los datos, debemos introducirlos en la tabla usando para ello el método *copiEsdTable*, Apéndice B, este método realiza un consulta SQl del tipo *INSERT INTO* para introducir los datos en la tabla ESD (Link, SlideId, EmbryonID, TreatmentID, Well, Slide Description). Cuando este proceso ha acabado cerramos la base de datos de donde estamos extrayendo los datos, archivo ".pdb" del Embryoscope, y la base de datos *BD_IVI_VALENCIA*.

Tabla: ESD — Nuevo registro — Borrar registro

	Link	SlideID	EmbryonID	TreatmentID	well	Slide Description
	Filtro	Filtro	Filtro	Filtro	Filtro	Filtro
1	v5 test1	D2014.11.26_S1545_I016	1	v5 test	1	
2	v5 test10	D2014.11.26_S1545_I016	10	v5 test	10	
3	v5 test11	D2014.11.26_S1545_I016	11	v5 test	11	
4	v5 test12	D2014.11.26_S1545_I016	12	v5 test	12	
5	v5 test2	D2014.11.26_S1545_I016	2	v5 test	2	
6	v5 test3	D2014.11.26_S1545_I016	3	v5 test	3	
7	v5 test4	D2014.11.26_S1545_I016	4	v5 test	4	
8	v5 test5	D2014.11.26_S1545_I016	5	v5 test	5	
9	v5 test6	D2014.11.26_S1545_I016	6	v5 test	6	
10	v5 test7	D2014.11.26_S1545_I016	7	v5 test	7	
11	v5 test8	D2014.11.26_S1545_I016	8	v5 test	8	
12	v5 test9	D2014.11.26_S1545_I016	9	v5 test	9	
13	21424361	D2014.11.26_S1546_I016	1	2142436	1	emb. si que se cambia
14	21424362	D2014.11.26_S1546_I016	2	2142436	2	emb. si que se cambia
15	21424363	D2014.11.26_S1546_I016	3	2142436	3	emb. si que se cambia
16	21424364	D2014.11.26_S1546_I016	4	2142436	4	emb. si que se cambia
17	21424365	D2014.11.26_S1546_I016	5	2142436	5	emb. si que se cambia
18	21424366	D2014.11.26_S1546_I016	6	2142436	6	emb. si que se cambia
19	21424367	D2014.11.26_S1546_I016	7	2142436	7	emb. si que se cambia
20	21424368	D2014.11.26_S1546_I016	8	2142436	8	emb. si que se cambia
21	21424369	D2014.11.26_S1547_I016	9	2142436	1	slide 2. step. no se cambia
22	214243610	D2014.11.26_S1547_I016	10	2142436	2	slide 2. step. no se cambia
23	214243611	D2014.11.26_S1547_I016	11	2142436	3	slide 2. step. no se cambia
24	214243612	D2014.11.26_S1547_I016	12	2142436	4	slide 2. step. no se cambia

Figura 2.13: Tabla ESD de BD_IVI_VALENCIA.

La Figura 2.13, ilustra el resultado de la extracción, tranformación y guardado de los datos de los Embryoscope. Se puede ver que la tabla contiene 6 campos: Link, SlideID, Embryon, TreatmentID, Well y Slide Description, aunque de estos datos solo necesitamos el link, el SlideID y el Well para poder Exportar las imágenes del embrión, el resto de campos es importante tenerlos porque nos aporta más información acerca del embrión y además nos da trazabilidad para poder hallar el ".pdb" asociado a ese embrión pudiendo así en un futuro incrementar esta tabla buscando más datos filtrando por el link.

2.4.1.3 Manejo y extracción de los datos de Sivis

De los datos del sivis disponemos de una exportación de todos los datos disponibles en el IVI desde 2014 a 2018 de la clínica de Valencia. Esta exportación a diferencia de las que suelen realizar los especialistas en medicina reproductiva no contiene datos morfocinéticos o demográficos ya que para nuestro modelo solo necesitamos información sobre embriones gestados. Así pues los datos de Sivis son:

- La gestación: que tiene un valor de 1 para los embriones gestados y 0 para los que no lo han hecho.
- La tasa de implementación: que nos indica qué porcentaje de embriones que han sido transferidos se han gestado. Las cifras habituales son el 0 %, 50 % y 100 % aunque en el caso de

los datos más antiguos se daba porcentajes del 33.3 % y 66.6 %. Esto se debe a que antes se implantaban 3 óvulos a la vez para incrementar las posibilidades de que alguno de ellos se gestara, mientras que hoy en día las técnicas de reproducción asistida han evolucionado elevando la tasa de éxito por lo que se implantan únicamente dos embriones.

- Número de Ovocito: este parámetro es como el ID del embrión.
- Fin Final: nos dice qué medidas se han tomado con el embrión, hay tres estados posibles:
  - Descartado.
  - Congelado.
  - Transferido.

  Nos centraremos únicamente en los transferidos, porque en esas bases de datos tendremos todo el proceso de crecimiento del embrión lo que es importante para poder medir de una forma correcta los tiempos de división.
- ID del paciente.
- Número de protocolo: es un número único que se relaciona un conjunto de embriones bajo el mismo ID de paciente y tratamiento.

Como resultado de la exportación de datos del sivis se tiene como resultado una hoja excel bastante grande, por lo que por comodidad y espacio en memoria durante la ejecución del programa se ha dividido en 6 hojas diferentes a las cuales se le ha llamado "Sivis"+"i" siendo i un número de hoja del 1 al 6 Figura A.1.

Los datos de estas hojas los vamos a agrupar en éxito y fracaso, para seleccionar los embriones en uno u otro grupo filtraremos por gestación, tasa de implementación y Fin final según se muestra en la tabla 2.1.

Grupo	Gestación	Tasa de implementación	Fin final
Éxito	1	100 %	Transferido
Fracaso	0	0 %	Transferido

Tabla 2.1: Filtrado de datos de la hoja Sivis.

Descartamos los embriones con 33.3 %, 50 % y 66.6 % de tasa de implementación por no poder saber de forma correcta cual de los embriones transferidos se ha gestado, por eso solo tomaremos los datos de casos en los que se han transferido dos o tres embriones y todos se han gestado (100 %) o no (0 %).

Para obtener los datos del Sivis, desde la clase principal Main (Apéndice A), recorremos las 6 hojas excel y para cada una de ellas extraemos los datos mediante el método *extraerDatosSivs*, figura A.4, en la cual creamos un objeto de la API "*Apache.POI*" para abrir el excel y manejar los datos. Las funciones y métodos para realizar este proceso lo hemos implementado en la clase *dataExcel* (Apéndice C). Una vez creado el objeto, usamos el método *dataBuscarGestación* figura C.1,la cual nos devuelve una lista(arrayList) con los datos que cumplen las características requeridas en el argumento de la función, (parámetros de la Tabla 2.1).

Una vez tenemos los datos filtrados debemos introducirlos en la tabla Sivis dentro de la base de datos *BD_IVI_VALENCIA*, proceso que se realiza mediante el método *copiSivisTable*, (Figura B.2, Apéndice C).

Tabla: Sivis — Nuevo registro — Borrar registro

	Link	Num Protocolo	Patient ID	EmbryionID	Gestacion	Tratamiento	LinkType	HojaExcel	FechaPuncion	FechaTransferencia
	Filtro	Filtro	Filtro	Filtro	Filtro	Filtro	Filtro	Filtro	Filtro	Filtro
1	71816356	7181635	999958	6	1	Ovodon/ICSI	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls	30-abr-2018	05-may-2018
2	71824544	7182454	999502	4	1	Ovodon/ICSI	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls	26-jun-2018	01-jul-2018
3	7181868D15	7181868D1	998967	5	1	T. congelados	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls		19-jun-2018
4	71821079	7182107	998644	9	1	Ovodon/ICSI	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls	30-may-2018	04-jun-2018
5	718161015	7181610	998550	15	1	Ovodon/ICSI	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls	27-abr-2018	02-may-2018
6	71816825	7181682	998378	5	1	Ovodon/ICSI	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls	02-may-2018	07-may-2018
7	71818162	7181816	997838	2	1	ICSI	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls	11-may-2018	17-may-2018
8	7181909D17	7181909D1	997812	7	1	T. congelados	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls		15-jun-2018
9	71821157	7182115	997454	7	1	Ovodon/ICSI	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls	31-may-2018	05-jun-2018
10	7181208D11	7181208D1	997268	1	1	T. congelados	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls		15-may-2018
11	71819936	7181993	997263	6	1	ICSI	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls	22-may-2018	27-may-2018
12	7181408D11	7181408D1	997242	1	1	T. congelados	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls		15-jun-2018
13	71819522	7181952	996894	2	0	ICSI	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls	19-may-2018	25-may-2018
14	71810655	7181065	996855	5	0	ICSI	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls	16-mar-2018	21-mar-2018
15	7181772D14	7181772D1	996604	4	0	T. congelados	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls		13-jun-2018
16	71815502	7181550	996559	2	1	ICSI	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls	24-abr-2018	29-abr-2018
17	71821056	7182105	996128	6	0	ICSI	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls	31-may-2018	05-jun-2018
18	718231810	7182318	995989	10	0	Ovodon/ICSI	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls	15-jun-2018	20-jun-2018
19	71817952	7181795	995589	2	1	ICSI	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls	10-may-2018	15-may-2018
20	71817958	7181795	995589	8	1	ICSI	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls	10-may-2018	15-may-2018
21	71819019	7181901	995451	9	1	Ovodon/ICSI	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls	16-may-2018	21-may-2018
22	7181492D11	7181492D1	995448	1	0	T. congelados	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls		11-jun-2018
23	7181353D12	7181353D1	995221	2	1	T. congelados	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls		25-may-2018
24	71820671	7182067	995207	1	0	Ovodon/ICSI	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls	28-may-2018	03-jun-2018
25	71820672	7182067	995207	2	0	Ovodon/ICSI	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls	28-may-2018	03-jun-2018
26	7181858D17	7181858D1	994894	7	1	T. congelados	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls		11-jun-2018
27	71815058	7181505	994634	8	0	Ovodon/ICSI	NUM_PROTOCOLO + OVOCITO	ExportCristian2014_2018_5.xls	20-abr-2018	25-abr-2018

Figura 2.14: Tabla Sivis de BD_IVI_VALENCIA.

El resultado de la extracción de los datos de Sivis se guardan en la tabla Sivis (Figura 2.14) en nuestra base de datos *BD_IVI_VALENCIA*, los campos de dicha tabla son:

- **Link**: es el número único que nos hemos creado para relacionar los datos del Sivis y de los Embryoscope.
- **Num Protocolo**: valor que relaciona el historial de un paciente y el tratamiento bajo el cual está sometido.
- **patienID**: numero del historia del paciente.
- **EmbryonID**: es el valor identificativo que se le da a un embrión dentro de un conjunto que están bajo el mimo tratamiento.
- **Gestacion**: valor binario que nos indica si el embrión al ser implantado en el útero de la paciente se ha gestado dando lugar a un embarazo (1) o por el contrario no lo ha hecho (0).
- **Tratamiento**: nos indica que tecnicas FIV se han usado con los gametos y también la procedencia de los óvulos:
  - Procedencia: el óvulo puede ser donado (OVODON) o propio. En la tabla Sivis cuando sea un óvulo donando aparecerá la etiqueta "*OVODON*" en el campo Tratamiento, por

el contrario si en dicho campo no hay ningún dato de procedencia es un óvulo propio. También existe la etiqueta "*T.congelado*"lo que nos indica que ese embrión es un cigoto cuyo crecimiento embrionario fue parado congelándolo.

- Técnica de fecundación :
  - FIV: en esta técnica se selecciona el óvulo y un conjunto de espermatozoides los cuales se sitúan junto al óvulo en un medio de fertilización dejando que sea el primer espermatozoide el que fecunde el óvulo.
  - ICSI: su siglas significan "inyección intracitoplasmática de espermatozoides", en esta técnica a diferencia del FIV solo se selecciona un espermatozoide con el que se fecunda directamente el embrión con una microinyección mediante una microaguja. Este tipo de tratamiento está indicado especialmente para diagnósticos de factor masculino severo, con espermatozoides con poca movilidad o bajo número de ellos.

- **LinkType**: este parámetro nos dice de que forma hemos realizado el link, qué datos hemos concatenado para la creación de este.
- **HojaExcel**: procedencindia de los datos en que excel están.
- **FechaPuncion**: la fecha en la que ha sido fecundado el embrión, como se puede apreciar en la Figura 2.14 solo los embriones "frescos" tienen este campo, los embriones de procedencia "T. congelados" no la tienen.
- **FechaTransferencia**: la fecha en la que ha sido transferido el embrión al útero de la paciente.

2.4.2 Fusión de los datos de Embryoscope y Sivis

Una vez tenemos las tablas ESD y Sivis completas tenemos que relacionarlas, para ello, como hemos comentado, hemos introducido en las tablas *ESD* y *SIVIS* el campo "*Link*". Con este parámetro relacionamos ambas tablas, asegurándonos de que esos datos pertenecen a un mismo embrión.

Los datos que vamos a buscar en la tabla ESD son: SlideID, Well y Link. De la tabla Sivis tomaremos: NombreSivis, Tratamiento y Gestacion y Link. De los datos tomados de ambas tablas filtramos por Gestacion y Link, con estos rellenaremos la tabla "*exportImag*", Figura 2.15, de la base de datos *BD_IVI_VALENCIA*, para ello, mediante Java hacemos uso de la API para enviar un comando "*executeUpdate*" con la siguiente instrucción SQLite.

```
INSERT INTO exportImag (
    SlideID,Well,Link,Gestacion,NombreSivis,Tratamiento
)
select
    ESD.SlideID,ESD.well,Sivis.Link,Sivis.Gestacion, Sivis.HojaExcel,
    Sivis.Tratamiento
    From Sivis inner Join ESD
    where Sivis.Link=ESD.Link AND Sivis.Gestacion=1
    AND Sivis.Tratamiento<>'T. congelados'
```

En dicha instrucción lo que se hace es introducir los datos de los campos; SlideID, Well, Link, Gestacion, NombreSivis y Tratamiento en la tabla *exportImag* seleccionados de las tablas *ESD* (Figura 2.13) y *SIVIS* (Figura 2.14) que tengan el mismo "link'($Sivis.Link = ESD.Link$),se hayan gestado (Sivis.Gestacion = 1) y que no sean embriones congelado ($Sivis.Tratamiento <> 'T. congelados'$).

Tabla: exportImag — Nuevo registro — Borrar r

	SlideID	Well	Link	Gestacion	NombreSivis	Tratamiento
	Filtro	Filtro	Filtro	Filtro	Filtro	Filtro
127	D2015.07.10_S1353_I106	1	21513861	1	ExportCristian2014_2018_2.xls	Ovodon/ICSI
128	D2015.07.10_S1353_I106	9	21513869	1	ExportCristian2014_2018_2.xls	Ovodon/ICSI
129	D2015.07.10_S1733_I016	3	21513433	1	ExportCristian2014_2018_2.xls	Ovodon/ICSI
130	D2015.07.10_S1733_I016	6	21513436	1	ExportCristian2014_2018_2.xls	Ovodon/ICSI
131	D2015.07.12_S1735_I016	4	21513833	1	ExportCristian2014_2018_2.xls	Ovodon/ICSI
132	D2015.07.12_S1736_I016	11	21513838	1	ExportCristian2014_2018_2.xls	Ovodon/ICSI
133	D2015.07.14_S1738_I016	2	21514032	1	ExportCristian2014_2018_2.xls	Ovodon/ICSI
134	D2015.07.14_S1738_I016	10	215140310	1	ExportCristian2014_2018_2.xls	Ovodon/ICSI
135	D2015.07.15_S1740_I016	2	21514122	1	ExportCristian2014_2018_2.xls	Ovodon/ICSI
136	D2015.07.15_S1740_I016	12	215141212	1	ExportCristian2014_2018_2.xls	Ovodon/ICSI
137	D2015.07.16_S1364_I106	2	21514312	1	ExportCristian2014_2018_2.xls	Ovodon/ICSI
138	D2015.07.16_S1364_I106	10	215143110	1	ExportCristian2014_2018_2.xls	Ovodon/ICSI
139	D2015.07.17_S1742_I016	2	21514572	1	ExportCristian2014_2018_2.xls	Ovodon/ICSI

Figura 2.15: Tabla exportImag de BD_IVI_VALENCIA.

La tabla Figura 2.15 contiene los datos necesarios para poder extraer las imágenes de los embriones fracaso y éxito de los archivos ".pdb", de esta forma realizando una consulta en la tabla *exportImag* de la base de datos *BD_IVI_ VALENCIA* obtenemos el nombre del ".pdb" y el *Well* donde están todas las imágenes del crecimiento embrionario de un determinado embrión.

2.4.3 Exportación de imágenes

Una vez ya hemos tratado todas las bases de datos, las hemos fusionado y las hemos filtrado, obtenido los datos que nos son de interés, estos datos se agrupan en fracaso y éxito, y nos dan información para poder trazar los datos de los embriones en las hojas del sivis y en las bases de datos de los Embryoscope.

Para el objetivo principal de este bloque lo que importa es la extracción de las imágenes éxito y fracaso contenidas en las bases de datos de los Embryoscope, para lo cual nos hemos creado la tabla *exportImag* (Figura 2.15).

Antes de exportar las imágenes tenemos que crear un sistema de directorios donde guardar las imágenes de forma estructurada y siguiendo un orden, para ello primero creamos una carpeta llamada Imágenes desde donde colgarán el resto de carpetas con la siguiente estructura:

- Imágenes
  - EXITO_FRESCOS: carpeta que contiene las imágenes de los embriones gestados.
    - Número de Link: carpeta donde están todas las imágenes de un embrión.

        - ◇ Link_Time-x_Frg-x.jpg: nombre con que se guardan las imágenes exportadas.
    - FRACASO_FRESCOS: carpeta que contiene las imágenes de los embriones no gestados.
        - ○ Número de Link
            - ◇ Link_Time-x_Frg-x.jpg.

La Figura 2.16 muestra un ejemplo de una carpeta donde se ha exportado todas las imágenes de un mismo embrión, en la carpeta cada imagen está etiquetada siguiendo la estructura que se ha mencionado, por ejemplo 11528672_Time.067_Frg1.jpg, lo que nos indica que todas las imágenes de esta carpeta corresponden al mismo embrión. Todas las imágenes se etiquetan igual, primero se pone el *Link*, a continuación está el tiempo del *frame*, este tiempo está en la escala de minutos por tanto el primer *frame* es tomado a los 0.067 lo que quiere decir que $60x0{,}067 = 4{,}02$ minutos el Embryoscope ha tomado la priemera imagen, por último tenemos el valor del *frame* que en esta caso al ser la primera imagen tenemos *Frg-1*.

Definida la estructura de carpetas exportamos las imágenes.

- 1: Tenemos que leer la base de datos para exportar las imágenes, para lo cual abrimos la base de datos creando un objeto de la clase *dataBase* y estableciendo la conexión.
- 2: Realizamos una consulta sql a la base de datos para buscar el tipo de imágenes que queremos (éxito o fracaso). Para ello ejecutamos una consulta sql filtrando con 1 (éxito) o un 0 (fracaso) en el campo gestación para obtener: SlideID, Well y Link con todas los embriones que cumplen la condición de gestación de la tabla *exportImag* (Figura 2.15). La respuesta con los datos de los campos los guardamos en una variable del tipo ArrayList<ArrayList>que llamamos ”datosExport”.
- 3: Usando la función *listarRuta*, Figura A.3 pasándole como argumento la ruta de la carpeta donde están todas las base de datos SQLite obtenemos un arrayList con las rutas de cada uno de los ”.pdb” de los Embryoscope.
- 4: para cada ”.pdb” del arrayList del punto anterior recorro la variable ”datosExport”.
    - 4.1: Como el nombre de las bases de datos contiene el nombre del slideID, comparo si la posición i del ”listaRutas”, el nombre de una base de datos, contiene un valor igual al valor SlideID ”datosExport”.

      ```
      boolean found = listaRutas.get(i).indexOf(datosExport.get(0).
      get(j).toString()) !=-1? true: false;
      ```

    - 4.2: Si la comparación es true tomamos la ruta de la base de datos y creamos una objeto *dataBase* para conectarnos a ella.
    - 4.3: Llamamos a la función *readPicture* pasándole como argumentos el Link, Well, Indicador y la ruta de la base de datos.
    - 4.4: La función *readPicture*, Figura B.3, realiza una consulta sql para obtener los datos y exportar las imágenes que cumplen la condición de gestación.

      ```
      SELECT run,Time,image FROM IMAGES WHERE focal=0 and well="+Well;
      ```

con la que se obtiene:

- run: es un parámetro que aumenta con cada foto que se toma, nos sirve para saber la secuencia de las imágenes.
- Time es el tiempo en el que se ha tomado la imagen, este tiempo lo normalizamos respecto al momento en el que es fecundo el óvulo, este parámetro está en la tabla "*GENERAL*" con la etiqueta "*Fertilization*" en los archivos ".pdb" exportados por el Embryoscope. La escala está en días por lo que la pasamos a minutos por comodidad, este cambio de escala lo realizamos con una simple regla de tres. Un día tiene $24x60$ minutos por lo que el tiempo normalizado multiplicado por 1440 nos da el tiempo en minutos de cada imagen en minutos.
- Imagen: son los datos en crudo de las imágenes, en las primeras pruebas se exportó a formato "TIFF" y se comparó si se perdía calidad con respecto a exportaciones en "JPG" y no fue así por lo que se decidió que a igual calidad exportaríamos en "JPG" ya que al ser un formato de compresión que pesa menos ahorraríamos espacio en memoria.

  A la hora de exportar se tiene en cuenta las horas que se ha dejado el embrión en el incubador, guardando únicamente las imágenes de crecimientos embrionarios que tiene una duración superior a 60 horas.

  Este límite nos garantiza que los embriones ya se han dividido en más de 5 células, además así evitamos exportar imágenes de embriones que han sido congelados o inviables.

  Por otro lado cuando se está exportando cortamos en las 80 horas ya que normalmente entre las 70-80 horas hay ya más de 5 células y es un margen bastante amplio para que los embriones más rezagados se hayan dividido, de esta forma ahorramos espacio no exportando imágenes de embriones demasiado desarrollados y que no nos aportan datos para nuestro modelo.

- 5: Una vez se han exportados todas las imágenes cerramos la base de datos y pasamos a la siguiente.

Las imágenes se exportan en las rutas correspondientse guardándolas con el run, link y el tiempo, esto es muy importante porque a la hora de pasar las muestras por el estimador de tiempo de división se tomará el tiempo del nombre de la imagen. Como resultado de este proceso obtenemos directorios como el de la Figura 2.16.

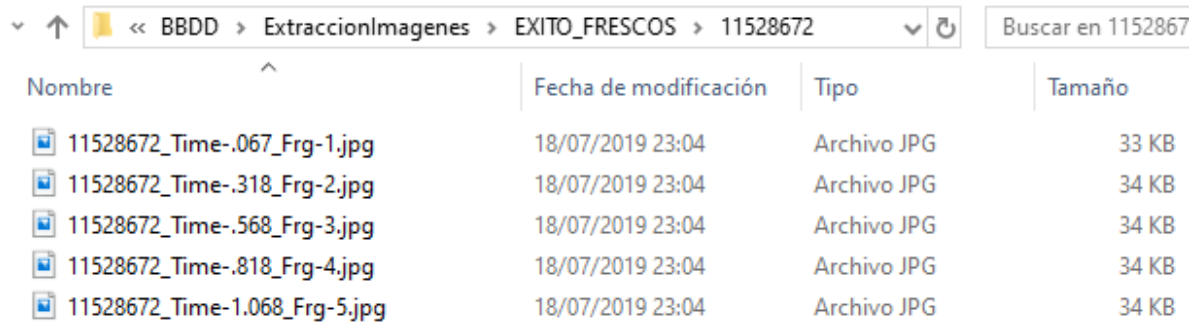


Figura 2.16: Directorio de una carpeta con imágenes de un embrión gestado.

Capítulo 3

Deep Learning para la estimación de la división célular en embriones

La inteligencia artificial nace a raíz del interés de realizar procesos de razonamiento y aprendizaje humano en máquinas. El momento fundacional de la IA fue una conferencia organizada por Jhon McCarthy en la Universidad de Darmouth en 1956, en esta conferencia Jhon McCarthy definió la inteligencia artificial (IA) como: "La ciencia y la ingeniería de crear máquinas inteligentes, especialmente programas de computación inteligentes. Está relacionada con la tarea similar de utilizar ordenadores para comprender la inteligencia humana, pero la IA no se limita a métodos que sean observables biológicamente"[23].

Podemos definir IA también como el objetivo de simular la inteligencia humana mediante procesos como el aprendizaje, el razonamiento y la auto correlación.

La IA busca dar solución a múltiples problemas de gran complejidad, como por ejemplo conseguir que una máquina aprenda de forma automática haciendo que actúe sin programación. Por la complejidad de los problemas que se enfrentan nacen subconjuntos de técnicas enmarcadas dentro del IA que pretende dar solución a problemas de menos complejida.

El Machine Learning nació en los años 50 cuando el informático teórico Arthur L.Samuel programó el primer algoritmo de Machine Learning. El programa de aprendizaje informático, consistía en un juego de damas en el que la máquina mejoraba conforme pasaban las partidas. Se considera Machine Learning al conjunto de técnicas mediante las cuales las máquinas tomando un conjunto de datos grande y detecta en ellos patrones con lo que poder hacer predicciones. Dentro de estás técnicas existen los modelos supervisados los cuales aprenden a partir de una base de datos tratada y con datos clasificada o los modelos no supervisados a los cuales se les da una base de datos y ellos mismo extraen y clasifican las muestras de datos.

Por otro lado tenemos el Deep Learning, estas técnicas son un subconjunto que hace parte del Machine Learning, su inicio lo encontramos en la creación del perceptrón multicapa, en 1957 Frank Rosenblatt comenzó el desarrollo de la primera red neuronal: el perceprón, en 1958 publicó en un articulo, [29],la capacidad de aprender y fue en ese momento donde nació la primera red neuronal de *deep learning*, este tipo de red neuronal fue discutida en 1969 por Minsky Marvin y Papert Seymour [21], demostrando que el perceptrón muticapa no era capaz de resolver problemas de no linealidad. Esto hizo que la investigación en este campo parara y no se le diera demasiada

importancia, hasta que con el avance tecnológico y matemático se produjo un empujón en este campo llegando a resolver el problema de no linealidad e investigando en nuevas redes como las redes convolucionales.

3.1 Deep Learning

Las redes neuronales intentan emular la capacidad de memorizar y relacionar hechos de los humanos, estas capacidades las adquirimos mediante experiencia es por ello que las redes neurales se centran en aprender usando la experiencia, esto quiere decir que buscan poder predecir y relacionar objetos a partir del conocimiento que han adquirido con mustras ya vistas por la red.

Los sistemas de *deep learning* se basa estructuras cuyo elemento base son las neuronas, es por ello que se les llama redes neuronales. En el cerebro humano las neuronas están interconectadas transmiten información a partir de impulsos nerviosos llamados sinapsis, este impulso es tomado por una neurona como entrada, se trata su información dentro de esta y retransmitida a la siguiente, este sistema compuesto por billones de neuronas es el que dota a los humanos de inteligencia pero no de conocimiento puesto que para adquirir conocimientos tenemos que nutrir nuestro cerebro con experiencias de las cuales extraer información aprendiendo de ellas y pudiendo relacionar distintos echos e incluso memorizar.

El comportamiento que se busca usando redes neuronales es el mismo que en nuestro cerebro, aprender a partir de la experiencia, en nuestro caso la experiencia es un conjunto de datos etiquetados bajo un *ground truth* con los cuales alimentamos la entrada de nuestra red, la cual se encarga de pasar la información de estos datos entre las distintas neuronas que van aprendiendo cada una algo de los datos con el fin de entre todas detectar patrones o elementos comunes entre los datos que pertenecen a una misma clase o etiqueta.

Las redes neuronales están compuestas por neuronas que se comunican unas con otras a través de la sinapsis en la que se van trasmitiendo unas a otras un pulso electromágnetico que tiene información, por lo que cada neurona tiene una entrada de la señal,se modifica o no en la neurona y se retrasmite por la salida a la siguiente.

Por tanto las redes neuronales:

- Están compuestas por "neuronas"que funcionan como unidades de procesamiento de datos encargadas de intercambiar datos.
- Gracias a su capacidad de aprender a partir de la experiencia pueden ser entradas para reconocer patrones como imágenes, manuscritos o tendencias de datos.
- Pueden aprender y mejorar su funcionamiento a medida que van entrando.

Para realizar el proceso de aprendizaje las neuronas se distribuyen en la red de la siguiente manera,figura 3.1:

- La capa de entrada que es por donde tomamos los datos con lo que vamos a entrenar nuesto modelo de predicción.

- *Hidden layers* o capas ocultas, son las capas que extraen información de los datos con el fin de alcanzar un conocimiento con el que poder predecir la clase de un dato de entrada de forma autónoma.
- La capa de salida nos etiqueta la entrada.

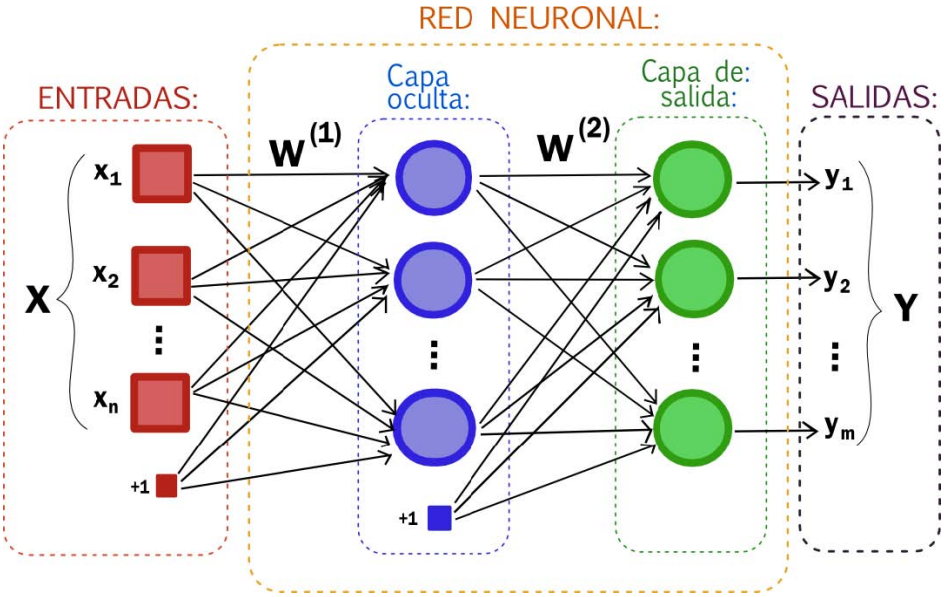


Figura 3.1: Estructura de una red Neuronal [1]

Una de las características más importantes de este tipo de estructuras es la capacidad de abstracción siendo capaz de centrarse en las *features* más importante de un dato de entrada, dejando de lados los irrelevantes, pero esta no es la única característica:

- Posee la capacidad de aprendizaje adaptativo ya que al igual que el cerebro humano aprende a partir de las experiencias o datos.
- Tolerancia a fallos: en caso de sufrir daños en una parte de la red, determinadas características de esta pueden mantenerse.
- Existe tecnología como microchips que hacen que se puedan integrar estos modelos de aprendizaje supervisado en sistemas ya existentes.
- Los cálculos de los parámetros de la red se pueden calcular en paralelo haciendo que la operación de aprendizaje sea más rápida.

3.1.1 Red neuronal convolucional

Las *Convolutional Neural Networks* o CNN son unas técnicas de *Deep learning* que se basan en el modelo del Neocognitron desarrollado por Kunihiko Fukushima [7] posteriormente en 1998 1998 Yann LeCu elaboró el método del backpropagation como técnica mediante la cual la red aprendiera [17], pero no fue hasta después de 2010 que se refinaría dicha técnica implementandolas en GPU's dando muy buenos resultados, por lo que en ese momento se popularizó este tipo de redes de *deep learning*.

Las CNN son usadas principalmente para clasificar imágenes, es te tipo de red Neuronal es una red de aprendizaje supervisado ya que para entrenarla se requiere de un gran número de datos previamente clasificados y etiquetados los cuales usa la red para aprender. Su funcionamiento imita al de el cortex del ojo humano ya que la red se compone de distintas capas encargadas de detectar formas diferentes con las que clasificar un objeto.

La arquitectura de las CNN se compone de la unión de diferentes capas que van extrayendo diferentes características o *features* de las imágenes con las cuales poder predecir de que tipo de objeto se trata, pudiendo así clasificarla, Figura 3.2.

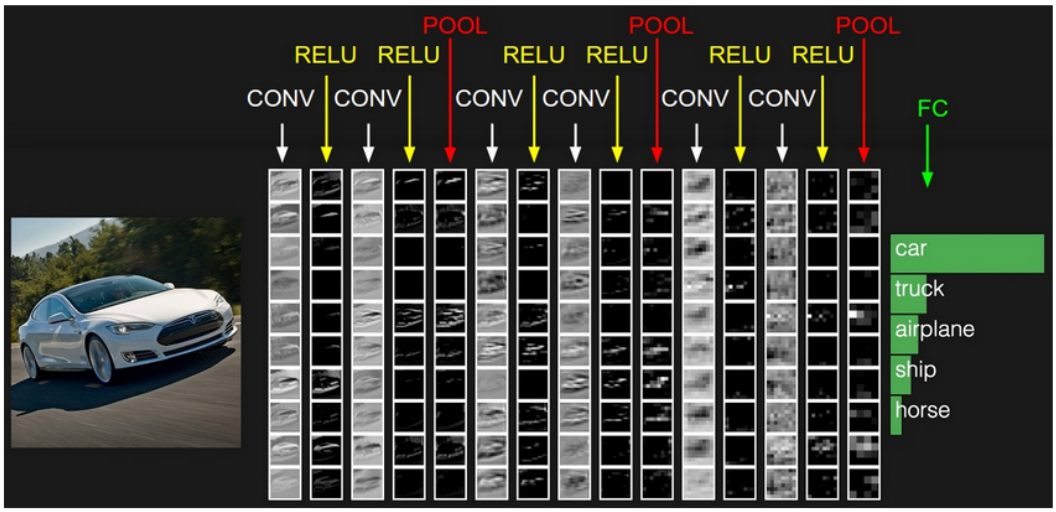


Figura 3.2: Ejemplo de una red CNN [12].

En la Figura 3.2, podemos ver como tenemos una imagen de un coche a la entrada, los píxeles de esta imagen son tomados en la primera capa, estos pixels se pasan por una capa de convolución y y una de activación (RELU), con esto conseguimos extraer las *.main* feautures de la imágen, despues de esto tenemos una capa de Pooling la cual nos hace mas pequeña la matriz de características que nos ha extraído las capas previas, de este modo nos quedamos con menos datos, dejando únicamente los más relevantes. Si repetimos esta estructura de capas al final tendremos una matriz con las características más importante de cada imágen y con esta podemos predecir a que clase pertenece el dato de entrada.

3.1.1.1 Capa covolucional.

Las capas convolucionales son útiles a la hora de analizar la imagen y detectar patrones con lo que poder clasificar una imágen, para esto se usan unos filtros llamados *kernel*, estos toman los píxeles de una imagen y se realiza la convolución con una ventana de NxN bits que se va deslizando sobre el mapa de píxeles generando como resultado una matriz que compone al mapa de *features* (Figura 3.3). A la hora de diseñar nuestra red deberemos usar más o menos capas convolucionales conforme al problema que nos enfrentamos, por ejemplo no es lo mismo diferenciar entre un plátano y una naranja que entre un balón y una naranja, en el segundo caso habría que poner más capas convolucionales ya que el balón se parece más a la naranja que el plátano.

El proceso de convolución es el que se puede apreciar en Figura 3.3, en dicha imagen tenemos una matriz de bits en verde sobre la cual se desliza la matriz convolución de color naranja, como resultado tenemos la matriz convolucionada de características, este proceso se define matemáticamente en la Ecuación 3.1.

$$y[m,n] = x[m,n] * h[m,n] = \sum_k \sum_l x[k,l] + h[m-k, n-l] \tag{3.1}$$

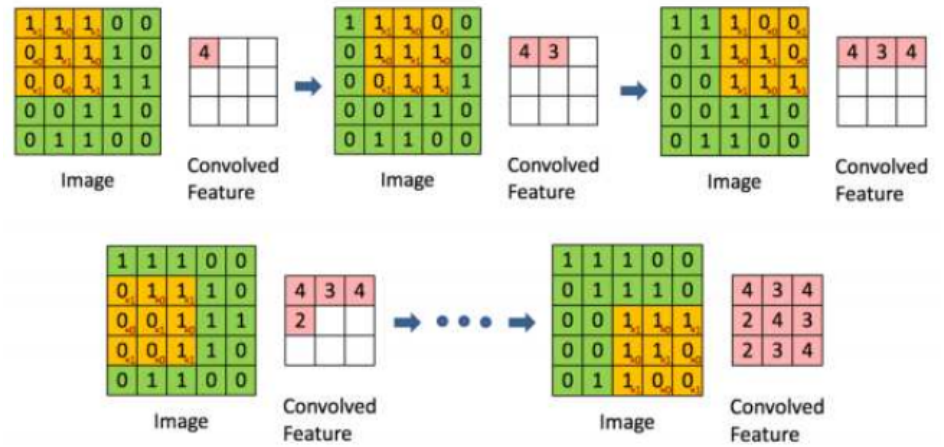


Figura 3.3: Capa Convolucional, [27].

3.1.1.2 Capa de activación

Estas capas se sitúan después de las capas de convolución y su función es la de aplicar un umbral o función de activación a la matriz de activación que ha generado previamente la capa de convolución.

Las funciones de activación se encarga de devolver una salida a partir de una entrada, su principal objetivo es el de dotar de no lienealidad a la red para que esta vaya aprendiendo de este tipo de valores, hay múltiples funciones que se utilizan para este propósito pero la más usadas son:

- **Softmax**: convierte la salida en probabilidad, sumando todas las salidas 1, se suele usar para normalizar múltiples clases y tiene un buen rendimiento en las últimas capas, su ecuación esta definida como:
$$\text{Softmax}(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}} \tag{3.2}$$

- **ReLu** esta función solo deja pasar los datos de entrada positivos anulando los negativos, esta función se comporta bien para problemas de clasificación de imágenes y hacen un buen trabajo en redes convolucionales, por el contrario es una función que no está acotada, su definición matemática es:
$$max(0,x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ x & \text{for} \quad x \geq 0 \end{cases} \tag{3.3}$$

- **Leaky ReLu**: Se basa en la función ReLu, tiene las mismas características de la ReLu pero a diferencia de ella no anula los valores negativos pero los penaliza.

$$f(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ a.x & \text{for} \quad x \geq 0 \end{cases} \tag{3.4}$$

- **Sigmoid**: con esta función se pasan los datos de entrada a 0 y 1, esta función se comporta muy bien en las últimas capas, no esta centrada en 0, por el contrario converge muy lentamente y satura en valores muy altos o muy bajos.

$$f(x) = \frac{1}{1 + e^{-x}} \tag{3.5}$$

3.1.1.3 Capa pooling

Esta capa se pone después de una capa de activación con ella realizamos una función de reducción de tamaño, por lo que conseguimos que a medida que se propague la información de una capa a otra nos quedamos con los datos más significativos optimizado el rendimiento del código, ya que manejamos menos información y no cargamos en exceso.

Este proceso se realiza tomando un conjunto se realiza tomando una matriz de bits NXN sobra la cual deslizamos una ventana de KxK con la que vamos seleccionamos los más significativos que están dentro de esa venta reduciendo la matriz de bits inicial en un una matriz con los datos más importantes, si se realiza un **Max Pooling** elegimos los *bits* de mayor peso que hay en la ventana que se deslizada y si se realiza un **Average Pooling** se toman la media todos los valores de activación de la ventana, Figura 3.4.

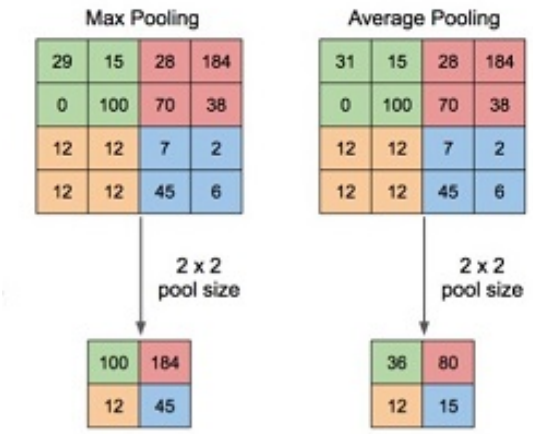


Figura 3.4: Capa Max y Average Pooling.

En la figura anterior (Figura 3.4) tenemos un mapa de *bits* de 4x4 al cual se le aplica un proceso de maxpoling con una ventana de 2x2 dando como resultado un mapa de *bits* más pequeños.

3.1.1.4 Capa fully connected

Agrupando capas convolucionales con capas de activación y capas de pooling (Figura 3.2) conseguimos extraer representaciones de características abstractas que se van moviendo por toda la estructura de la red convolucional. Las capas fully connected toma todas las características extraídas por la red y aplica una función dando lugar una predicción sobre la clase a la que la red interpreta que pertenece la imagen de estrada, normalmente se suele usar la función de activación Sigmoid (Ecuación 3.5) si el problema es de clasificación binaria o la función softmax (Ecuación 3.2) si se quiere intepretar la salida y clasificicación como una probabilidad, normalmente esta se suele usar para problemas de multiclase como por ejemplo el de la figura 3.2.

La desventaja de las capas *fully connected* es que están definidas con pocas neuronas que concentran todas las características de la imagen, a diferencia de las capas previas en las cada neurona se centraba únicamente en una parte de la imagen de entrada, esta última capa tiene que generalizar usando cada parámetro extraído para toda la imagen.

3.1.1.5 Entrenamiento y aprendizaje.

Cuando se diseña una red neuronal los pesos de esta son en un principio aleatorio por lo que las primeras predicciones no suelen ser muy cercanas a lo que buscamos, pero a medida que pasan las épocas de entrenamiento esto pesos se van ajustando dejando ser aleatorios, esto se consigue evaluando el error de predicción sobre el *ground truth* y penalizando las neuronas para ajustar los pesos de cada una de ellas y de esta forma ir aprendiendo conforme se van analizando imágenes por la red.

En primer lugar lo que se realiza tras la salida de la capa *fully connected* es un calculo del error sobre la predicción que se ha hecho, para ello se usa una función de perdidas o *loss function*, las más comunes son:

- Error cuadrático medio:
$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y_i})^2 \tag{3.6}$$
- Error medio absoluto:
$$MAE = \frac{\sum_{i=1}^{n}|y_i - \tilde{y_i}|}{n} \tag{3.7}$$
- Categorical Cross-Entropy:
$$CE = -\log_{10}\left(\frac{e^{y_i}}{\sum_j e^{\tilde{y_i}}}\right) \tag{3.8}$$
- Binary crossentropy:
$$BC = \frac{1}{N}\sum_{i=0}^{N}(y_i * \log_{10}(\tilde{y_i}) + (1 - y_i) * \log_{10}(1 - \tilde{y_i})) \tag{3.9}$$

En las ecuaciones 3.6, 3.7, 3.8, 3.9 y_i es el valor real de la clase a la que pertenece el dato de entrada e $\tilde{y_i}$ es el valor de la predicción de la red. A parte de estas ecuaciones existen muchas otras, utilizándose cada una de ellas para resolver un tipo determinado de problema [15].

Una vez hemos evaluado el error que se ha cometido tomamos este para entrenar la red, penalizando cada neurona y ajustando los pesos de esta para que en la siguiente época de entrenamiento la red funcione mejor dando valores más bajos al calcular la función de error.

El proceso con el que se penaliza las neuronas cambiando sus pesos y haciendo que la red aprenda se llama *backpropagation* o propagación hacia atrás. Con la técnica de *backpropagation* se responsabiliza a cada las neuronas del error final en la predicción de la etiqueta, para ello se realiza una propagación del error total de atrás hacia adelante, esto tiene sentido por que el error de una capa depende directamente de la capa anterior, este método se llama retropropagación de errores.

Al aplicar *backpropagation* se está operando de forma recursiva capa tras capa moviendo el error hacia atrás, con esto conseguimos que al finalizar la propagación sepamos cual ha sido el error de cada neurona, con lo que podemos ajustar cada uno de los pesos y además al propagar el error solo una vez hacia atrás conseguimos ser más eficientes en la ejecución.

Backprogation es un método para calcular cada una de las derivadas parciales de los parámetros de nuestra red respecto a los pesos, esto lo necesitamos para poder optimizar la red haciendo uso del algoritmo del descenso del gradiente [17], para realizar esto se evalúa la matriz de pesos de una capa de la red con la derivada parcial del error respecto a los pesos como se muestra la Ecuación 3.10.

$$W_i(t+1) = W_i(t) - \eta \frac{\partial Loss}{\partial W_i} \tag{3.10}$$

En la Ecuación 3.10 hay el termino η al cual llamamos tasa de aprendizaje, con este termino podemos ajustar en que medida queremos que la red aprenda, pero tenemos que tener cuidado con él porque si usamos un valor muy grande puede que el gradiente no converja a un mínimo pudiendo incluso quedar oscilando entre dos valores, si la tasa de aprendizaje es muy baja se tendría que dar muchísimas épocas a la red para que aprendiera siendo poco eficiente su aprendizaje.

3.2 Modelo de división celular

Tenemos que crear un modelo que pasáandole una imagen de un embrión nos etiquete la imagen con; 1, 2, 3-4 o +5 según el número de células presentes en la imágen, este tipo de clasificador debe extraer características propias de cada clase, para este tipo de problemas lo mejor es usar red neuronal convolucional.

Para nuestro modelo tomaremos los datos del divisor celular [16] en el que se propone un marco para contar automáticamente el número de células en embriones humanos en desarrollo haciendo uso de técnicas de *deep learning* para reproducirlos en paython [13] haciendo uso de bibliotecas como Keras [15] y tensorflow [15].

Primero vamos a tomar los datos con lo que vamos a entrenar y validar el modelo en python. Este conjunto de datos cuenta con un total de 263 videos de unas 100 horas de duración cada uno, los cuales fueron ya tratados en su día para extraer los fotogramas de cada video [28]. Los fotogramas de estos videos están subdivididos en dos grupos validación y entrenamiento, a su

vez cada grupo contiene 4 subcarpetas una por cada etiqueta (1, 2, 3-4 o +5) y dentro los *frames* que son clasificados con esatas etiquetas.

A la hora de diseñar redes neuronales se necesitan muchos datos de entrada para que la red aprenda lo máximo posible, esto hace que la cola de muestras a la entrada de la red pueda saturar el sistema haciendo que este vaya mucho más lento, para ello librerías como Keras tienen funciones que permiten realizar un preprocesado de las imágenes de entrada con el objetivo de no sobrecargar la GPU de un equipo, esta clase es *ImageDataGenerator*,[15]. Esta clase nos permite tomar las imágenes con escala de grises y reescalar las imágenes al tamaño que deseemos y mezclarlas (*shuffle*) para entrenar la red con datos mezclados y de esta forma no aprende con datos "perfectos", en nuestro caso la usamos para ir tomando muestras de 16 imágenes mezclando clases.

Una vez tenemos definidos el conjunto de datos de entrenamiento se realiza el diseño de las capas de nuestra CNN Tabla 3.1.

N	Capa	Neuronas	Tamaño	Otros
1	Convolucional	24	[11 11]	Paso = 1 , *Padding* = 10
2	ReLu	-	-	-
3	*Max-pooling*	-	[3,3]	Paso = 1
4	Convolucional	64	[5 5]	Paso = 1 , *Padding* = 4
5	ReLu	-	-	-
6	*Max-pooling*	-	[3,3]	Paso = 1
7	Convolucional	96	[3 3]	Paso = 1 , *Padding* = 2
8	ReLu	-	-	-
9	Convolucional	96	[3 3]	Paso = 1 , *Padding* = 2
10	ReLu	-	-	-
11	Convolucional	64	[3 3]	Paso = 1 , *Padding* = 2
12	ReLu	-	-	-
13	*Max-pooling*	-	[3,3]	Paso = 1
14	Fully-connected	512	-	Ratio *dropout* 50 %
15	ReLu	-	-	-
16	Fully-connected	512	-	Ratio *dropout* 50 %
17	ReLu	-	-	-
18	Fully-connected	512	-	Ratio *dropout* 50 %
19	ReLu	-	-	-
20	Fully-connected	4	-	-
21	Softmax	-	-	-
22	Clasificación	-	-	-

Tabla 3.1: Arquitectura de la CNN utilizada,[16]

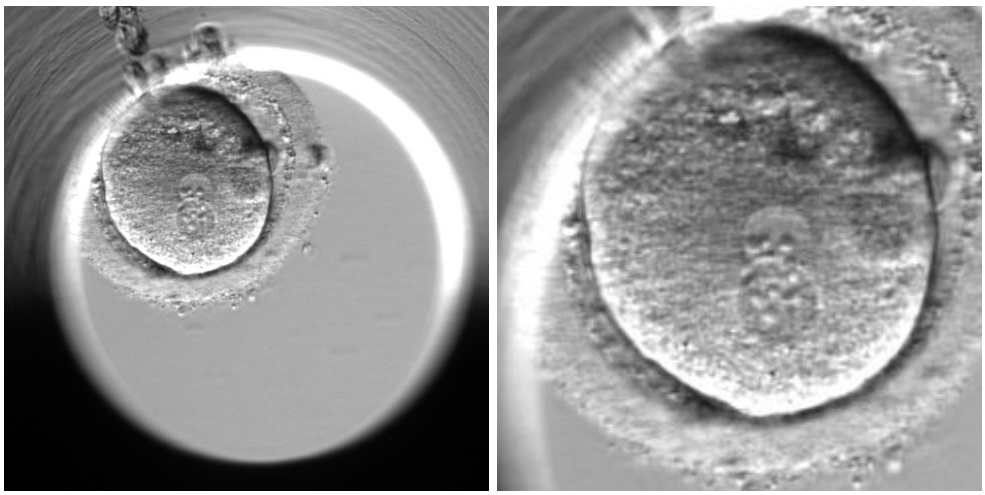
Definida nuestra red procedemos a compilarla para crear su estructura entrenarla y validarla, para ello hacemos uso de la API *sequential* [14] que tiene como métodos:

- *Compile*: con esta función nos crea la red neuronal que hemos definido previamente para ello debemos definir la función de perdidas y el optimizador que va a utilizar nuestra red para el *backpropagation*.
- *Fit*: con este método vamos entrenando la función, para ello tenemos que definir el número de épocas, el conjunto de datos de entrenamiento y los de validación, además nos permite definir una condición de parada, por ejemplo si la función no mejora las perdidas de validación tras 15 épocas la red para de entrenar salvando los mejores pesos de la red hasta ese momento.

```
callbacks=[
EarlyStopping(monitor='val_loss', patience=15),
ModelCheckpoint(filepath='divisor_cel.npy',
monitor='val_loss', verbose=1, save_best_only=True)]
```

- *Evaluate*: nos calcula los valores de pérdidas y las metricas del modelo durante su entrenamiento.
- *Predict*: una vez el modelo esta entrenado, con este método podemos tomar un con junto de datos y predecir a que clase pertenecen.

Con la red entrenada, tenemos que pasarle las imágenes de nuestra base de datos *BD_IVI_VALENCIA* en la cual tenemos gran cantidad de imágenes que clasificar por lo que usaremos nuestro modelo de predicción para saber cuantas células hay en cada fotograma, pero primero tenenos que tratar las imágenes antes de pasárselas a la red, ya que la red aprendió a clasificar con imágenes en las que se encontraba exclusivamente el embrión y las imágenes que hemos exportado de la base de datos *BD_IVI_VALENCIA* son fotogramas en los que sale todo el espacio del Well sin estar necesariamente centrado el embrión como se puede apreciar en la Figura 3.5a. Para realizar el proceso de *cropping* usamos un programa de matlab realizado en [28] en el que detectando el centro del embrión en la imágen y midiendo el radio de este se corta la imágen entorno al embrión como en la Figura 3.5b.



(a) Fotograma tomado por el Embryoscope **(b)** Fotograma recortado

Figura 3.5: Ejemplo de proceso del resultado de cropear las imágenes de la base de datos

3.3 Tiempos de división célular

Tomamos las imágenes ya clasificadas por el número de células para estimar el momento exacto en el que se dividen las células, parámetros fundamentales para elegir el embrión más viable como hemos indicado en la introducción [22] [8].

Para calcular los tiempos de división tomaremos el modelo de estimación de movimiento entre imágenes (SAD), [28], con el que obtenemos una gráfica con un eje temporal en el eje de las abscisas y el valores de SAD en el de las ordenas, detectando cuando se produce picos en la gráfica podemos saber en que momentos hay más movimiento de un *frame* a otro siendo ese momento donde se ha producido una división celular.

Para realizar esto usamos el resultado obtenido del clasificador de células el cual nos clasifica todas imágenes que hemos exportado de nuestra base de datos *IVI_ VALENCIA* obteniendo como resultado un excel como el de la Figura 4.3 y el directorio donde están todas estas imágenes cropeadas, Figura 3.5.

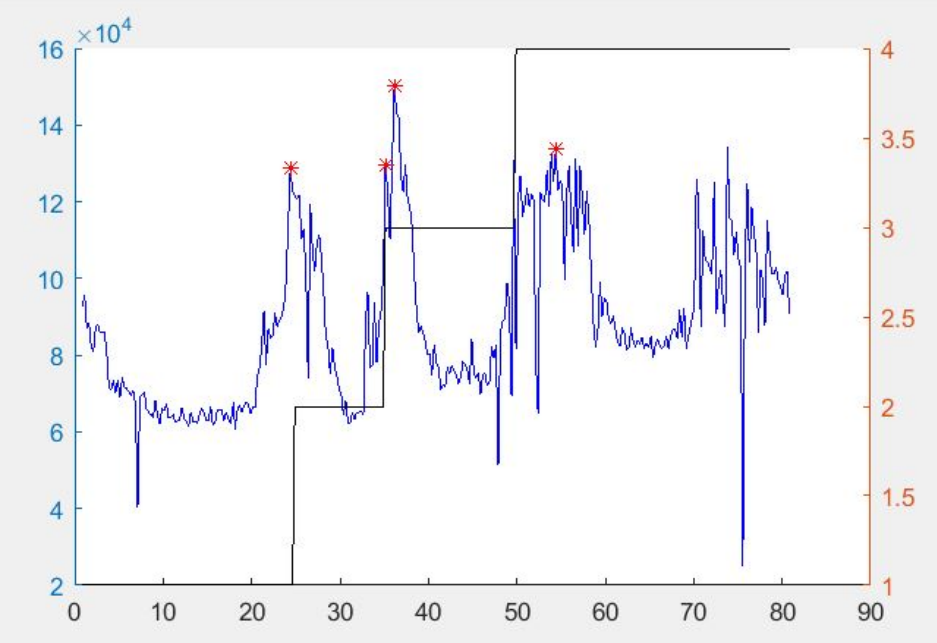


Figura 3.6: Calculo de la cantidad de movimiento (SAD).

Como se aprecia en la Figura 3.6, tenemos una gráfica con dos funciones: la función de color azul nos indica la cantidad de movimiento según el tiempo detectados en los *frames*, con asteriscos rojos tenemos los picos con más movimiento por lo que nos índica que en ese momento se ha producido una división, por otro lado tenemos una función con cuatro escalones los cuales corresponden a la franja de tiempo en la que tenemos una, dos, tres-cuatro o cinco células y esta se extrae de la clasificación realizada por el divisor de células tomando el tiempo de la primera imagen clasificada en una clase. Como se puede ver los picos de la función de cantidad de movimiento coinciden con los momentos en los que se pasa de un estado a otro, estos instantes son los momentos de división, por lo que podemos afirmar que la clasificación de las imagens son las correctas, con esta información obtenemos los tiempos con los que se recoge en [8] que se

puede estimar la calidad de un embrión para ser implantado en el útero de la paciente con altas posibilidades de gestación:

- **t2**: tiempo desde fertilización hasta el momento en el que se produce la primera mitosis, pasando de 1 a 2 células y observándose dos blastocitos completamente separados por membranas.
- **t3**: instante en el cual se produce la división de un blastocito, pasando de 2 a 3 células.
- **t4**: instante en el cual se produce una nueva división. Se tienen entonces 4 blastocitos diferenciados completamente en el embrión. En ocasiones, según la frecuencia de adquisición de imágenes puede coincidir con t3.
- **t5**: instante en el cual, tras una nueva división en un blastocito, se tienen 5 células en el embrión.
- **cc2**: segundo periodo de interfase. Un periodo de interfase es aquel en el cual la célula se desarrolla previo a una división celular. En este caso, se trata de la interfase de dos células, hasta dividirse a tres células. Es decir, cc2 = t3 - t2.
- **cc3**: tercer periodo de interfase, desde que se tienen 4 células diferenciadas hasta que comienza la mitosis de los blastocios pasando de 4 a 8 células. Este instante se define en la aparición de la célula 5. cc3 = t5 - t4.
- **s1**: duración de la primera mitosis. Tiempo que tarda en estabilizarse la presencia de dos células desde el inicio de la división.
- **s2**: duración de la segunda mitosis. Al no presentar periodo de interfase entre tres y cuatro células, se tiene que s2 = t4 - t3.
- **s3**: duración de la tercera mitosis.

Estos parámetros son calculados para cada conjunto imágenes correspondientes al mismo link y se exporta el resultado a un excel, de esta forma tenemos los tiempos de división celular durante el crecimiento embrionario para un determinado embrión.

Capítulo 4

Resultados

El principal resultado obtenido es el diseño e implementación de una aplicación con la que poder manejar grandes cantidades de datos pudiendo clasificarlos, tratarlos y filtrarlos con el fin de sacar de ellos la información que se crea conveniente. Como consecuencia se ha logrado crear una base de datos muy amplia y robusta que contiene los datos más importantes para el estudio y desarrollo de nuevas técnicas con las que mejorar la tasa de éxito en tratamientos FIV.

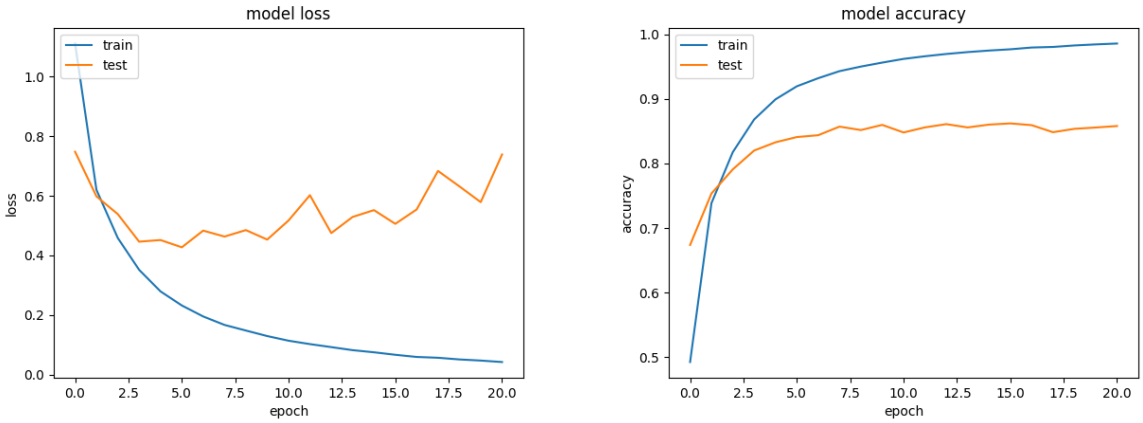
Por otro lado, la aplicación con la que se extraen los datos se puede ampliar fácilmente con nuevos métodos y funciones, pudiendo usar ya la estructura hecha para buscar o filtrar otro datos que puedan ser relevantes para otros estudios, además esta aplicación al estar en un lenguaje tan flexible, estandarizado y universal como es Java es fácilmente integrable dentro del sistema informático del IVI, lo que permitiría relacionar de forma inmediata datos de un Embryoscope y del sistema de gestión Sivis, obteniendo así un sistema automático de recolección de información evitando de este modo posibles errores humanos a la hora de tomar los datos.

En nuestro caso gracias a desarrollo de la base de datos *IVI_ VALENCIA* hemos ampliado los casos de éxito y fracaso en la implantaciób pasando de un conjunto de 263 casos a 983 (572 casos de éxito y 456 casos de fallo) recogidos desde el 2014 al 2018, esto supone un aumuento del 373 % de datos lo que es un gran ventaja a la hora de realizar un buen modelo de predicción ya que a la hora de desarrollar el modelo es muy importante tener gran cantidad de datos fiables.

En una segunda etapa hemos diseñado un modelo de clasificación de células entranando con el conjunto de 263 datos que ya estaban previamente etiquetados con las clases (1, 2, 3-4, +5).

A la hora de entrenar la red, como esta ha sido validada en [16], entrenamos con el conjunto de entrenamiento y validamos directamente con el test.

Tras el entrenamiento el modelo para en la época 20, como hemos puesto una condición de parada tras 15 épocas sin mejorar las pérdidas, el mejor valor es el que tenemos en la época 5, por otro lado vemos en la Figura 4.1a que cuando la de gráfica test disminuye y por más que pasan las épocas no aumenta evidencia que no existe overfitting, además en la Figura 4.1b vemos como el modelo aprende tras varias épocas oscilando el valor de *acuraccy* entorno a los 0.85.



(a) Función de pérdidas.

(b) Función de precisión.

Figura 4.1: Funciones de entramiento del modelo.

Generando el reporte de clasificación (Tabla 4.1) y la matriz de confusión (Tabla 4.2) vemos de una forma más visual las predicciones obtenidas por parte de la CNN sobre el conjunto de datos de test.

Clase	Precision	Recall	f1-Score	Support
1 Célula	0.97	0.99	0.98	5191
2 Células	0.82	0.78	0.8	2493
3-4 Células	0.75	0.71	0.73	3305
+5 Células	0.87	0.9	0.89	5533

Tabla 4.1: *Clasification Report* del clasificador de imágenes.

En la Tabla 4.1, tenemos el campo precisión que nos indica que porcentaje el etiquetas predecidas han sido correctas, nuestro modelo predice mejor las clases 1 célula y +5 células. El campo Recall se define como la división entre el número de verdaderos y la suma de los verdaderos positivos más los falsos negativos. F1-Score es la media ponderada de recall y precisión de tal forma que los valores más altos demuestran una mayoría acierto del modelo y un valor un mayor error.

Este *classification report* nos indica que el modelo en global clasifica muy bien siendo la clase 1 célula la que mayor tasa de acierto tiene seguida de la clase +5, por lo que el modelo predice de una forma más acertada estas dos clases, este hecho se puede ver en la Tabla 4.2, en la que vemos que para la clase 1 hay muy fallos mientras que para la clase dos si aumentan un poco pero no son comparables a la cantidad de imágenes mal clasificadas para las clases 2 y 3-4 en las que se producen más clasificaciones erróneas datos que son corroborados en la *precision* de la Tabla 4.1 en la que se puede ver que clasifica algo pero las clases intermedias.

Etiqueta real	**Etiqueta predecida**			
-	**1**	**2**	**3-4**	**+5**
1	5117	59	6	9
2	103	1955	335	100
3-4	37	303	2348	617
+5	28	79	452	4974

Tabla 4.2: Matriz de confusión del clasificador de imágenes.

Tras observar los resultados del entrenamiento podemos decir que el modelo clasifica de manera robusta y precisa las imágenes por lo que podemos emplearlo para que clasifique el conjunto de imágenes (éxito y fracaso) extraídas de nuestra base de datos *IVI_ VALENCIA*.

Una vez entrenada la red calculamos los datos de sensibilidad, comparando los datos obtenidos vemos que; nuestro divisor es menos sensible detectando las imágenes con una y con cinco células, especialmente detectando una ya que pierde un 2.3 % con respecto al clasificador en matlab, pero sin embargo mejora en un 6 % detectando los grupos de 2 células y de 3-4 en un 3 %, en global el modelo de Python mejora, pasando de media de un 84 % de sensibilidad a un 87 % , por lo tanto conseguimos un predictor ligeramente mejor,Tabla 4.1.

Como resultado de pasar las imágenes por nuestro divisor celular obtenemos las probabilidades que la red le asigna a la imagen de pertenecer a cada una de las clases. Haciendo uso de *Pandas* [18] exportamos a excel los resultados. En el Excel tenemos 4 columnas , cada una etiquetada con el nombre de la clase (1, 2, 3-4, +5), y una última columna con el nombre de la imagen (link-tiempo-nº de fotograma) como se puede ver en la Figura 4.2.

	A	B	C	D	E	F
		1	2	3-4	+5	Names
0		0,999958396	4,15531E-05	3,97378E-08	2,04384E-08	0/11527192_Time-.335_Frg-2.jpg
1		0,999997258	2,80049E-06	7,14126E-10	4,35995E-10	0/11527192_Time-.585_Frg-3.jpg
2		0,999992251	7,69003E-06	3,17873E-09	1,93007E-09	0/11527192_Time-.836_Frg-4.jpg
3		0,999885917	0,000113999	7,51506E-08	3,69398E-08	0/11527192_Time-1.086_Frg-5.jpg
4		0,999962687	3,73147E-05	2,88373E-08	1,72147E-08	0/11527192_Time-1.336_Frg-6.jpg
5		0,988068521	0,011807296	9,02928E-05	3,38572E-05	0/11527192_Time-1.586_Frg-7.jpg
6		0,999068081	0,000930354	1,17325E-06	4,08262E-07	0/11527192_Time-1.836_Frg-8.jpg
7		0,999989748	1,02428E-05	8,09165E-09	6,41768E-09	0/11527192_Time-10.089_Frg-41.jpg
8		0,999986887	1,31462E-05	1,06256E-08	8,32523E-09	0/11527192_Time-10.339_Frg-42.jpg
9		0,999987364	1,26805E-05	5,48226E-09	2,91733E-09	0/11527192_Time-10.589_Frg-43.jpg
10		0,999977946	2,20394E-05	1,51855E-08	8,39852E-09	0/11527192_Time-10.839_Frg-44.jpg
11		0,999798715	0,000201153	8,60171E-08	5,13245E-08	0/11527192_Time-11.089_Frg-45.jpg
12		0,999970675	2,92909E-05	2,2574E-08	1,56487E-08	0/11527192_Time-11.339_Frg-46.jpg
13		0,999263227	0,000736139	3,87708E-07	2,12823E-07	0/11527192_Time-11.590_Frg-47.jpg
14		0,999910951	8,89152E-05	9,35512E-08	5,12742E-08	0/11527192_Time-11.840_Frg-48.jpg
15		0,99997735	2,27084E-05	1,50652E-08	1,0765E-08	0/11527192_Time-12.090_Frg-49.jpg
16		0,999623656	0,00037311	1,83313E-06	1,38703E-06	0/11527192_Time-12.340_Frg-50.jpg
17		0,999670863	0,00032811	6,90999E-07	3,72356E-07	0/11527192_Time-12.590_Frg-51.jpg
18		0,999007642	0,000972284	1,02764E-05	9,90768E-06	0/11527192_Time-12.840_Frg-52.jpg
19		0,999561965	0,000432573	3,04333E-06	2,40024E-06	0/11527192_Time-13.090_Frg-53.jpg
20		0,999593079	0,000405237	1,05927E-06	5,61814E-07	0/11527192_Time-13.340_Frg-54.jpg
21		0,999812663	0,000186748	4,0416E-07	1,88015E-07	0/11527192_Time-13.590_Frg-55.jpg
22		0,999602139	0,000396505	1,01084E-06	4,83428E-07	0/11527192_Time-13.840_Frg-56.jpg
23		0,999714911	0,000284033	7,33484E-07	3,94709E-07	0/11527192_Time-14.091_Frg-57.jpg
24		0,999170303	0,000826377	2,2995E-06	1,01576E-06	0/11527192_Time-14.341_Frg-58.jpg
25		0,99961251	0,000385557	1,36777E-06	6,4953E-07	0/11527192_Time-14.591_Frg-59.jpg
26		0,99990046	9,89619E-05	3,17351E-07	2,22643E-07	0/11527192_Time-14.841_Frg-60.jpg
27		0,999503613	0,000491585	2,83014E-06	1,91544E-06	0/11527192_Time-15.091_Frg-61.jpg

Figura 4.2: Excel con los resultados de las imágenes clasificadas.

Este excel resultado de predecir con el modelo entranando nos da una 276350 imágenes clasificadas, esto es un gran avance puesto que ahora disponemos de una gran cantidad de datos con las

cuales extraer parámetros morfocinéticos desarrollando mejores modelos, esto gracias a la base de datos que hemos elaborado.

Con el SAD obtenemos los tiempos de división descritos en sección 3.3 (Figura 4.3).

	A	B	C	D	E	F	G	H
1	ID	t2	t3	t4	t5	cc2	cc3	implantation
2	11527192	28,175	42,935	43,435	64,128	14,76	20,693	1
3	11527359	25,565	37,328	37,825	52,19	11,763	14,365	0
4	11528011	29,878	49,152	49,152	54,74	19,274	5,588	0
5	11528015	33,389	46,658	46,658	54,176	13,269	7,518	0
6	11528428	22,845	34,874	35,873	49,811	12,029	13,938	0
7	11528429	35,625	39,877	40,376	48,562	4,252	8,186	0
8	11528592	24,087	35,345	37,847	49,851	11,258	12,004	1
9	11528672	18,325	39,679	39,679	48,183	21,354	8,504	1
10	11528678	32,687	40,941	40,941	46,444	8,254	5,503	1
11	11528683	20,321	25,852	26,602	37,357	5,531	10,755	1
12	11529332	23,47	35,724	36,724	50,18	12,254	13,456	0
13	11529336	31,979	42,983	44,741	62,193	11,004	17,452	0
14	11529403	29,744	43,711	45,214	63,72	13,967	18,506	0
15	11529405	27,997	39,251	40,752	53,721	11,254	12,969	0
16	11529583	35,571	49,339	52,091	67,854	13,768	15,763	0
17	11529642	25,881	38,89	39,89	43,868	13,009	3,978	1
18	11529648	26,642	39,401	41,404	57,905	12,759	16,501	1
19	11529654	27,088	35,341	35,341	44,611	8,253	9,27	1
20	11529656	29,843	37,097	37,097	49,603	7,254	12,506	1
21	11530136	17,808	23,27	23,27	28,272	5,462	5,002	1
22	115304211	24,486	33,24	35,24	49,338	8,754	14,098	1
23	11530746	26,884	40,64	42,137	58,268	13,756	16,131	0
24	11530747	18,878	27,638	34,641	56,019	8,76	21,378	0
25	11531225	25,548	39,302	40,053	52,144	13,754	12,091	0
26	11531415	28,049	39,555	40,305	54,269	11,506	13,964	1
27	11531419	36,06	52,775	52,775	68,567	16,715	15,792	1
28	115320313	27,789	43,295	43,295	56,55	15,506	13,255	1
29	11532036	25,295	37,299	39,799	50,053	12,004	10,254	1
30	115320610	22,875	35,138	35,635	52,177	12,263	16,542	1
31	11532068	23,874	33,382	37,632	45,67	9,508	8,038	1

Figura 4.3: Excel con los tiempos de división

La Figura 4.3 muestra la hoja excel que se exporta con la estimación de los tiempos de división celular, en cada fila vemos que para un ID (link) se muestra sus tiempos de división y la etiqueta de implantación que hace referencia al *label* éxito (1) o fracaso (0) con los que hemos definido desde un principio nuestra BD_IVI_VALENCIA.

Capítulo 5

Conclusiones y propuesta de trabajo futuro

Tras analizar los resultados obtenidos podemos concluir que:

- Se ha conseguido manejar una gran cantidad de datos almacenada en distintos tipos de formatos, para ello se ha hecho uso de API's con las que se ha extraído la información. Como resultado de este proceso se ha alcanzado un sistema de gestón de información capaz de relacionar y almacenar datos de diferentes bases de datos y en diferentes formatos uniéndolos en una misma base de datos. La aplicación diseñada trabaja con el formato con el que el IVI guarda su información (Sivis y ".pdb" exportados por los Embryoscope) por lo que el proceso de tratado, análisis y filtrado hecho para los datos de la clínica de valencia son extrapolables para otras clínicas en consecuencia la aplicación diseñada puede ser usada para tratar bases de datos de otras clínicas, ampliando así el conjunto de datos del cual extraer datos morfocinéticos para el estudio de técnicas de predicción de éxito en la implantación en tratamientos FIV.
- Se ha creado una base de datos robusta con los datos de la clínica de Valencia en total se ha logrado agrupar datos para 572 casos de tratamientos éxitos y 456 de fracaso lo que nos da un conjunto de 1028 casos de tratamientos FIV para estudiar.
- Con el desarrollo y posterior entrenamiento de la red neuronal convolucional para la clasificación de células se ha demostrado que el diseño de la red propuesto [16] se comporta bastante bien con nuestro conjunto de muestras, además se ha verificado el buen rendimiento de las CNN para abordar problemas de clasificación y predicción con imágenes evidenciando el motivo de porque se usan cada vez más técnicas de *deep learning* para resolver este tipo de problemas.
- Adaptando el modelo SAD [28] a las imágenes exportadas de la base de datos creada hemos logrado obtener los tiempos propuesto por [8] para poder desarrollar un modelo de predicción de éxito en la implantación basado en los tiempos de división.

Como se ha visto el buen comportamiento de algoritmos de *deep learning* para problemas de predicción se podría hacer uso de un perceptrón multicapa par el desarrollo de un modelo de predicción de éxito de implantación que tome los tiempos de división obtenidos por el SAD. Otro

tipo de modelo que se podría usar sería el LSTM o *long short-term memory* que a diferencia de las CNN son redes neuronales que tienen ”memoria” por lo que pueden trabajar con vídeos extrayendo hiperparámetros que dependen del instante del vídeo.

Por otro lado tras trabajar con un volumen de datos bastante considerable se ha llegado a la conclusión de lo importante que es ser rígido a la hora de seguir un modelo estandarizado de adquisición de datos, ya que el principal problema con el que me he encontrado es con la ambigüedad con la que se toman los datos, lo que dificulta la elaboración de un sistema de filtrado de datos, además este tipo de problemas no solo repercuten a la hora de elaborar una aplicación, también afecta en la propia gestión del tipo de tratamientos lo que afecta a la eficacia en la implantación de embriones o incluso a la trazabilidad de los embriones, por ello y viviendo en una sociedad en la que cada vez es más importante los datos, recomiendo al IVI estandarizar la forma en la que se toman los datos siendo estricto en el tipo de abreviación y en los parámetros que se miden y se introducen tanto el SIVI como en los incubadores Embryoscope.

Esta apreciación es aplicable a muchos ámbitos de la industria y de la sociedad ya que cada vez es más habitual el desarrollo de modelos predictivos o de clasificación para enfrentar problemas de la vida diaria o para optimizar determinados procesos y para poder construir estos de una forma robusta es necesario tener acceso a una base de datos organizada que tenga el máximo de datos posibles.

Bibliografía

[1] C. Amorim. *Red Neuronal en Python con Numpy*. 2017. URL: https://artfromcode.wordpress.com/2017/04/18/red-neuronal-en-python-con-numpy-parte-1/.

[2] Cvblab. *Descripción del grupo de investigaci'on Cvblab*. 2019. URL: http://www.cvblab.webs.upv.es/es/inicio/ (visitado 2018).

[3] Guía Dev. *MySQL vs SQLite*. 2019. URL: https://guiadev.com/mysql-vs-sqlite/ (visitado).

[4] Oracle Doocs. *Documentación package java.sql*. 2019. URL: https://sqlitebrowser.org/.

[5] Apache Software Foundation. *Documentación de la la API para manajer los Excel*. 2019. URL: https://poi.apache.org/ (visitado 09-04-2019).

[6] Eclipse Foundation. *link de descarga Eclipse*. 2019. URL: https://www.java.com/es/download/ (visitado 24-09-2010).

[7] Kunihiko Fukushima. "Neocognitron: A Self-Organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position". En: *Biological Cybernetics* (1980).

[8] D. Hlinka y col. "Time-Lapse Cleavage Rating Predicts Human Embryo Viability". En: *Physiological Research* (2012).

[9] IVI. *Pàgina web IVI*. 2019. URL: https://ivi.es/tratamientos-reproduccion-asistida/inseminacion-artificial/.

[10] IVI. *Pàgina web IVI*. 2019. URL: https://ivi.es/tratamientos-reproduccion-asistida/fecundacion-in-vitro/.

[11] IVI. *Pàgina web IVI*. 2019. URL: https://ivi.es/preguntas-frecuentes/tasas-de-exito/.

[12] A. Karpathy J. Johnson. *CS231n: Convolutional Neural Networks for Visual Recognition.* 2019. URL: http://cs231n.github.io/convolutional-networks/ (visitado 24-05-2019).

[13] jetbrains. *Entorno pycharm*. 2019. URL: https://www.jetbrains.com/pycharm/ (visitado 01-05-2019).

[14] Keras.io. *Keras: API sequential*. 2019. URL: https://keras.io/models/sequential/ (visitado 01-05-2019).

[15] Keras.io. *Keras: The Python Deep Learning library*. 2019. URL: https://keras.io/ (visitado 01-05-2019).

[16] A. Khan, S. Gould y M. Salzmann. *Deep Convolutional Neural Networks for Human Embryonic Cell Counting*. 2016.

[17] Y. Lecun y col. *Gradient-based learning applied to document recognition*. 1998, págs. 2278-2324.

[18] Wes McKinney. *Pandas documentation*. 2019. URL: https://pandas.pydata.org/pandas-docs/stable/ (visitado 01-05-2019).

[19] J. Melton y A. Eisenberg. *SQL y Java: guía para SQLJ, JDBC y tecnologías relacionadas*. Ra-ma, 2002. ISBN: 8478975063.

[20] M. Mesenguer y col. "The use of morphokinetics as a predictor of embryo implantation". En: *Human Reproduction* (2011).

[21] P Minsky. *Perceptrons: An Introduction to Computational Geometry*. MIT Press, 1969.

[22] A.P.A Van Montfoort y col. "Early cleavage is a valuable addition to existing embryo selection parameters: a study using single embryo transfers". En: *Human Reproduction* (2004).

[23] J Moor. "The Dartmouth College Artificial Intelligence Conference: The Next Fifty Years." En: *AI Magazine* 27 (ene. de 2006), págs. 87-91.

[24] Y. Motato y col. "Morphokinetic analysis and embryonic prediction for blastocyst formation through an integrated time-lapse system". En: *Fertility and Sterility* (2015).

[25] Oracle. *Documentación de la la API para conectarnos a las bases de datos sql*. 2019. URL: https://docs.oracle.com/javase/7/docs/api/java/sql/package-summary.html.

[26] Oracle. *link de descarga Java*. 2019. URL: https://www.java.com/es/download/ (visitado 08-05-2019).

[27] "Redes Neuronales Convolucionales e." En: *Escuela Técnica Superior de IngenieríaUniversidad de Sevilla* (2017).

[28] J.J Siva Rodríguez. *DESARROLLO DE UN SISTEMA DE PROCESAMIENTO DE IMAGENES PARA LA PREDICCION SOBRE EL EXITO EN IMPLANTACION DE BLAS-*

TOCITOS CON PARAMETROS MORFOCINETICOS. 2018. URL: https://riunet.upv.es/handle/10251/110084.

[29] F. Rosenblatt. "The Perceptron: A Probabilistic Model for Information Storage and Organization in The Brain". En: *Psychological Review* (1958), págs. 65-386.

Apéndice A

Programa principal

Programa principal

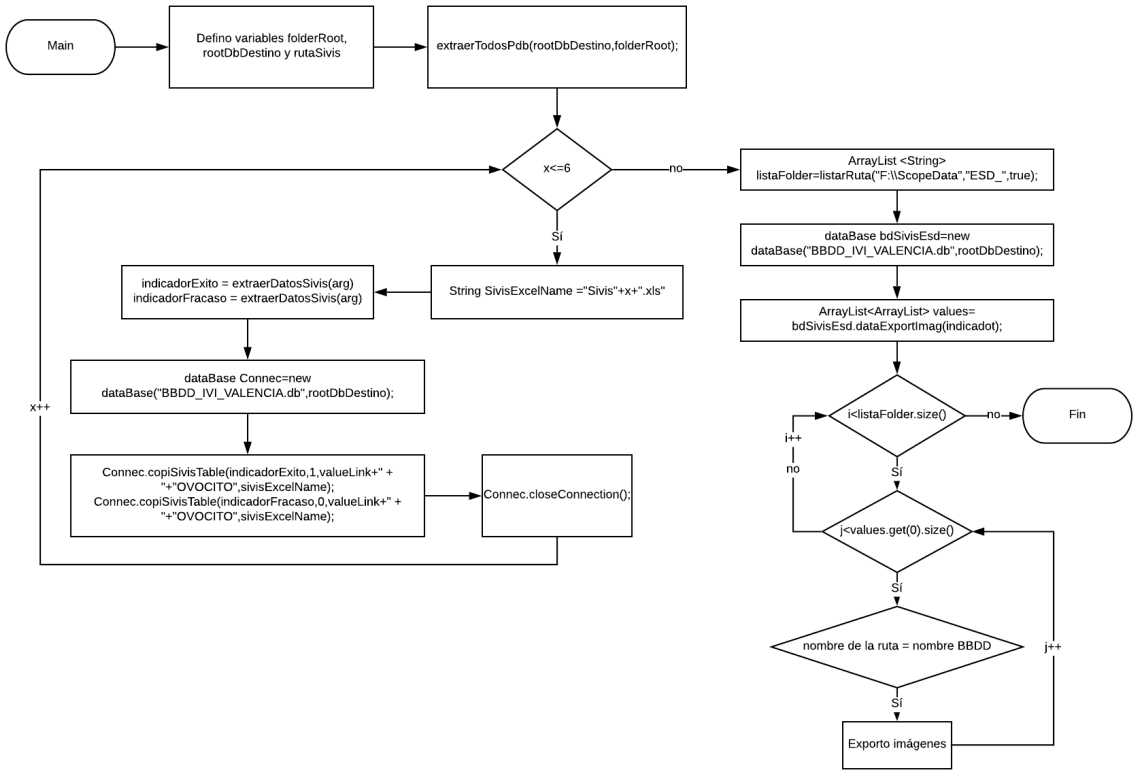


Figura A.1: Diagrama de flujo del Main

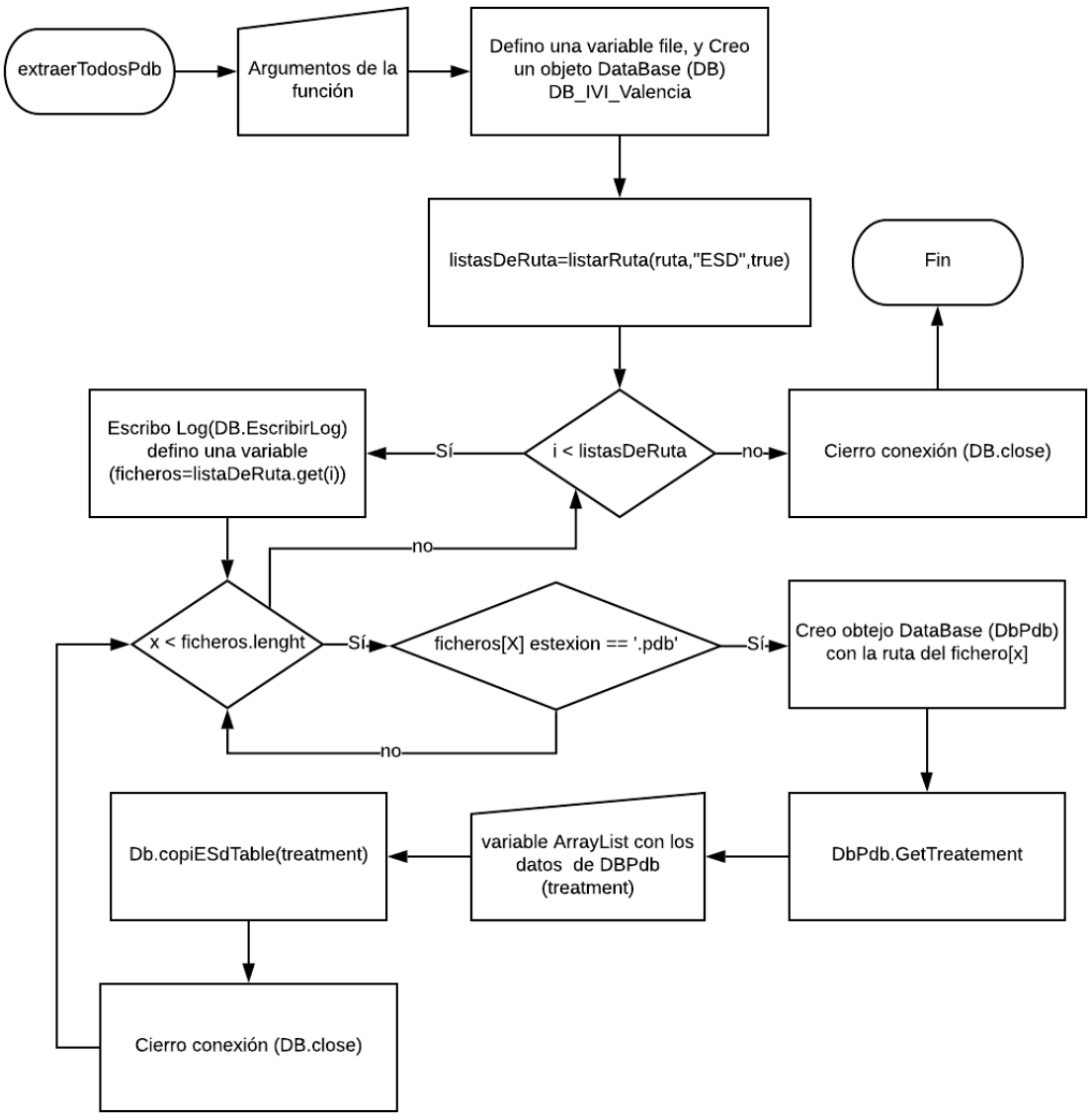


Figura A.2: Método para extraer los datos de los ”pdb”.

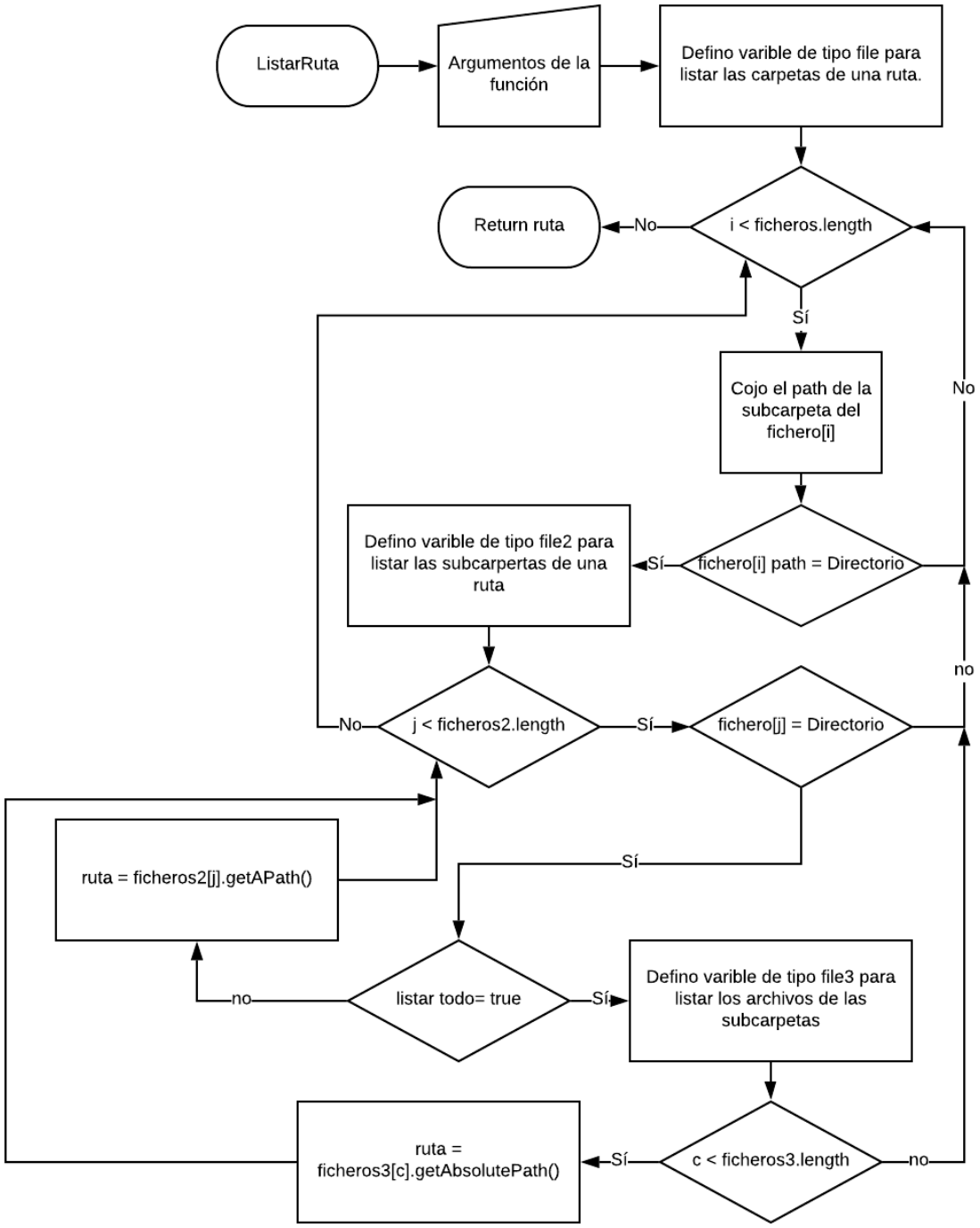


Figura A.3: Función para listar la ruta de loss archivos de una ruta

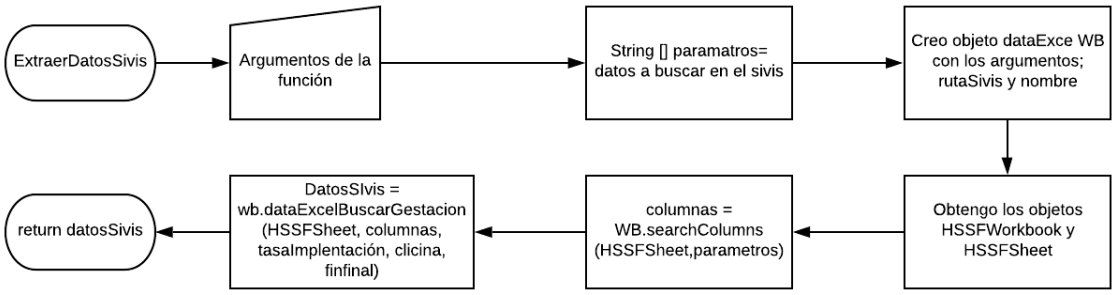


Figura A.4: Función que flitra los datos de las hojas del sivis.

Apéndice B

Clase dataBase

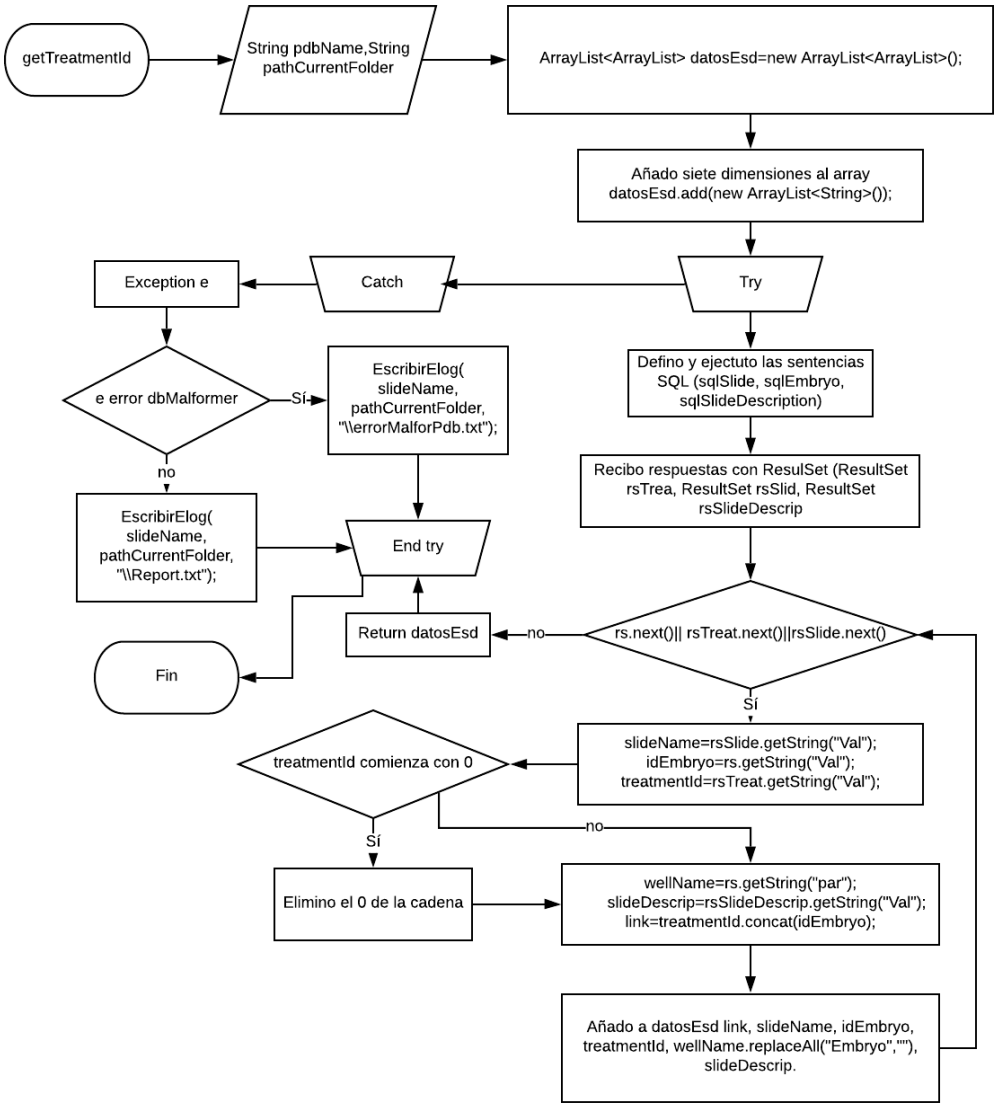


Figura B.1: Diagrama de flujo de la función getTreatmentId

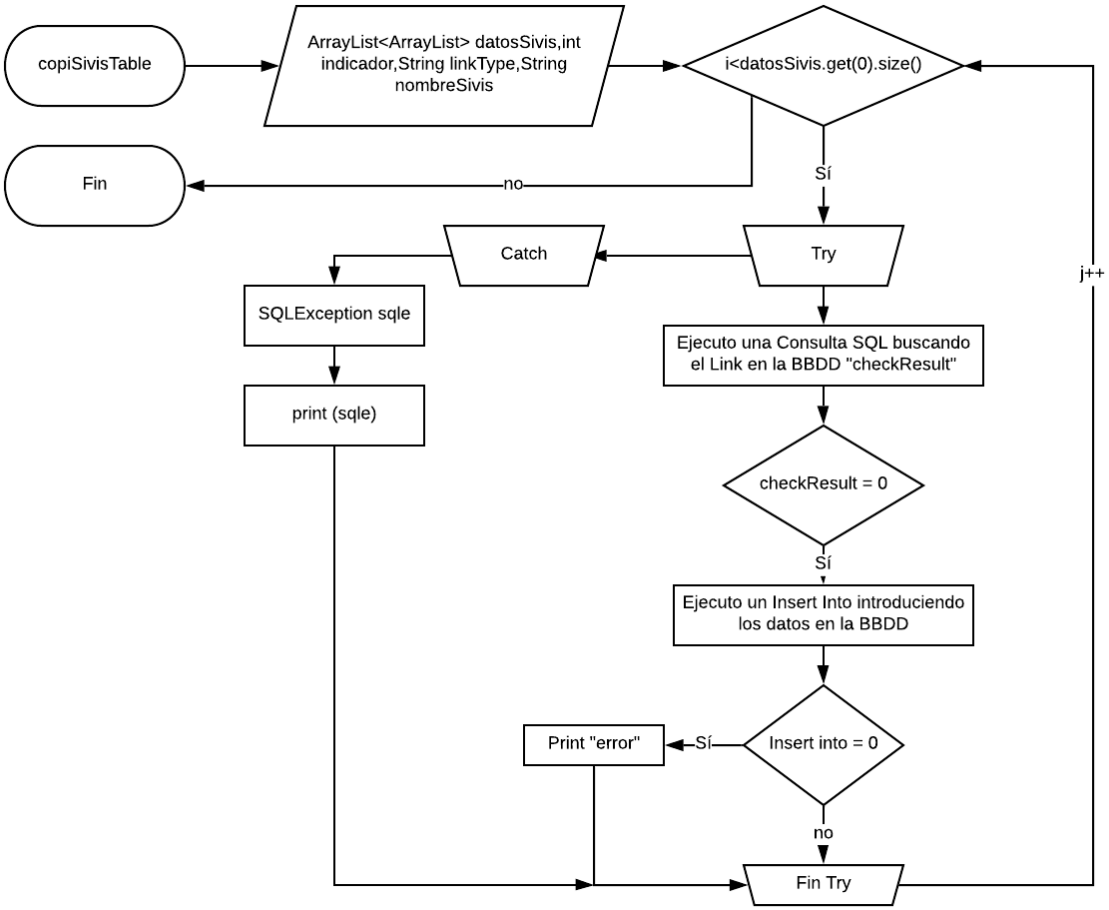


Figura B.2: Diagrama de flujo del método copiSivisTable

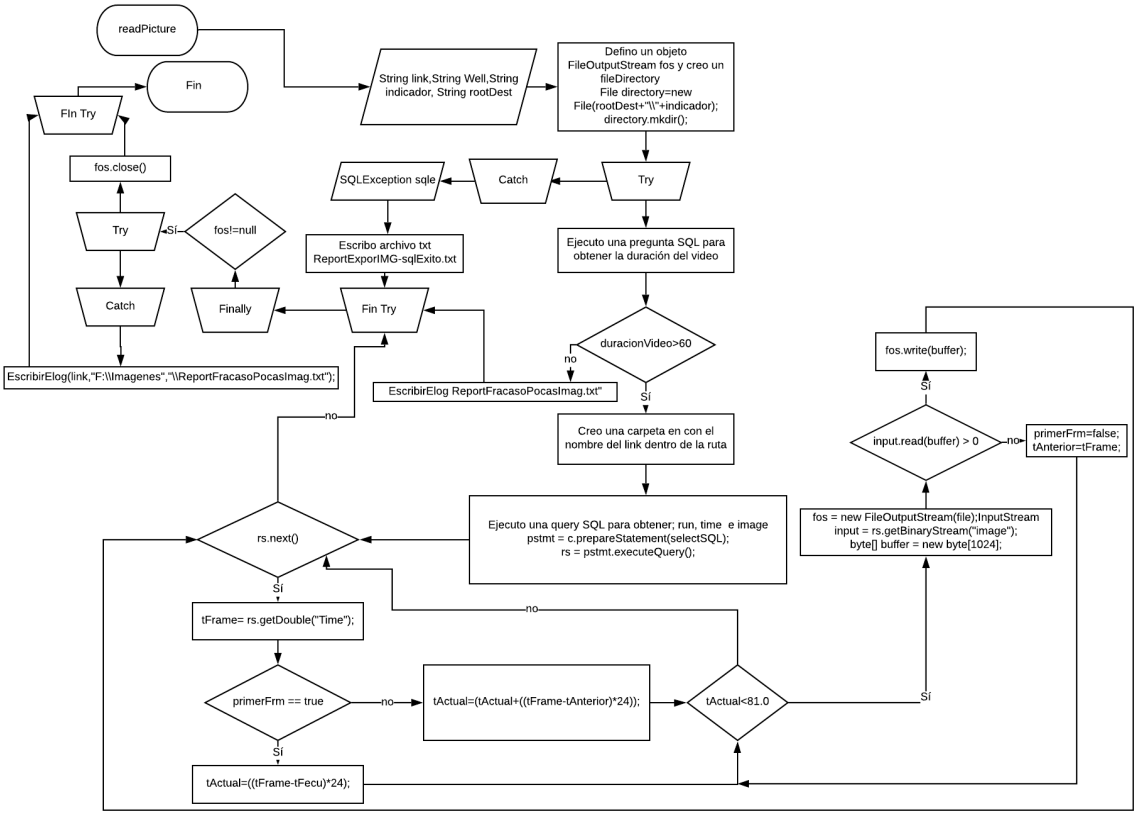


Figura B.3: Método readPicture

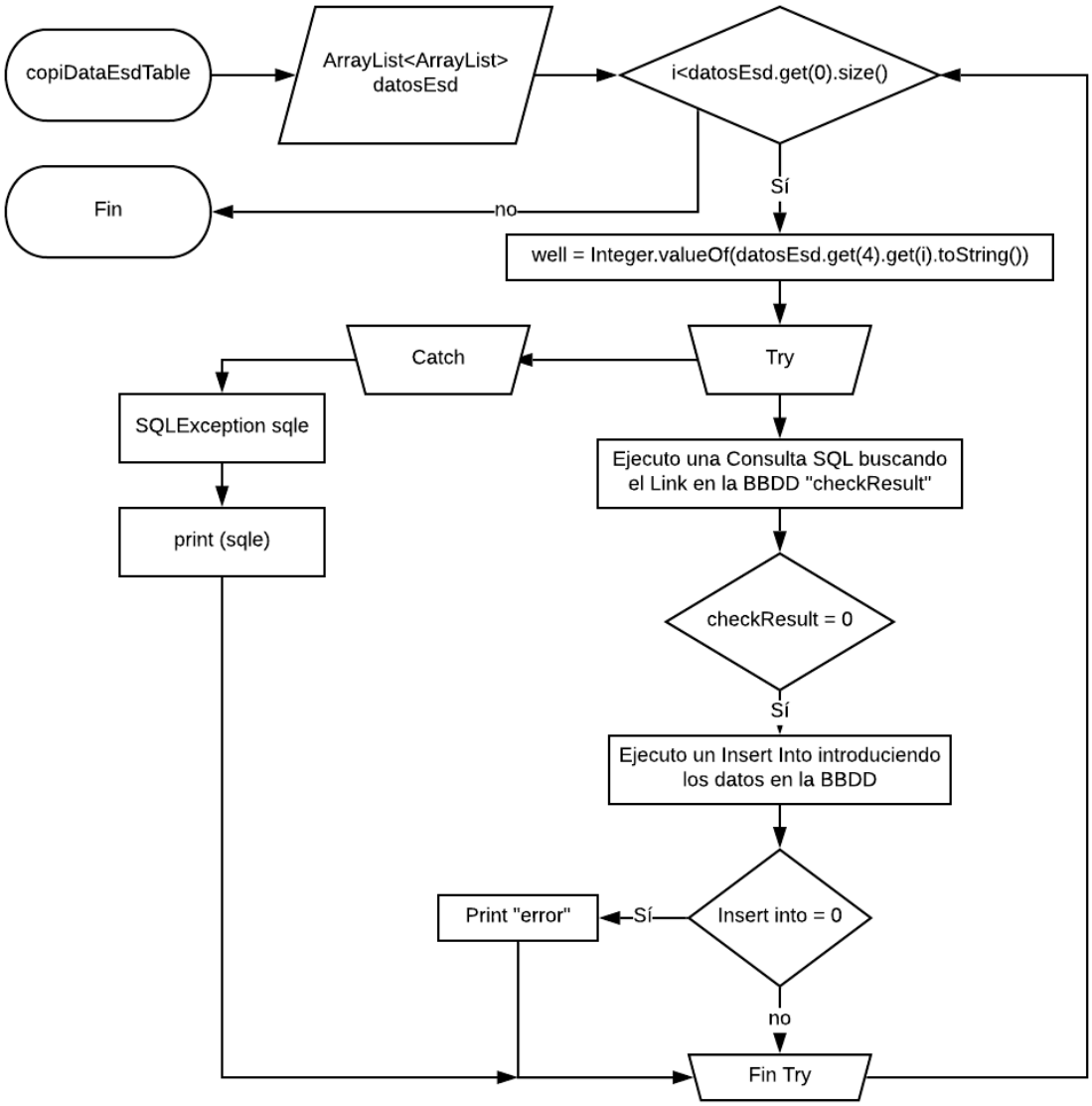


Figura B.4: Método copiESdTable

Apéndice C

Clase dataExcel

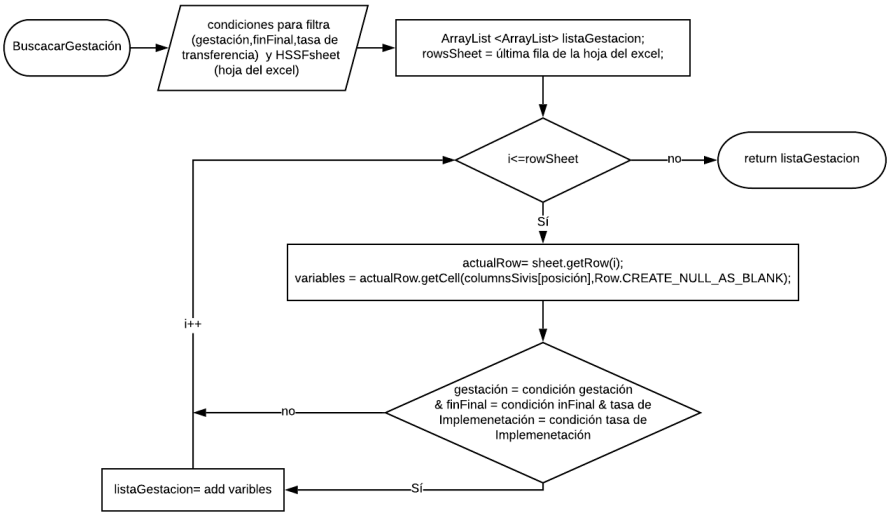


Figura C.1: Diagrama buscar gestación

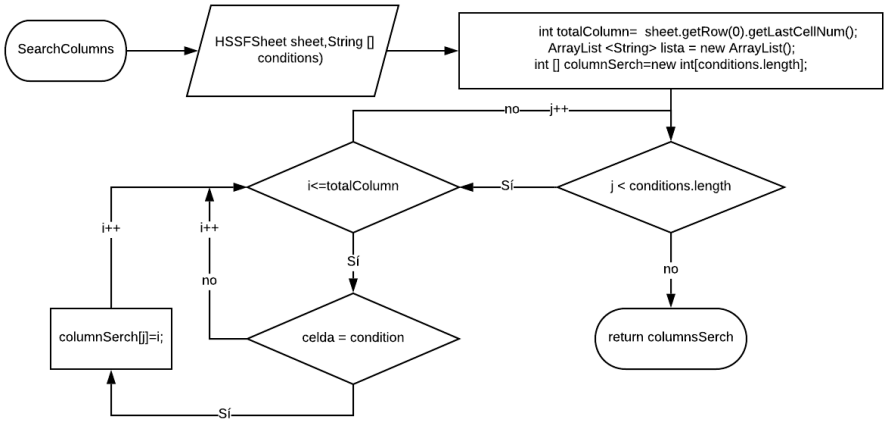


Figura C.2: Diagrama para buscar la posición de las columnas