Escuela Politécnica Superior

Clasificación de imágenes TC de tórax afectados por tuberculosis

Grado en Ingeniería Robótica

Trabajo Fin de Grado

Autor:
Irene Llopis Quereda

Tutor/es:
Andrés Fuster Guillo

Pablo Gil Vázquez

Julio 2018

Universitat d'Alacant
Universidad de Alicante

MOTIVACIÓN PERSONAL Y CONTEXTO

Motivación Personal

Hay dos cosas que me interesan especialmente. Una son las enfermedades y el daño que provocan y la segunda es, precisamente, emplear la tecnología para prevenir o paliar sus efectos. Considero que actualmente, con los numerosos avances en esa tecnología, se puede ayudar mucho en el diagnóstico precoz de enfermedades e incluso en apoyar a los especialistas en la toma de decisiones sobre los tratamientos más adecuados.

Por eso cuando me propusieron colaborar hace un año en un trabajo sobre el diagnóstico de la tuberculosis, me interesé mucho por el tema. Desconocía que una enfermedad que parecía antigua seguía causando tantas muertes y afectando la vida de muchas personas. Pensaba que era complicado sin tener unos profundos conocimientos sobre la enfermedad poder aportar un granito de arena en la lucha contra ella. Pero empecé a leer artículos sobre esta, así como de formas de detección de otros tipos de enfermedades mediante imágenes. Aquel primer trabajo en entorno competitivo fue muy interesante ya que me permitió reforzar mis conocimientos en tratamiento de imágenes con una herramienta que había utilizado mucho durante la carrera, el Matlab. También me permitió introducirme en el mundo del Deep Learning. Los primeros resultados no fueron los esperados, pero me permitieron conocer en mayor profundidad otros modelos utilizados por otros grupos de investigación y aprender de los mismos. Este trabajo final de grado recoge parte de ese camino de aprendizaje en un mundo que me apasiona y del que estoy convencida que va a ser referencia en los próximos años.

Contexto

Lo que hace unos años era inviable, ahora con la cantidad de información disponible y la enorme potencia de los sistemas informáticos, ya empieza a dar resultados. El análisis en un tiempo razonable de imágenes médicas puede dar una información de vital importancia para detectar enfermedades e incluso subtipos dentro de las mismas, con el objeto de ayudar a los profesionales de la medicina a la elección de los mejores tratamientos.

Es un campo de estudio en plena ebullición dada la importancia del tema y los esperanzadores resultados que está produciendo. Una de las enfermedades que se resiste a desaparecer es la tuberculosis, entre otros motivos por la dificultad del diagnóstico del tipo de enfermedad o su resistencia al tratamiento con determinados tipos de antibióticos.

Se han desarrollado varios foros de trabajo para el estudio de las tomografías computarizadas de pulmones para conseguir dar la mayor información posible. En las conferencias CLEF se ha dedicado una serie de tareas específicas a la lucha contra la tuberculosis desde el análisis de dichas imágenes. La tecnología es un aliado fiel de la sanidad ya que es capaz de dar mucha información adicional a la meramente visual. Entre otros modelos, el análisis de imágenes con el cálculo del *optical flow* o el uso del Deep Learning pueden permitirnos lograr grandes mejoras en el este campo. Es parte del objetivo de este trabajo, el estudio y análisis automático de imágenes para obtener la mayor información posible.

OBJETIVOS

Objetivo principal

El objetivo principal de este trabajo es el análisis de imágenes médicas, en concreto las tomografías computarizadas de pecho proporcionadas por las conferencias CLEF.

Para ello me centraré en la resolución de la primera tarea propuesta: Asignación de la severidad de la tuberculosis en distintos pacientes.

Objetivos específicos

- Probar la eficacia de nuevas técnicas que todavía no se habían aplicado a este campo de estudio, como son el cálculo del ADV y el flujo óptico.
- Profundizar en el conocimiento del uso de aprendizaje automático.
- Estudio de modelos de tratamiento de imágenes que ayuden en la detección de enfermedades.

AGRADECIMIENTOS

Quiero agradecer a mis tutores, Pablo Gil y Andrés Fuster, así como al profesor Jorge Azorín por su apoyo y guía en todo momento. Hubiese sido imposible llegar al final sin su ayuda, paciencia y rapidez a la hora de resolver mis dudas o plantear otras.

También me gustaría darle las gracias al mejor padre del mundo (criterio totalmente objetivo) el cual me ha estado dando ánimos todos y cada uno de los días, además de aguantar mis quejas y frustraciones. Y no nos olvidemos de mi querida madre, que aunque no entendiera mucho lo que estaba haciendo, se preocupaba y me preguntaba incansablemente por cómo me iba.

CONTENIDO

1. INTRODUCCIÓN

El objetivo de este trabajo es el de desarrollar modelos que permitan la detección de forma automática del grado y tipo de tuberculosis a través de la información obtenida a partir de imágenes tomográficas.

1.1. Planificación

El desarrollo del trabajo ha estado formado por una serie de fases que como son la definición del problema, estudio del estado del arte, elección de los sistemas para solucionar dicho problema planteado, evaluación de los mismos y estudio de las conclusiones finales.

La temporalización de las fases ha sido la siguiente:

a. **Introducción del problema a resolver (abril- mayo 2018)**

   La idea surge con mi integración en un grupo de trabajo de la Universidad de Alicante, que participaba en unas conferencias-competiciones con el objetivo de desarrollar sistemas que permitieran detectar grados y tipos de tuberculosis en imágenes tomográficas (Llopis, Fuster-Guilló, Ramón Rico-Juan, Azorín-López, & Llopis, n.d.).

   Durante esta fase descubrí la importancia de la lucha contra una enfermedad que apenas conocía y que parecía del pasado. También me permitió realizar mis primeros trabajos sobre procesamiento de imágenes utilizando Matlab, así como el conocimiento del trabajo con redes neuronales. Fruto del trabajo en esta fase es mi participación en un artículo [2] para las conferencias CLEF 2018

b. **Determinación del Trabajo de Fin de Grado a realizar (junio 2018)**

   El tema me parecía muy interesante como trabajo de ingeniería, por lo que me decidí seguir trabajando con el mismo dentro del marco del Trabajo Final de Grado de Ingeniería Robótica.

c. Formación en redes neuronales (julio 2018)

Junto con mis tutores determinamos que sería una buena idea recibir más formación en el mundo de las redes neuronales, por ello asistí a un curso realizado en la Universidad de Alicante: Introducción al Deep Learning.

d. Estudio del estado del arte (enero – marzo 2019)

Durante esta fase realicé estudios en dos ámbitos fundamentales, por una parte en el conocimiento de la enfermedad que se desea tratar y por otro, un análisis de otros sistemas que han propuesto soluciones para el mismo problema.

e. Análisis de propuestas a realizar, desarrollo de las mismas y estudio de los resultados (abril – julio 2019)

A partir del estudio realizado previamente, se hicieron una serie de propuestas a implementar apara analizar su eficacia.

El modelo de trabajo ha sido la definición de tareas a realizar, las cuales incluían tanto las propias de la escritura de la memoria final como el desarrollo de los distintos experimentos realizados durante el trabajo. De forma consensuada con los directores del proyecto se priorizaban las tareas a realizar. Mediante la herramienta Trello, hemos podido no solo especificar de forma clara las tareas y el orden, sino también las que iban siendo finalizadas y revisadas por los tutores.

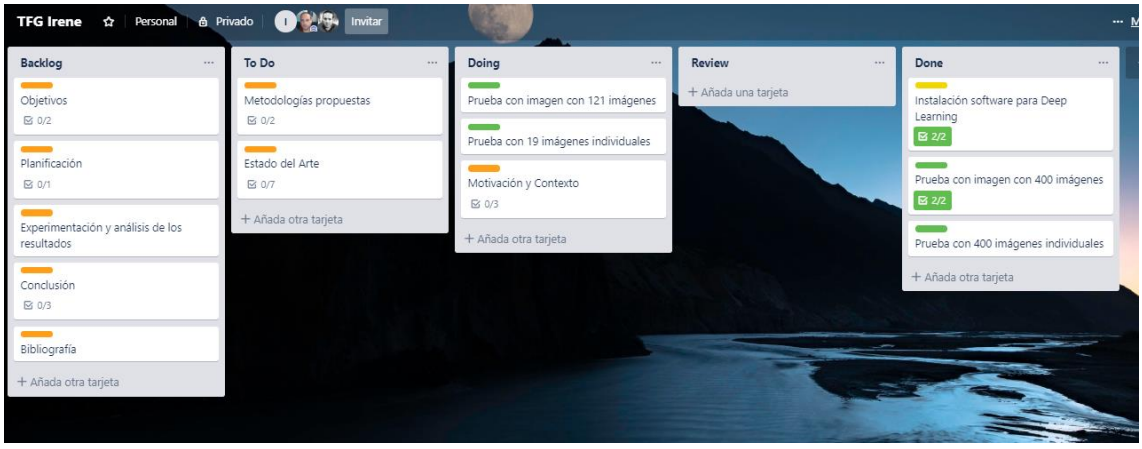


Figura 1: Herramienta Trello

Con el uso de etiquetas de colores se podía diferenciar el tipo de tarea y, por otro lado, el estado de cada una de ellas.

Ha supuesto una a la hora de planificar el trabajo y un conocimiento del estado del mismo de cara a cumplir los plazos de entrega.

f. **Obtención de conclusiones (julio 2019)**

Con las pruebas realizadas y contrastando resultados, he podido obtener una serie de conclusiones acerca de los modelos propuestos.

1.2. Estructura del proyecto

El Trabajo de Fin de Grado está estructurado en una serie de capítulos. En este primero se han definido los objetivos a conseguir, así como una planificación para poder conseguir dichos objetivos.

En el siguiente, se ha realizado un estudio del estado del arte sobre el tema propuesto en una serie de ámbitos muy diferenciados. En el primer apartado se han descrito las características fundamentales de la tuberculosis, los modelos de detección tradicionales y el grado y tipo de enfermedad. En un segundo subapartado he realizado un estudio de los diferentes tipos de imágenes médicas que se pueden utilizar, ya que son parte fundamental del proyecto. Dentro de este mismo capítulo se describe la parte teórica de lo que van a ser las propuestas en las que se base el trabajo, que son el uso de técnicas de Deep Learning y de flujo óptico. Por último, se ha dedicado un apartado a explicar qué son las conferencias CLEF y su importancia, ya que me han permitido disponer de colecciones de datos para poder evaluar el sistema. Finalmente, se explican las tareas presentadas y las medidas de evaluación utilizadas en dicha conferencia para medir la eficacia de las propuestas realizadas para resolver las distintas tareas.

En el capítulo tres describo los dos modelos propuestos dentro de este trabajo, el uso de redes neuronales y el ADV. En este capítulo, se detalla cómo he decidido preparar la información para procesarla y obtener los resultados.

En el capítulo cuatro se describen los resultados obtenidos para cada una de las pruebas realizadas, descritas previamente en el capítulo anterior.

A continuación, en el siguiente capítulo, se detallan las conclusiones obtenidas tras la realización del trabajo y se contrasta la eficacia de cada uno de los modelos propuestos.

Por último, en el capítulo seis se incluyen las referencias bibliográficas utilizadas.

2. ESTADO DEL ARTE

2.1. La Tuberculosis

La tuberculosis es una enfermedad causada por la bacteria *Mycobacterium Tuberculosis* o bacilo de Koch. El principal órgano afectado son los pulmones, pero también podemos encontrar afecciones en los riñones, la columna vertebral y el cerebro.

Se trata de una de las enfermedades que causan más muertes en el mundo:

- Un cuarto de la población mundial la padece
- En 2017 se registraron 10 millones de afectados y 1.3 millones de muertes a nivel mundial
- Causa la muerte de más personas que la malaria y el SIDA combinados

2.1.1 ¿Cómo se transmite la tuberculosis?

La transmisión se realiza a través del aire, cuando una persona que padece esta enfermedad tose, estornuda, habla o escupe, de forma que las personas que se encuentran cerca pueden inhalar estas bacterias.

También puede transmitirse a través de las heces y mediante la orina.

Una vez inhaladas, se pueden alojar en los pulmones, donde comienzan a multiplicarse. Además, como hemos mencionado previamente, pueden moverse a través de la sangre para llegar a otros órganos.

Hay que tener claro que la única forma de contagiar la tuberculosis entre personas es a través de las bacterias alojadas en pulmones y garganta. La que afecta a los riñones o la columna vertebral, entre otras zonas, no sería contagiosa.

2.1.2 Tipos clínico-morfológicos de tuberculosis

1. Tuberculosis primaria

   Infección de un individuo que no ha sido previamente infectado o inmunizado

2. Tuberculosis secundaria

   La lesión aumenta de tamaño y se localiza en la zona apical de uno o los dos pulmones.

   Existen varios estadios de tuberculosis pulmonar secundaria:

   - Tuberculosis aguda aguda
   - Tuberculosis fibrofocal
   - Tuberculosis infiltrativa
   - Tuberculoma
   - Neumonía tuberculosa caseosa
   - Tuberculosis cavernosa aguda
   - Fibrocavei-nous tuberculosis
   - Tuberculosis cirrótica

3. Tuberculosis hematógena

   Propagación de la tuberculosis a partir de la tuberculosis pulmonar, debida a la infección de la vena pulmonar, produciendo una lesión orgánica diseminada o aislada en diferentes órganos extra pulmonares o en la arteria pulmonar, produciendo lesiones miliares dentro del pulmón.

2.1.3 Signos y síntomas

La infección inicial suele ser asintomática.

A partir de ahí, el 95% de las personas entran en fase de latencia y para el 5% restante la infección inicial evoluciona hacia la enfermedad.

Los síntomas más comunes son:

- Debilidad o fatiga
- Pérdida de peso sin causa conocida
- Falta de apetito
- Escalofríos
- Fiebre
- Sudores nocturnos
- Astenia
- Anorexia

Además, se pueden presentar otros síntomas dependiendo del área afectada. Por ejemplo:

- Tuberculosis pulmonar:

  Es la que mayores manifestaciones ofrece, ya que es la más frecuente.

  - Tos intensa que dura 3 o más semanas
  - Dolor en el pecho
  - Hemoptisis (tos con sangre) o esputo
  - Disnea

- Sistema circulatorio:
    - Taquicardia
    - Disnea
    - Sudoración
    - Anemia

- Sistema digestivo:
    - Náuseas
    - Constipación
    - Diarreas
    - Irregularidades en la menstruación o amenorrea

- Sistema nervioso:
    - Nerviosismo
    - Irritabilidad
    - Depresión
    - Rasgos de psicosis
    - Alteración en los reflejos vasomotores

2.1.4 Personas con alto riesgo de padecer tuberculosis

- Personas con infección por el VIH
- Personas infectadas por tuberculosis en los últimos 2 años
- Personas que tienen infección de tuberculosis latente, aunque nunca presenten la enfermedad, son más propensos a enfermar de otras variedades de tuberculosis
- Bebés y niños pequeños
- Personas que se inyectan drogas ilícitas
- Personas que padecen enfermedades que debilitan el sistema inmunitario, como lupus o la esclerosis múltiple.
- Ancianos
- Personas que no han recibido un tratamiento adecuado para dicha enfermedad

2.1.5 Tratamiento

Tuberculosis latente

Las personas con este tipo de tuberculosis no tienen síntomas ni pueden contagiar a otras personas. Sin embargo, si las bacterias se activan y comienzan a multiplicarse, habrá que tratarla para evitar que acabe padeciendo la enfermedad de la tuberculosis.

Actualmente existen cuatro tratamientos para la infección de tuberculosis latente. La elección de uno frente a otro depende de la duración, eligiendo el más corto siempre que sea posible, ya que hay más probabilidades de que los pacientes completen los tratamientos cuando tienen una menor duración.

Medicamento	Duración	Intervalo
Isoniacida y rifapentina	3 meses	Una vez a la semana
Rifampina	4 meses	Diariamente
Isoniacida	6 meses	Diariamente o dos veces a la semana
Isoniacida	9 meses	Diariamente

Enfermedad de la tuberculosis

Para el correcto funcionamiento del tratamiento de la enfermedad es importante cumplir con el plazo y forma en la que se le ha indicado, ya que en caso contrario las bacterias podrían desarrollar resistencia al medicamento, complicando el tratamiento.

El periodo varía entre 6 a 9 meses distinguiendo 2 fases. Una fase inicial de 2 meses y una fase de continuación con una duración variable entre 4 o 7 meses, dependiendo del tratamiento elegido.

Entre los 10 medicamentos aprobados podemos distinguir la Isoniazida, Rifampina, Etambutol y Pirazinamida como los de primera línea contra la tuberculosis.

Tuberculosis resistente a los medicamentos

Encontramos dos tipos diferentes:

- Tuberculosis Multirresistente (MDR TB): TB resistente al isoniazida (INH) y rifampicina (RMP)
- Tuberculosis Extremadamente Resistente a los medicamentos (XDR TB): MDR-TB con una resistencia adicional a la fluoroquinolona y a uno de los tres fármacos de segunda línea usados en el tratamiento como mínimo.

Los motivos por los que una persona puede desarrollar una de estas variantes de la tuberculosis son:

  - Diagnóstico erróneo o aplicación del tratamiento indebida
  - Pacientes que no toman correctamente los medicamentos necesarios para su tratamiento
  - El uso de medicamentos de baja calidad y recorte de medicinas
  - Ser infectado por alguien que ya padece tuberculosis resistente a los medicamentos

Es muy complicado tratar y curar este tipo de tuberculosis ya que un manejo inadecuado puede poner en riesgo la vida del paciente. Además, este tratamiento es mucho más caro y prolongado, suponiendo una carga adicional para los pacientes y los sistemas sanitarios.

2.1.6 Pruebas y diagnóstico

Usualmente, los enfermos por tuberculosis presentan uno o más síntomas de los mencionados anteriormente.

Las personas que tengan síntomas (aunque muestre un resultado negativo en alguna de las pruebas) o un resultado positivo en la prueba de detección de tuberculosis, deben ser evaluadas para detectar si realmente padecen o no tuberculosis.

Existen dos tipos de pruebas para detectar si una persona ha sido infectada con bacterias de la tuberculosis:

- La prueba cutánea de tuberculina

  Se inyecta en la parte inferior del brazo una pequeña cantidad de tuberculina y, tras 48-72 horas, el paciente debe regresar para que el personal médico analice el tamaño del área elevada, endurecida o hinchada.

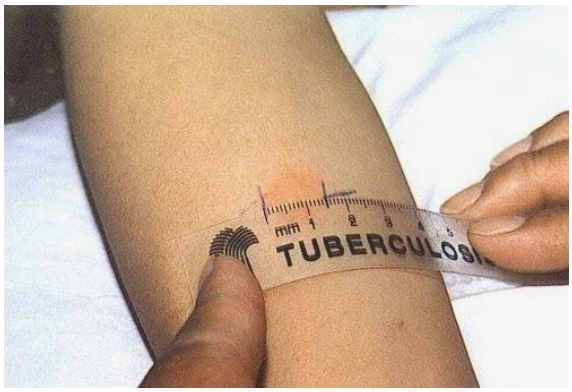


Figura 2: Prueba cutánea de tuberculina

  La interpretación se hará de la siguiente forma:

  - 0 – 5 mm: no reactor
  - 6 – 14 mm: reactor → puede deberse a una vacunación BCG o una infección por una micobacteria atípica
  - Más de 14 mm: hiperérgico → solo se produce por infección de *M. tuberculosis*

- *Pruebas de sangre*

En caso de un resultado positivo, sería necesario realizar otras pruebas, ya que con las mencionadas anteriormente no se puede confirmar si una persona padece una infección de tuberculosis latente o la enfermedad de tuberculosis.

Para ello utilizan otros métodos de diagnóstico:

- Antecedentes médicos

  Hay que tener en cuenta:

  - ✓ Antecedentes de exposición a tuberculosis
  - ✓ Factores demográficos como país de origen, edad, raza, ocupación… ya que éstos pueden aumentar el riesgo de exposición a la enfermedad
  - ✓ Paciente con otras afecciones como VIH o diabetes

- Examen físico

  Proporciona información sobre el estado del paciente y otros factores que podrían influir en el tratamiento contra la tuberculosis.

- Radiografía de tórax anteroposterior

  Se utiliza para detectar anormalidades en el pecho, lesiones que pueden aparecer en cualquier parte de los pulmones, con distintos tamaños, formas, densidad y cavitación, siendo más común la lesión apical. Aunque esta prueba no puede usarse como diagnóstico definitivo, se utiliza para descartar la posibilidad de tuberculosis pulmonar en una persona que haya tenido una reacción positiva a la prueba cutánea de la tuberculina o a la prueba de sangre.

  La radiografía de tórax se considera fundamental en el diagnóstico. Por eso más adelante nos centraremos en dicha prueba, utilizando imágenes de diferentes radiografías, las cuales procesaremos para determinar si un paciente padece tuberculosis o no, y en caso afirmativo, cuál de todos los tipos.

- Microbiología diagnóstica o baciloscopia

  Se realiza un cultivo de varias muestras de un frotis de esputo u otras muestras para comprobar la presencia de bacilos acidorresistentes (BAAR), los cuales deben ser *M. tuberculosis.*

  Un resultado positivo de este confirmaría el diagnóstico.

- Resistencia a los medicamentos

  Cuando la *M. tuberculosis* se aísla, debe analizarse para determinar su resistencia a los medicamentos ya que es de gran importancia identificar esta resistencia lo antes posible para garantizar un tratamiento eficaz.

2.2 Imágenes médicas

Las imágenes médicas se utilizan como medida del cuerpo humano a diferentes escalas empleando distintas modalidades y basándose en las propiedades físicas del mismo.

Usualmente se trata de imágenes volumétricas (3D), aunque a veces se leS incorpora una dimensión adicional de tiempo (4D) o múltiples canales (4-5D).

En la actualidad, el uso de imágenes médicas en el diagnóstico de enfermedades se ha incrementado, ya que se trata de una fuente muy confiable y sencilla.

Existen diferentes tipos en función de la región a analizar y el dispositivo empleado para su captura:

- Radiografía

  Consiste en la obtención de una imagen de una zona anatómica utilizando rayos X, los cuales se hacen pasar a través del cuerpo y se absorben en distintas cantidades según la densidad del material que atraviesan, de forma que los materiales densos (huesos o metales) aparecen de color blanco, el aire de color negro, y la grasa y músculos se representan como sombras de color gris.
  Principalmente se utiliza en huesos y dientes para detectar fracturas, infecciones, artritis, caries dentales, osteoporosis, cáncer de huesos…, en tórax para el diagnóstico de enfermedades pulmonares como la tuberculosis, el cáncer de mama, agrandamiento de corazón o la obstrucción de vasos sanguíneos, y en el abdomen, en caso de problemas en el sistema digestivo u objetos tragados.

Figura 3: Radiografías de columna vertebral, tórax, cráneo y rodilla

- Resonancia magnética

Se trata de una tecnología de imágenes no invasiva que produce imágenes anatómicas tridimensionales sin el uso de radiación dañina.

La IRM (Imagen por Resonancia Magnética) utiliza imanes que producen un potente campo magnético, obligando a los protones que se encuentran en el agua que compone los tejidos vivos a alinearse con el cuerpo.

Al generar una corriente a través del paciente, los protones se estimulan y giran fuera del equilibrio, luchando contra la fuerza del campo magnético.

Cuando esta corriente cesa, los sensores de la IRM detectan la energía liberada mientras los protones se realinean con dicho campo.

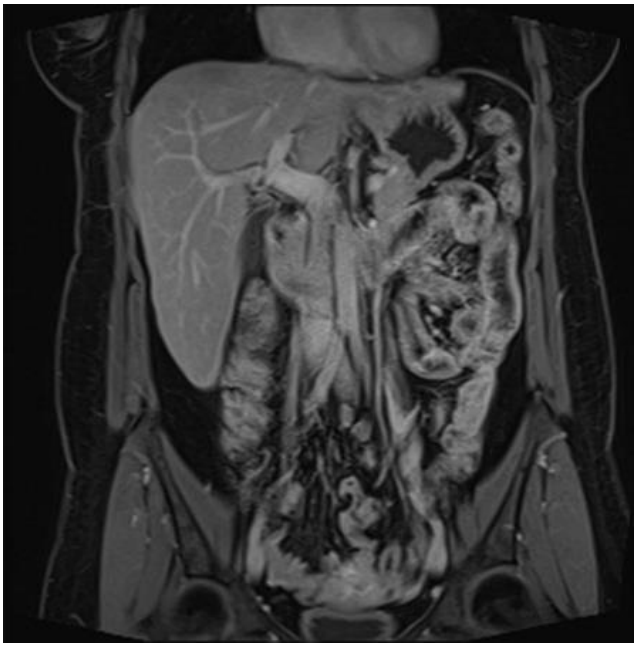


Figura 5: Resonancia magnética de la zona del abdomen

El tiempo en realinearse y la cantidad de energía liberada depende del tejido.

- Tomografías computarizadas

TC (Tomografía Computarizada) es un procedimiento computarizado de imágenes mediante el uso de rayos X.

Se proyecta un haz de rayos X al paciente y se gira rápidamente alrededor del cuerpo, produciendo señales que son procesadas por la computadora de la máquina para generar imágenes transversales del cuerpo denominadas "cortes" o imágenes tomográficas.

Contienen información más detallada que una radiografía ya que los cortes sucesivos se pueden unir digitalmente para formar una imagen tridimensional, permitiendo la identificación y ubicación de las estructuras a analizar más fácilmente.

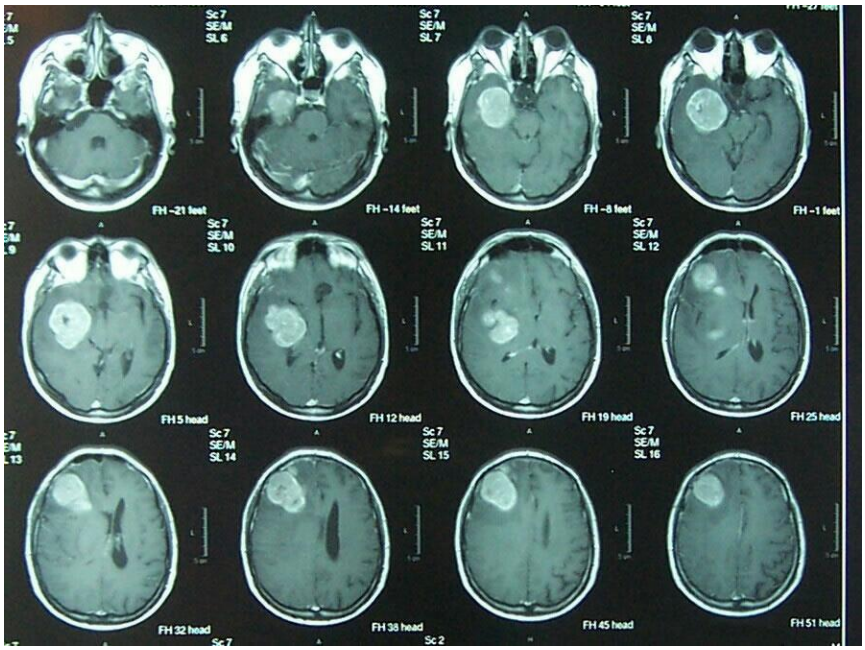


Figura 6: Tomografía computarizada de cráneo

- Ecografías

La ecografía es una técnica diagnóstica que se utiliza para explorar los órganos del interior del organismo.

Se basa en la reflexión que experimenta el sonido cuando choca con una superficie en función de su densidad.

El ecógrafo emite un sonido de frecuencia muy alta (ultrasonido), inaudible por el oído humano, y éste, al chocar con los distintos órganos, genera una imagen que se representa en un monitor de televisión.

Figura 7: Ecografía obstétrica

2.3 Detección de tuberculosis mediante imágenes médicas

Aunque la radiografía de pecho se mantiene como modalidad de imágenes básica para la detección de tuberculosis pulmonar, cierto es que la tomografía computarizada es más útil en la evaluación de tuberculosis pulmonar y extrapulmonar ya que proporciona información más detallada que la primera.

El papel del uso de imágenes en la tuberculosis ha mostrado un crecimiento exponencial ya que permite comprobar el grado de la enfermedad, la respuesta a la terapia o la detección de infección residual tras el tratamiento (Bomanji, Gupta, Gulati, & Das, 2015).

Además, con estas pruebas, se pueden percibir manchas blancas en los pulmones, las cuales indicarían las zonas donde el sistema inmunitario ha encapsulado las bacterias causantes de esta enfermedad o mostrar los cambios en el pulmón debido a la tuberculosis.

Figura 8: Imágenes radiológicas de tórax

El diagnóstico de detección asistida por ordenador (CAD) se trata de una herramienta en el procesamiento de imágenes médicas. Los objetivos del CAD son obtener un mejor resultado en la exactitud del diagnóstico y una buena interpretación de las imágenes radiológicas utilizando un ordenador como herramienta.

Existen dos tipos de enfoques utilizados: encontrar la localización de las lesiones y la cuantificación de las características de las radiografías, ya sean patrones normales o anormales.

Para ello, se le realizan una serie de procedimientos a las imágenes.(Nyein Naing & Z. Htike, 2015)

2.3.1 Preprocesamiento

La finalidad es mejorar la calidad de la imagen y reducir las partes no deseadas del fondo.

La segmentación de la imagen es realmente importante a la hora de extraer características de gran calidad para la posterior clasificación.

Existen diferentes técnicas que se utilizan para realizar la segmentación del pulmón como la segmentación pulmonar basada en corte gráfico, la segmentación de cuencas, el método de umbral de Otsu, el modelo de forma activa y modelo de apariencia activa o el modelo de forma activa multisegmento.

La extracción de características no es más que la reducción de una imagen a valores de características y la obtención de mejores resultados en la clasificación. Contamos tanto con características geométricas como de textura.

2.3.2 Clasificación

Existen dos tipos de clasificación:

- Clasificación supervisada: utiliza los resultados obtenidos a partir de muestras de entrenamiento para clasificar una imagen.
- Clasificación no supervisada: encuentra clústeres de la imagen multibanda sin el análisis de intervención.

Se utilizan distintos clasificadores como podrían ser:

a) Fuzzy – ART Neural Networks

Este tipo de redes neuronales integran las ventajas de los operadores de lógica difusa (fuzzy) y las características básicas de la implementación de ART (Adaptative Resonance Theory), donde las operaciones de intersección y unión utilizadas en ART1 se sustituyen por los operadores mencionados anteriormente.

b) Árboles de decisión

Un árbol de decisión contiene nodos y ramas donde cada nodo representa una decisión o prueba.

c) Support Vector Machine (SVM)

Se trata de un clasificador supervisado no-probabilístico que genera hiperplanos para separar muestras de dos diferentes clases en un espacio con infinitas dimensiones.

En este caso, se propuso el algoritmo para identificar y contar el número de bacterias de tuberculosis en las imágenes microscópicas.

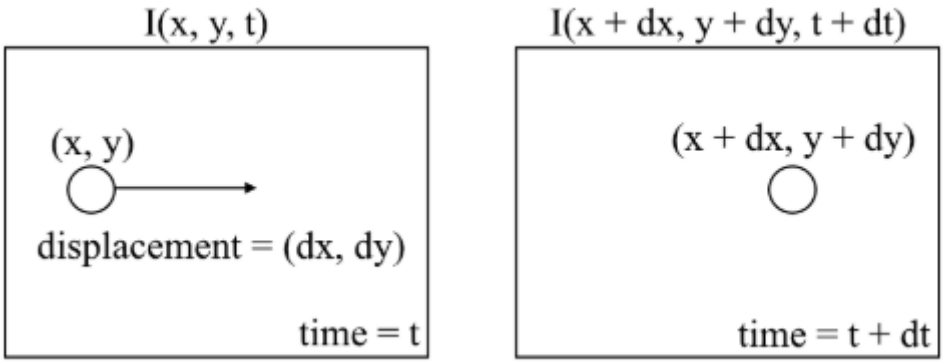
d) Principal Component Analysis (PCA)

Se utiliza como método para la clasificación de imágenes radiológicas basado en las características extraídas mediante SIFT.

PCA identifica las principales características o componentes, conocidos como Componentes Principales, que en su conjunto representan el objeto completo.

2.4 Flujo óptico

En una escena con cámara fija, el flujo óptico es el movimiento de los objetos entre dos fotogramas consecutivos de una secuencia, causado por el movimiento relativo de los objetos con respecto a la cámara.



La intensidad de la imagen (I) se puede expresar en función del espacio (x,y) y el tiempo (t). De forma que si tomamos la primera imagen I(x,y,t) y el movimiento de sus píxeles (dx,dy) en función del tiempo (t), obtenemos el siguiente fotograma como I(x+dx, y+ dy, t+ dt).

Figura 9: Flujo óptico en una carretera

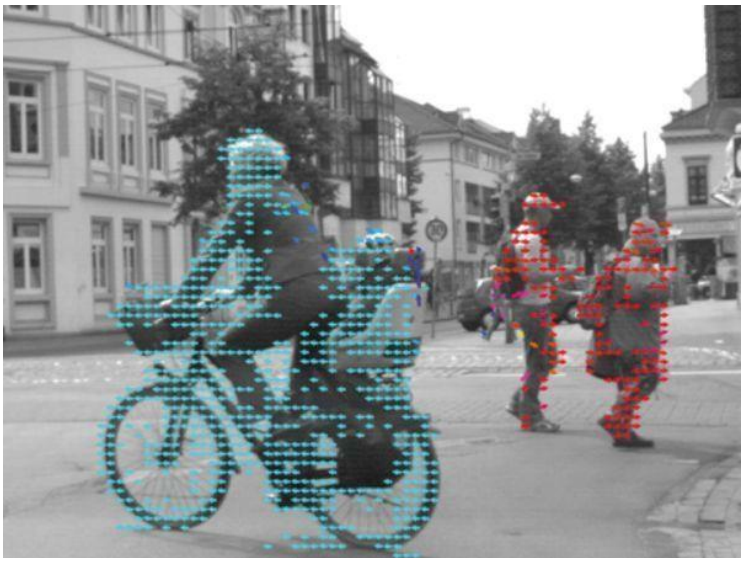


Figura 10: Flujo óptico del movimiento de las personas

Mayormente el flujo óptico se utiliza en del seguimiento de objetos con aplicaciones como podrían ser la detección de movimiento, la segmentación de objetos, el tiempo hasta la colisión, pero también se puede servir como herramienta en el campo de la medicina.

Por ejemplo, en la estimación del movimiento del pulmón usando tomografías computarizadas.

La idea es aplicarlo en pacientes que están recibiendo radioterapia para eliminar el cáncer de pulmón, con la finalidad de encontrar la máxima dosis que se le puede dar al paciente sin dañar el tejido sano que rodea al tumor.

El algoritmo para la "estimación del flujo óptico de alta precisión" propuesto se utiliza con una secuencia de imágenes CT del pulmón del paciente, las cuales capturan las distintas fases de la respiración de este (Hermann & Werner, n.d.).

En concreto, el experimento se realizó con 10 sets de imágenes torácicas CT 4D (3D + tiempo), de forma que por paciente había 10 muestras de imágenes 3D (una para cada fase distinta de la respiración).

Primeramente, se registran el final de la inspiración (EI) y el final de la expiración (EE) y se generan 300 puntos de referencia para cada par, los cuales se irán analizando a lo largo de la secuencia.

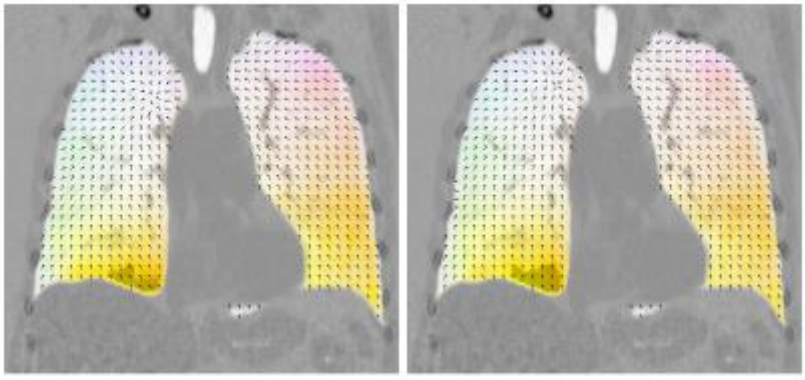


Figura 11: Vista coronal o frontal del tórax

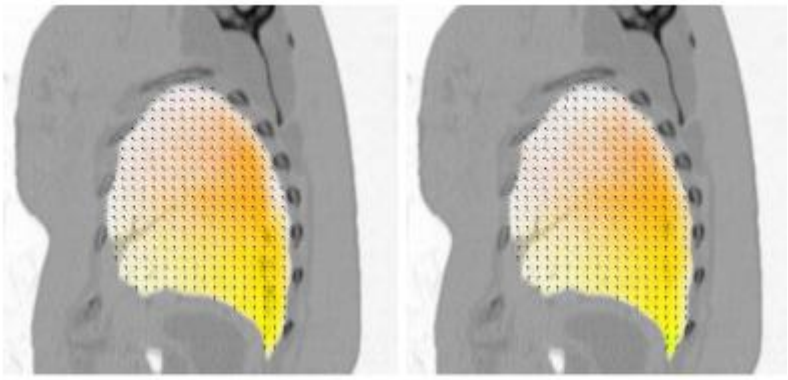


Figura 12: Vista sagital del tórax

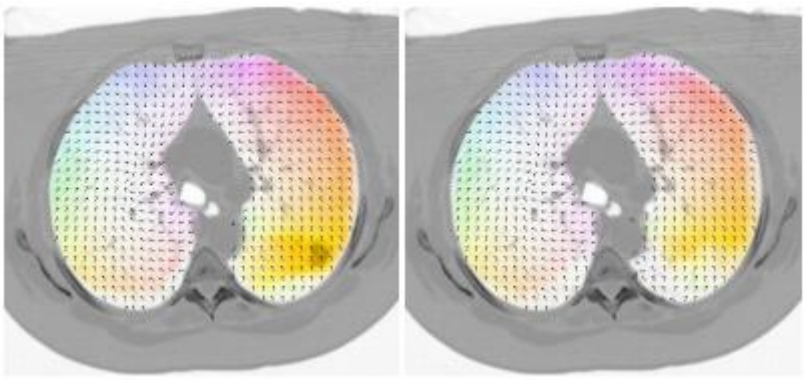


Figura 13: Vista axial del tórax

En las imágenes anteriores se muestra el movimiento estimado entre EI y EE.

La conclusión es que el algoritmo presentado es tan rápido y eficaz como los mejores métodos presentados hasta el momento, mostrando un alto potencial para su uso en aplicaciones clínicas.

2.5 ADV

El ADV (Activity Description Vector) es un vector que se utiliza para el análisis de trayectorias. Éste, describe la actividad en una secuencia de imágenes contando los movimientos en cuatro direcciones del espacio 2D (derecha, izquierda, arriba, abajo) producidos en cada región de la imagen.

Es capaz de superar otros métodos actuales ya que tiene en cuenta la actividad de una región específica sin tener en cuenta la vecindad.

Por ejemplo, se puede utilizar como herramienta para evaluar la habilidad de una red neuronal para la clasificación del comportamiento humano a partir de información extraída de una escena (Azorin-Lopez, Saval-Calvo, Fuster-Guillo, Garcia-Rodriguez, & Mora-Mora, 2017).

Las trayectorias se describen dividiendo la escena en regiones y comprimiendo la información valores acumulativos.

Como el ADV integra toda la información de la trayectoria sin restricciones de tamaño ni secuenciales, es muy apropiado para hacer predicciones. En concreto, el ADV utiliza el número de veces que una persona aparece en un punto específico y los movimientos locales que realiza. Para cada celda, el ADV es la acumulación de los histogramas de movimientos en las cuatro direcciones (arriba, abajo, izquierda, derecha) y la frecuencia.

Los cuatro primeros valores se obtienen con el vector de desplazamiento entre dos puntos consecutivos analizados y la normal correspondiente para cada eje. Con esto conseguimos información sobre la dirección de la trayectoria y la velocidad.

Por otro lado, la frecuencia se calcula con el número de veces que el centroide está en un punto específico del escenario.

Finalmente, el ADV describe la trayectoria de una persona usando la colección de ADVs obtenidos para cada celda.

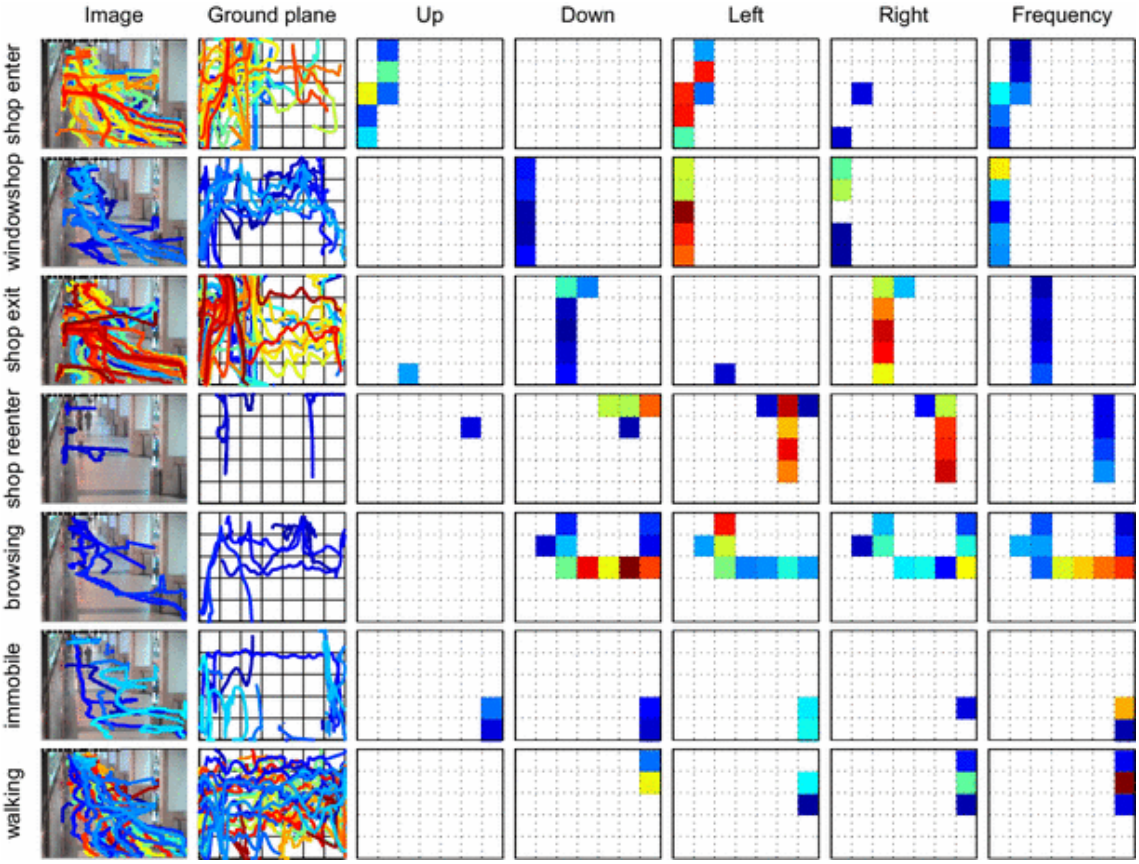


Figura 14: Ejemplo de las trayectorias con los movimientos U, D, L y R y la frecuencia F

A continuación se construye una matriz de dos dimensiones como colección de todos los vectores que forman el mapa de características (FM).Esta matriz, denominada Activity Descriptor Map (ADM) contiene el mismo número de elementos que celdas en las que se divide la escena.

$$\mathrm{ADM} = \begin{pmatrix} \mathrm{ADV}_{1,1} & \mathrm{ADV}_{1,2} & \dots & \mathrm{ADV}_{1,n} \\ \mathrm{ADV}_{2,1} & \mathrm{ADV}_{2,2} & \dots & \mathrm{ADV}_{2,n} \\ \vdots & \vdots & \vdots & \vdots \\ \mathrm{ADV}_{m,1} & \mathrm{ADV}_{m,2} & \dots & \mathrm{ADV}_{m,n} \end{pmatrix}$$

Dicha matriz se utiliza como entrada para una red neuronal, en concreto la 2D-SOM-PINT. Para poder entrenar dicha red, las muestras deber normalizarse en un rango (0,1). Esto se consigue dividiendo cada componente del vector ADV por el máximo valor de cada componente.

Cada región del espacio tiene asociadas unas neuronas concretas, de forma que cada entrada es capaz de activar la neurona relacionada al área correspondiente.

Tras el entrenamiento, esta red muestra un mayor nivel de entendimiento.

Las pruebas realizadas se llevaron a cabo utilizando dos conjuntos de datos incluidos en la base de datos CAVIAR, la entrada de un laboratorio en Francia y un centro comercial en Portugal.

Por un lado, los comportamientos de las personas en el primer dataset son: caminar, hojear, permanecer inmóviles o agacharse. En el segundo caso: entrar en la tienda, mirar el escaparate, salir de la tienda, volver a entrar en una tienda, hojear, permanecer inmóviles o caminar.

Finalmente, se realizaron varias pruebas variando el tamaño de la ADM, analizando la sensibilidad, la especificidad y la exactitud.

Las conclusión obtenida fue que el rendimiento de la red neuronal aumenta a medida que aumenta el número de neuronas y de celdas, obteniendo el mejor resultado con un tamaño de 9 x13 celdas y 27x 39 neuronas para todas las muestras. En cambio, para el caso del laboratorio, a partir de un tamaño de la matriz de 5x7 el resultado siempre es el mismo.

A pesar de ello, el método utilizado tiene un gran resultado en el análisis del comportamiento humano aunque la entrada de la red neuronal sea de un tamaño de 1x1.

Además, se realizaron las mismas pruebas con el conjunto de datos del pasillo del centro comercial utilizando diversos métodos de clasificación (HSMM, MC, PN…), que mostraron que la red 2D-SOM-PINT presenta una mejora respecto a estos en cuanto a sensibilidad.

2.6 Deep Learning

Machine Learning es una rama de la informática que estudia el diseño de algoritmos con la capacidad de "aprender". Un subcampo sería el Deep Learning (aprendizaje profundo), que es una serie de técnicas que hacen uso de las redes neuronales artificiales profundas, es decir con más de una capa oculta, para imitar computacionalmente la estructura y funcionamiento del cerebro.

Una de las librerías implementadas en Phyton más eficientes para la evaluación de modelos de aprendizaje profundo es Tensor Flow, siendo Keras una librería que es capaz de ejecutarse sobre Tensor Flow, facilitando su empleo.

2.6.1 Introducción a las redes neuronales

Perceptrones

La red neuronal más simple es un "perceptrón", que consiste en una sola neurona.

De forma similar a una neurona biológica, una neurona artificial imita la estructura de un árbol, con un nodo de entrada y un nodo de salida, conectados entre ellos.

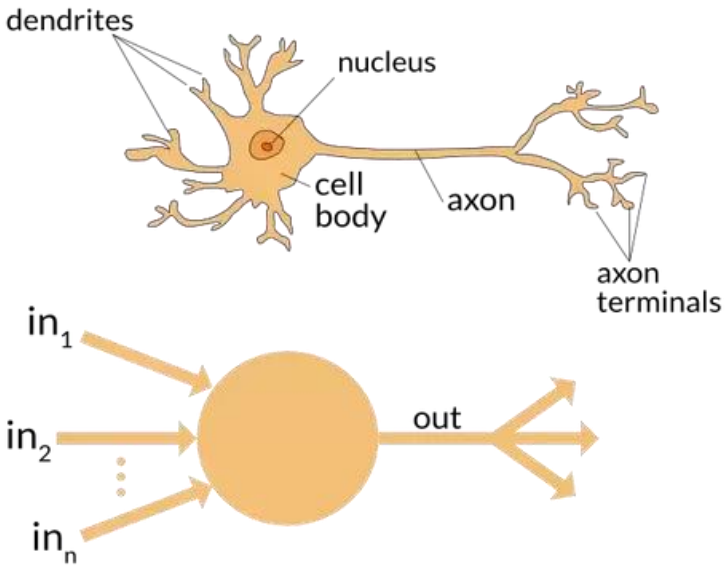


Figura 15: Comparación de una neurona biológica con la neurona artificial

Existen los siguientes seis componentes en las neuronas artificiales, como se puede observar en la (imagen de abajo):

1. Nodos de entrada: cada nodo va asociado a un valor numérico, que puede ser cualquier número real.
2. Conexiones: las conexiones entre nodos llevan asignados un peso, que también es un número real.
3. Sumatorio: teniendo en cuenta los valores de los nodos de entrada y los pesos se computa la suma ponderada de la siguiente forma:

$$y = f(\sum_{i=1}^{n} w_i * x_i)$$

4. Función de activación o transferencia: la cual se encarga de devolver una salida a partir de un valor de entrada, en este caso, el sumatorio obtenido previamente.
   Las funciones que se utilizan buscan que las derivadas sean simples, reduciendo de este modo, el coste computacional.
   Podemos dividirlas en dos tipos:

- ***Funciones de Activación Lineales***

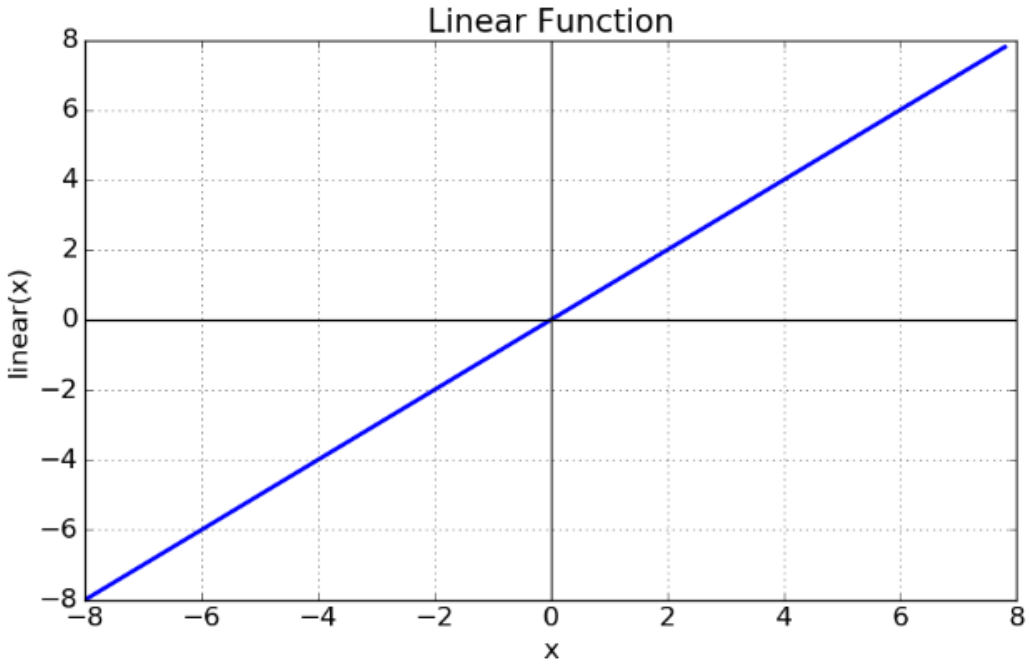


Figura 16: Función lineal

Al utilizar redes neuronales, lo que se suele buscar es que los resultados se encuentren entre 0 y 1 o entre -1 y 1, es decir, que la salida esté limitada dentro de un rango.

En este caso, al tratarse de una función lineal, ocurre todo lo contrario, por lo tanto, no es común utilizar este tipo de funciones como funciones de activación.

- ***Funciones de Activación No-Lineales***

Este tipo de funciones facilitan la generalización del modelo o adaptación de un conjunto de datos y diferenciación entre la salida.

A su vez, se dividen basándose en los rangos utilizados:

a. Sigmoide

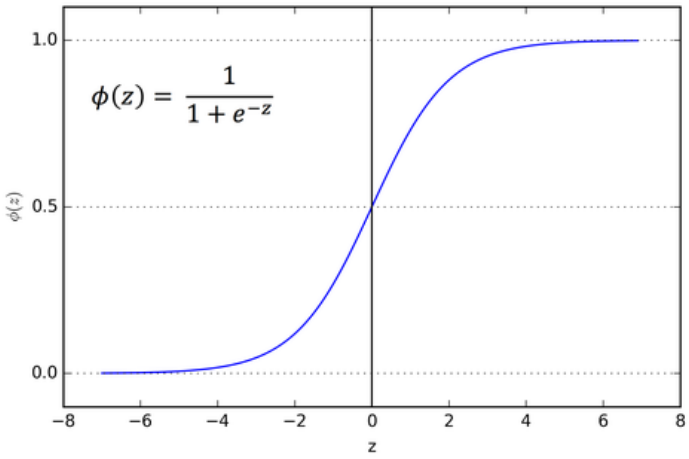


Figura 17: Función sigmoide

Transforma los valores introducidos a una escala (0,1), donde los valores altos tienden asintóticamente a 1 y los valores muy bajos a 0.

Por esta razón, se suele utilizar en modelos que necesitan computar la probabilidad como salida.

Por otro lado, la función sigmoidea pude producir un estancamiento durante el periodo de entrenamiento.

b. Tangente hiperbólica

Esta función tiene la misma logística que la función sigmoidea ya que transforma los valores introducidos a una escala (-1,1), donde los valores altos tienden asintóticamente a 1 y los valores muy bajos a -1, pero funciona mejor.

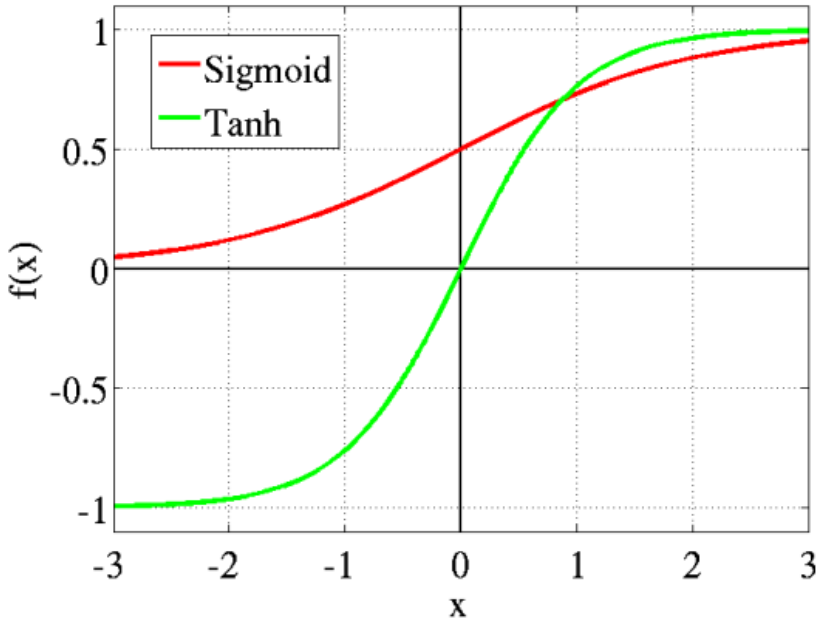


Figura 18: Tangente hiperbólica

La ventaja es que las entradas negativas serán asignadas como negativas y las entradas que sean cero, se asignarán cerca del cero.

Este tipo de funciones se suele utilizar para la clasificación entre dos clases y tiene buen desempeño en redes recurrentes

$$f(x) = \frac{2}{1 - e^{-2x}} - 1$$

c. ReLU (Rectified Lineal Unit)

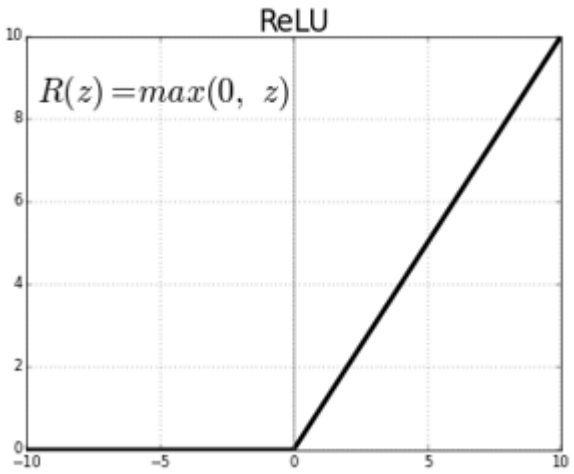


Figura 19: Función ReLU

Se trata de la función de activación más utilizada actualmente ya que se usa en casi todas las redes convolucionales (por su buen desempeño en ellas y el buen comportamiento con el tratamiento de imágenes).

Transforma los valores introducidos anulando los valores negativos y dejando los positivos (activación "Sparse"), consiguiendo un rango de [0, +∞).

El problema reside en que cualquier valor negativo se transforma en 0, de forma que dichos valores no se mapean de manera apropiada.

$$f(x) = \max(0, x) = \begin{cases} 0\ for\ x < 0 \\ x\ for\ x \geq 0 \end{cases}$$

d. Leaky ReLU

Con esta función de activación se intenta resolver la problemática del caso anterior, incrementando el rango de la función ReLU.

Para ello los valores negativos se multiplican por un coeficiente rectificativo y los positivos se dejan como entran.

$$f(x) = \begin{cases} 0\ for\ x < 0 \\ a * x\ for\ x\ \geq 0 \end{cases}$$

$$f(x) = \begin{cases} a * x \; for \; x < 0 \\ x \; for \; x \geq 0 \end{cases}$$

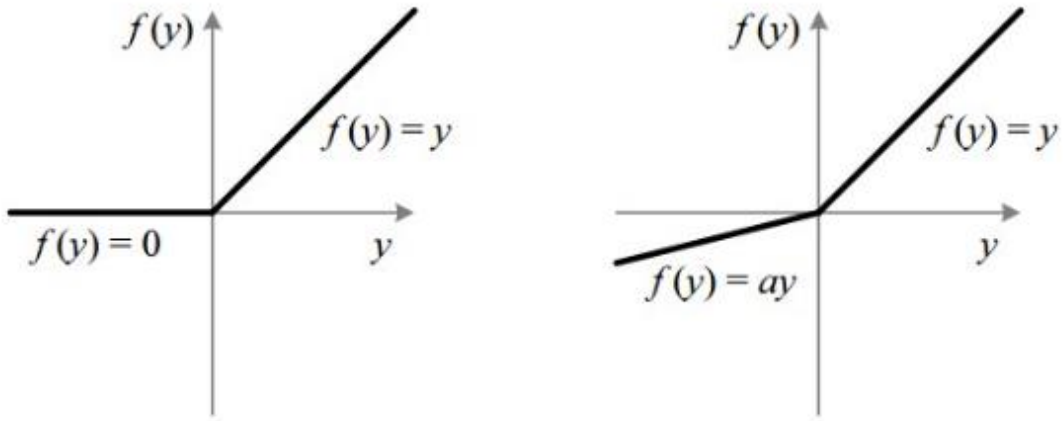


Figura 20: Función ReLU (izquierda) VS función ReLU rectificada (derecha)

Normalmente el valor de a suele ser 0.01 o un número cercano a este. Cuando se trata de un número distinto, la función se denomina Randomized ReLU.

Por tanto, el rango pasa a ser de (-∞, +∞).

Al igual que la función ReLU, se comporta bien con imágenes y además tiene un buen desempeño en redes convolucionales.

e. <u>Softmax</u>

Se utiliza cuando se quiere obtener una representación en forma de probabilidades, ya que el sumatorio de todas las probabilidades de las salidas es 1. Por tanto, resulta muy útil en tareas de normalización de tipos multiclase.

El rango utilizado es de [0,1].

$$f(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

5. Nodo de salida: valor asociado al valor de salida de la función de activación.
6. Bias (sesgo): es un parámetro crítico adicional que permite mover la función de activación hacia la derecha o la izquierda, dependiendo de ello el éxito del aprendizaje.

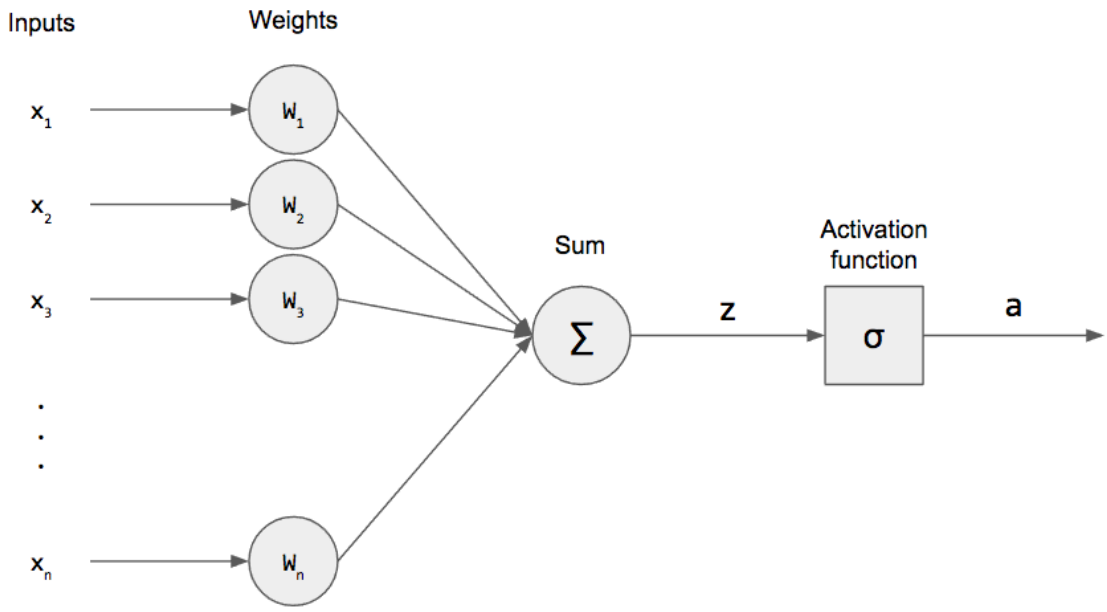


Figura 21: Diagrama de una red neuronal

Los perceptrones funcionan con valores umbral, que se utilizan con la finalidad de clasificar. Por ejemplo, una salida por encima del valor umbral indicaría que esa instancia pertenece a una clase, y una salida con un valor por debajo del umbral, significa que esa entrada forma parte de la otra clase.

Perceptrón Multicapa (MLP)

Los perceptrones de varias capas, también conocidos como "redes neuronales feed-forward", consisten en un conjunto de neuronas que se encuentran organizadas en distintas capas, normalmente dos o tres, pero no hay límite.

Dichas neuronas están conectadas entre ellas de forma que las salidas de una capa sirven como entrada de la siguiente. Usualmente, los perceptrones de varias capas están totalmente conectados (fully connected), lo que quiere decir que cada perceptrón de una capa específica se une a cada perceptrón de la capa siguiente.

La arquitectura más utilizada actualmente sería:

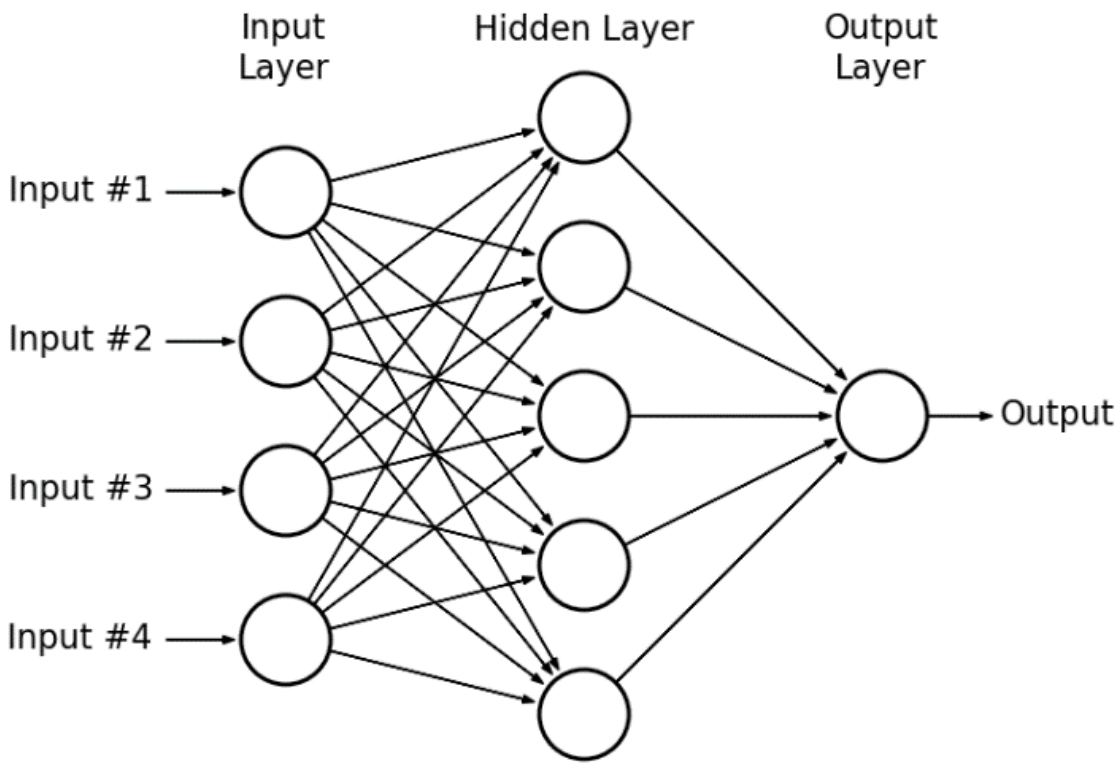


Figura 22: Capas en una red neuronal

1. Una primera capa de entrada, que recibe la información del exterior.
2. Una serie de capas ocultas (intermedias), las cuales son las encargadas de realizar el trabajo de la red.
3. Una capa de salida, que proporciona el resultado.

El número de capas y neuronas dependerá del tipo de aplicación a la cual se vaya a destinar la red neuronal.

2.6.2 Tipos de aprendizaje

Para poder aprender, las redes neuronales se sirven de un algoritmo de aprendizaje, formado por un conjunto de reglas que permiten a la red aprender a partir de los datos suministrados mediante la modificación de los pesos sinápticos de las conexiones entre neuronas y el umbral.

Los tipos de aprendizaje se dividen en tres:

- **Aprendizaje supervisado**

  Se introducen unos valores de entrada a la red, y los valores de salida que se generan se comparan con los valores de salida correctos. Si existe alguna diferencia, se ajusta la red en consecuencia.
- **Aprendizaje de refuerzo**

  Se introducen valores de entrada y lo único que se le indica a la red es si las salidas proporcionados por esta son correctas o no.
- **Aprendizaje no supervisado**

  Para este tipo de aprendizaje no hay ningún tipo de guía, así que la red lo único que hace es reconocer patrones en los datos de entrada y crear distintas categorías partiendo de estos patrones. De esta forma, cuando entra algún dato, tras el entrenamiento, la red podrá clasificarlo en una de las categorías generadas.

2.6.3 Redes Neuronales Convolucionales

Las Redes Neuronales Convolucionales (CNNs) son un tipo de Redes Neuronales muy efectivas en las tareas de reconocimiento y clasificación.

La diferencia respecto al resto, es que se asume que la entrada será una imagen, lo que permite codificar una serie de propiedades en la arquitectura.

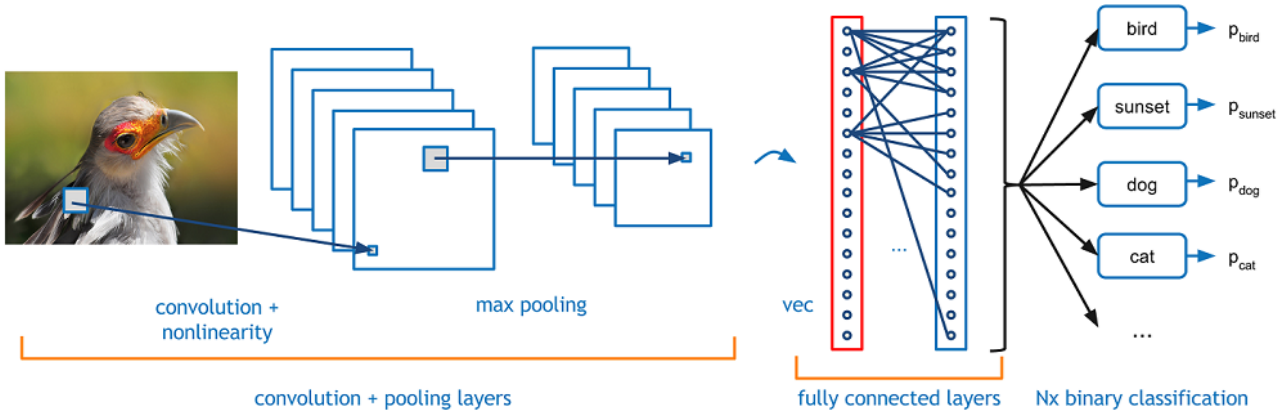


Figura 23: Red neuronal convolucional

Como se puede ver en la Figura 23, la red convolucional recibe una imagen de un pájaro, la cual debe clasificar en una de las categorías definidas (pájaro, atardecer, perro, gato…). El resultado es la asignación de una probabilidad a cada una de las distintas categorías. La suma de todas las probabilidades debería ser 1.

Las cuatro principales operaciones realizadas en una red convolucional son:

I. Convolución
II. Non Linearity (ReLU)
III. Pooling o submuestreo
IV. Clasificación (Fully Connected Layer)

i. Convolución

La finalidad de este paso es extraer características de la imagen de entrada (generando un mapa de características) utilizando un conjunto de filtros que se pueden aprender.

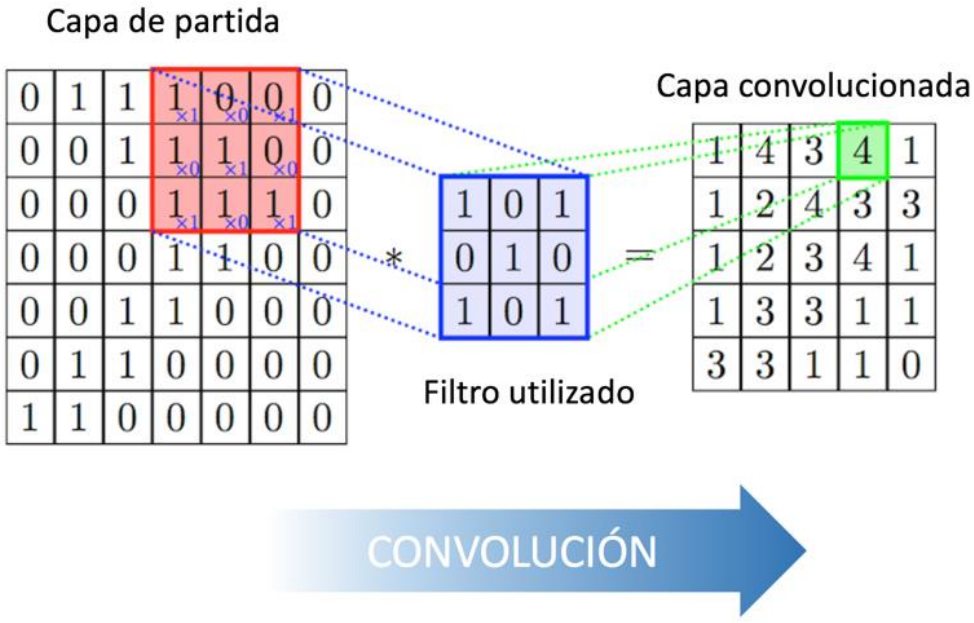


Figura 24: Proceso e convolución

En la siguiente imagen se pueden ver los distintos efectos de la convolución sobre una imagen tras aplicarle distintos filtr

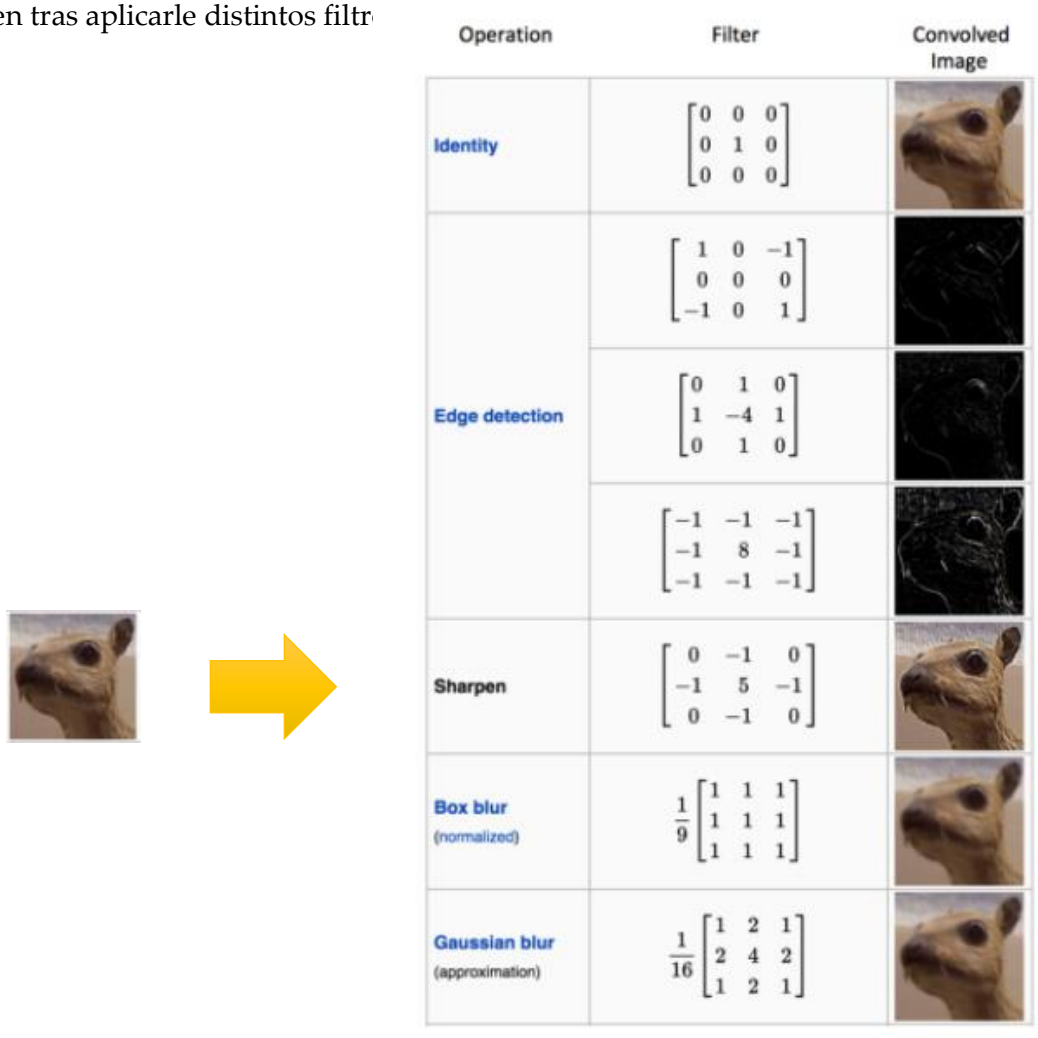


Figura 25: Imagen tras aplicarles distintos filtros en el proceso de convolución

Como ya hemos dicho, la red neuronal aprende los valores de los filtros a aplicar por si sola durante el proceso de entrenamiento, pero hay que especificarle una serie de parámetros como el número de filtros, el tamaño del filtro, la arquitectura de la red…

El Mapa de Características (Feature Map) se controla mediante tres parámetros:

- **Profundidad (Depth)**: corresponde al número de filtros utilizados en la operación de convolución. Por ejemplo, si aplicamos 3 filtros, se obtienen 3 mapas de profundidad distintos, que se guardan en una matriz, la cual tendrá una profundidad de 3.

- **Paso (Stride):** número de píxeles por los que movemos el filtro sobre la matriz de entrada. Cuanto mayor es el número de pasos, se producen mapas de características más pequeños.

- **Zero-padding**: este parámetro nos permite conservar el tamaño original de entrada. Se utiliza para añadir un borde de píxeles con valor de cero alrededor de la imagen de entrada.

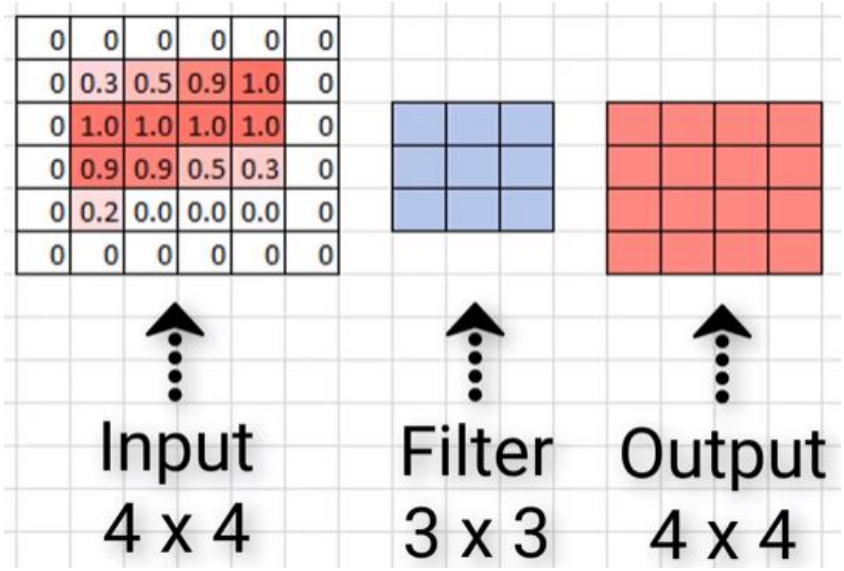


Figura 26: Zero-padding

Existen dos categorías para realizar el código a la hora de definir la capa convolucional:

Tipo	Descripción	Resultado
Valid	No padding	La dimensión se reduce
Same	Se añaden ceros a los bordes	La dimensión inicial se mantiene

ii. ReLu

La idea de esta operación es introducir la no linealidad en la red convolucional, ya que normalmente los datos que queremos que la red aprenda no son lineales.

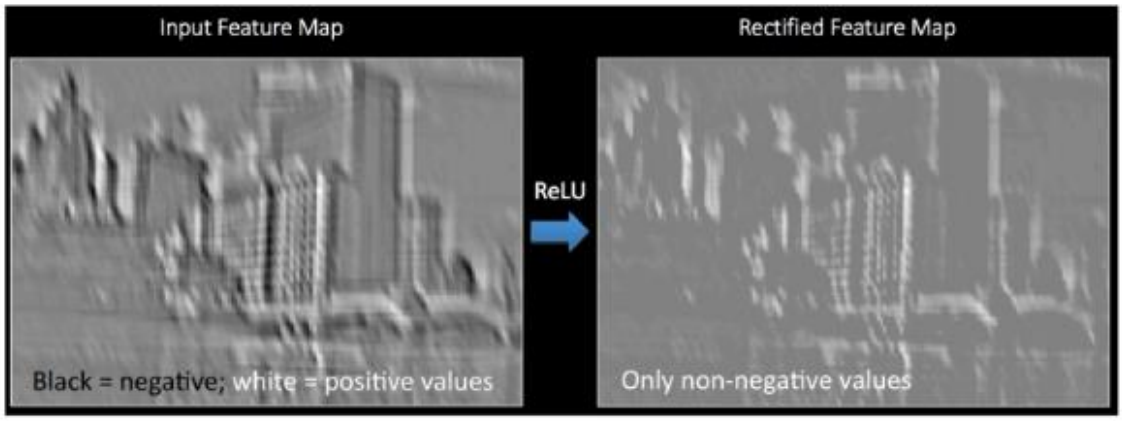


Figura 27: Imagen original a la izquierda y rectificada mediante la operación de ReLU a la derecha

Como se ha visto en los apartados anteriores, existen otras funciones no lineales que también se pueden utilizar en lugar de ReLU, pero en la mayoría de las pruebas esta es la que mejor funciona.

iii. Capa de reducción o pooling

Se suele colocar después de la capa convolucional y reduce la dimensión de cada mapa de características (sin afectar a la profundidad del mismo), manteniendo la información más relevante.

Ventajas:

- La disminución del tamaño conduce a una menor sobrecarga de cálculo para las siguientes capas de la red.
- Reduce el sobreajuste.
- Hace que la red sea invariante a pequeñas transformaciones, distorsiones y traslaciones de la imagen de entrada.

La capa de reducción puede ser de distintos tipos: *max-pooling*, a*verage-pooling*, *globlaMax-pooling*, *globalAverage-pooling*, *sum-pooling*…

La más utilizada es *max-pooling* ya que se ha demostrado que es la que mejor funciona. Este método divide la imagen de entrada en un conjunto de rectángulos de los cuales se queda con el máximo valor de cada uno.

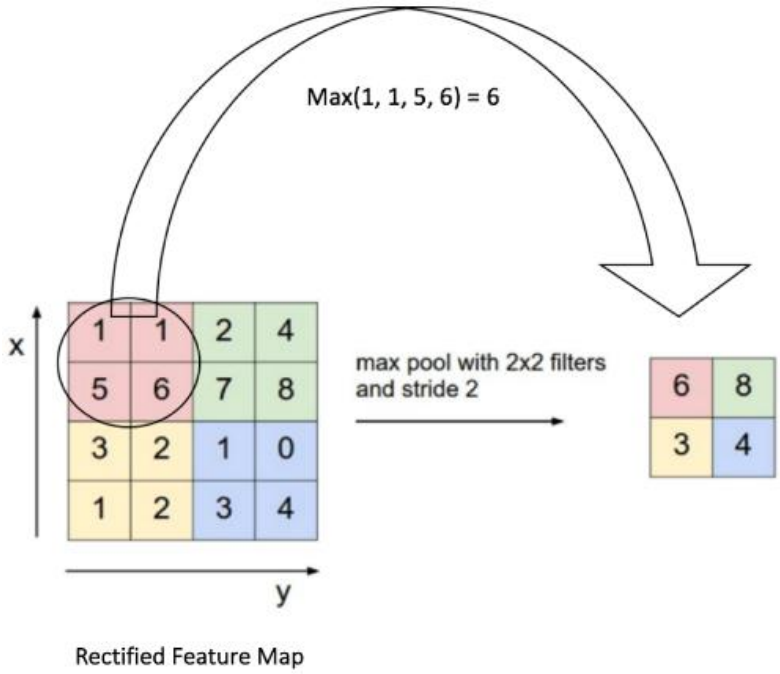


Figura 28: Pooling

En el caso de *average-pooling* sería el mismo procedimiento pero, en este caso, obteniendo la media de cada rectángulo.

4. *Capa clasificadora totalmente conectada (Fully Connected Layer)*

Antes de esta capa, se utiliza *Flatten,* que realiza un aplanamiento para convertir datos multidimensionales en un vector de características 1D para poder ser utilizado por esta la capa clasificadora densamente conectada.

Se trata de una capa típica de los perceptrones multicapa que utiliza la función de activación *softmax* en la capa de salida.

Como el nombre indica, cada neurona de la capa anterior se encuentra conectada a cada neurona de la siguiente. El número de neuronas de la capa anterior será igual al número de píxeles, y el número de neuronas de la capa Fully Connected tendrá el valor del número de clases que se deben predecir.

La salida de las capas convolucionales y de reducción genera un mapa de características de alto nivel de la imagen de entrada, que permite a la capa Fully Connected clasificarla en distintas clases tras entrenar la red asignando a casa clase una probabilidad.

La suma de todas las probabilidades es 1, hecho que se consigue al utilizar la función de activación *softmax*.

Además de las distintas capas mencionadas anteriormente, en muchas ocasiones se suele utilizar un *Dropout.* Se trata de una técnica de regularización con la finalidad de evitar el sobreajuste de la red neuronal, que consiste en no considerar todas las conexiones entre neuronas y capas.

Para ello, se seleccionan neuronas aleatoriamente en cada iteración, las cuales se ignorarán para el forwardpropagation y backwardpropagation.

A medida que la red neuronal aprende, los pesos de las neuronas se van ajustando y especializando, por lo tanto, si se dejan de actualizar los pesos de las neuronas desactivadas, el resto de las neuronas deberán intervenir para realizar las predicciones para las que faltan.

El resultado es que la red se vuelve menos sensible a los pesos específicos de las neuronas, se reduce la dependencia de las relaciones con las neuronas vecinas y es capaz de una

mayor generalización, haciendo menos probable la adaptación a los datos de entrenamiento.

El Dropout tiene un parámetro que oscila entre 0 a 1, siendo este la probabilidad de que las neuronas se queden activadas. Usualmente se suele emplear un valor de 0.5, que corresponde a una desactivación de la mitad de las neuronas.

Actualmente, está en auge el uso de las redes neuronales en el campo de la medicina, sobretodo en el análisis de datos e imágenes médicas. Podemos ver algunos ejemplos en el siguiente apartado.

2.7 Las Conferencias CLEF

La informática puede suponer un apoyo para la detección automática de pacientes que padecen tuberculosis.

En esta línea el CLEF (Conference and Labs of the Evaluation Forum) ha desarrollado diversas tareas dentro de este ámbito.

Se trata de una serie de campañas que se llevan realizando desde el año 2000, centrándose en la evaluación sistemática de información, a través de diversas tareas.

La mayoría de las tareas están relacionadas con la clasificación y anotación de imágenes (ImageCLEF).

2.7.1 ImageCLEF

ImageCLEF es el nombre que se le dan a las tareas que utilizan el tratamiento de imágenes.

Comenzaron a proponerse en 2003, y desde 2004 se añaden tareas médicas cada año.

2.7.2 ImageCLEF Tuberculosis

La primera vez que se propuso esta tarea fue en 2017, contando con la participación de 9 grupos.

Como hemos comentado anteriormente, podemos encontrar pacientes que han desarrollado resistencia a los medicamentos que tratan la tuberculosis (MDR), de forma que su recuperación es mucho más compleja y cara. Por esta razón, una detección temprana sería fundamental a la hora de lograr un tratamiento efectivo.

El principal problema es que los métodos utilizados en la detección de MDR son también muy caros y llevan varios meses, de forma que actualmente se precisa el desarrollo de un nuevo método que permita reducir costes y tiempo de procesamiento.

Con esta idea, ImageCLEF introdujo la tarea en la que se analizan imágenes médicas de pulmones de distintos pacientes.

2.7.3 Tareas edición 2019

a) Objetivos

En las ediciones anteriores, los participantes debían de evaluar la probabilidad de que un paciente con tuberculosis presentara resistencia a medicamentos (MDR subtask) y, por otro lado, clasificar automáticamente la tuberculosis del paciente en uno de los tipos mencionados anteriormente (TBT subtask). En ambos casos solo se podían utilizar las imágenes CT del pecho proporcionadas.

Por otro lado, el año pasado se introdujo una nueva subtarea donde, a diferencia de las otras dos, la mayoría de los participantes obtuvieron grandes resultados en la evaluación del grado de gravedad de la tuberculosis (SVR subtask). Por esta razón, esta tarea también se ha propuesto este año.

Tras dos ediciones se llegaron a las siguientes conclusiones:

- La tarea MDR era imposible basándose únicamente en imágenes.

- Para la tarea de TBT hubo cierta mejora en 2018 respecto a los resultados de 2017, pero no suficiente, teniendo en cuenta los datos extra proporcionados en esa segunda edición.
- La subtarea MDR no era factible y las subtareas de TBT y SVR pueden resolverse rápidamente por un radiólogo especializado. Por lo tanto, en la conferencia de 2019 se ha presentado una nueva subtarea que tendrá una mayor utilidad e impacto en las clínicas.

Finalmente, las tareas propuestas este año son las siguientes:

- Subtarea 1: Severity scoring (SVR)

  Como hemos explicado anteriormente, esta subtarea se obtener el grado de gravedad de la TB, utilizando únicamente las imágenes CT del pecho del paciente.

  La clasificación sería la misma que realizaría el personal médico basándose en diversos factores como las lesiones, los resultados de los tests microbiológicos, la duración del tratamiento… La finalidad es asignar al paciente una puntuación entre 1 ("crítico/muy malo") y 5 ("muy bueno"), para ello se realiza una clasificación binaria, de forma que las puntuaciones se agrupan en "HIGH" (puntuaciones de 1,2 y 3) y "LOW" (4 y 5).

- Subtarea 2: CT report (CTR)

  Los participantes tienen generar un informe automático basado en las imágenes proporcionadas, que debe incluir los siguientes datos en binario: pulmón izquierdo afectado, pulmón derecho afectado, presencia de calcificaciones, presencia de cavernas, pleuritis (inflamación de la pleura, membrana que recubre la cavidad torácica y pulmones) y disminución de la capacidad del pulmón.

b) Corpus de evaluación

Para esta edición, las dos subtareas usan el mismo conjunto de datos, que consiste en 335 escáneres de pecho de los pacientes con tuberculosis y un set de información médica relevante para cada uno de ellos. De estos 335 escáneres, 218 se utilizarán para entrenar y 117 para las pruebas.

Dicha información médica incluye los siguientes datos en binario: discapacidad, recaída, síntomas, comorbilidad, bacilaridad, farmacorresistencia, educación superior, ex recluso, alcohólico, tabaquismo y gravedad.

Para todos los pacientes de presentan imágenes 3D con un tamaño de 512x512 píxeles con un número variable de rebanas (entre 50 y 400) y están guardadas con el formato NIfTI.

c) Método de evaluación

- Subtarea 1: Severity scoring (SVR)

  Se evalúa como un problema de clasificación binaria, tomando como medidas el área bajo la curva ROC (AUC) y precisión (accuracy).

  A la hora de obtener el ranking, se prioriza el AUC frente a la precisión.

- Subtarea 2: CT report (CTR)

  En este caso, se trata de un problema de clasificación multi binaria (6 resultados). Al igual que en la subtarea anterior, las medidas de evaluación incluyen AUC y accuracy.

  El ranking también se basa primero en la media de AUC obtenida y el AUC mínimo obtenido (ambos criterios aplicados sobre los 6 resultados).

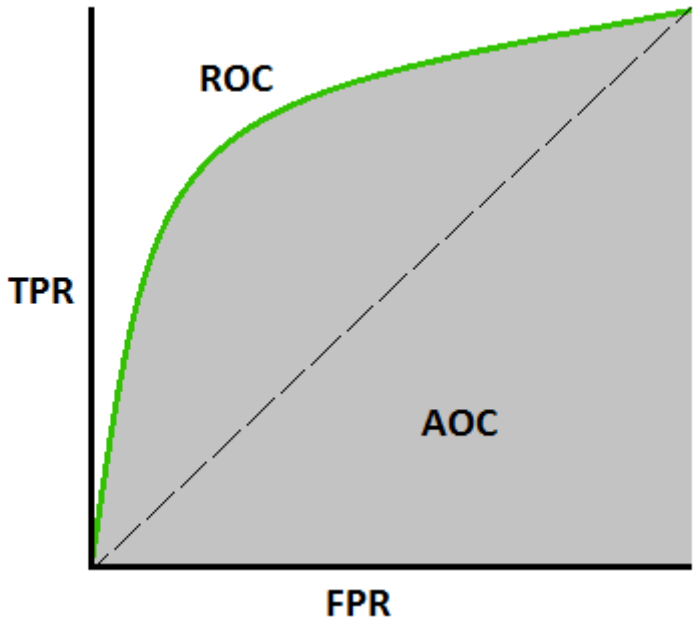


Figura 29: Gráfica que representa el área bajo la curva ROC (AUC)

AUC

AUC indica cómo de capaz es el modelo de distinguir entre clases. Por tanto, un mayor valor de AUC quiere decir que el modelo utilizado es mejor prediciendo los 0s como 0s y los 1s como 1s.

ROC es una curva de probabilidad que se representa en función de VPR y FPR (relación entre verdaderos positivos y falsos positivos).

La diagonal divide el espacio ROC, de forma que los puntos que se encuentran por encima de esta se consideran buenos resultados de clasificación, y los puntos por debajo, resultados peores que el azar.

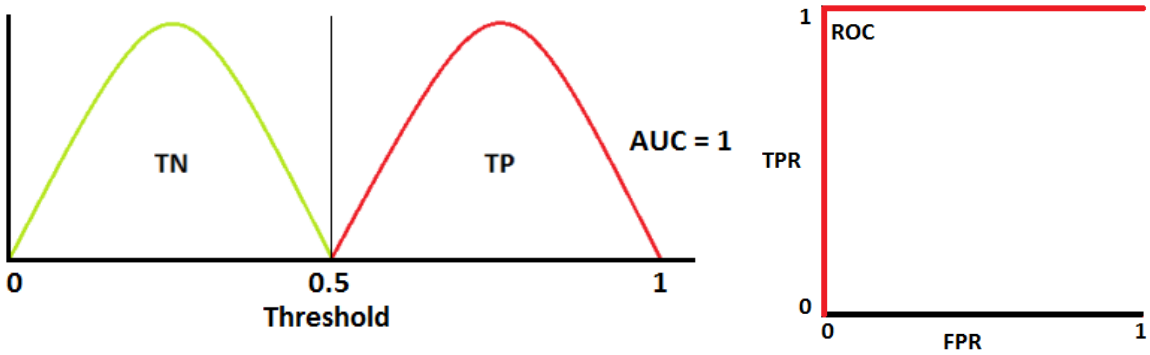


Figura 30: Representación de los verdaderos positivos y verdaderos negativos (derecha) y curva ROC (izquierda) cuando AUC = 1

Esta sería la situación ideal (AUC = 1) ya que el modelo es capaz de distinguir entre las dos posibilidades, obteniendo un diagnóstico perfecto, donde se obtienen todos los valores como verdaderos positivos y verdaderos negativos.

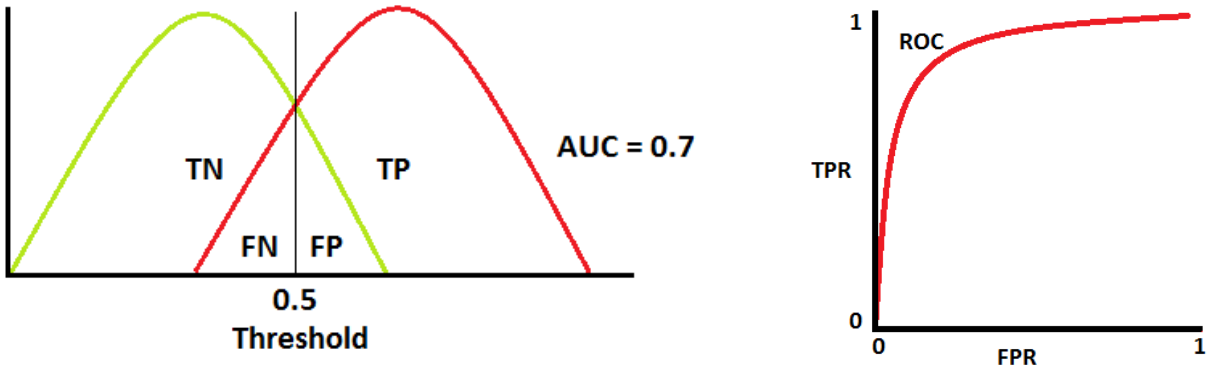


Figura 31: Representación de los verdaderos positivos y verdaderos negativos (derecha) y curva ROC (izquierda) cuando AUC = 0.7

En este caso, las dos distribuciones se solapan, dando lugar a falsos negativos y falsos positivos. Un valor de 0.7 implica que el modelo será capaz de realizar la diferenciación entre las dos clases con un 70% de acierto, considerándolo apto para el diagnóstico.

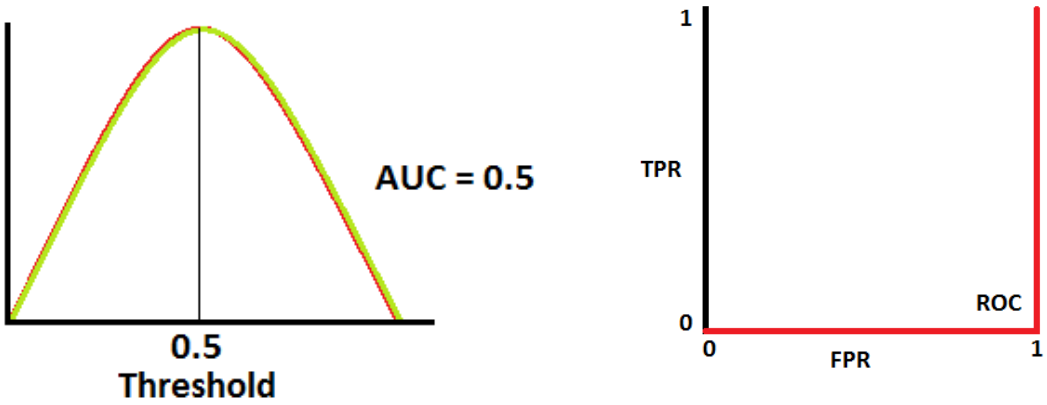


Figura 32: Representación de los verdaderos positivos y verdaderos negativos (derecha) y curva ROC (izquierda) cuando AUC = 0.5

Un valor del 0.5 da lugar a la peor situación, ya que quiere decir que el modelo no tiene capacidad de clasificación.

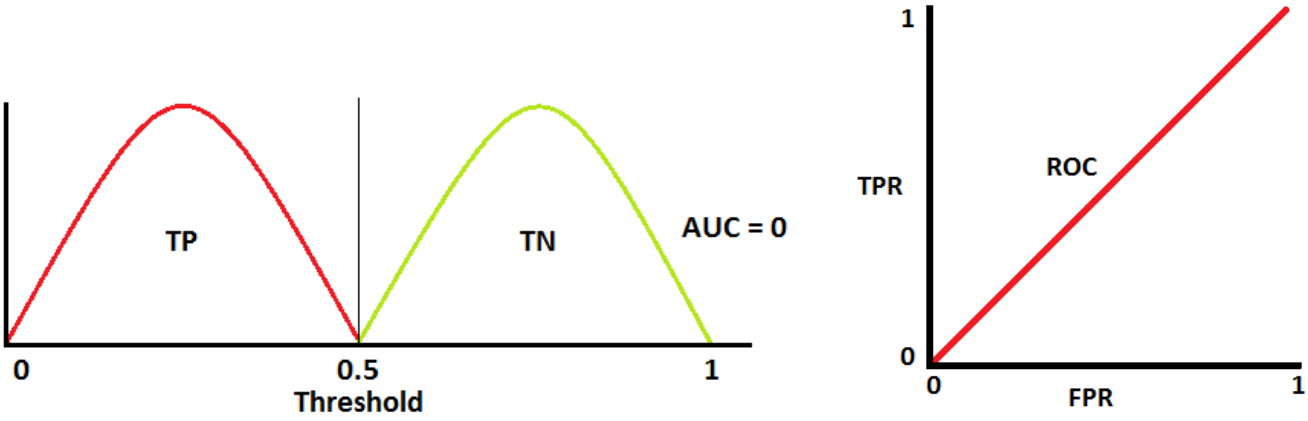


Figura 33: Representación de los verdaderos positivos y verdaderos negativos (derecha) y curva ROC (izquierda) cuando AUC = 0

Cuando AUC alcanza un valor de 0 implica que el modelo está intercambiando las clases de forma que realiza la predicción al revés.

VPR (Razón de Verdaderos Positivos) / Sensibilidad

Mide cómo el modelo es capaz de clasificar los casos positivos correctamente calculando la proporción de verdaderos positivos:

$$VPR = \frac{VP}{P} = \frac{VP}{(VP + FN)}$$

Donde:

- VP = Verdaderos Positivos / Éxitos
- FN = Falsos Negativos

SPC (Especificidad) /Razón de Verdaderos Negativos

Obtiene cuántos resultados positivos realmente no los son, teniendo en cuenta todos los casos negativos:

$$SPC = \frac{VN}{N} = \frac{VN}{(FP + VN)}$$

Donde:

- VN = Verdaderos negativos
- FP = Falsos Positivos /Error tipo I

FPR (Razón de Falsos Positivos)

$$FPR = 1 - SP = \frac{FP}{(FP + FN)}$$

Donde:

- FN = Falsos Negativos /Error tipo II

Accuracy

Se calcula con la finalidad de obtener la habilidad el modelo para diferenciar los casos positivos de los negativos correctamente.

ACC (Accuracy) / Exactitud

Para ello se calcula la proporción de verdaderos positivos y verdaderos negativos entre todos los casos:

$$ACC = \frac{VP + VN}{P + N} = \frac{VP + VN}{VP + FN + FP + VN}$$

Donde:

- P (condición positiva) = número de positivos reales (VP + FN)
- N (condición negativa)= número de negativos reales (FP + VN)

Ambas medidas tienen como objetivo principal valorar la eficacia del sistema en cuestión, pero divergen a la hora de calcularlas, ya que en el caso de la exactitud (ACC) se valora el número de aciertos respecto al total y para AUC se tienen en cuenta los falsos positivos y los falsos negativos también.

Dichas diferencias podemos ver con el siguiente ejemplo:

Contamos con 100 pacientes diferentes, de los cuales 50 padecerían tuberculosis y 50 no.

CASO 1

El resultado obtenido es que todos los pacientes que sufren la enfermedad dan positivo en TB y los que no, dan un resultado negativo.

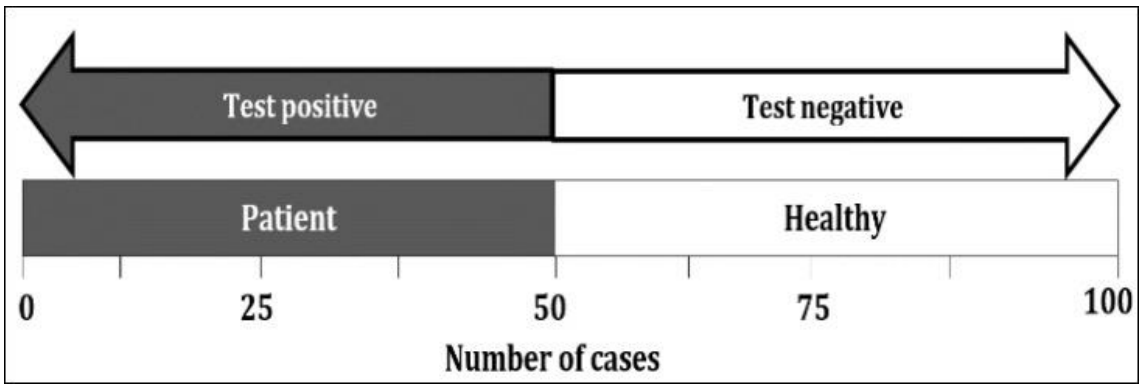


Figura 34:

En este caso obtendríamos:

- 100% en Exactitud (Accuracy)
- 100 % en Sensibilidad (VPR)
- 100% en Especificidad (SPC)

CASO 2

El modelo detecta 25 de los 50 con tuberculosis como positivos, y los otros 25 como negativos.

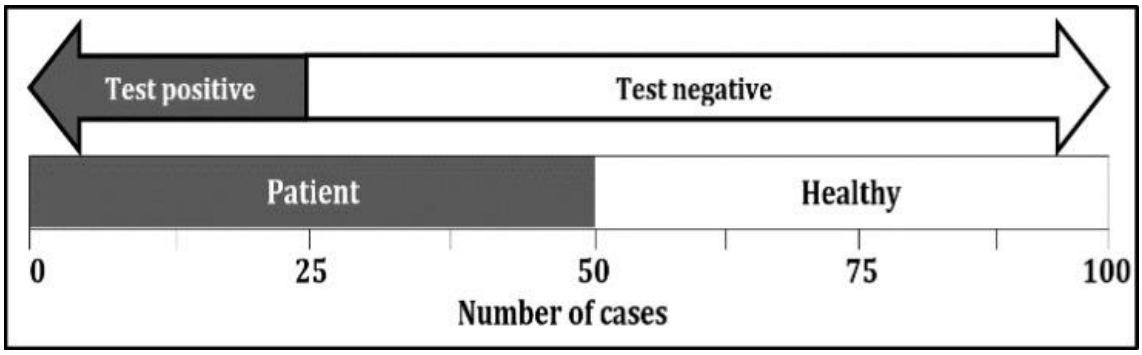


Figura 35

Exactitud

De los 100 casos disponibles, se han determinado 25 enfermos y 50 no enfermos correctamente. Aplicando la fórmula expuesta anteriormente obtendríamos un valor de 75 sobre 100.

Sensibilidad

Contábamos con 50 enfermos, y el sistema sólo ha podido reconocer 25, obteniendo un valor de 25 sobre 50.

Especificidad

De los 100 pacientes, 50 no tenían enfermedad y se han reconocido todos ellos, lo que equivale a un 50 de 50.

Por tanto, los resultados serían los siguientes:

- 75 % en Exactitud
- 50% en Sensibilidad
- 50% en Especificidad

Teniendo en cuenta estos tres parámetros se puede concluir que este modelo sería eficiente a la hora de confirmar la enfermedad, pero no es adecuado para la detección de la misma.

CASO 3

Por último, supongamos que de entre 50 pacientes que no sufren enfermedad, se diagnostican solo 25, y el resto se clasifican como enfermos.

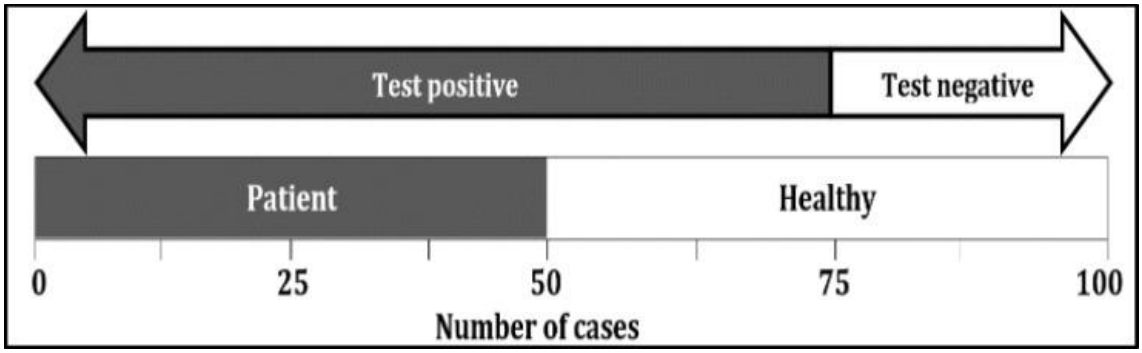


Figura 36

El valor de la exactitud sería la misma que en el caso anterior, pero la sensibilidad y la especificidad serían totalmente contrarias a las vistas en dicho caso.

Sensibilidad

Contábamos con 50 enfermos y todos ellos han sido diagnosticados como tal (50 de 50)

Especificidad

Se han detectado 25 pacientes como personas que no padecen tuberculosis cuando en realidad eran 50 (25 de 50)

Por lo tanto:

- 75 % en Exactitud
- 50% en Sensibilidad
- 50% en Especificidad

Al contrario que en el caso 2, este modelo podría ser útil para la detección de la TB pero no para la confirmar si un paciente padece o no dicha enfermedad.

2.7.4 Modelo para la detección de la tuberculosis

En 2018, 11 grupos de 9 países distintos fueron los que participaron en la edición de ese mismo año.

Todos ellos abordaron las tres tareas propuestas con distintas metodologías, las cuales analizaremos a continuación más detenidamente.

DETECCIÓN DE MDR

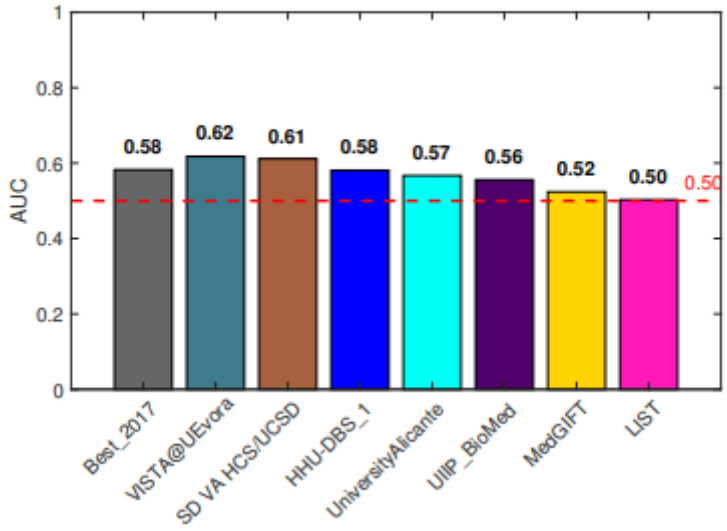


Figura 37: Resultados obtenidos en la tarea de la detección de la tuberculosis multirresistente

Para esta primera subtarea se obtuvieron los siguientes resultados:

En el gráfico de la derecha está representada el área bajo la curva ROC (AUC) con los mejores resultados obtenidos, donde "Best_2017" representa la mejor puntuación obtenida en la edición de 2017.

Teniendo en cuenta que la línea roja representa una clasificación aleatoria (AUC = 0.5), podemos comprobar que las metodologías empleadas no consiguen unos resultados más prometedores que lanzar una moneda al aire.

Por tanto, la detección de la tuberculosis resistente a medicamentos basada en el uso de imágenes no ofrece una solución que pueda ser de utilidad en la práctica clínica.

El mejor resultado obtenido bajo el criterio de AUC es de 0.6178, alcanzado por el equipo VISTA@UEvora.

Dicho equipo utilizó un modelo basado en modelado 3D y extracción de patrones de textura (Ahmed, Obaidullah, Jayatilake, Gonçalves, & Rato, 2018).

El esquema seguido es el siguiente:

1. Los datos de entrada se pre procesan de forma que los datos 2D se convierten en 3D y se seleccionan las zonas de interés del pulmón
2. Se extraen los bronquios utilizando un valor umbral predefinido
3. Se realiza un análisis de textura sobre las partes elegidas
4. Se computa un vector de características al que se le aplican múltiples clasificadores para efectuar el análisis y posterior clasificación

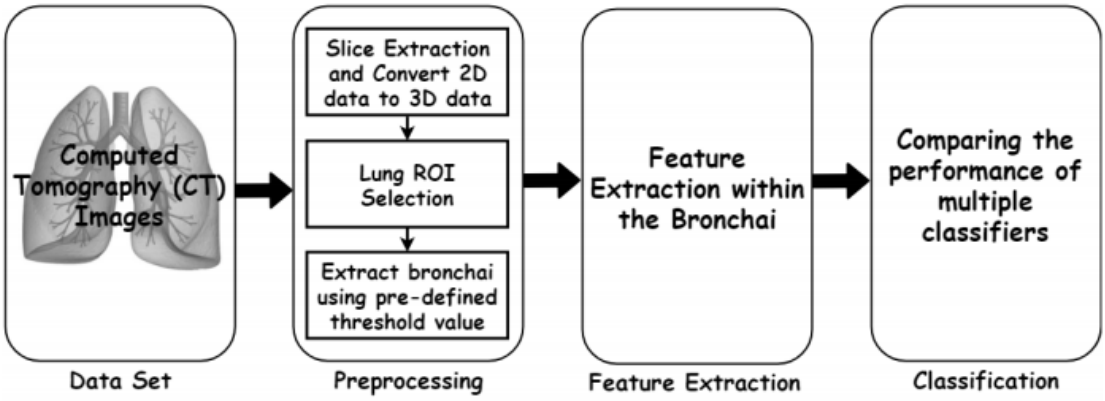


Figura 38: Proceso seguido por el grupo VISTA@UEvora para superar la tarea de la detección de la tuberculosis multirresistente

Por otro lado, si nos basamos en ACC, tenemos al equipo San Diego VA HCS/UCSD como grupo con mejor resultado (0.6144), el cual utilizó una red neuronal convolucional (CNN) para hacer frente a las tareas propuestas(Gentili, 2018).

Podemos dividir su trabajo en dos partes:

1. Preprocesamiento de las imágenes

   Las imágenes proporcionadas estaban en formato NIfTI, por lo tanto, se utilizó una función de Phyton3 (med2image) para convertirlas a png y jpg y poder trabajar con ellas.

   Se reconstruyeron utilizando un plano coronal ya que se observó que de esta forma se abarca más

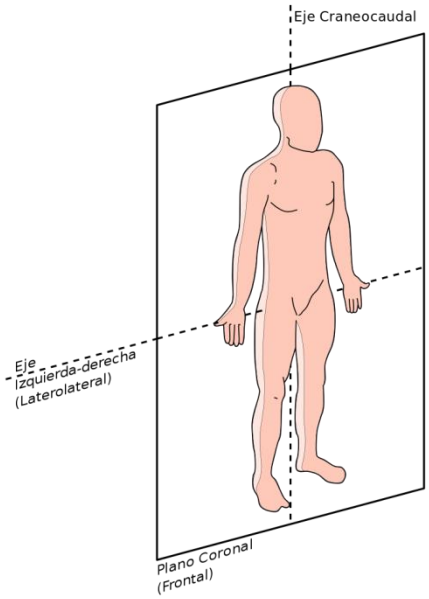


Figura 39: Plano coronal

área del pulmón y se podían detectar las zonas anormales del mismo con más facilidad.

2. Entrenamiento de la red neuronal

La red neuronal elegida (ResNet 50) fue entrenada con imágenes de tamaño 64x64 obteniendo el siguiente diagrama, donde podemos observar la tasa de aprendizaje frente a la pérdida.

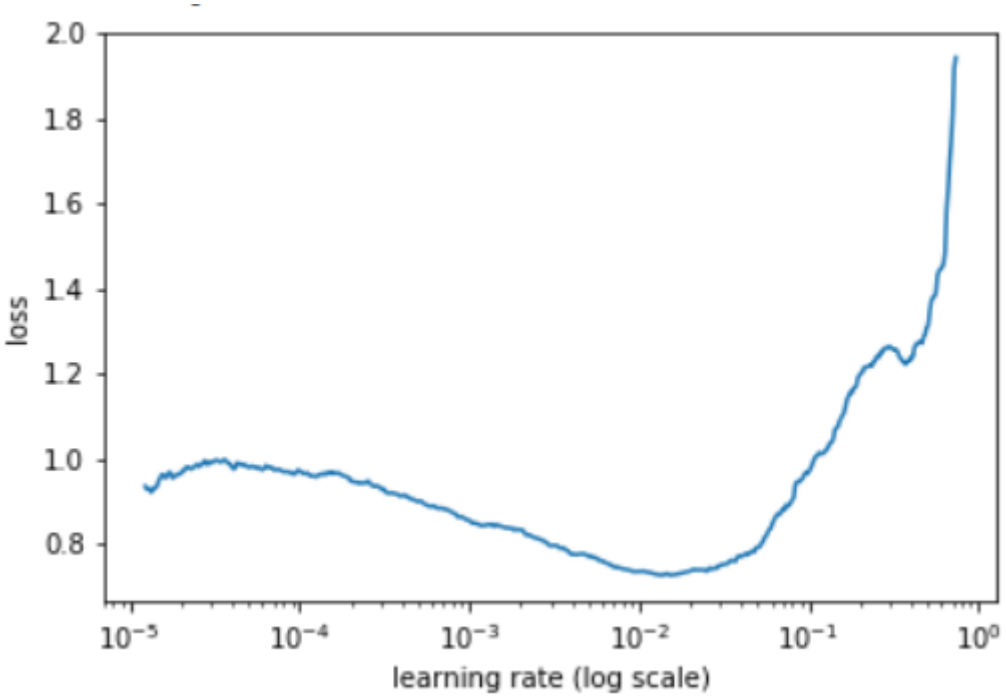


Figura 40: Pérdida en función de la tasa de aprendizaje

Basándose en el gráfico anterior, se eligió una tasa de aprendizaje del 0.002 para las últimas capas, que fueron entrenadas durante dos épocas sin aumento de datos y, después, dos veces más aumentado los datos utilizando diversos recursos como rotaciones, cambios aleatorios en la intensidad y voltear la imagen horizontalmente.

Por otro lado, las capas intermedias fueron entrenadas con una tasa de aprendizaje de un tercio de las últimas capas y las primeras, con un noveno del valor de las últimas.

Debido a que las imágenes se analizaron por separado, se obtenían 200 resultados diferentes para cada paciente, haciendo que disminuyera la probabilidad de detectar la tuberculosis MDR.

Para solucionar este problema, se utilizó un Excel para escalar el valor obtenido ya que se conocía el número de casos positivos y negativos que se habían proporcionado.

CLASIFICACIÓN DEL TIPO DE TUBERCULOSIS

Para esta segunda tarea, en lugar de utilizar AUC como método de evaluación, se empleó el coeficiente Cohen Kappa para ajustar el efecto del azar, además de ACC.

En la siguiente gráfica se ha representado la tasa de verdaderos positivos para cada uno de los cinco tipos de tuberculosis a determinar, donde la línea roja representa los resultados esperados si se aplica una clasificación aleatoria de cinco elementos.

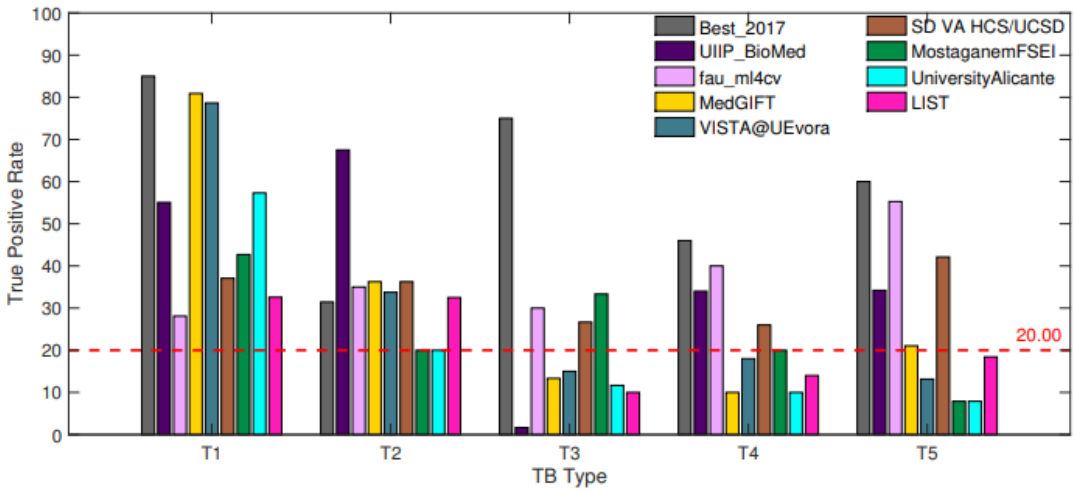


Figura 41: Resultados obtenidos en la tarea de clasificación en los distintos tipos de tuberculosis en base a los verdaderos positivos

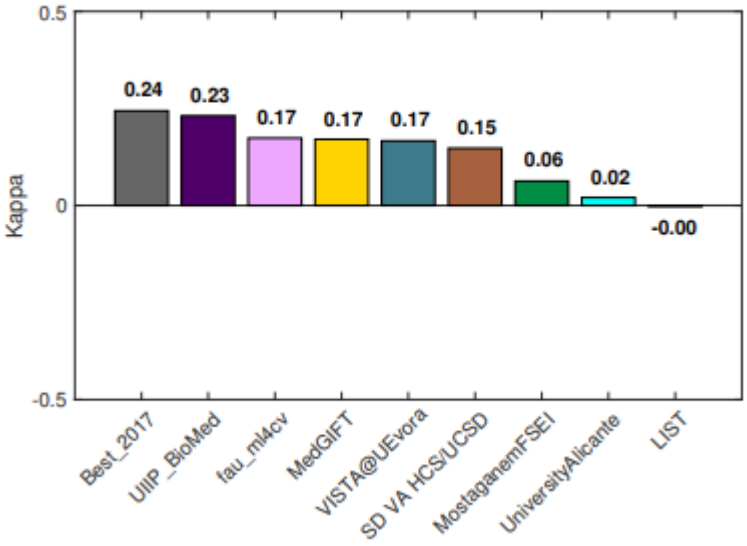


Figura 42: Resultados obtenidos en la tarea de clasificación en los distintos tipos de tuberculosis utilizando el coeficiente Kappa

El mejor resultado conseguido en este caso fue obtenido por el grupo UIIP_BioMed, tanto basándose en el coeficiente Cohen Kappa (0.23) como en la exactitud (0.4227).

Al igual que el grupo mencionado anteriormente, este equipo utilizó una red convolucional para realizar la clasificación en los distintos tipos de tuberculosis.

El procedimiento a seguir fue el siguiente (Liauchuk, Tarasau, Snezhko, & Kovalev, 2018):

1. Preparación de las imágenes

   Las lesiones producidas por la tuberculosis fueron etiquetadas manualmente en 198 escáneres CT 3D en dos etapas diferentes. Primeramente, gracias a la ayuda de radiólogos especializados para realizar una localización general y, posteriormente, haciendo una segmentación más precisa de las lesiones.

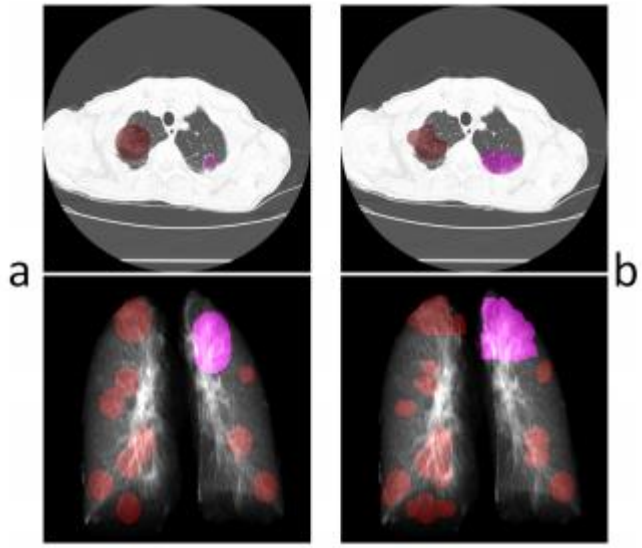


Figura 43: En la parte superior se muestran imágenes con un corte axial y en la inferior, un plano frontal de los pulmones.

Para ello se hizo uso de una herramienta de software diseñada por los propios autores, el cual permite un etiquetado de hasta 10 tipos de lesiones de tuberculosis diferentes.

Por un lado, las dos imágenes de la derecha (a) son el resultado tras la primera etapa, y las dos de la izquierda (b) cómo queda el pulmón tras la segunda etapa de etiquetado más preciso.

Las regiones más allá del pulmón se etiquetaron como “no importa” (color gris) para que la red neuronal omitiera dichas zonas a la hora de entrenar y validar, así se presta atención a aquellas zonas que sean de real interés.

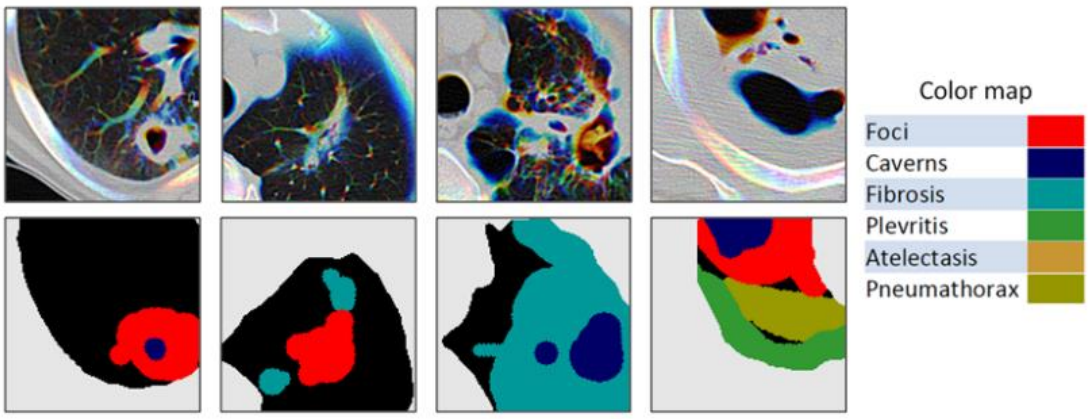


Figura 44: Asignación de diferentes colores en función del tipo de tuberculosis

2. Segmentación de las regiones del pulmón

   Se realizaron proyecciones a lo largo de los ejes X, Y, Z, calculadas para cada referencia del escáner de pecho. Las tres proyecciones normalizadas se concatenan formando un descriptor cuantitativo de la imagen de referencia.
   Se seleccionan las 5 imágenes más similares basándose en dicho descriptor obtenido.

3. Entrenamiento de la red neuronal

   Las imágenes utilizadas para entrenar la red neuronal se redimensionaron a 128x128. A continuación, se cogieron tres rebanadas vecinas para formar una imagen RGB de forma que se pueda utilizar la información espacial a lo largo del eje Z.
   Finalmente, utilizando una interpolación cúbica, se dimensionaron las imágenes de nuevo a 256x256 para mejorar la detección de las pequeñas lesiones mediante la red neuronal AlexNet.

De los 198 escáneres etiquetados previamente, 149 se utilizaron con algoritmos de entrenamiento y los 49 restantes para la validación.

Para mejorar el ratio de convergencia y la exactitud, se hizo uso del modelo ILSVRC2012-trained, inicializando los pesos de la red, la cual fue programada para detectar diversos tipos de lesión al mismo tiempo.

Los parámetros con los que se obtuvieron mejores resultados fueron los siguientes:

- Número de épocas = 60
- Función de activación = ReLu
- Batch size = 64
- Solver type = SGD Caffe Solver
- Tasa de aprendizaje para las 20 primeras épocas = 0.001
- Tasa de aprendizaje para las 20 siguientes = 0.0001
- Tasa de aprendizaje para las 20 últimas = 0.00001

4. Obtención de los mapas de probabilidad

Con la idea de reducir el número de detecciones falsas se generaron mapas de probabilidad en los cuales, cualquier probabilidad por debajo de un umbral permitido, en este caso 0.5, se valoraban como 0.

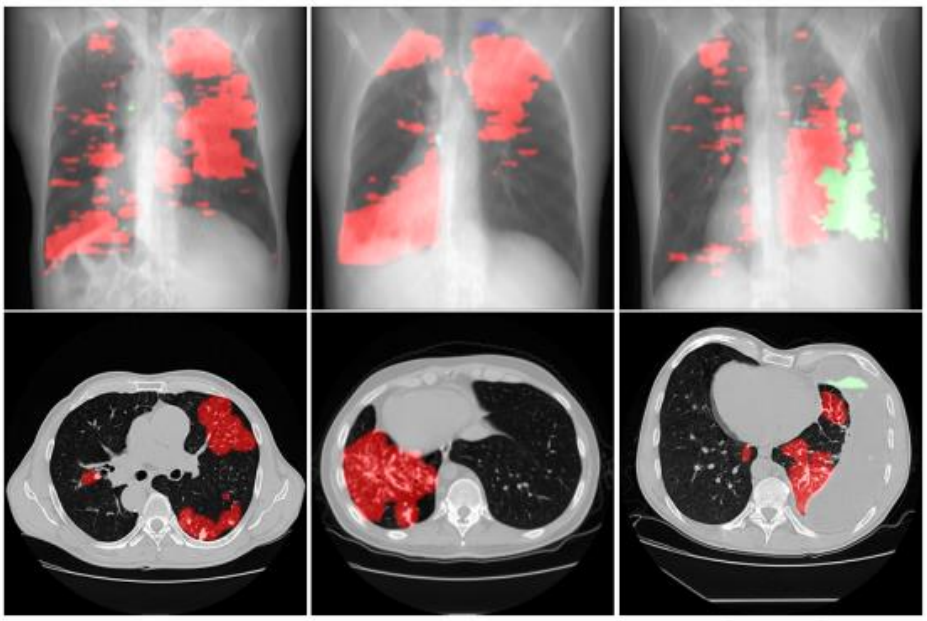


Figura 45: CT de tórax desde diferentes puntos de vista con los puntos de interés coloreados en rojo

5. Construcción del TB-descriptor

Una vez obtenido el mapa de probabilidad, se construye un descriptor (TB-descriptor).

Para cada tipo de lesión, la presencia en cada una de las 6 partes en las que se divide el pulmón (imagen de abajo) se calcula como la suma de las probabilidades en los vóxels correspondientes dividido por el número de vóxels del pulmón correspondiente a dicha parte.

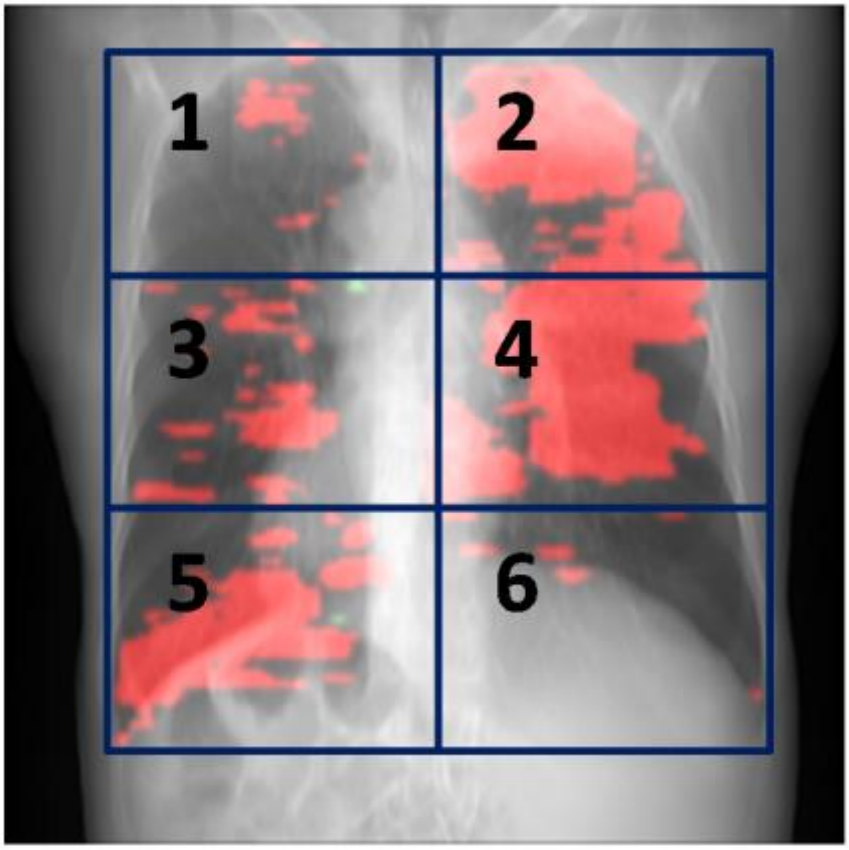


Figura 46: División del pulmón en 6 zonas diferentes para la posterior asignación de un tipo de tuberculosis asociada a cada una de ellas

Por lo tanto, este descriptor nos indica la presencia de un tipo concreto de lesión en cada una de las partes, asignándosele una puntuación entre 0 y 1 para cada una de ellas.

PUNTUACIÓN DE SEVERIDAD

A diferencia de las dos subtareas anteriores, los resultados en esta fueron notablemente mejores alcanzado valor mínimo de RMSE (Raíz del error cuadrático) de 0.7840, obtenido por el grupo UIIP_BioMed, al igual que en la tarea 2.

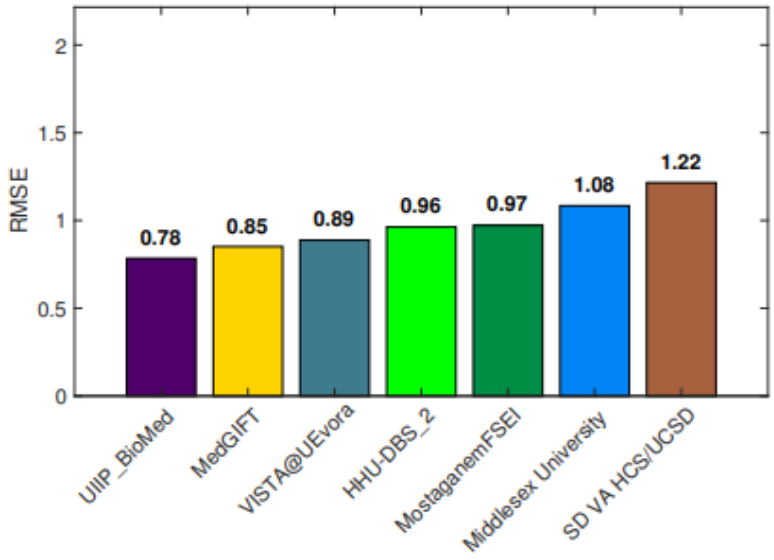


Figura 47: Resultados obtenidos en la tarea de puntuación de severidad de la tuberculosis presente en el paciente

Por otro lado, valorando el máximo valor AUC obtenido (0.7708), encontramos al equipo MedGIFT.

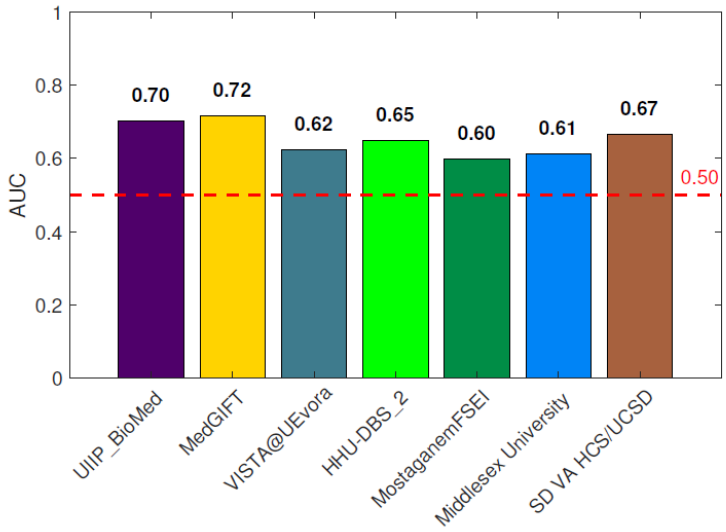


Figura 48: Resultados obtenidos en la tarea de puntuación de severidad de la tuberculosis presente en el paciente en función del AUC

El método utilizado consistía en la creación de un modelo gráfico de los pulmones, basando los nodos del mismo en atlas geométricos y aristas ponderadas, codificando las diferencias entre los descriptores de textura 3D de cada región.

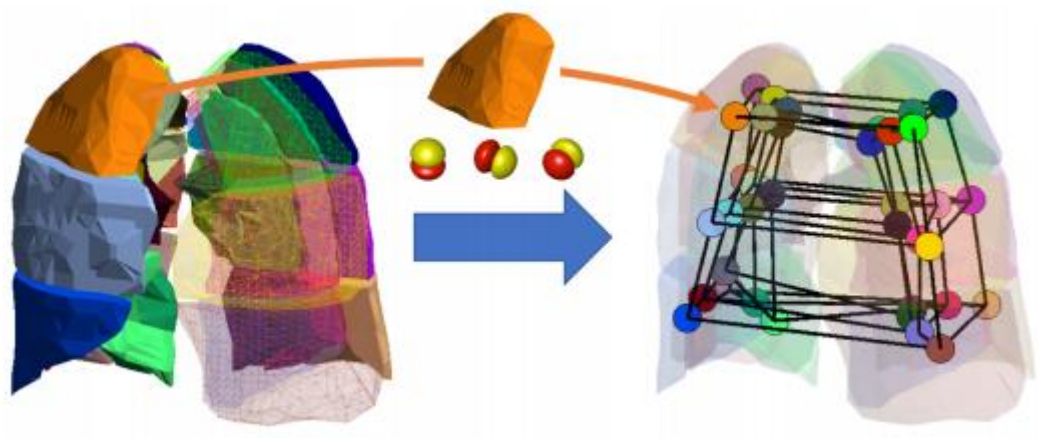


Figura 49: Modelo de los pulmones obtenido a partir de los descriptores extraídos de estos

El procedimiento a seguir fue el siguiente (Cid & Müller, 2018):

1. Preprocesamiento

   El primer paso fue la obtención de imágenes 3D y las máscaras del pulmón, utilizando un tamaño de vóxel de 1mm evitando, así, la pérdida de información.

2. Atlas del pulmón

   Para la obtención de la estructura de la anatomía del pulmón se optó por una división geométrica con un número fijo de 36 regiones utilizando únicamente las máscaras de los pulmones (Figura 49).

3. Características de textura en 3D

   La primera prueba resultó en el cálculo de un histograma de gradientes basado en la transformada de Fourier HOG. Se utilizaron 28 direcciones 3D, obteniendo un vector de características de 28 dimensiones para cada vóxel.

También se utilizó la transformada 3D de Rieesz-wavelet, con la cual se consiguieron mejores resultados. En este caso el vector de características tenía 40 dimensiones para cada vóxel, las cuales se redujeron a 10 dimensiones.

4. Modelo gráfico del pulmón basado en la textura

   Se utilizó un grafo ponderado no dirigido que contaba con 36 nodos y 84 aristas para modelar el pulmón.

   Cada nodo corresponde a una de las 36 regiones en las que se había dividido previamente. Las aristas se colocaron en función de la adyacencia definida por el atlas, de forma que se añadía una entre dos nodos si las dos regiones donde se situaban dichos nodos eran vecinas. Además de éstas, se añadieron 18 aristas adicionales, conectado cada par de nodos que representaban regiones opuestas dentro del atlas respecto a la división derecha/izquierda de los pulmones (Figura 49).

   Los pesos asignados a las distintas aristas se definieron utilizando la distancia Euclídea y de correlación, dando como resultado grafos diferentes en función de la medida usada.

5. Descriptor del paciente basado en el descriptor

   El descriptor de características de cada paciente se definía en función de los pesos del grafo. Como se generaron 84 aristas, haciendo un total de 84 pesos, el descriptor era de 84 dimensiones.

   Los vectores se normalizaron utilizando una normalización Z-score, basada en los pacientes del set de entrenamiento.

6. Clasificación

   Se usó una máquina de soporte vectorial (SVM) con la ayuda de RBF kernel.

Para esta tarea en concreto, se consideró la puntuación de severidad como una clase en el algoritmo de SVM, asignando la clase más probable ("high" con las puntuaciones de1 a 3) como la puntuación predicha.

Por tanto, en este caso, para cada paciente, se obtenía la probabilidad de pertenecer a la clase "high" añadiendo las probabilidades SVM de pertenecer a las clases 1-3.

2.8 Conclusiones

Sin duda alguna, todas estas técnicas van a suponer un avance notable en el desarrollo de la lucha contra las enfermedades. No obstante como se ha visto en los análisis de algunos casos todavía queda un largo camino por recorrer ya que los resultados no son definitivos y hay margen para conseguir que sean realmente relevantes.

Tras el análisis de las metodologías propuestas para las tareas a resolver en las conferencias CLEF se puede comprobar que el uso del Deep Learning está muy extendido. En cambio, el análisis del movimiento mediante el flujo óptico es innovador en este campo y podría aportar grandes resultados.

3. METODOLOGÍAS PROPUESTAS

3.1. Redes Neuronales

3.1.1 Preparación de los datos

Partimos de un conjunto de 218 pacientes que se utilizan para realizar el entrenamiento. Cada uno tiene un número variable de rebanadas de entre 50 y 400 que forman el escáner de pecho.

Las imágenes se encuentran en formato NIfTI (Neuroimaging Informatics Technology Initiative), por ello, he empleado la biblioteca de Matlab que permite cargar, salvar, ver y analizar este tipo de formato.

```
niiImage= sprintf ('CTR_TRN_%03d.nii',j);
S=  load_nii (niiImage);1
A = S.img;2
```

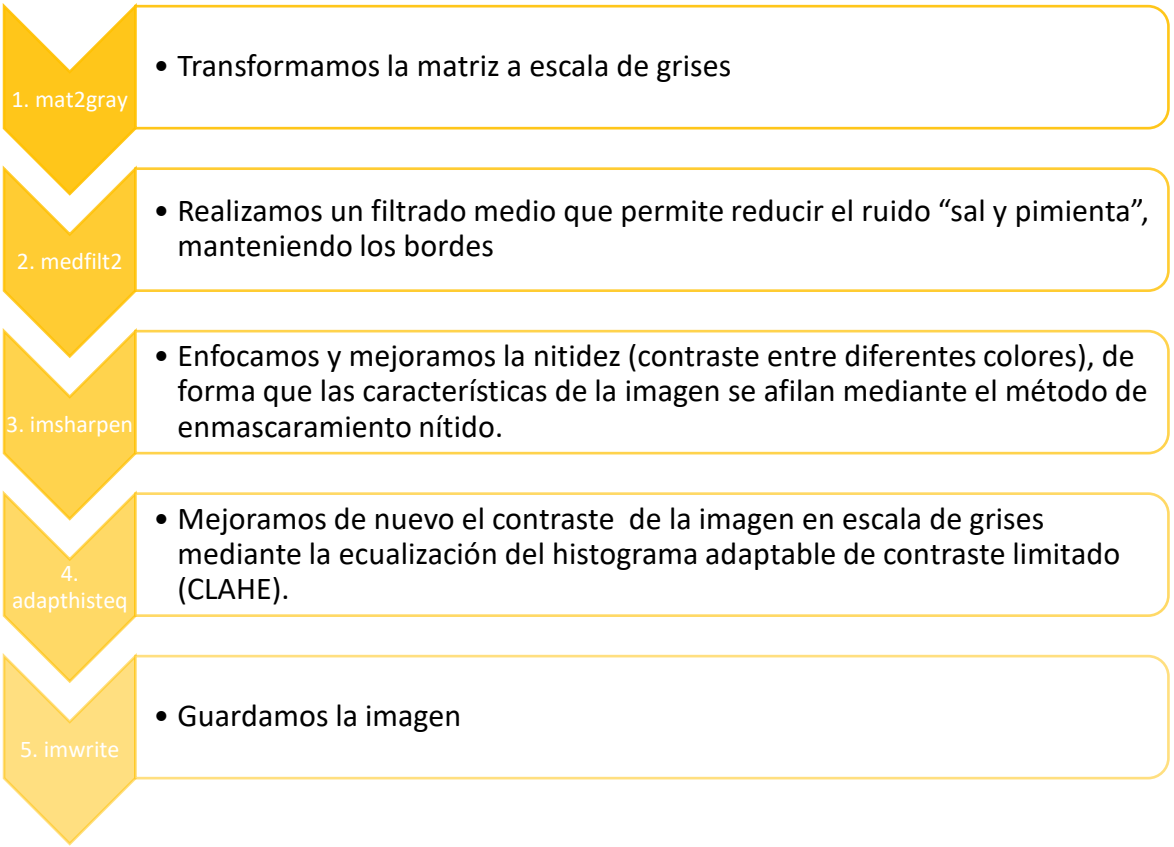
Para cada uno de los 218 pacientes, cargamos las imágenes (1) y almacenamos los datos de las mismas en una matriz.

A continuación, he empleado dos métodos distintos para procesar las imágenes que posteriormente serán la entrada de la red neuronal.

3.1.1.1 Prueba 1: DL 19

En esta primera prueba seleccionamos 19 imágenes de entre las 50-400 que forman el escáner de cada paciente y las guardamos.

Antes de guardarlas hay que seguir una serie de pasos para transformar la matriz que habíamos sacado previamente en una imagen de nuevo, pero, en este caso, en formato jpg.




```
imagen= A(:,:,i);
    imagenGris = mat2gray(imagen); 1
    imagenMedFilter = medfilt2(imagenGris);2
    imagenSharpFilter = imsharpen(imagenMedFilter); 3
    imagenAdjust = adapthisteq(imagenSharpFilter); 4
    nombre = sprintf ('AdapHisteq_CTR_TRN_%03d_%03d.jpg',j,slides);
    imwrite(imagenAdjust, nombre)5
```

Además de *adapthisteq*, utilicé otros métodos para mejorar la imagen, como *imadjust* e *histeq*, pero los resultados obtenidos fueron muy similares.

Figura 50: Por orden, de izquierda a derecha, Imagen original, Imadjust, Histeq y AdaptHisteq

Por ejemplo, algunas de las imágenes generadas para el paciente 1 serían:

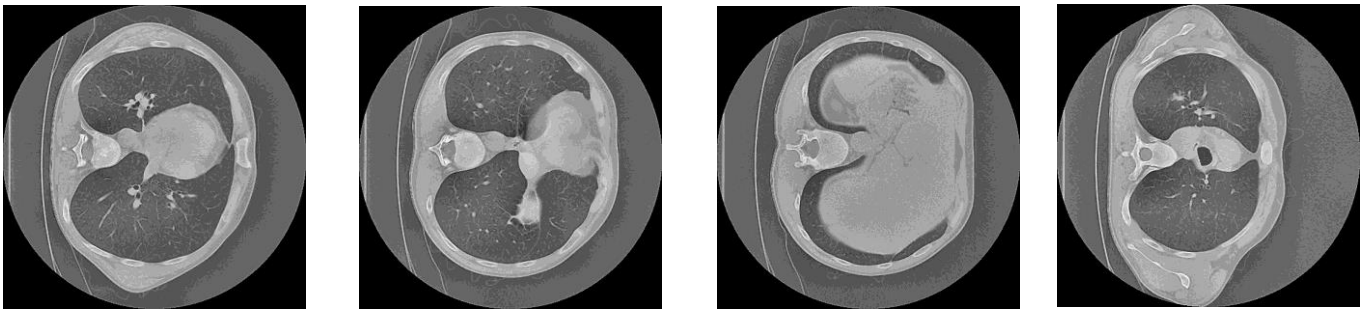


Figura 51: Imágenes obtenidas de algunos cortes del primer paciente

3.1.1.2 Prueba 2: DL conjunto 400 duplicadas

Por otro lado, también generé una imagen formada por todas las imágenes que componen el escáner de cada paciente.

Como cada uno de ellos tenía un número variable de rebanadas lo primero era normalizar de forma que todos ellos tuvieran una imagen del mismo tamaño.

Había dos opciones principalmente:

1. Eliminar rebanadas en los pacientes que tuvieran más de 50 rebanadas hasta que todos ellos obtuvieran una imagen formada por 50 imágenes.
2. Hacer copias de algunas rebanadas en pacientes con menos de 400 hasta que la imagen resultante para cada paciente tuviera 400.

La primera opción supondría la pérdida de información, así que opté por la segunda, generando para cada paciente una imagen con las 400 rebanadas, colocadas en 4 filas de 100 imágenes cada una.

Para ello se iban concatenando las matrices obtenidas de cada imagen en 4 vectores distintos y, finalmente se concatenaban entre ellos también.

```
    VectorImagenes1= A(:,:,i);
    [VectorImagenes1,i] = obtenerVector(VectorImagenes1,NumSlices,i,A, intervalo);
    VectorImagenes2= A(:,:,i);
    [VectorImagenes2,i] = obtenerVector(VectorImagenes2,NumSlices,i,A, intervalo);
    VectorImagenes3= A(:,:,i);
    [VectorImagenes3,i] = obtenerVector(VectorImagenes3,NumSlices,i,A, intervalo);
    VectorImagenes4= A(:,:,i);
    [VectorImagenes4,i] = obtenerVector(VectorImagenes4,NumSlices,i,A, intervalo);

 Imagen = [VectorImagenes1; VectorImagenes2; VectorImagenes3; VectorImagenes4];
```

obtenerVector, es una función creada para ir concatenando la imagen siguiente con las anteriores, teniendo en cuenta cuando es necesario duplicar alguna de ellas para conseguir que cada uno de los vectores esté formado por 100 imágenes.

Por último, seguí el mismo procedimiento mencionado en el apartado anterior para conseguir una mejora de la imagen.

La imagen obtenida para el primer paciente se muestra en la Figura 52.

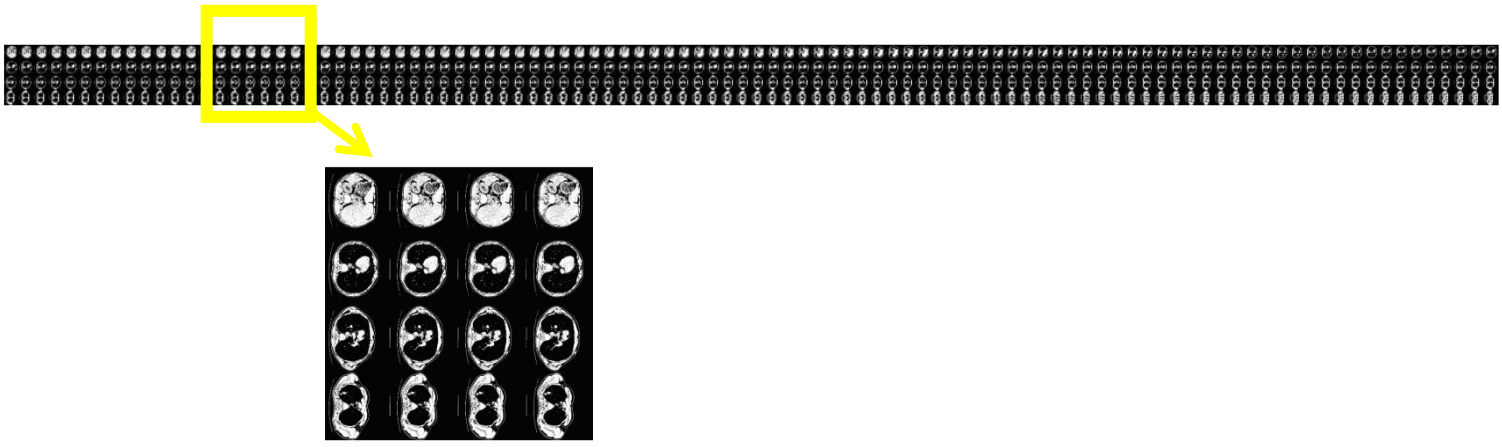


Figura 52: Imagen generada con todas las secciones del primer paciente, duplicando alguna de ellas

3.1.1.3 Prueba 3: DL conjunto 400 negro

Siguiendo la idea de la prueba anterior, generé una imagen formada por las 400 rebanadas de cada paciente. Pero, en este caso, en lugar de completar las imágenes que faltan en los distintos pacientes con copias de las rebanadas, concatené todas las imágenes facilitadas y rellené el espacio restante con imágenes completamente en negro.

Para esta prueba, decidí no mejorar la imagen ya que este proceso puede hacer que la red neuronal pierda datos de los cuales podría sacar información durante el proceso de entrenamiento.

Por tanto, las imágenes generadas tendrían un aspecto similar a la Figura 53.

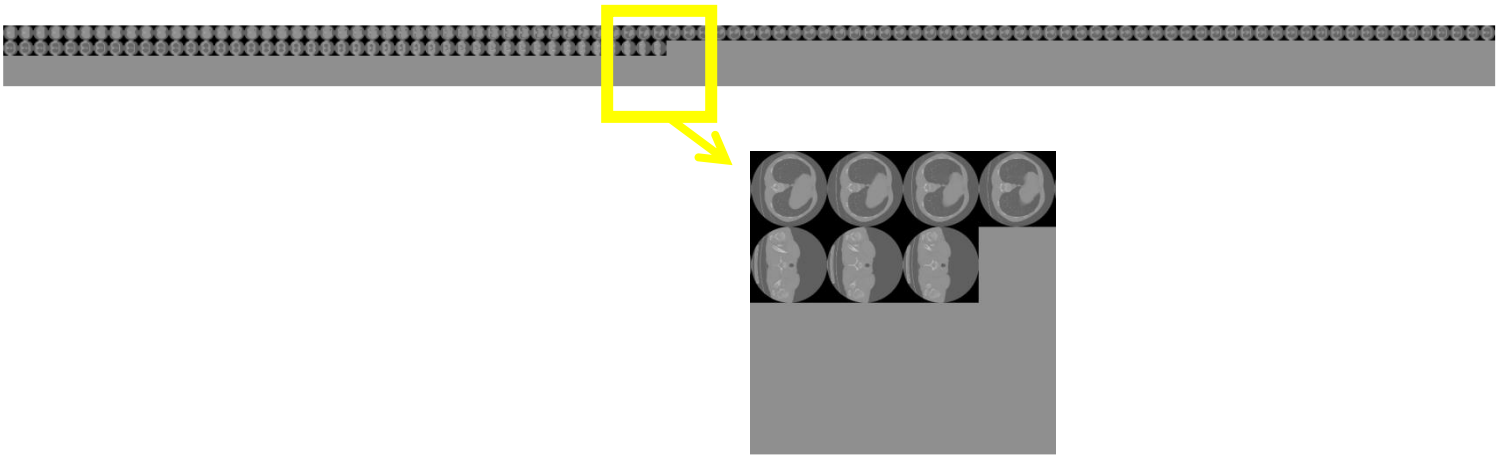


Figura 53: Imagen generada con todas las secciones del primer paciente, completando con imágenes en negro

3.1.1.4 Prueba 4: DL 400

Continuando con la idea anterior, entrenamos la red con 400 imágenes individuales para cada paciente, completando con imágenes totalmente en negro en el caso de que dicho paciente no llegue a las 400 muestras necesarias.

3.1.1.5 Prueba 5: DL conjunto 121 duplicadas

Tras unas cuantas pruebas, me di cuenta que entrenando a la red con una sola imagen formada por la composición de un conjunto de ellas daba mejor resultado que utilizando el conjunto directamente como entrada. Por esta razón continué el estudio basándome en estas conclusiones.

Analizando el número de muestras proporcionadas por cada paciente comprobé que la media es de 128 imágenes para cada uno de ellos. Para poder formar una imagen cuadrada, en lugar de rectangular como en los casos anteriores, extraje 121, que concatené conformando una matriz de 11x11.

Para los pacientes con menos de 121 cortes, dupliqué algunos de ellos y para aquellos con muestras de más fui quitando. En ambos casos, siguiendo un intervalo previamente calculado en función del número de muestras inicial.

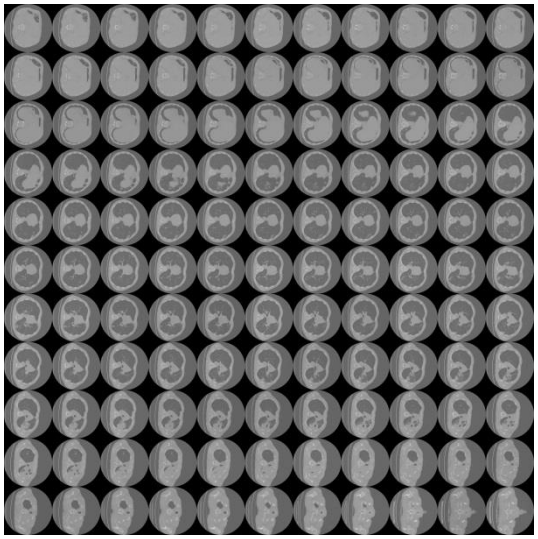


Figura 55: Paciente 212

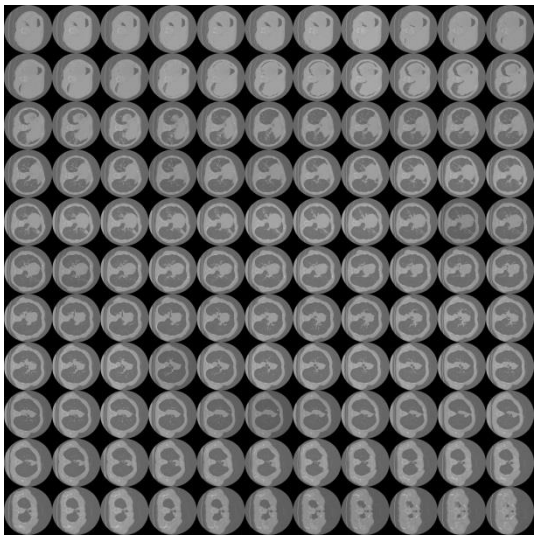


Figura 54: Paciente 96

3.1.1.6 Prueba 6: DL conjunto 121 negro

Con la misma base que la prueba anterior, también generé imágenes formadas por 121 imágenes para cada paciente. La diferencia es que para aquellos con menos muestras de las necesarias, en lugar de duplicar, completaba con imágenes completamente en negro.

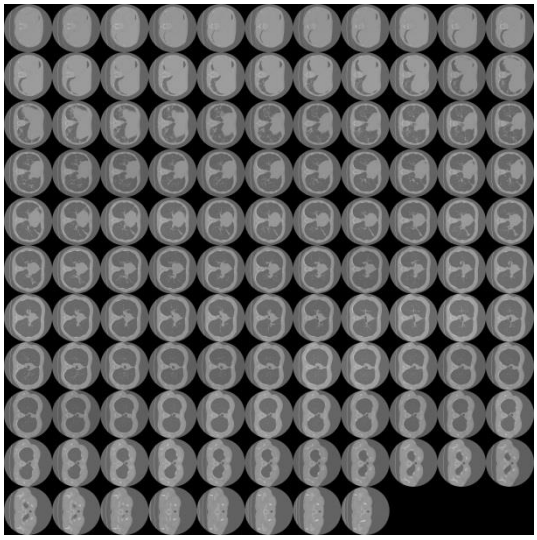


Figura 57: Paciente 78

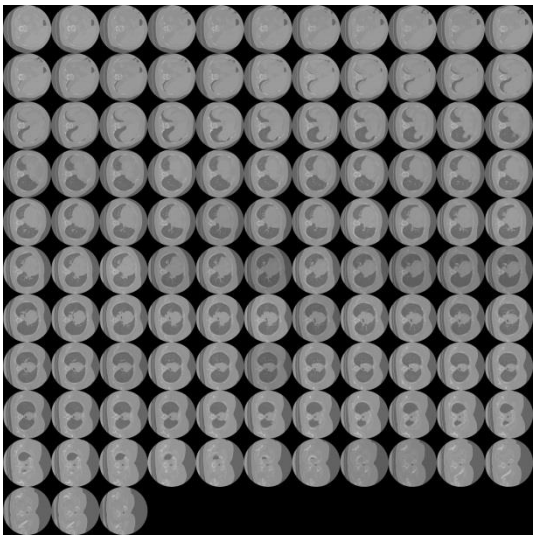


Figura 56: Paciente 3

3.1.1.7 Prueba 7: DL 25%

La siguiente pregunta a contestar era: ¿Es necesario utilizar muestras de todas las secciones para obtener un buen resultado? ¿Hay intervalos específicos que aportan más información o hay intervalos que no aportan información o incluso enmascaran el resultado?

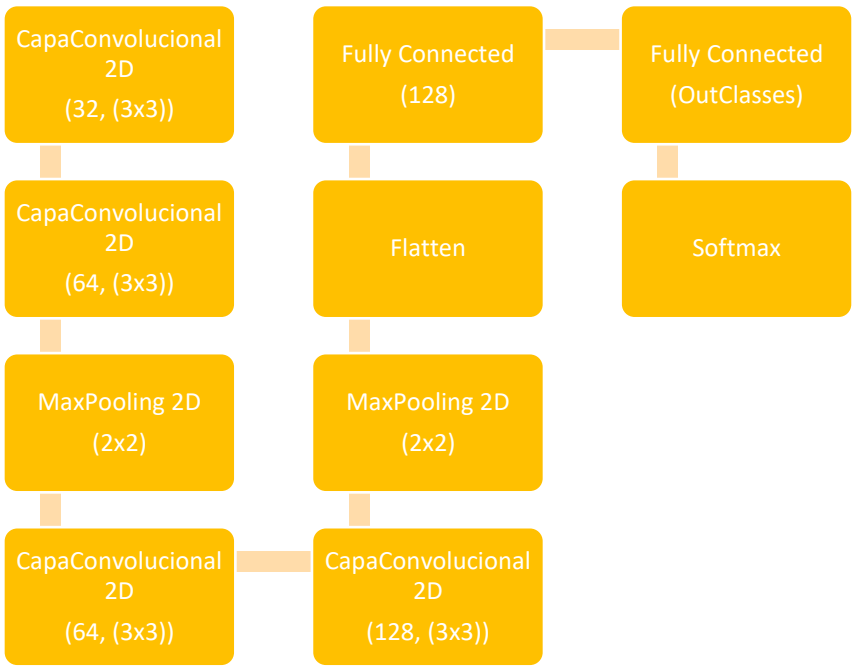
Para obtener una respuesta dividí el número de cortes de cada paciente en 4 intervalos de forma equitativa, por lo que cada uno de ellos contiene un 25 por ciento de las imágenes. Así se generan cuatro imágenes distintas, una para cada uno de los intervalos. De esta forma una imagen analizada estará formada por el primer cuarto de las imágenes totales, otra por el segundo cuarto y así sucesivamente.

Figura 58: Primer 25 , Segundo 25%, Tercer 25% y Cuarto 25 % de las tomas del paciente 27

3.1.2 Entrenamiento de la red neuronal

Utilicé dos variaciones de la misma red neuronal, la cual estaba formada por las siguientes capas:

3.1.2.1 Red Neuronal 1

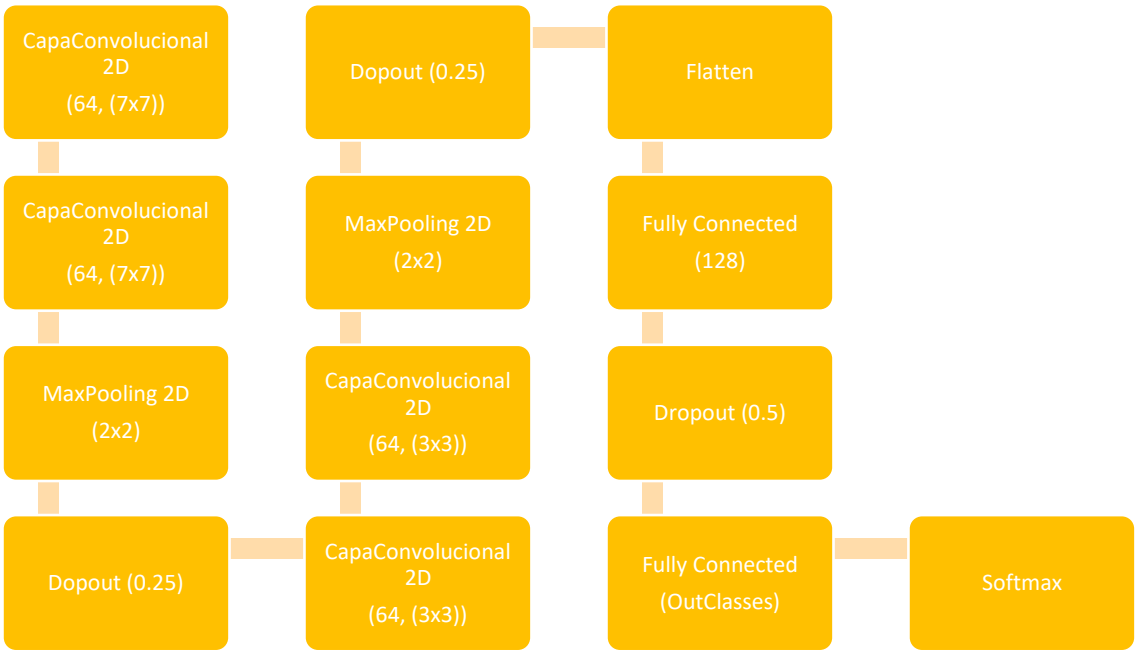


```python
def build_model3(input_shape):
    model = Sequential()

    model.add(Conv2D(32, (3, 3), padding='same', activation='relu', input_shape=input_shape, name='inputs'))
    model.add(Conv2D(64, (3, 3), activation='relu'))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Conv2D(64, (3, 3), padding='same', activation='relu'))
    model.add(Conv2D(128, (3, 3), activation='relu'))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Flatten())
    model.add(Dense(128, activation='relu', name='outputs'))
    model.add(Dense(nb_classes))
    model.add(Activation('softmax'))

    model.compile(loss='categorical_crossentropy', optimizer=keras.optimizers.Adam(), metrics=['accuracy'])

    return model
```

Con esta red neuronal se reduce el tamaño de la máscara de los filtros al principio, y a medida que se añaden capas, se va incrementando poco a poco.

Al trabajar con imágenes tan grandes, pude disminuir el tamaño de entrada de la imagen, ya que poco a poco se irá aumentado.

3.1.2.2 Red Neuronal 2



```
def build_model3(input_shape):
  model = Sequential()

  model.add(Conv2D(64, (7, 7), padding='same', activation='relu', input_shape=input_shape, name='inputs'))
  model.add(Conv2D(64, (7, 7), activation='relu'))
  model.add(MaxPooling2D(pool_size=(2, 2)))
  model.add(Dropout(0.25))

  model.add(Conv2D(64, (3, 3), padding='same', activation='relu'))
  model.add(Conv2D(64, (3, 3), activation='relu'))
  model.add(MaxPooling2D(pool_size=(2, 2)))
  model.add(Dropout(0.25))

  model.add(Flatten())
  model.add(Dense(128, activation='relu', name='outputs'))
  model.add(Dropout(0.5))
  model.add(Dense(nb_classes))
  model.add(Activation('softmax'))

  model.compile(loss='categorical_crossentropy', optimizer=keras.optimizers.Adam(), metrics=['accuracy'])

  return model
```

Manteniendo las capas utilizadas en la red neuronal anterior, en este caso, se mantiene el mismo tamaño de la máscara de los filtros desde el principio.

Además, se utiliza un dropout del 25% para reducir el sobreajuste que pueda producirse.

Para estas dos redes neuronales se ha utilizado el set de funciones de keras *callbacks. Por un lado ModelCheckpoint,* permitiéndome guardar el modelo después de cada época, y por otro, *EarlyStopping,* la cual cesa el entrenamiento cuando un parámetro especificado se monitorea y éste deja de mejorar. En este caso me he centrado en la pérdida, de forma que si tras 50 épocas dicho valor no disminuye se deja de entrenar la red.

```
callbacks = [ModelCheckpoint(filename, monitor='val_loss'), EarlyStopping(monitor='val_loss', patience=50)]
history = model.fit(X_train, y_train_nn, batch_size=batch_size, epochs=epochs, verbose=2, validation_split=0.1, callbacks=callbacks)
```

3.1.3 Entrenamiento

Para el entrenamiento de la red neuronal se utilizaron las imágenes generadas de los 218 pacientes.

En función del número de muestras que se introduzcan para cada uno de ellos se redimensiona la entrada. En las CNN esta suele ser de un solo frame con un tamaño de *"ancho x alto x 3"* en caso de una imagen RGB. Pero, por ejemplo, en la primera prueba se utilizan como entrenamiento 19 imágenes por paciente, de forma que cambiamos el tamaño de entrada a *"ancho x alto x número de muestras"*.

De entre todos los pacientes se seleccionaron de forma aleatoria el 90% de ellos para entrenar la red neuronal, utilizando el método de validación cruzada (cross-validation).

Este consiste en dividir los datos en dos partes, una que se utiliza como conjunto de entrenamiento para determinar los distintos parámetros de la red, y la otra, denominada como *conjunto de validación*, se emplea para obtener la tasa de clasificación incorrecta.

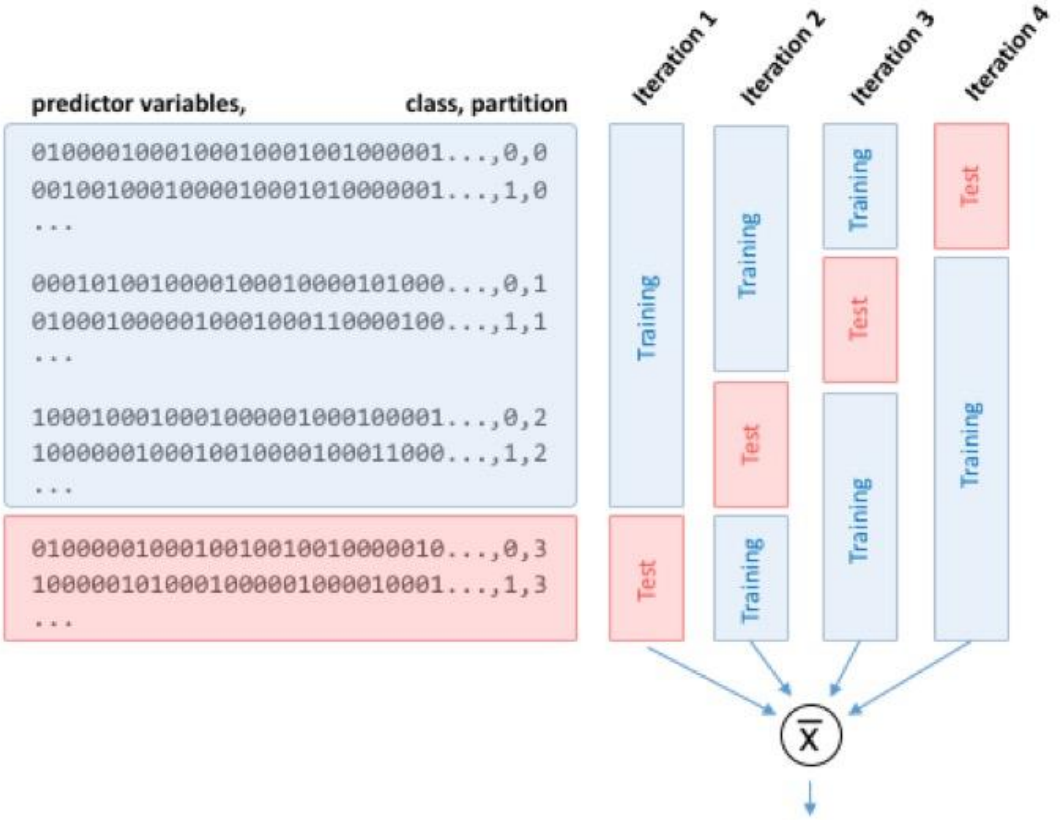
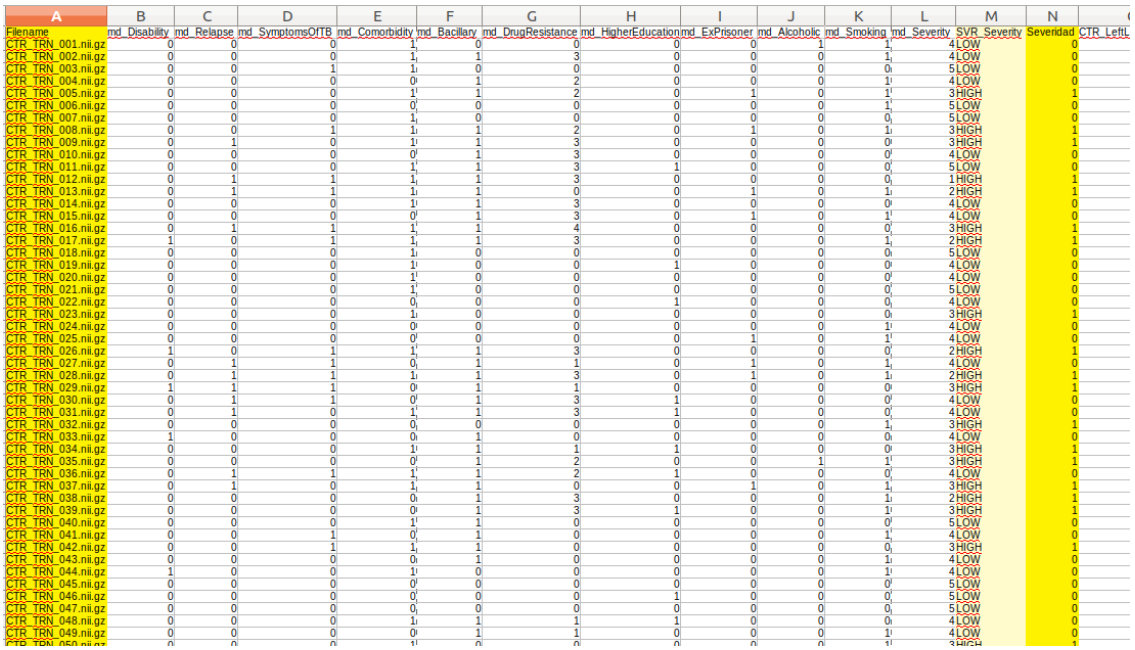


Figura 59: Proceso de cross-validation

El 10% restante de los pacientes se guarda para que, una vez entrenada la red, se pruebe su eficacia.

Además de las imágenes, también se ha de introducir en la red neuronal las etiquetas con la información relevante para analizar.

En este caso, se ha utilizado un Excel, donde para cada paciente se le tiene asignada una puntuación de la severidad de la tuberculosis: “LOW” (0) o “HIGH” (1).



Filename	md_Disability	md_Relapse	md_SymptomsOfTB	md_Comorbidity	md_Bacillary	md_DrugResistance	md_HigherEducation	md_ExPrisoner	md_Alcoholic	md_Smoking	md_Severity	SVR_Severity	Severidad	CTR_LeftL
CTR_TRN_001.nii.gz	0	0	0	1	0	0	0	0	1	1	4	LOW	0	
CTR_TRN_002.nii.gz	0	0	0	1	1	3	0	0	0	1	4	LOW	0	
CTR_TRN_003.nii.gz	0	0	1	1	0	0	0	0	0	0	5	LOW	0	
CTR_TRN_004.nii.gz	0	0	0	0	1	2	0	0	0	1	4	LOW	0	
CTR_TRN_005.nii.gz	0	0	0	1	1	2	0	1	0	1	3	HIGH	1	
CTR_TRN_006.nii.gz	0	0	0	0	0	0	0	0	0	1	5	LOW	0	
CTR_TRN_007.nii.gz	0	0	0	1	0	0	0	0	0	0	5	LOW	0	
CTR_TRN_008.nii.gz	0	0	1	1	1	2	0	1	0	1	3	HIGH	1	
CTR_TRN_009.nii.gz	0	1	0	1	1	3	0	0	0	0	3	HIGH	1	
CTR_TRN_010.nii.gz	0	0	0	0	1	3	0	0	0	0	4	LOW	0	
CTR_TRN_011.nii.gz	0	0	0	1	1	3	1	0	0	0	5	LOW	0	
CTR_TRN_012.nii.gz	0	1	1	1	1	3	0	0	0	0	1	HIGH	1	
CTR_TRN_013.nii.gz	0	1	1	1	1	0	0	1	0	1	2	HIGH	1	
CTR_TRN_014.nii.gz	0	0	0	1	1	3	0	0	0	0	4	LOW	0	
CTR_TRN_015.nii.gz	0	0	0	0	1	3	0	1	0	1	4	LOW	0	
CTR_TRN_016.nii.gz	0	1	1	1	1	4	0	0	0	0	3	HIGH	1	
CTR_TRN_017.nii.gz	1	0	1	1	1	3	0	0	0	1	2	HIGH	1	
CTR_TRN_018.nii.gz	0	0	0	1	0	0	0	0	0	0	5	LOW	0	
CTR_TRN_019.nii.gz	0	0	0	1	0	0	1	0	0	0	4	LOW	0	
CTR_TRN_020.nii.gz	0	0	0	1	0	0	0	0	0	0	4	LOW	0	
CTR_TRN_021.nii.gz	0	0	0	1	0	0	0	0	0	0	5	LOW	0	
CTR_TRN_022.nii.gz	0	0	0	0	0	0	1	0	0	0	4	LOW	0	
CTR_TRN_023.nii.gz	0	0	0	1	0	0	0	0	0	0	3	HIGH	1	
CTR_TRN_024.nii.gz	0	0	0	0	0	0	0	0	0	1	4	LOW	0	
CTR_TRN_025.nii.gz	0	0	0	0	0	0	0	1	0	1	4	LOW	0	
CTR_TRN_026.nii.gz	1	0	1	1	1	3	0	0	0	0	2	HIGH	1	
CTR_TRN_027.nii.gz	0	1	1	0	1	1	0	1	0	1	4	LOW	0	
CTR_TRN_028.nii.gz	0	1	1	1	1	3	0	1	0	1	2	HIGH	1	
CTR_TRN_029.nii.gz	1	1	1	0	1	1	0	0	0	0	3	HIGH	1	
CTR_TRN_030.nii.gz	0	1	1	0	1	3	1	0	0	0	4	LOW	0	
CTR_TRN_031.nii.gz	0	1	0	1	1	3	1	0	0	0	4	LOW	0	
CTR_TRN_032.nii.gz	0	0	0	0	0	0	0	0	0	1	3	HIGH	1	
CTR_TRN_033.nii.gz	1	0	0	0	1	0	0	0	0	0	4	LOW	0	
CTR_TRN_034.nii.gz	0	0	0	1	1	1	1	0	0	0	3	HIGH	1	
CTR_TRN_035.nii.gz	0	0	0	0	1	2	0	0	1	1	3	HIGH	1	
CTR_TRN_036.nii.gz	0	1	1	1	1	2	1	0	0	0	4	LOW	0	
CTR_TRN_037.nii.gz	0	1	0	1	1	0	0	1	0	1	3	HIGH	1	
CTR_TRN_038.nii.gz	0	0	0	0	1	3	0	0	0	1	2	HIGH	1	
CTR_TRN_039.nii.gz	0	0	0	0	1	3	1	0	0	1	3	HIGH	1	
CTR_TRN_040.nii.gz	0	0	0	1	1	0	0	0	0	0	5	LOW	0	
CTR_TRN_041.nii.gz	0	0	1	0	1	0	0	0	0	1	4	LOW	0	
CTR_TRN_042.nii.gz	0	0	1	1	1	0	0	0	0	0	3	HIGH	1	
CTR_TRN_043.nii.gz	0	0	0	0	1	0	0	0	0	1	4	LOW	0	
CTR_TRN_044.nii.gz	1	0	0	1	0	0	0	0	0	1	4	LOW	0	
CTR_TRN_045.nii.gz	0	0	0	0	0	0	0	0	0	0	5	LOW	0	
CTR_TRN_046.nii.gz	0	0	0	0	0	0	1	0	0	0	5	LOW	0	
CTR_TRN_047.nii.gz	0	0	0	0	0	0	0	0	0	0	5	LOW	0	
CTR_TRN_048.nii.gz	0	0	0	1	1	1	1	0	0	0	4	LOW	0	
CTR_TRN_049.nii.gz	0	0	0	0	1	1	0	0	0	1	4	LOW	0	

Figura 60: Etiquetas para los distintos pacientes

La forma de obtener la eficacia se realiza comparando los valores predichos por la red con los valores reales que se encuentran dentro de dicho Excel.

3.2 Optical Flow y ADV

En esta segunda prueba se ha utilizado un método en el que se utiliza una combinación del optical flow y un método de extracción de características llamado ADV.

La idea principal se basa en interpretar las imágenes proporcionadas como una secuencia de vídeo, de forma que se puedan extraer descriptores al analizar los movimientos y deformaciones producidos.

El ADV (Activity Description Vector) es un vector que se utiliza para el análisis de trayectorias. Este, describe la actividad en una secuencia de imágenes contando los movimientos en 4 direcciones del espacio 2D (derecha, izquierda, arriba, abajo) producidos en cada región de la imagen.

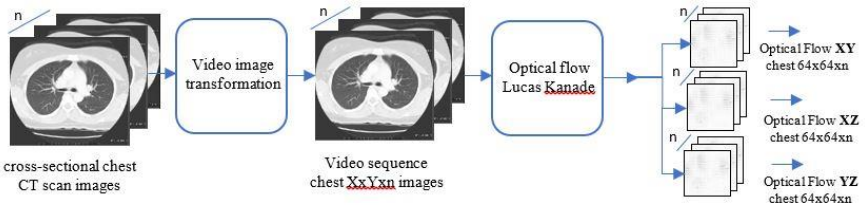


Figura 61: Proceso seguido para el cálculo del optical flow en cada uno de los planos

A partir de las tomografías de cada paciente, se generan tres secuencias de vídeo. Cada uno de ellos corresponde a la sección del volumen del escáner en cada eje: XY, XZ y YZ. A continuación, se calcula el optical flow para cada vídeo por separado utilizando el método Lucas Kanade(Rojas, n.d.).

Este algoritmo puede proporcionar una estimación del movimiento de unas características en concreto en una sucesión de imágenes de una escena. Se asocia un vector de movimiento (u, v) para cada píxel de interés en la escena, el cual se obtiene por la comparación de dos imágenes consecutivas.

Se basa en dos suposiciones:

1. Las dos imágenes están separadas por un incremento pequeño de tiempo, de forma que los objetos de la escena no se han desplazado significativamente. Por lo tanto, este algoritmo funciona mejor con objetos que se mueven lentamente.

2. Las imágenes representan una escena que contiene objetos con textura en tonos grises con diferentes niveles de intensidad, que cambian suavemente.

El algoritmo en sí no utiliza información del color, sino que escanea la segunda imagen en busca de una coincidencia para un determinado píxel. Funciona estimando la dirección en la que un objeto se ha movido, pudiendo explicar, de esta forma, los cambios en la intensidad producidos.

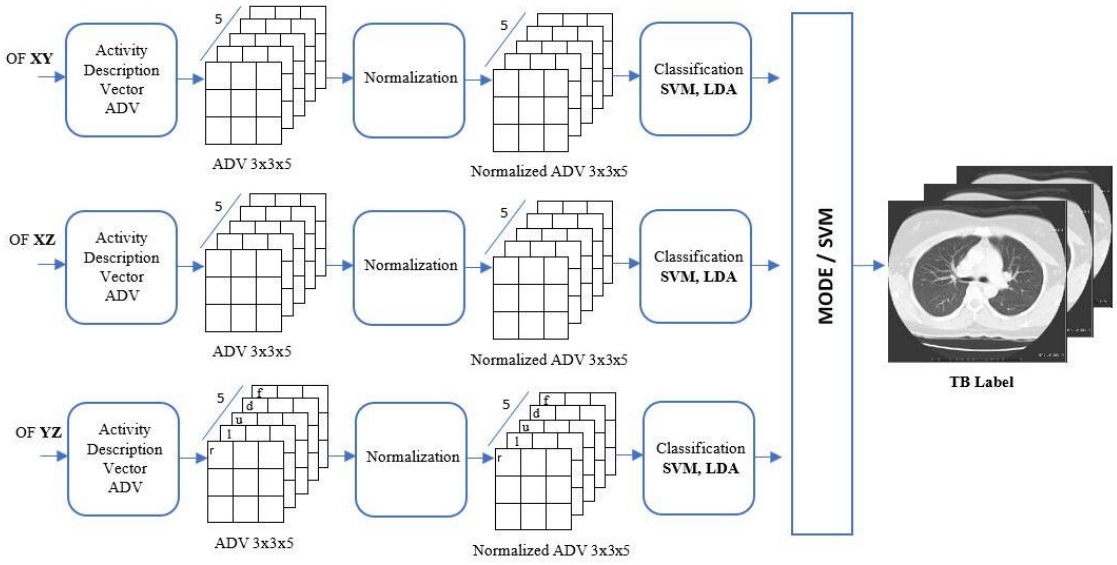


Figura 62: Proceso seguido para la clasificación de las imágenes a partir de la obtención del ADV

El siguiente paso es obtener el vector de características ADV. La dimensión de dicho vector es 3x3x5, donde los 4 primeros componentes acumulan los descriptores extraídos para cada optical flow obtenido en las 4 direcciones del espacio 2D en cada región 3x3 de la imagen y el último calcula las frecuencias en los cambios de dirección.

Tras la normalización del vector ADV generado, se utiliza este como entrada a un clasificador genérico como es SVM y LDA.

Finalmente, los resultados individuales para cada eje obtenidos tras la clasificación, se combinan proporcionando un único resultado utilizando un modelo estadístico o el clasificador SVM, empleado en el paso anterior.

El procedimiento descrito anteriormente fue el utilizado para resolver las tareas propuestas por el CLEF este año. A partir de este se han realizado unas modificaciones en este Trabajo de Fin de Grado con la idea de mejorar los resultados obtenidos.

Inicialmente, en lugar de cambiar el formato original de las imágenes de de *NIfTI* a *jpg*, estas se han guardado como *tiff*, ya que se trata de un formato sin pérdida. Para ello se ha hecho uso de la librería de Python *tiffile* y *nibabel*.

```
import os
import numpy as np


import nibabel as nib
import scipy.misc

from tifffile import imsave



for user in range(1,219):
    filename = "/home/irene/Documentos/ADV/FormatoNii/CTR_TRN_{:03}.nii".format(user)
    path = "/home/irene/Documentos/ADV/FormatoTiff/Paciente_{:03d}".format(user)

    try:
        os.mkdir(path)
    except OSError:
        print("Creation of the directory %s failed" %path)
    else: ("Successfully created the directory %s " %path)

    img = nib.load(filename)
    data = img.get_data()
    [row, colum, deep] = data.shape

    for x in range (0,deep):
        matrix = data[:,:,x]
        directorio = os.path.join(path,'CTR_TRN_{:03d}_{:03d}.tiff'.format(user,x))
        scipy.misc.imsave(directorio,matrix)

```

Con las imágenes obtenidas, se calcula el ADV siguiendo los mismos pasos explicados previamente. En cambio, en este caso, en lugar de obtener una única matriz LRUDF, se generan dos matrices, una con los movimientos hacia arriba y hacia abajo, junto con la frecuencia (UDF) y otra con los movimientos hacia la derecha, la izquierda y la frecuencia (LRD).

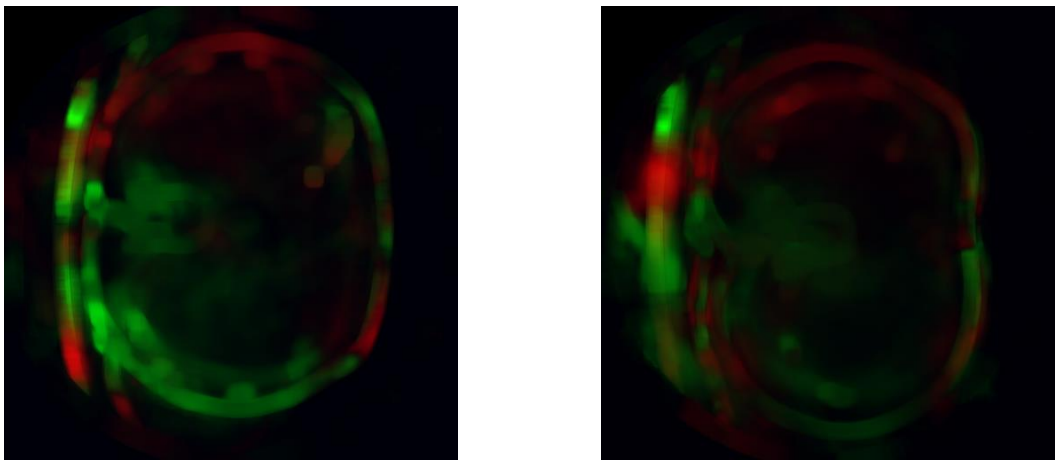


Figura 63: UDF Paciente 1 (izq) y Paciente 2 (dcha)

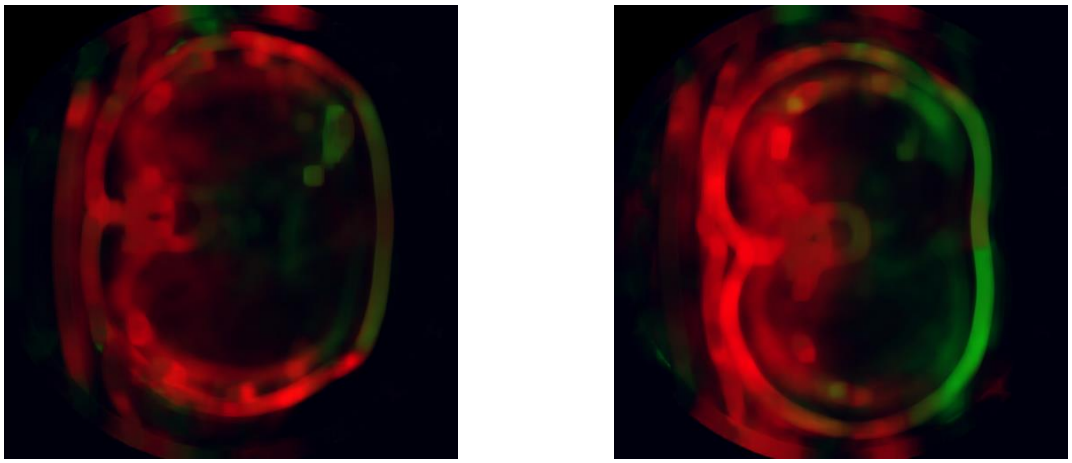


Figura 64: LRF Paciente 1 (izq) y Paciente 2 (dcha)

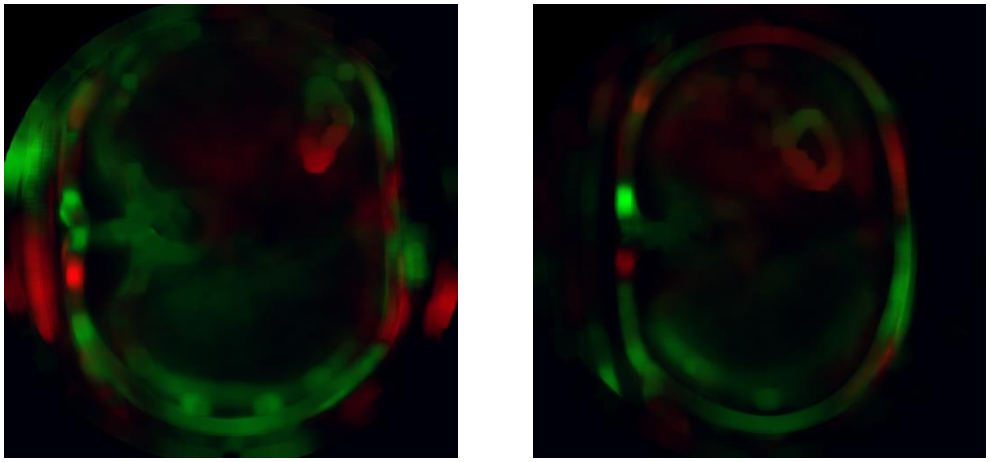


Figura 65: UDF Paciente 6 (izq) y Paciente 7 (dcha)

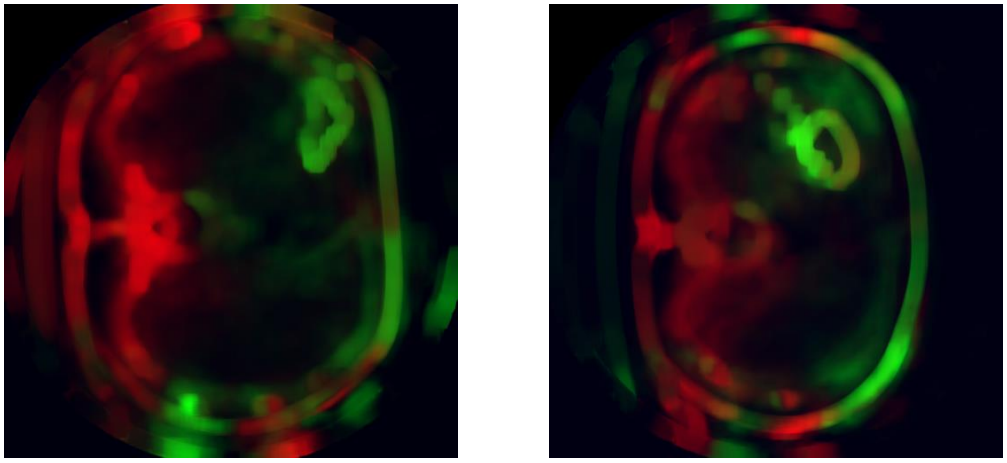


Figura 66: LRF Paciente 1 (izq) y Paciente 2 (dcha)

Además de las matrices UDF y LRF individuales para cada paciente, se generaron otras dos con los valores máximos, que se utiliarán en el siguente paso para normalizar las anteriores .

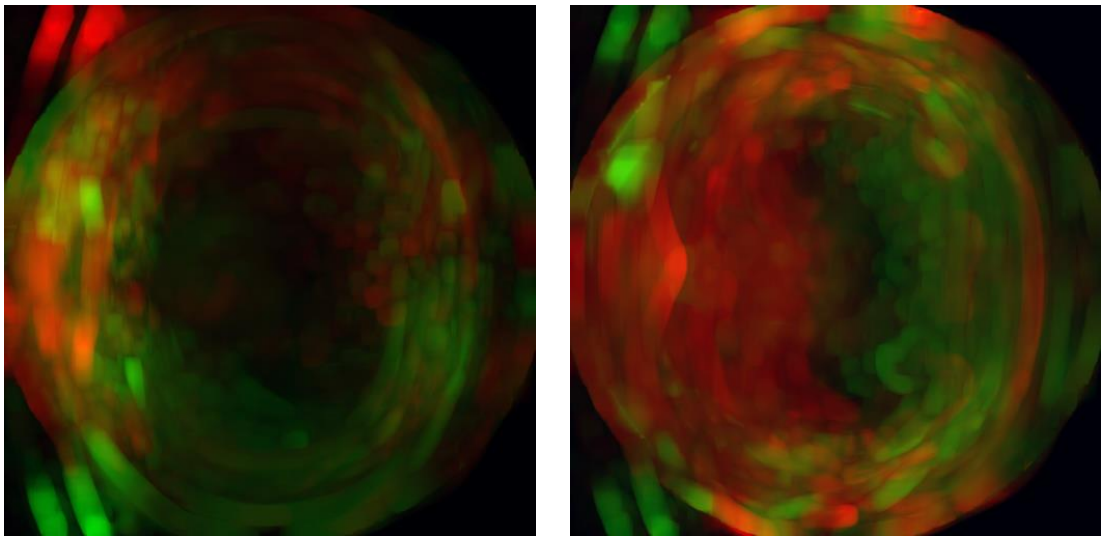


Figura 67: MaxUDF (izq) y MaxLRF (dcha)

Por último, una vez normalizadas las matrices (Figura 68, 69, 70 y 71), estas se clasifican utilizando la ResNet50, una red ya preentrenada que cuenta con 50 capas. Para ello primero hay que cambiar la última capa de forma que la salida clasifique entre las dos clases posibles: HIGH o LOW. A continuación, se congelan el resto de capas y se entrena la red para ajustar los pesos para la tarea a realizar. Finalmente, ya se puede utilizar para clasificar las matrices calculadas.

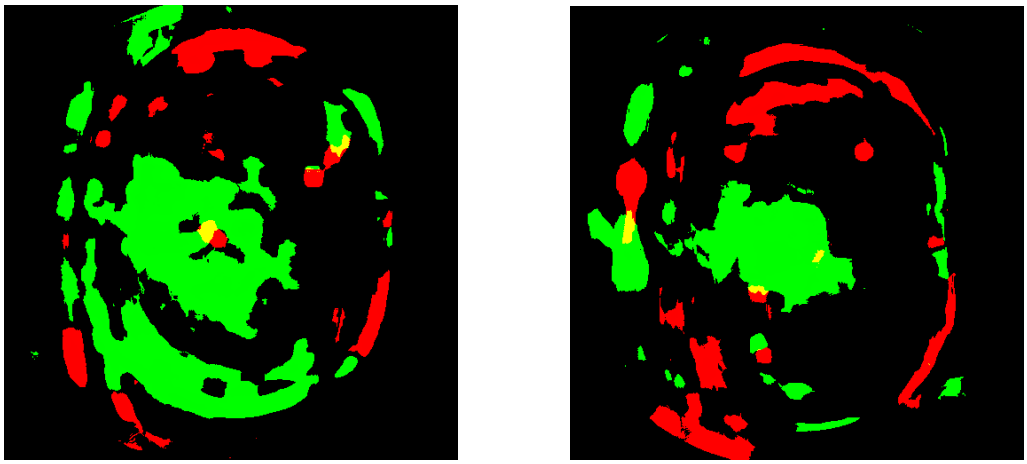


Figura 68: UDF normalizado Paciente 1 y 2

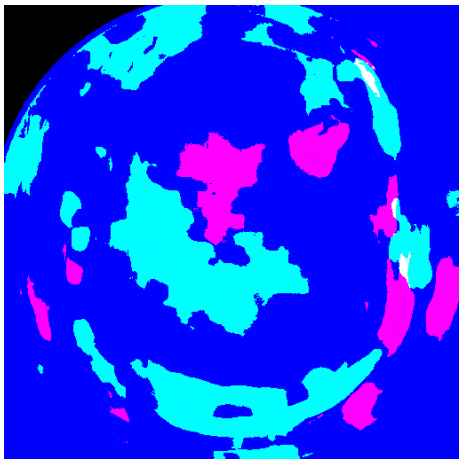
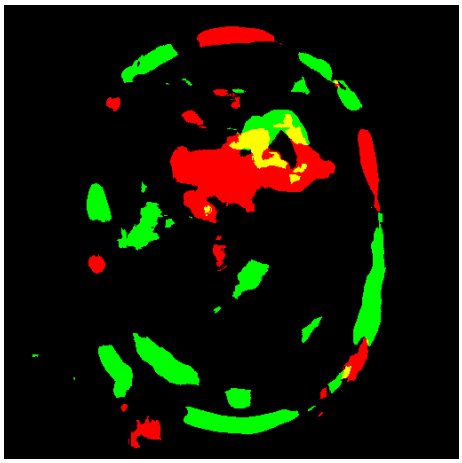



Figura 69: UDF normalizado Paciente 6 y 7

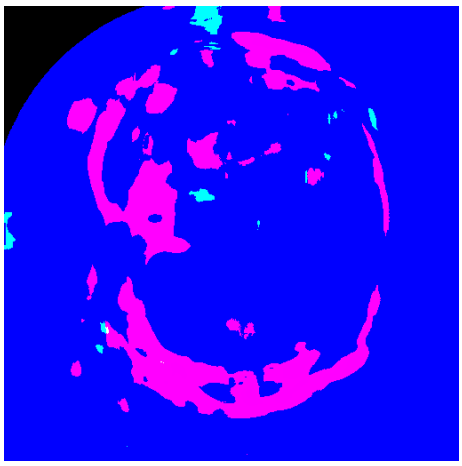
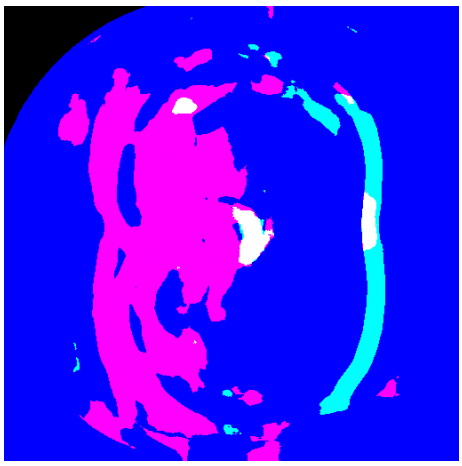



Figura 70: LRF normalizado Paciente 1 y 2

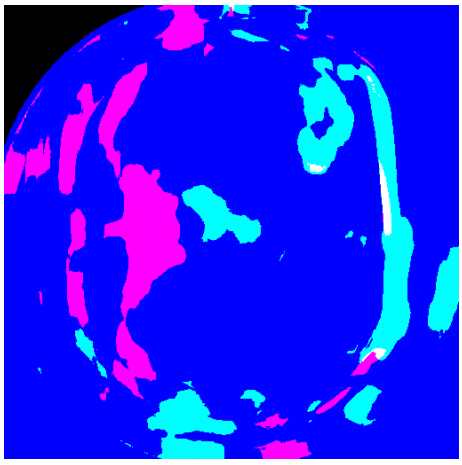
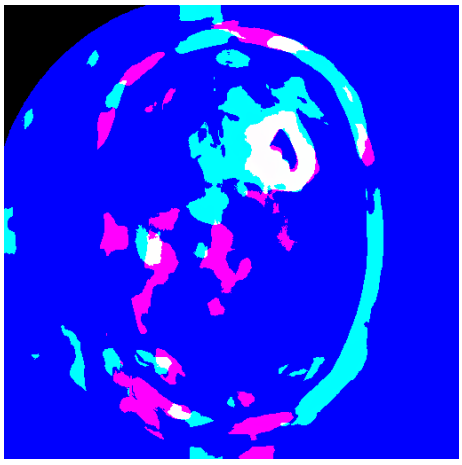



Figura 71: LRF normalizado Paciente 6 y 7

Para realizar una comparación del funcionamiento de las redes neuronales, se han utilizado las dos redes neuronales propuestas en el apartado anterior, además de la ResNet50. También se intentó probar la VGG, pero no se obtuvieron resultados debido a problemas de memoria con el ordenador.

4. EXPERIMENTACIÓN Y ANÁLISIS DE LOS RESULTADOS

4.1. Redes Neuronales

Para entender el funcionamiento de las redes neuronales se estudian las curvas de aprendizaje, donde se representa el rendimiento frente al tiempo.

Al revisar los resultados durante el entrenamiento se pueden diagnosticar problemas con el aprendizaje, como el underfitting u overfitting o conjuntos de datos de entrenamiento y validación poco representativos.

En dichas gráficas se muestran dos curvas:

- C**urva de aprendizaje durante el entrenamiento**: se obtiene con el conjunto de datos de entrenamiento y ofrece una idea de cómo el modelo está aprendiendo.
- **Curva de aprendizaje durante la validación:** se obtiene con el set de validación, al cual no ha tenido acceso la red neuronal previamente, y muestra cómo el modelo generaliza para nuevos casos.

Comportamientos del modelo

Figura 72: Distintos comportamientos de las redes neuronales durante el entrenamiento

i. Underfitting

Ocurre cuando un modelo no aprende del conjunto de datos utilizados para el aprendizaje ya que hay pocos datos de entrenamiento y la máquina es incapaz de generalizar el conocimiento.

Este tipo de comportamiento solo se puede identificar en las gráficas donde se representa la pérdida durante el entrenamiento, y se puede reconocer si cumple alguna de las siguientes características:

- Se muestra una línea horizontal o valores con ruido que muestran una gran pérdida.
- La pérdida disminuye constantemente hasta el final del entrenamiento.

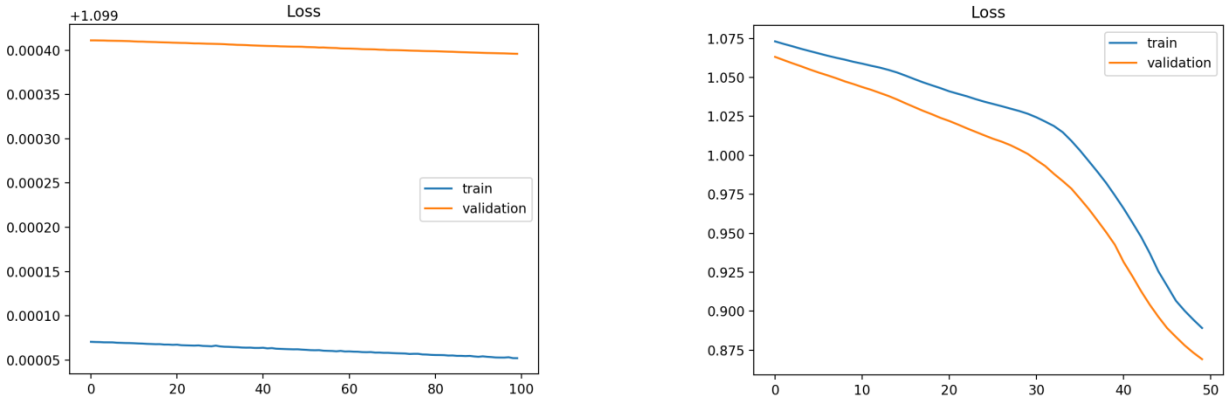


Figura 73: Funciones de pérdida en caso de underfitting

ii. Overfitting

En este caso el modelo aprende muy bien de los datos de entrenamiento, pero la máquina se ajusta solo a aprender los casos particulares que se le enseñan y es incapaz de reconocer nuevos datos. De esta forma, cuanto más se especializa el modelo al conjunto de datos, es menos capaz de generalizar con los nuevos.

Esto ocurre cuando la red neuronal es muy compleja para el problema a analizar o se entrena por mucho tiempo.

Se caracteriza por:

- La curva de pérdida durante el aprendizaje disminuye a medida que aumenta la experiencia.
- La curva de pérdida durante la validación disminuye hasta cierto punto, donde vuelve a aumentar de nuevo. Ese punto de inflexión puede ser el punto donde se detenga el entrenamiento, ya que a partir de él se produce el sobreajuste.

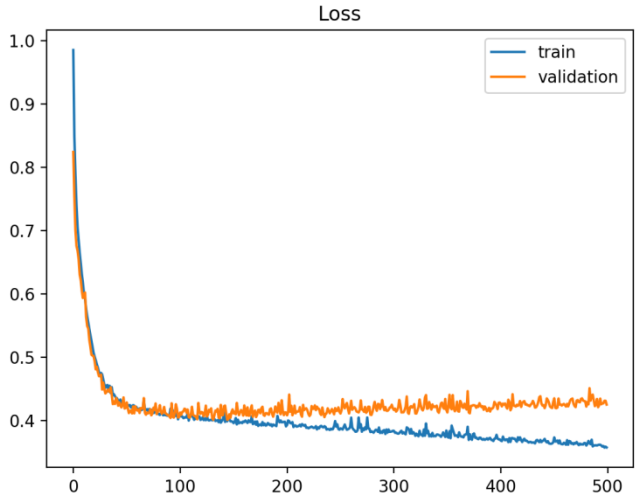


Figura 74: Función de pérdida en caso de overfitting

iii. Buenas curvas de aprendizaje

Se trata de la curva ideal y se identifica cuando la pérdida de validación y entrenamiento disminuyen hasta un punto de estabilidad con una diferencia mínima entre ellas conocida como "generalization gap" o brecha de generalización. Normalmente, la curva de entrenamiento se encuentra por debajo de la de validación.

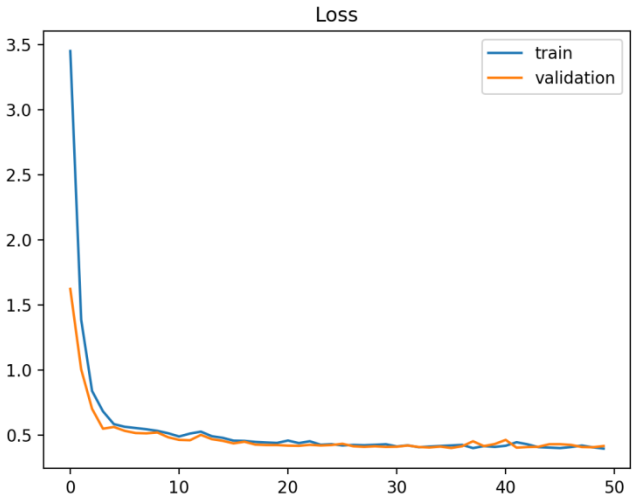
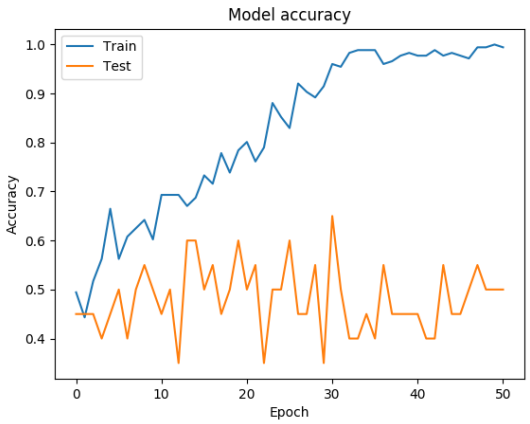
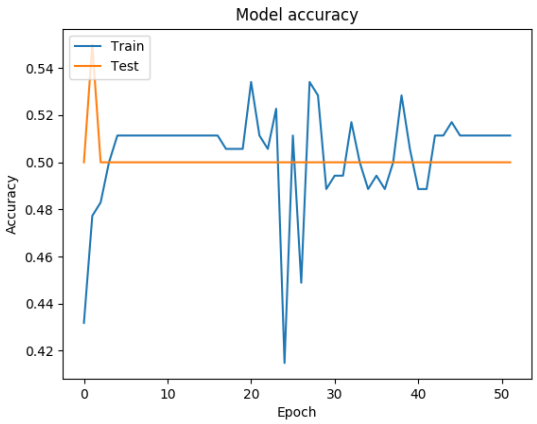
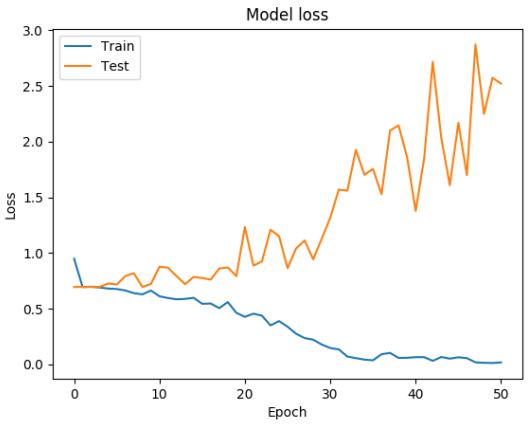


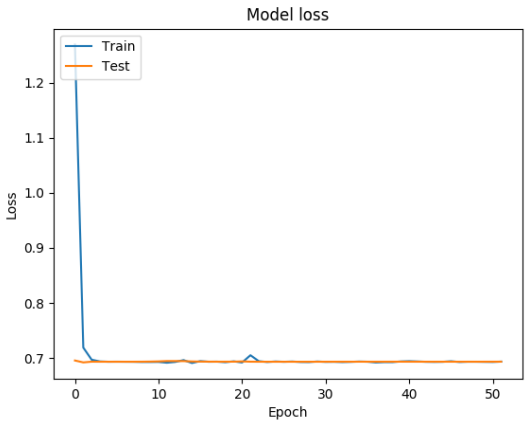
Figura 75: Función de pérdida cuando hay una buena tasa de aprendizaje

A continuación analizaremos los resultados obtenidos para cada una de las pruebas realizadas, estudiando las gráficas de pérdida (loss) y de rendimiento (accuracy).

4.1.1 Prueba 1: DL 19

RED NEURONAL 1	RED NEURONAL 2
	





RED NEURONAL 1

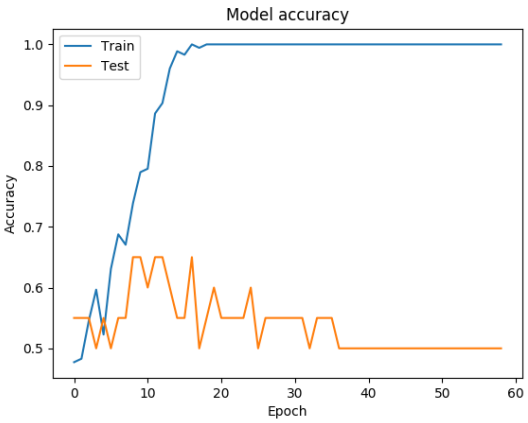
```
CNN accuracy: 0.52      [0.65 0.55 0.59 0.45 0.64 0.41 0.59 0.29 0.43 0.57]
```

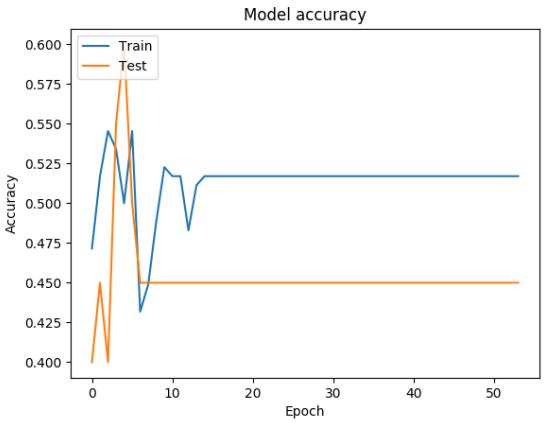
RED NEURONAL 2

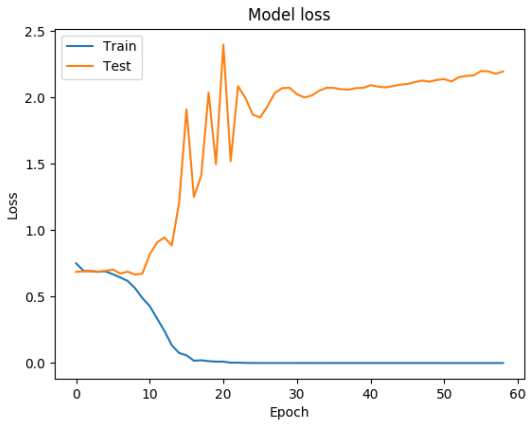
```
CNN accuracy: 0.49      [0.52 0.5  0.5  0.5  0.5  0.5  0.32 0.52 0.52 0.52]
```

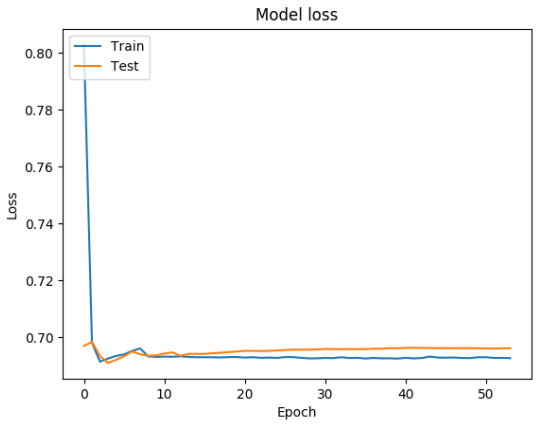
4.1.2 Prueba 2: DL conjunto 400 duplicadas

RED NEURONAL 1	RED NEURONAL 2









RED NEURONAL 1

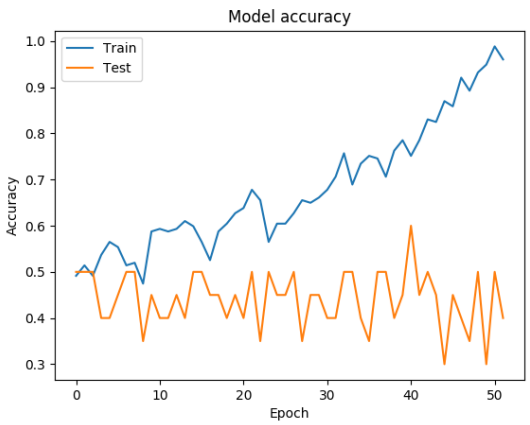
```
CNN accuracy: 0.50      [0.61 0.41 0.5  0.36 0.5  0.36 0.59 0.52 0.48 0.62]
```

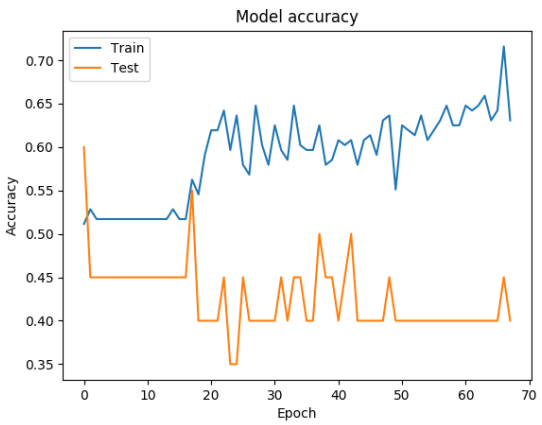
RED NEURONAL 2

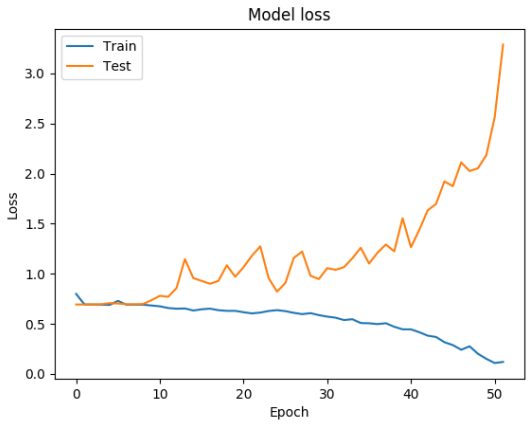
```
CNN accuracy: 0.51      [0.43 0.5  0.45 0.5  0.41 0.55 0.59 0.52 0.48 0.67]
```

4.1.3 Prueba3: DL conjunto 400 negro

RED NEURONAL 1	RED NEURONAL 2







RED NEURONAL 1

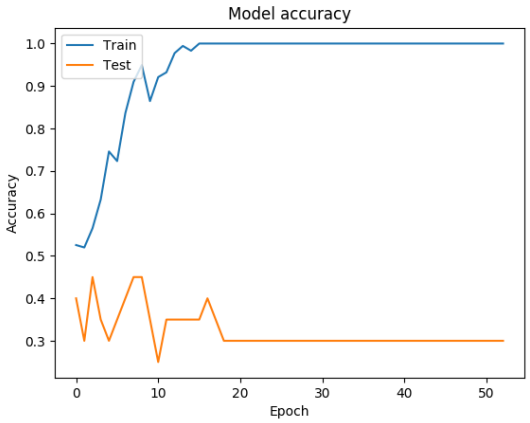
```
CNN accuracy: 0.58      [0.78 0.59 0.64 0.41 0.36 0.59 0.64 0.57 0.57 0.67]
```

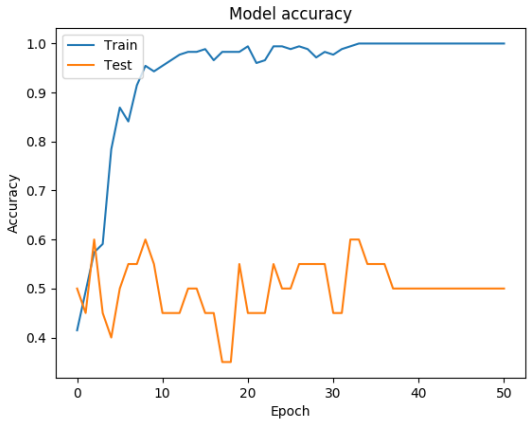
RED NEURONAL 2

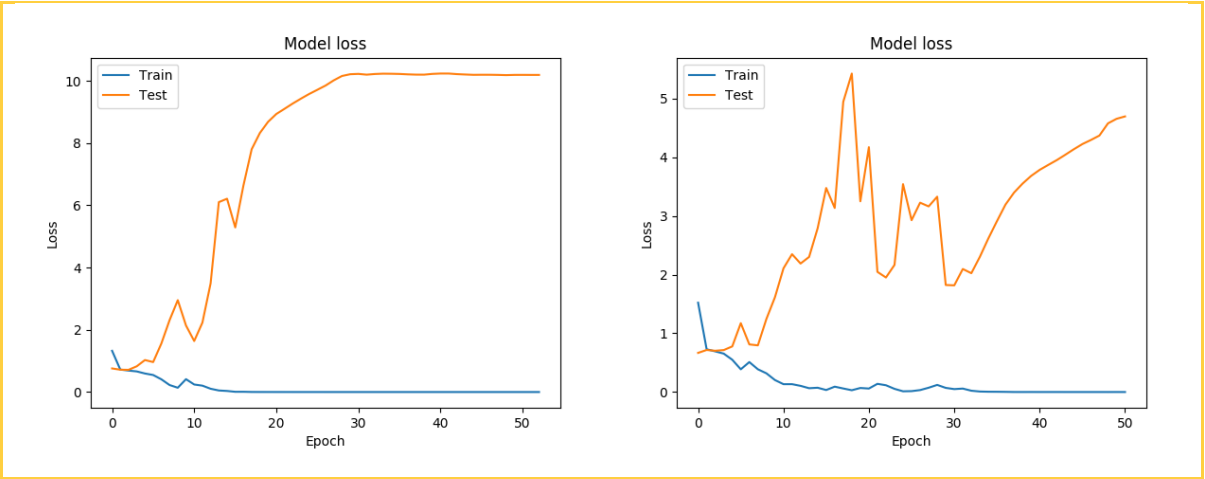
```
CNN accuracy: 0.56      [0.57 0.5  0.5  0.55 0.55 0.77 0.5  0.52 0.62 0.52]
```

4.1.4 Prueba 4: DL 400

RED NEURONAL 1	RED NEURONAL 2







RED NEURONAL 1

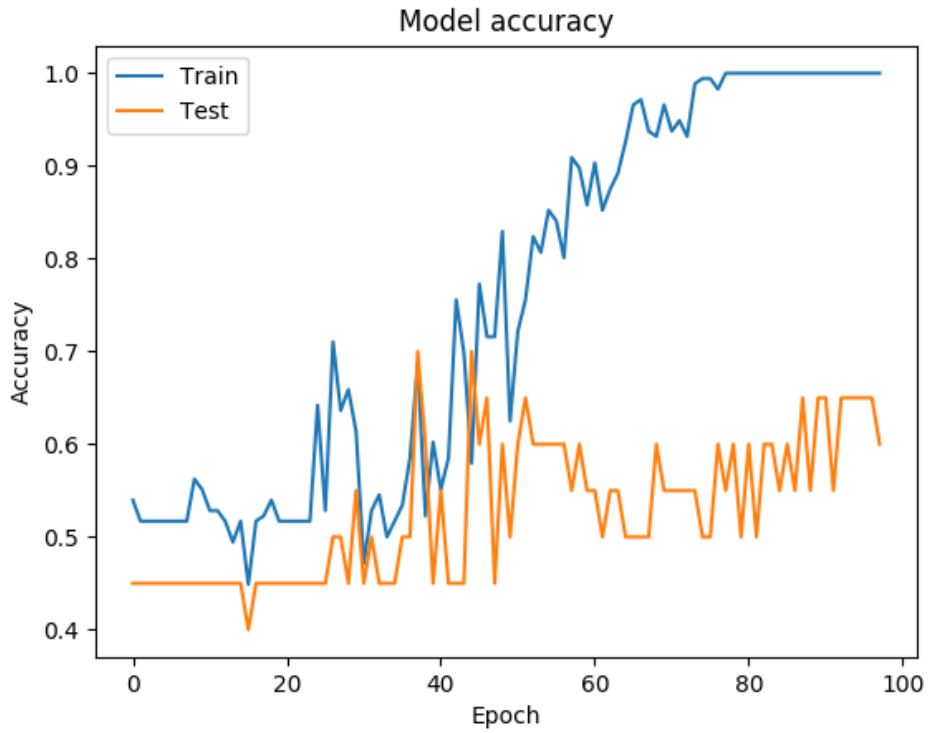
```
CNN accuracy: 0.48      [0.43 0.5  0.64 0.45 0.59 0.45 0.36 0.38 0.43 0.52]
```

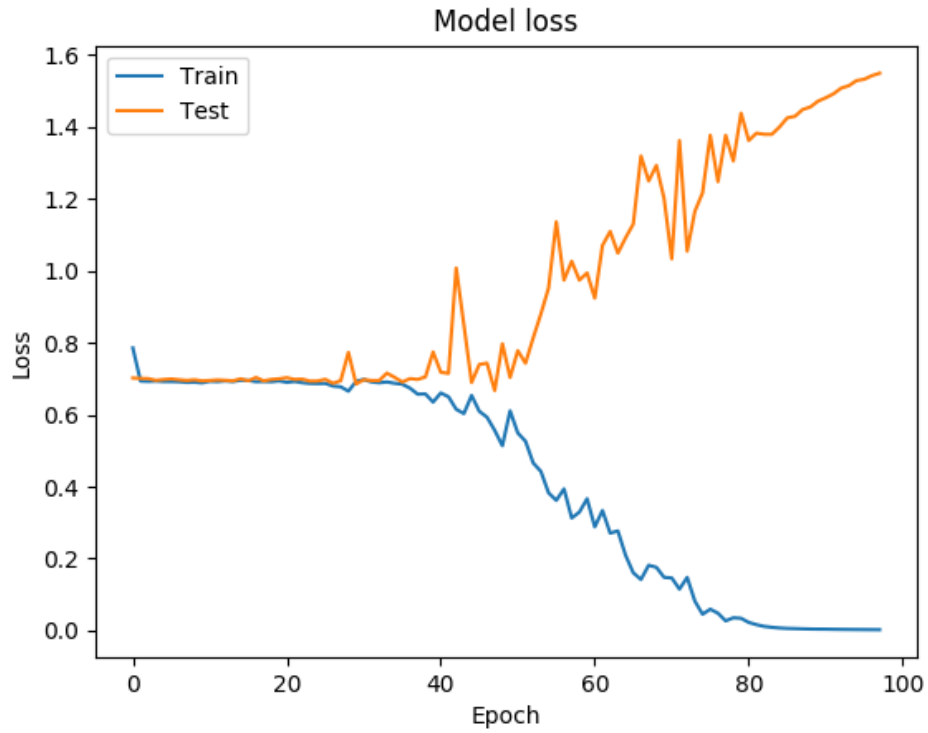
RED NEURONAL 2

```
CNN accuracy: 0.50      [0.52 0.45 0.68 0.32 0.5  0.36 0.59 0.43 0.48 0.62]
```

4.1.5 Prueba 5: DL conjunto 121 duplicadas

RED NEURONAL 1





RED NEURONAL 1

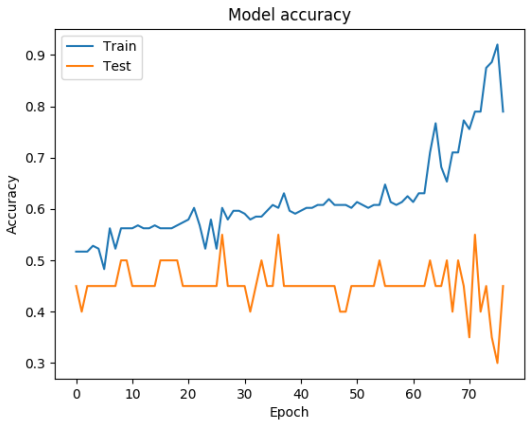
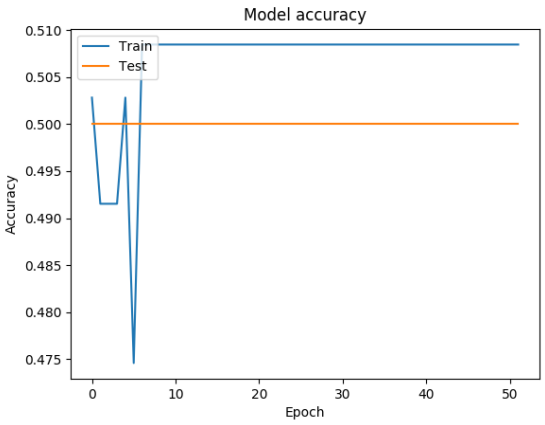
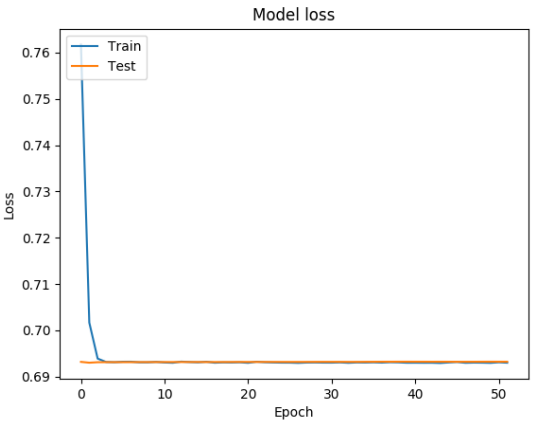
```
CNN accuracy: 0.59      [0.52 0.55 0.5  0.59 0.59 0.64 0.64 0.52 0.62 0.76]
```

RED NEURONAL 2

Al probar con la imagen formada por 121 cortes del paciente, duplicando algunos de ellos en el caso de los pacientes que poseen menos de 121 muestras, como entrada de la segunda red neuronal los resultados obtenidos muestran que ésta no aprende nada de dichas imágenes.

Aunque se modifique el tamaño de la imagen de entrada , el resultado sigue siendo el mismo.

4.1.6 Prueba 6: DL conjunto 121 negro

RED NEURONAL 1	RED NEURONAL 2
	
	

RED NEURONAL 1

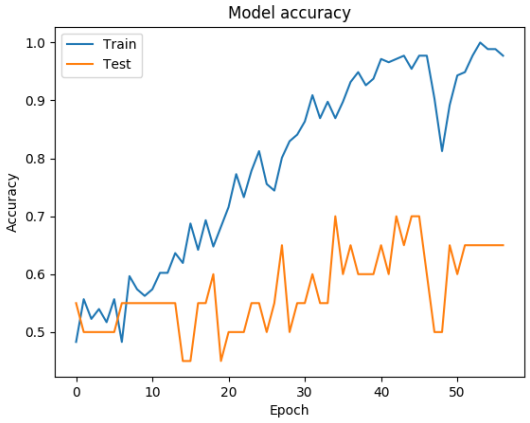
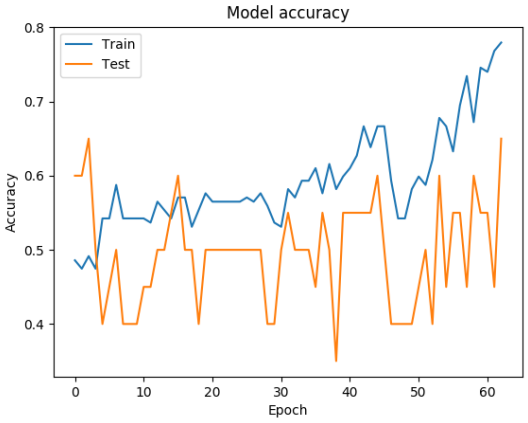
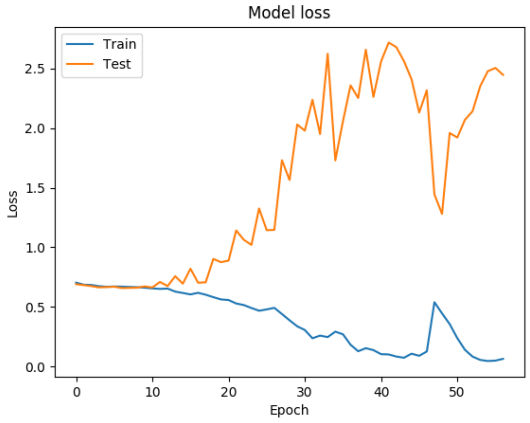
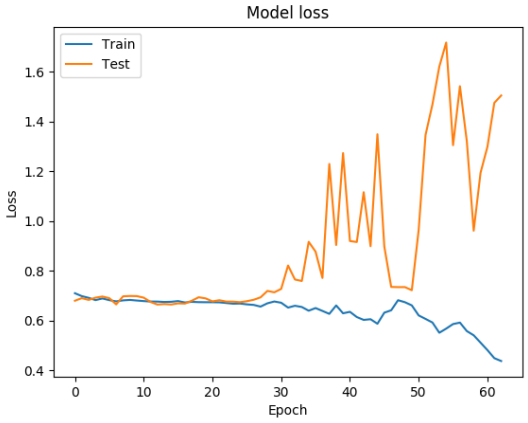
```
CNN accuracy: 0.57      [0.57  0.5 0.59 0.55 0.68 0.55  0.5 0.71 0.48 0.57]
```

RED NEURONAL 2

```
CNN accuracy: 0.52      [0.52 0.5  0.5  0.5  0.5  0.5  0.5  0.52 0.52 0.67]
```

4.1.7 Prueba 7: DL 25%

4.1.7.1 Primer intervalo

RED NEURONAL 1	RED NEURONAL 2
	
	

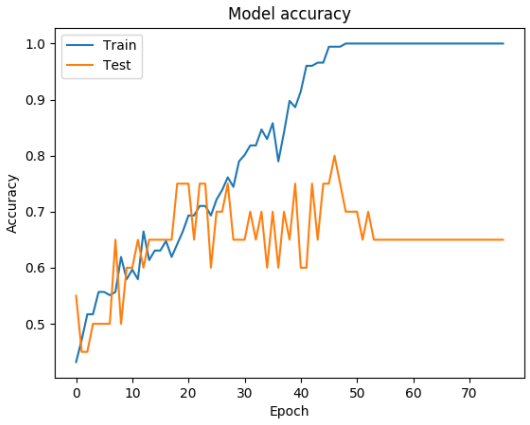
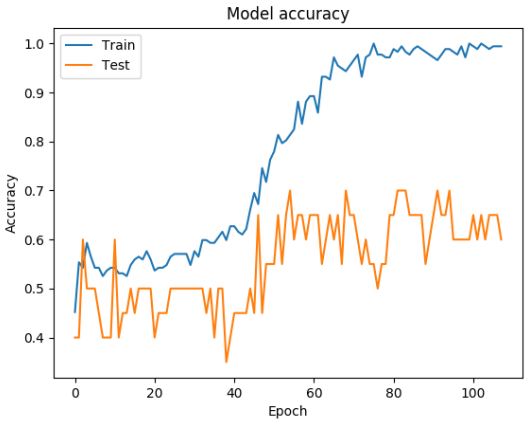
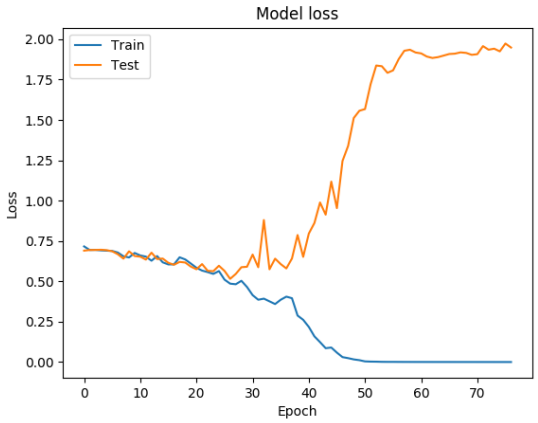
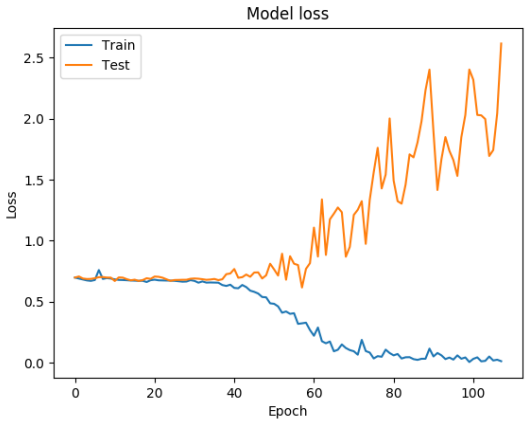
RED NEURONAL 1

```
CNN accuracy: 0.53       [0.57 0.5  0.59 0.55 0.59 0.5  0.5  0.38 0.67 0.48]
```

RED NEURONAL 2

```
CNN accuracy: 0.61       [0.57 0.68 0.73 0.59 0.64 0.64 0.45 0.57 0.62 0.62]
```

4.1.7.2 Segundo intervalo

RED NEURONAL 1	RED NEURONAL 2
	
	

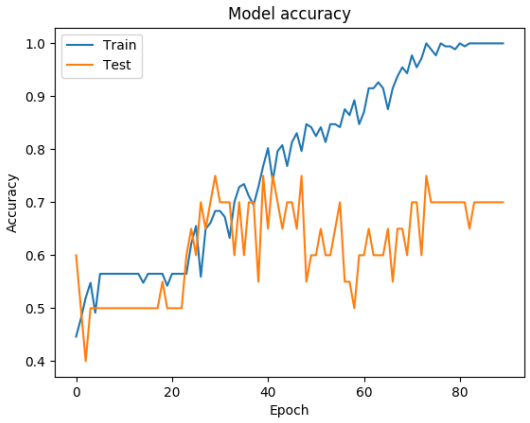
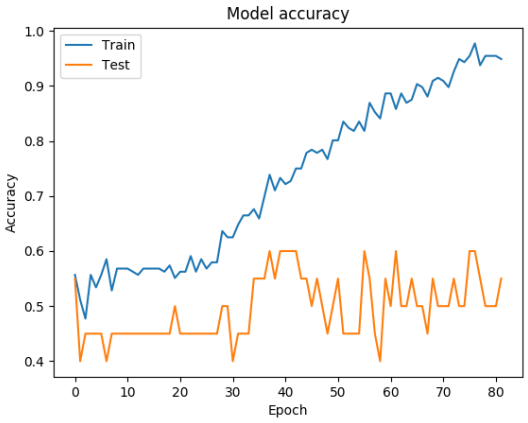
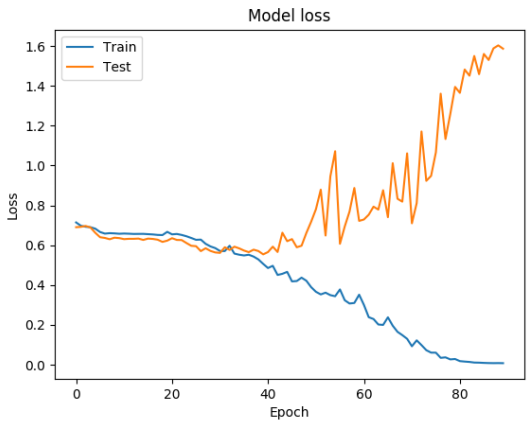
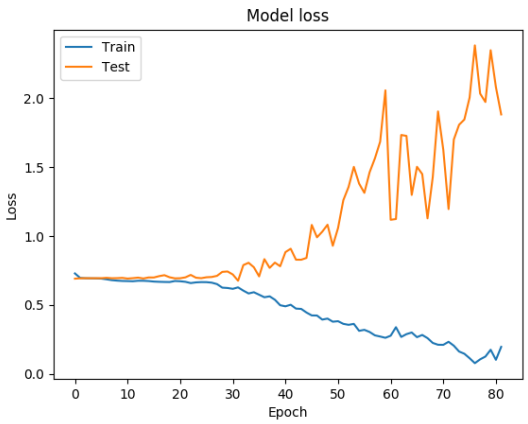
RED NEURONAL 1

```
CNN accuracy: 0.57       [0.48 0.59 0.59 0.5  0.45 0.55 0.68 0.52 0.52 0.76]
```

RED NEURONAL 2

```
CNN accuracy: 0.57      [0.61 0.59 0.77 0.36 0.5  0.64  0.5  0.43 0.71 0.62]
```

4.1.7.3 Tercer intervalo

RED NEURONAL 1	RED NEURONAL 2
	
	

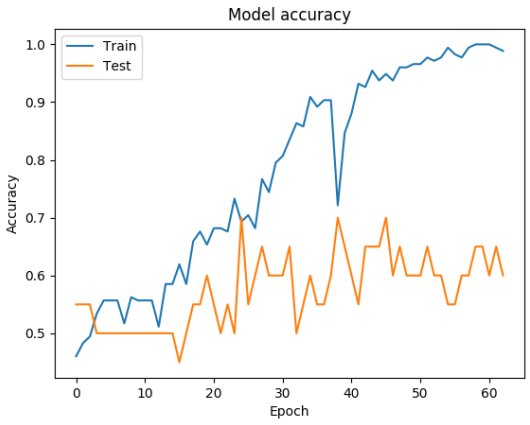
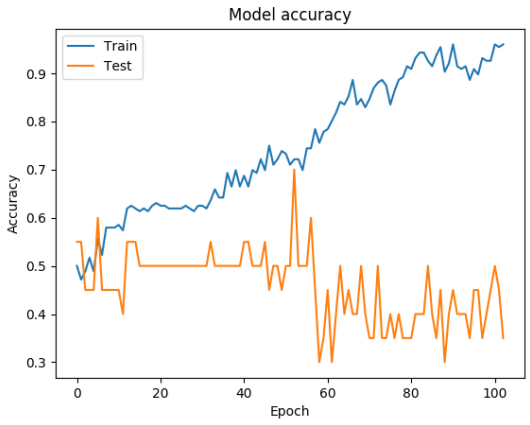
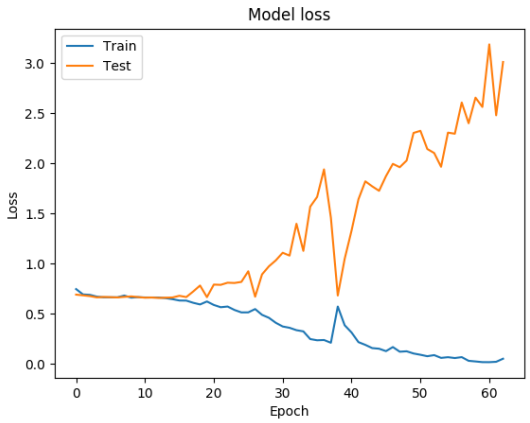
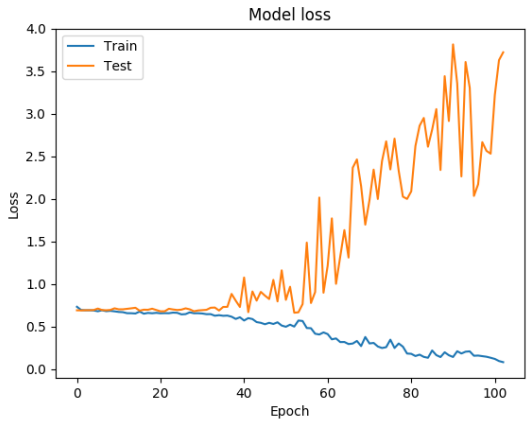
RED NEURONAL 1

```
CNN accuracy: 0.52      [0.48 0.55 0.45 0.64 0.64  0.45  0.45  0.67 0.52 0.33]
```

RED NEURONAL 2

```
CNN accuracy: 0.60      [0.61 0.73 0.55 0.59 0.5  0.5  0.68  0.48 0.76 0.62]
```

4.1.7.4 Cuarto intervalo

RED NEURONAL 1	RED NEURONAL 2
	
	

RED NEURONAL 1

```
CNN accuracy: 0.55      [0.57 0.55 0.36 0.59 0.59 0.59 0.59 0.48 0.57 0.57]
```

RED NEURONAL 2

```
CNN accuracy: 0.56      [0.65 0.59 0.45 0.5  0.41 0.77 0.55 0.52 0.71 0.48]
```

Si analizamos la función de pérdida en las distintas pruebas realizadas se puede comprobar que en la mayoría de ellas a partir de 20 épocas las dos curvas, de entrenamiento y de validación, se separan, de forma que deja de reducirse la pérdida,

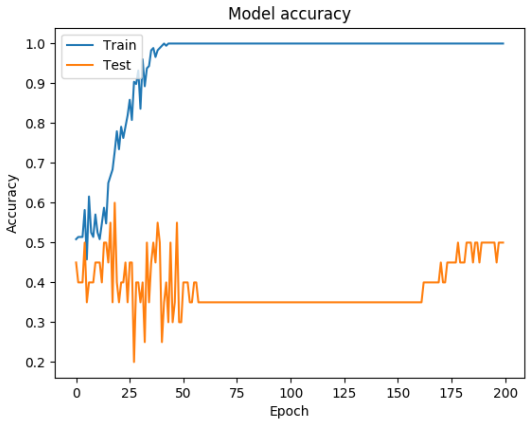
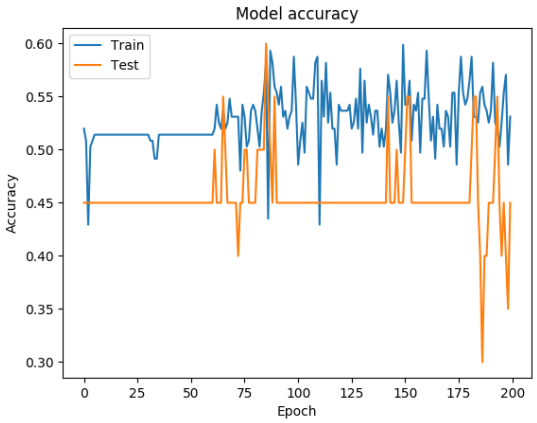
incluso aumenta. Además, en la gráfica del rendimiento se ve que éste no aumenta notablemente a lo largo del entrenamiento.

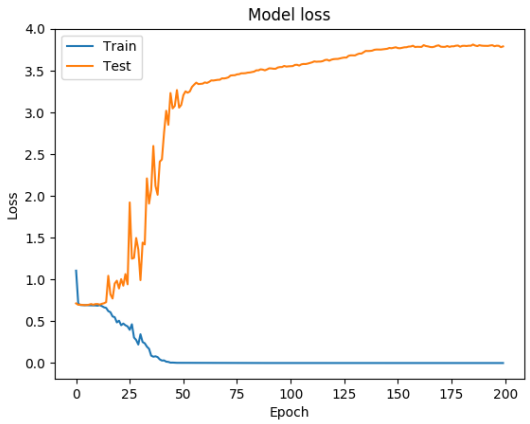
A la vista de los resultados se hicieron dos pruebas más.

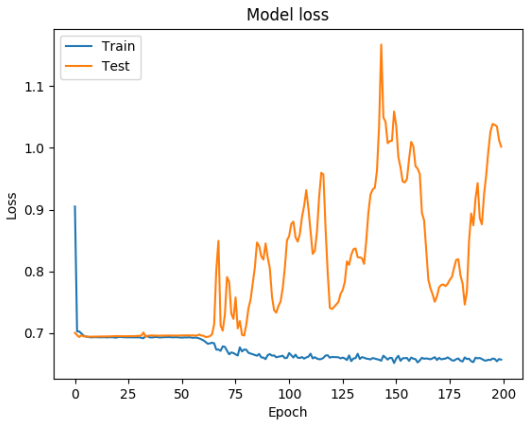
1. Aumentar el número de épocas a 200, ya que la gráfica se muestra ampliada y podría darse el caso de que las curvas que parecen divergir a partir de 20 épocas, realmente converja más adelante.
2. Reducir las épocas hasta el valor para el cual en cada prueba las dos curvas se separan debido a partir de este punto la red podría estar dejando de aprender y la estaríamos sobreentrenando.

200 épocas

4.1.8 Prueba 1: DL 19

RED NEURONAL 1	RED NEURONAL 2
	





RED NEURONAL 1

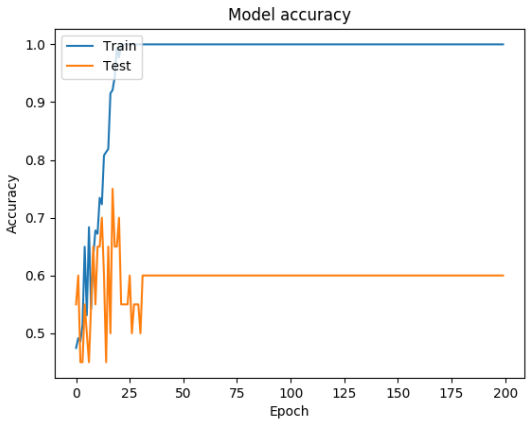
```
CNN accuracy: 0.90      [1.   1.   1.   1.   1.   1.   1.   0.9  0.48 0.57]
```

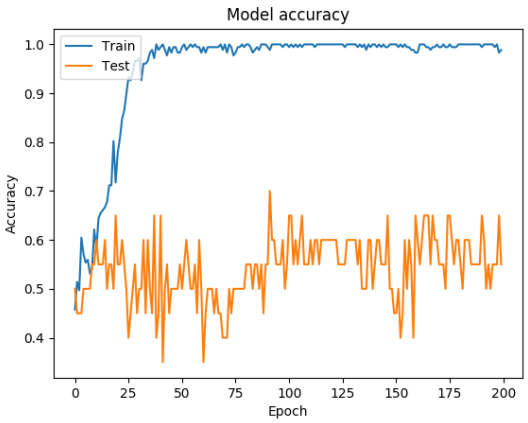
RED NEURONAL 2

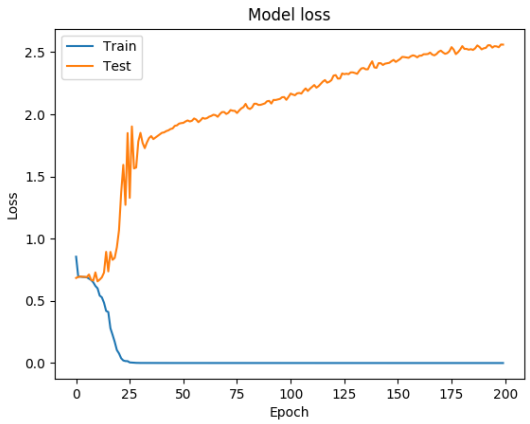
```
CNN accuracy: 0.90      [1.   1.   1.   1.   1.   1.   1.   0.9  0.48 0.57]
```

4.1.9 Prueba 2: DL conjunto 400 duplicadas

RED NEURONAL 1	RED NEURONAL 2







RED NEURONAL 1

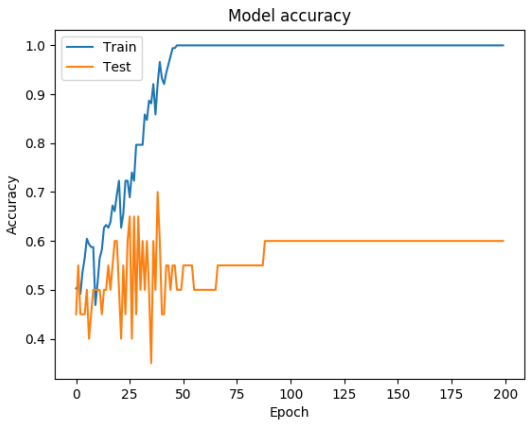
```
CNN accuracy: 0.89      [1.   1.   1.   1.   1.   1.   1.   0.95 0.48 0.48]
```

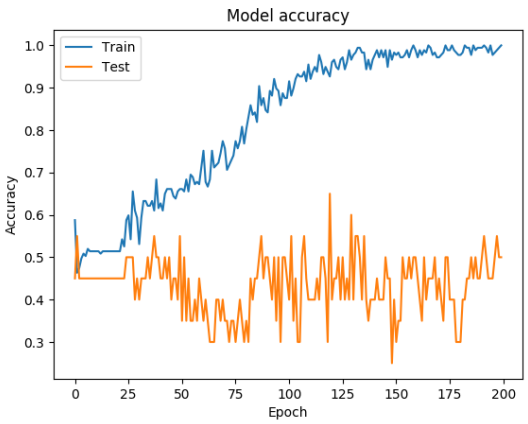
RED NEURONAL 2

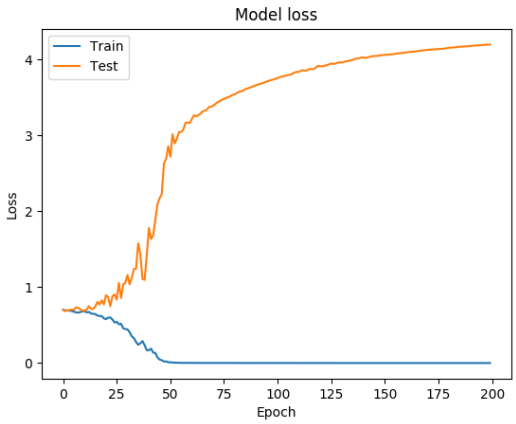
```
CNN accuracy: 0.89      [1.   1.   1.   1.   1.   1.   1.   0.95 0.48 0.48]
```

4.1.10 Prueba3: DL conjunto 400 negro

RED NEURONAL 1	RED NEURONAL 2







RED NEURONAL 1

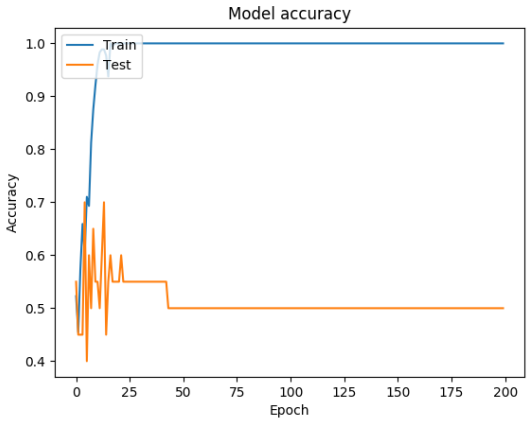
```
CNN accuracy: 0.92      [1.   1.   1.   1.   1.   1.   1.   0.95 0.57 0.67]
```

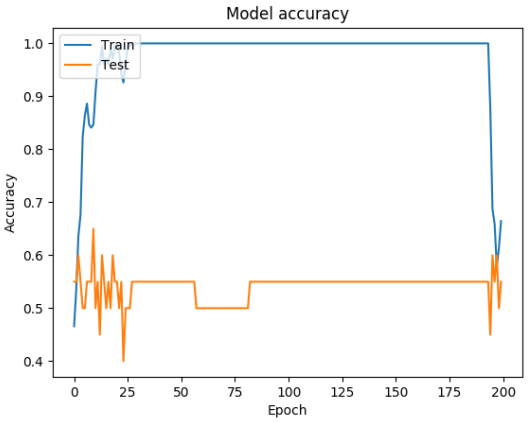
RED NEURONAL 2

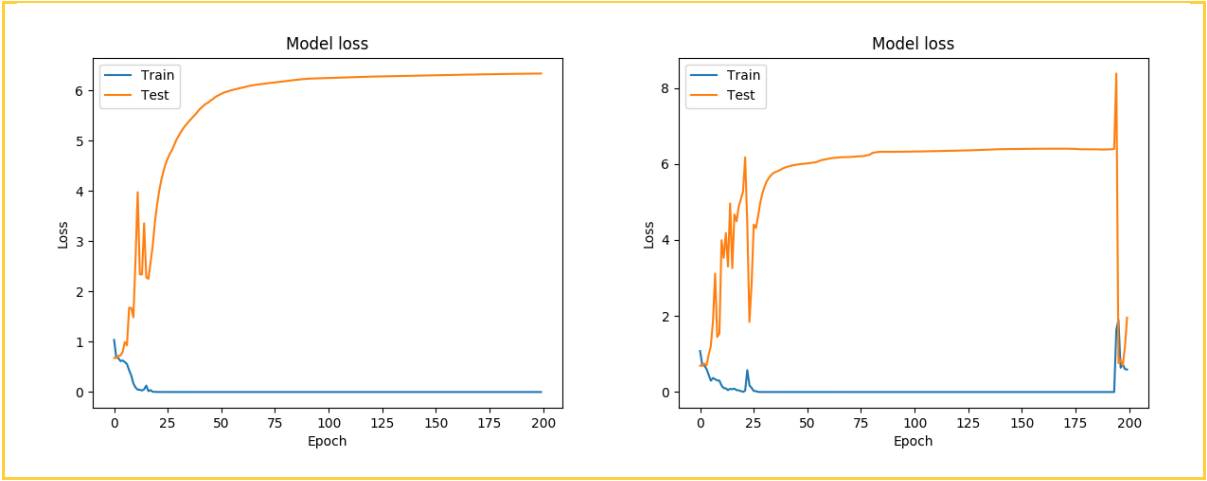
```
CNN accuracy: 0.92      [1.   1.   1.   1.   1.   1.   1.   0.95 0.57 0.67]
```

4.1.11 Prueba 4: DL 400

RED NEURONAL 1	RED NEURONAL 2







RED NEURONAL 1

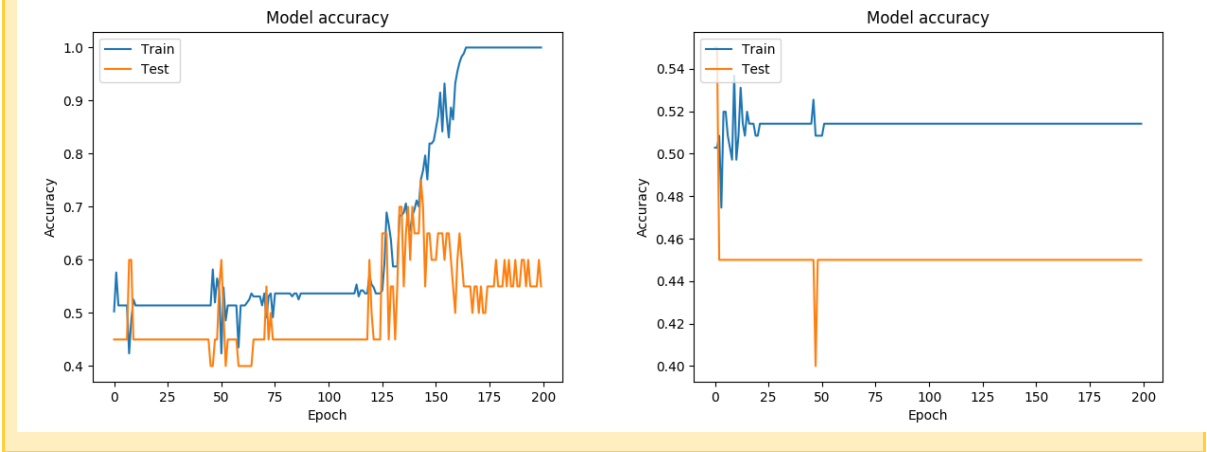
```
CNN accuracy: 0.89      [1.   1.   1.   1.   1.   1.   1.   1.   0.33 0.57]
```

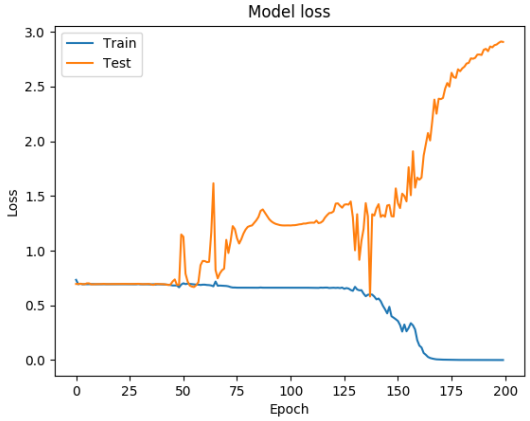
RED NEURONAL 2

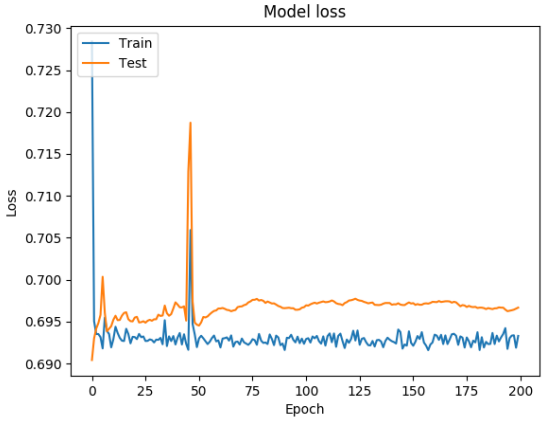
```
CNN accuracy: 0.89      [1.   1.   1.   1.   1.   1.   1.   1.   0.33 0.57]
```

4.1.12 Prueba 5: DL conjunto 121 duplicadas

RED NEURONAL 1	RED NEURONAL 2







RED NEURONAL 1

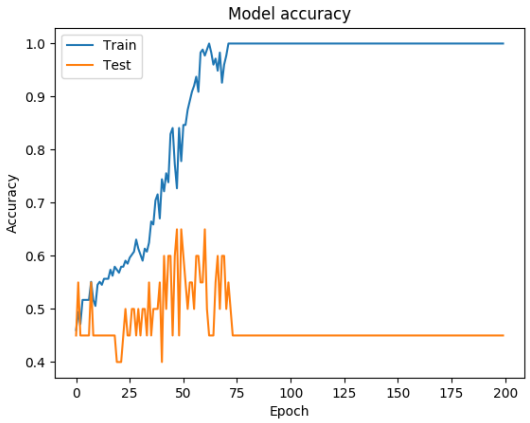
```
CNN accuracy: 0.94      [1.   1.   1.   1.   1.   1.   1.   0.95 0.71 0.76]
```

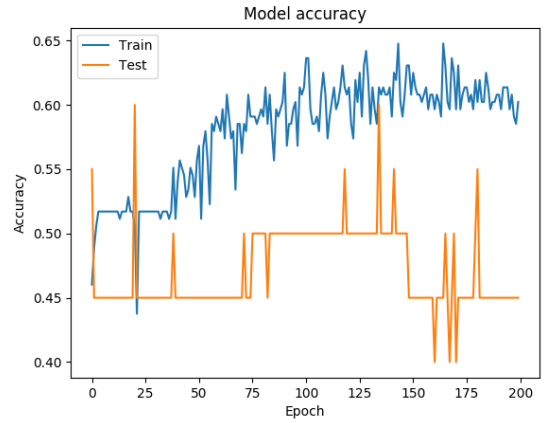
RED NEURONAL 2

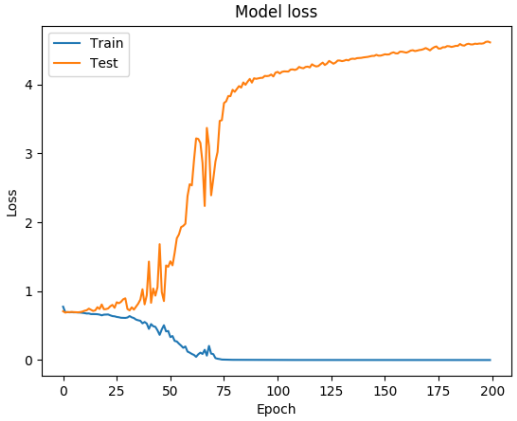
```
CNN accuracy: 0.94      [1.   1.   1.   1.   1.   1.   1.   0.95 0.71 0.76]
```

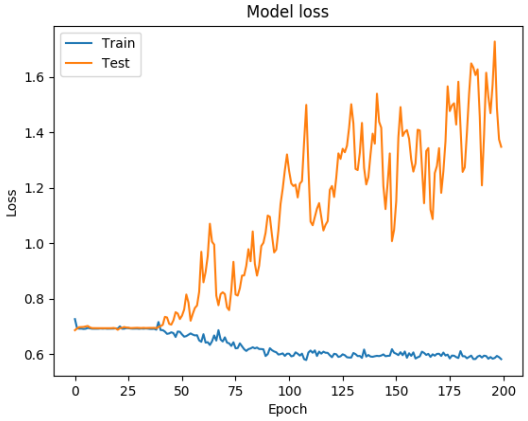
4.1.13 Prueba 6: DL conjunto 121 negro

RED NEURONAL 1	RED NEURONAL 2









RED NEURONAL 1

```
CNN accuracy: 0.85      [0.96 0.86 1.   0.95 0.95 0.91 0.91 0.86 0.57 0.52]
```

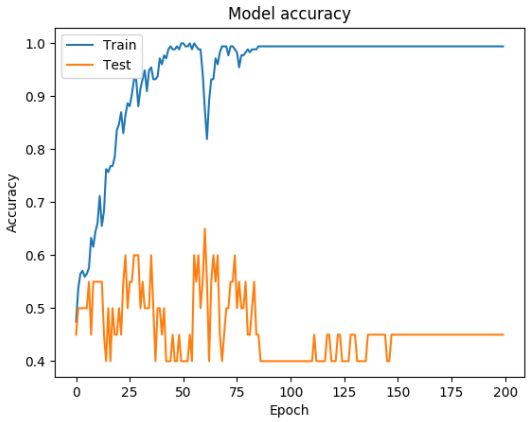
RED NEURONAL 2

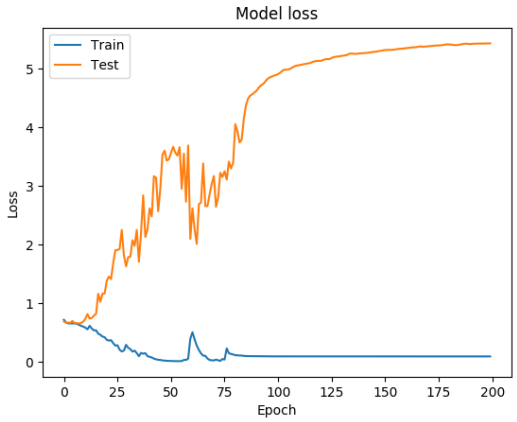
```
CNN accuracy: 0.85      [0.96 0.86 1.   0.95 0.95 0.91 0.91 0.86 0.57 0.52]
```

4.1.14 Prueba 7: DL 25%

4.1.14.1 Primer intervalo

RED NEURONAL 1



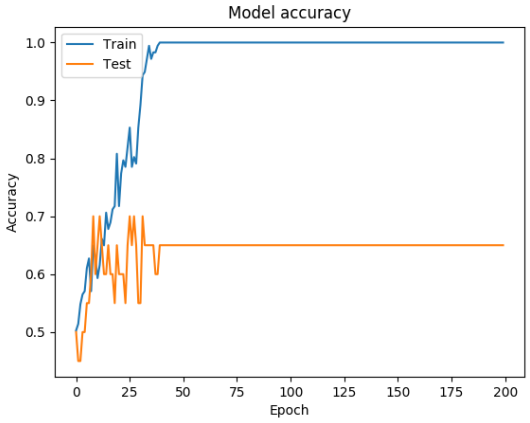


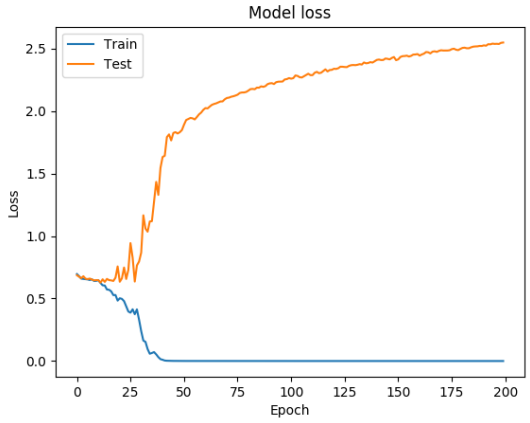
RED NEURONAL 1

```
CNN accuracy: 0.90      [1. | 1.   1.   1.   1.   1.   1.   0.95 0.67 0.43]
```

4.1.14.2 Segundo intervalo

RED NEURONAL 1



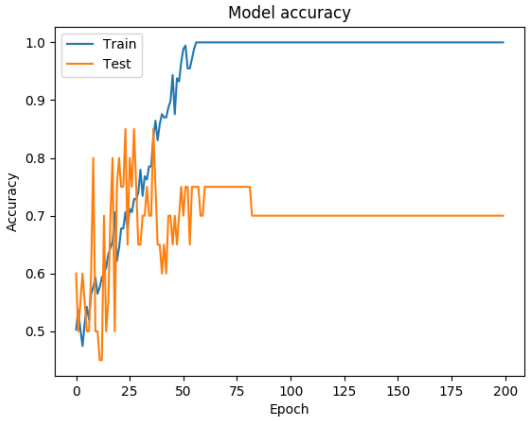


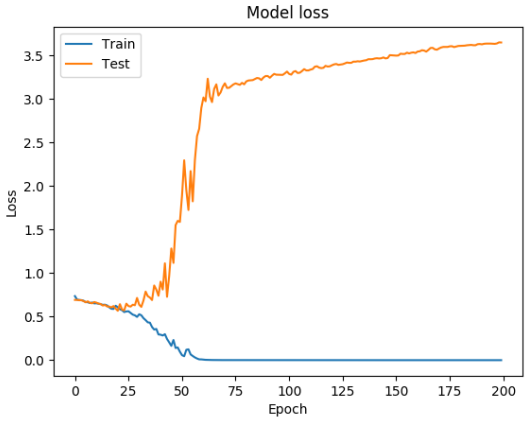
RED NEURONAL 1

```
CNN accuracy: 0.92      [1.   1.   1.   1.   1.   1.   1.   0.95 0.57 0.71]
```

4.1.14.3 Tercer intervalo

RED NEURONAL 1



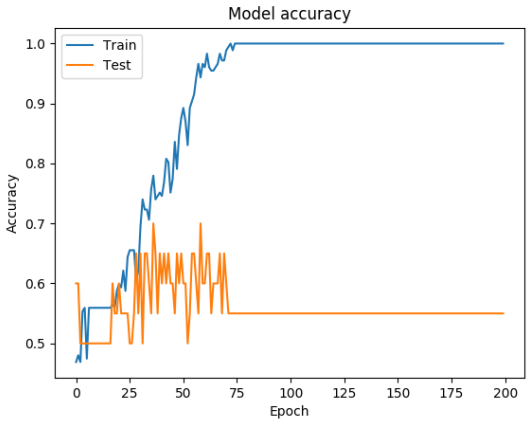


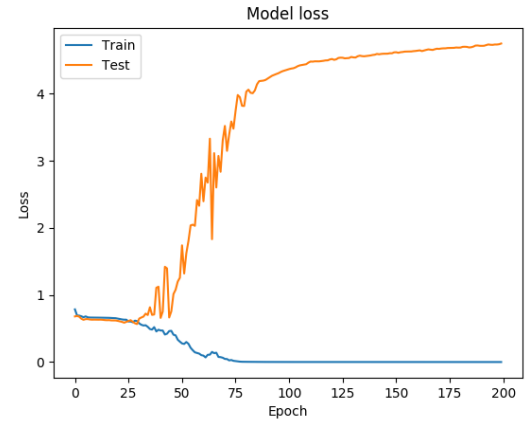
RED NEURONAL 1

```
CNN accuracy: 0.91      [1.   1.   1.   1.   1.   1.   1.   0.95 0.57 0.57]
```

4.1.14.4 Cuarto intervalo

RED NEURONAL 1





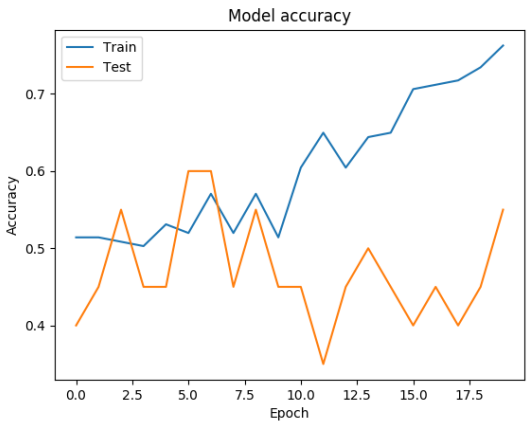
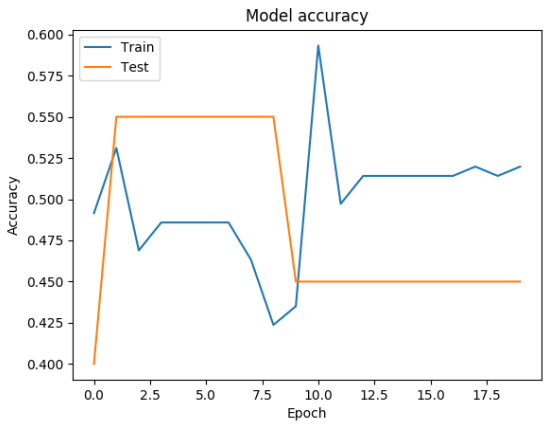
RED NEURONAL 1

```
CNN accuracy: 0.93      [1.   1.   1.   1.   1.   1.   1.   1.   0.71 0.57]
```

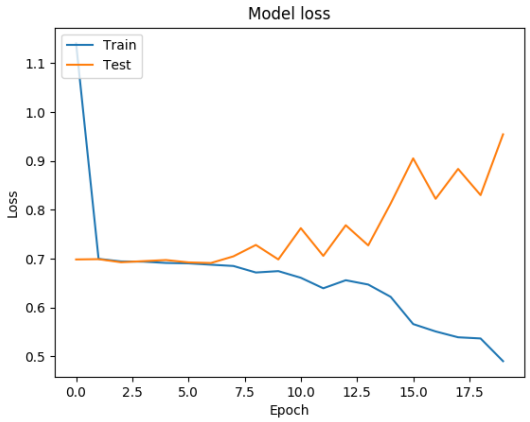
Observando las gráficas que representan la pérdida podemos comprobar que la red neuronal se está sobreentrenando, ya que a medida que nos desplazamos en el eje x, la pérdida en el entrenamiento disminuye y, en cambio, durante la validación, aumenta.

Por lo tanto, se decide entrenar la red neuronal hasta el número de épocas en el cual las dos curvas se separan para cada prueba realizada anteriormente.

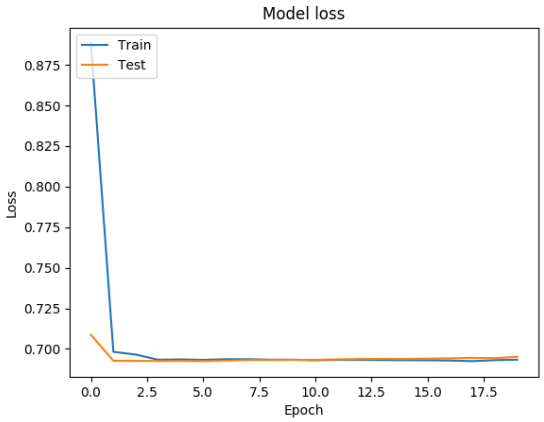
4.1.15 Prueba 1: DL 19

RED NEURONAL 1 20 épocas	RED NEURONAL 2 20 épocas
	

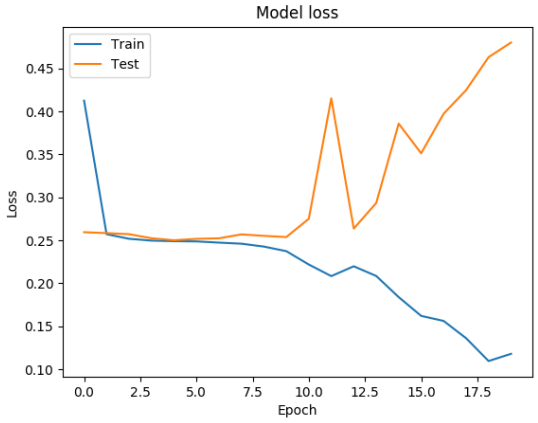
1: Entropía cruzada



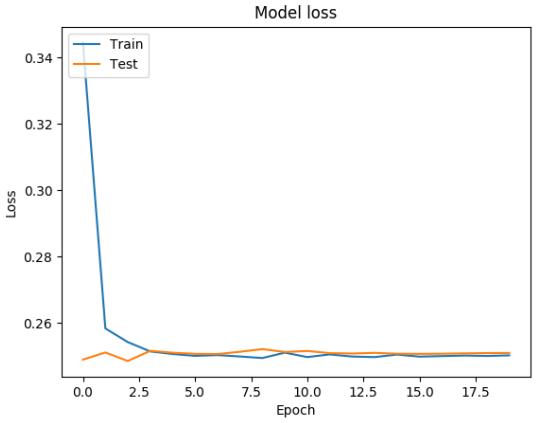
2:Entropía cruzada



3: Error cuadrático



4: Error cuadrático



RED NEURONAL 1

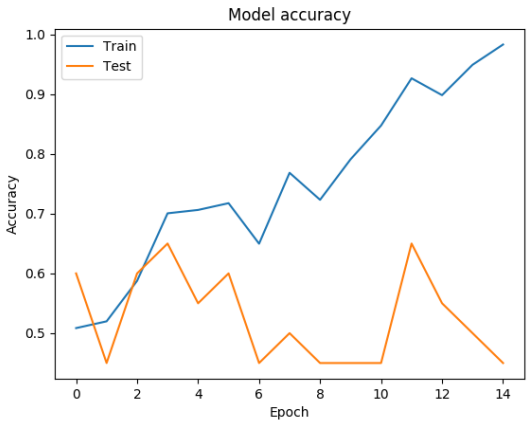
```
CNN accuracy: 0.90      [1.   1.   1.   1.   1.   1.   1.   0.9  0.48 0.57]
```

RED NEURONAL 2

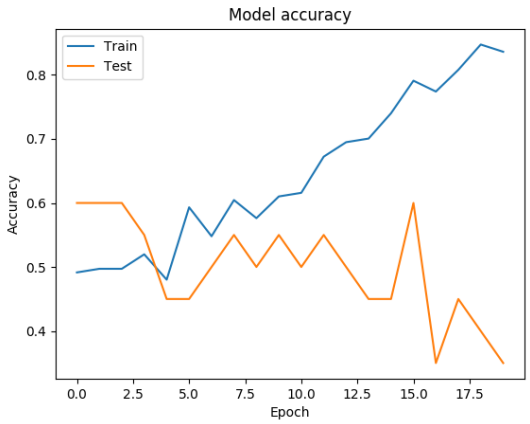
```
CNN accuracy: 0.90      [1.   1.   1.   1.   1.   1.   1.   0.9  0.48 0.57]
```

4.1.16 Prueba 2: DL conjunto 400 duplicadas

RED NEURONAL 1	RED NEURONAL 2
15 épocas	**20 épocas**

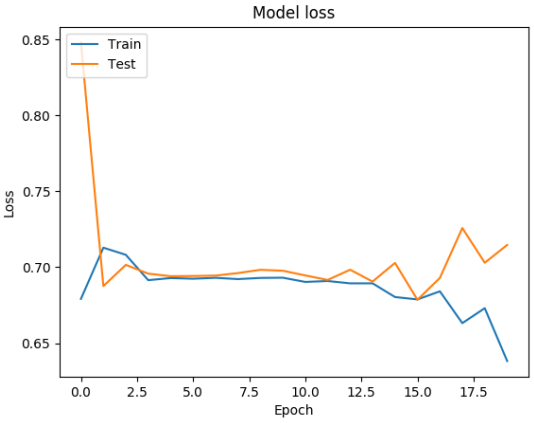


5: Entropía cruzada

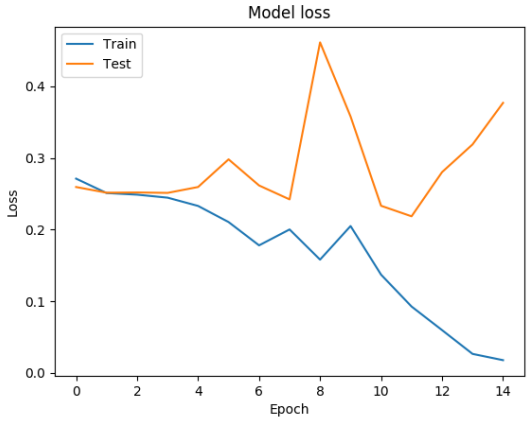


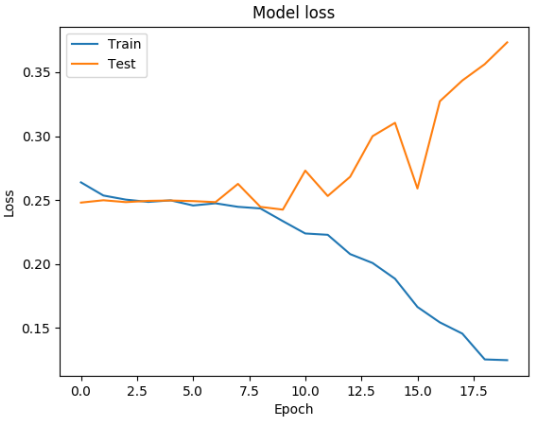
6: Entropía cruzada

7: Error cuadrático



8: Error cuadrático





RED NEURONAL 1

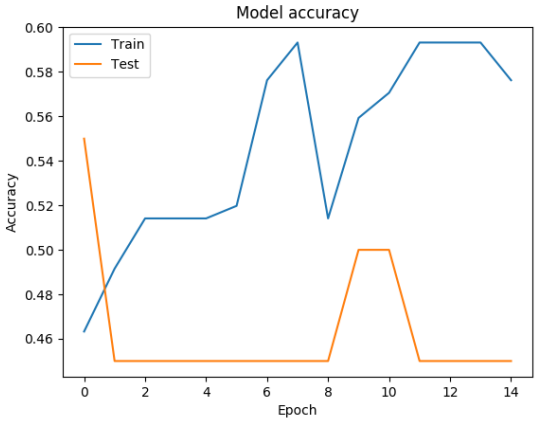
```
CNN accuracy: 0.89      [1.   1.   1.   1.   1.   1.   1.   0.95 0.48 0.48]
```

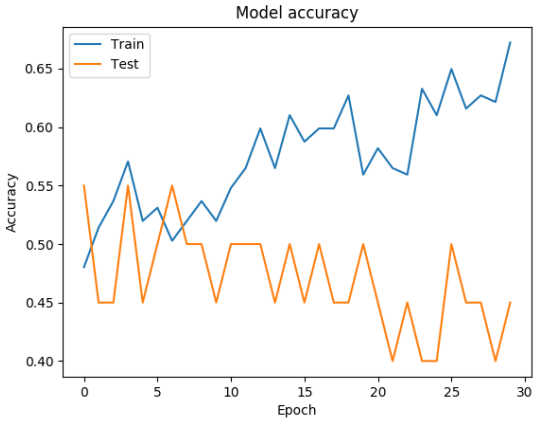
RED NEURONAL 2

```
CNN accuracy: 0.89      [1.   1.   1.   1.   1.   1.   1.   0.95 0.48 0.48]
```

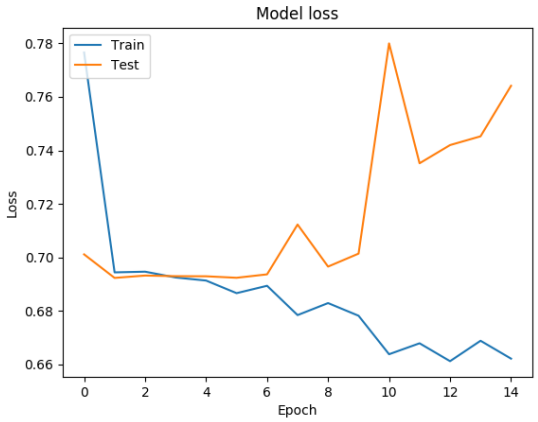
4.1.17 Prueba3: DL conjunto 400 negro

RED NEURONAL 1	RED NEURONAL 2
15 épocas	30 épocas

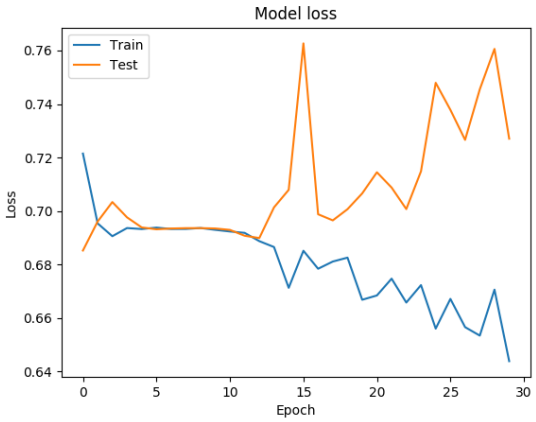




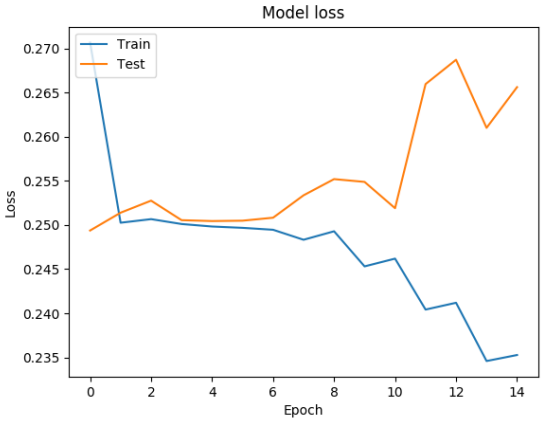
9: Entropía cruzada



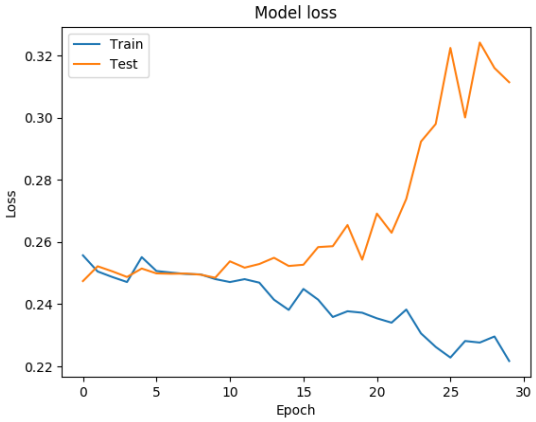
10: Entropía cruzada



11: Error cuadrático



12: Error cuadrático



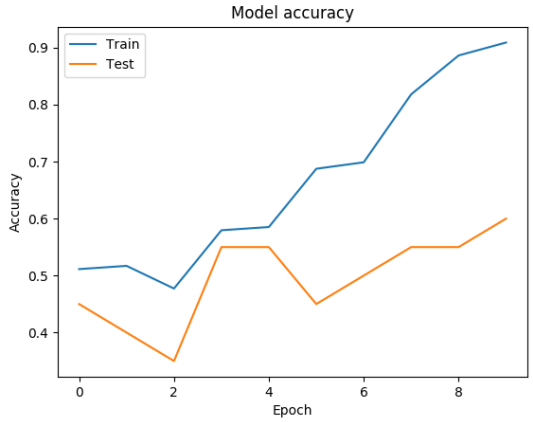
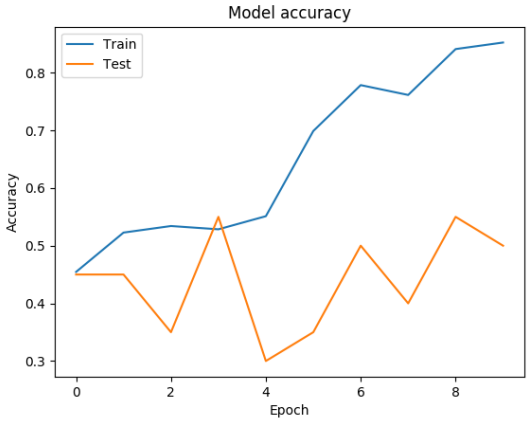
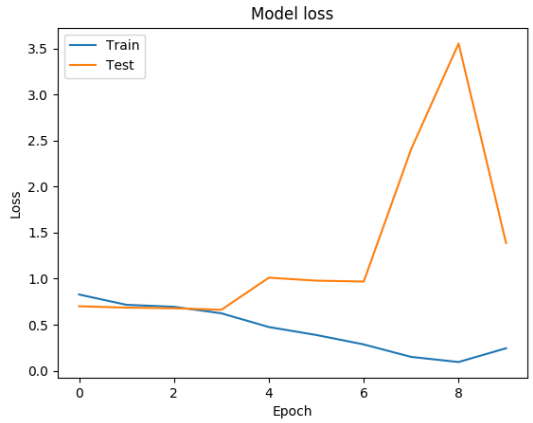
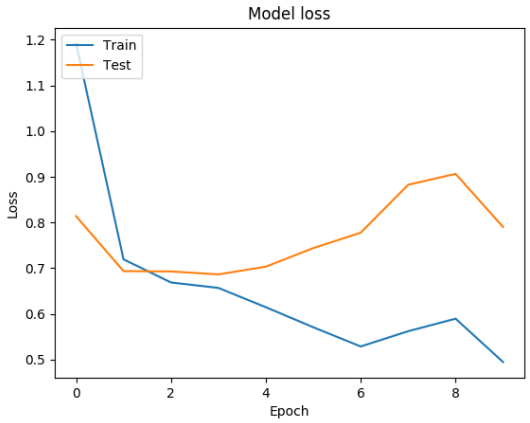
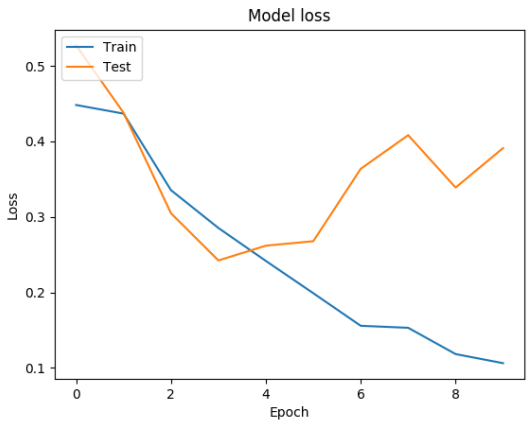
RED NEURONAL 1

```
CNN accuracy: 0.92      [1.   1.   1.   1.   1.   1.   1.   0.95 0.57 0.67]
```

RED NEURONAL 2

```
CNN accuracy: 0.92      [1.   1.   1.   1.   1.   1.   1.   0.95 0.57 0.67]
```

4.1.18 Prueba 4: DL 400

RED NEURONAL 1 10 épocas	RED NEURONAL 2 10 épocas
	
13: Entropía cruzada	*14: Entropía cruzada*
	
15: Error cuadrático	*16: Error cuadrático*
	

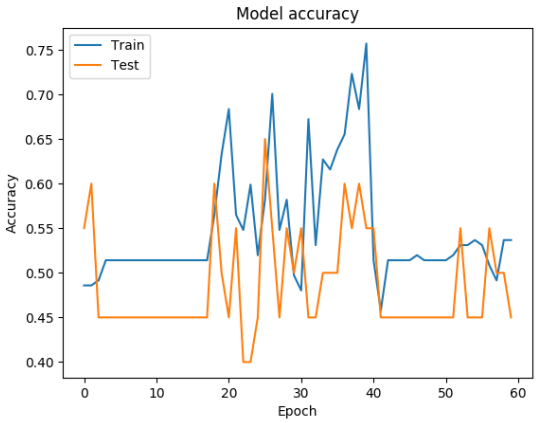
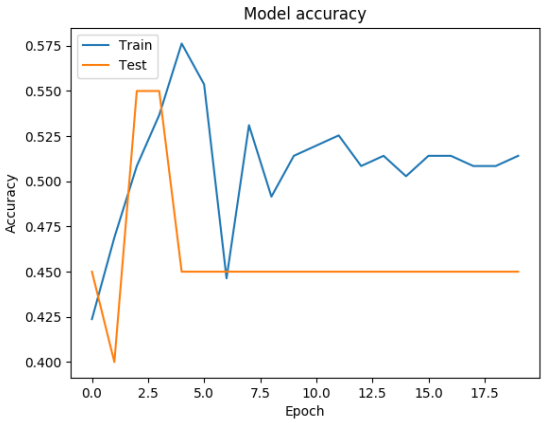
RED NEURONAL 1

```
CNN accuracy: 0.89      [1.   1.   1.   1.   1.   1.   1.   1.   0.33 0.57]
```

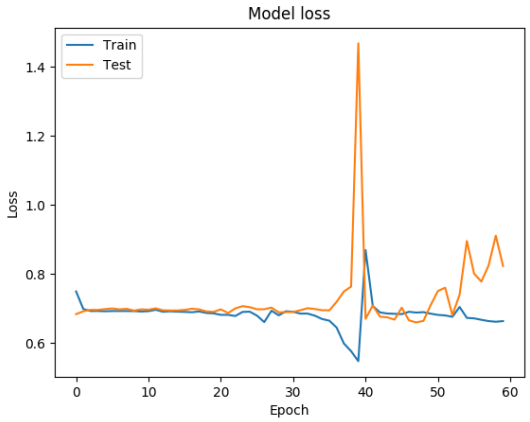
RED NEURONAL 2

```
CNN accuracy: 0.89      [1.   1.   1.   1.   1.   1.   1.   1.   0.33 0.57]
```

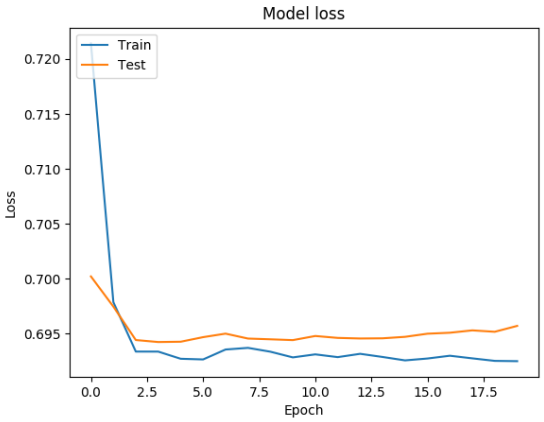
4.1.19 Prueba 5: DL conjunto 121 duplicadas

RED NEURONAL 1 60 épocas	RED NEURONAL 2 20 épocas
	

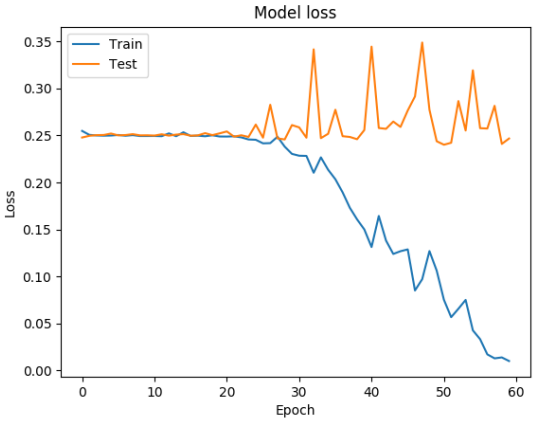
17: Entropía cruzada



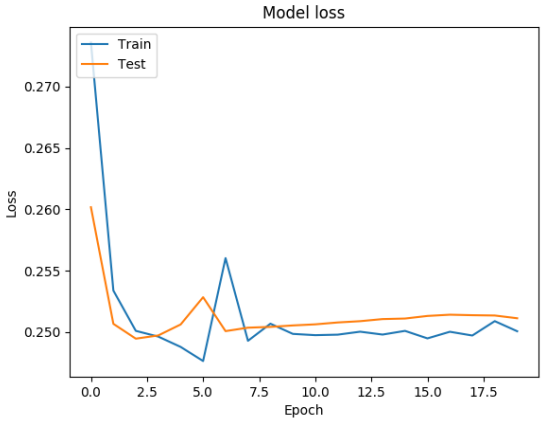
18: Entropía cruzada



19: Error cuadrático



20: Error cuadrático



RED NEURONAL 1

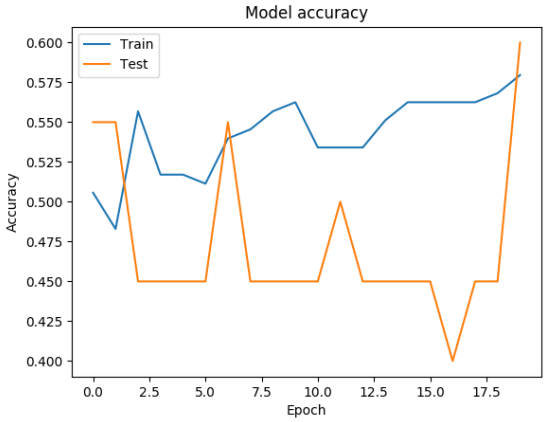
```
CNN accuracy: 0.94      [1.   1.   1.   1.   1.   1.   1.   0.95 0.71 0.76]
```

RED NEURONAL 2

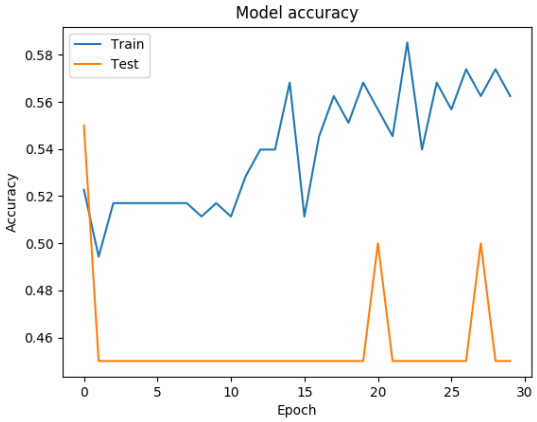
```
CNN accuracy: 0.94      [1.   1.   1.   1.   1.   1.   1.   0.95 0.71 0.76]
```

4.1.20 Prueba 6: DL conjunto 121 negro

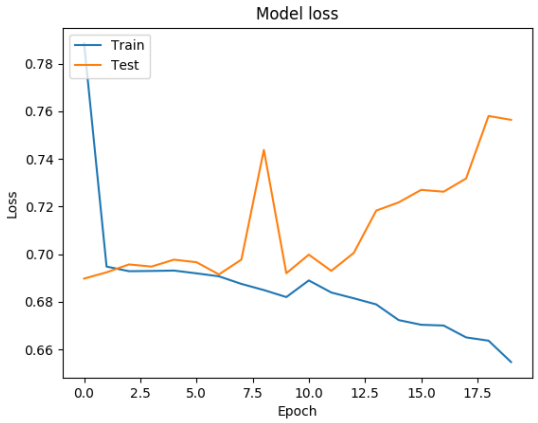
RED NEURONAL 1	RED NEURONAL 2
20 épocas	30 épocas



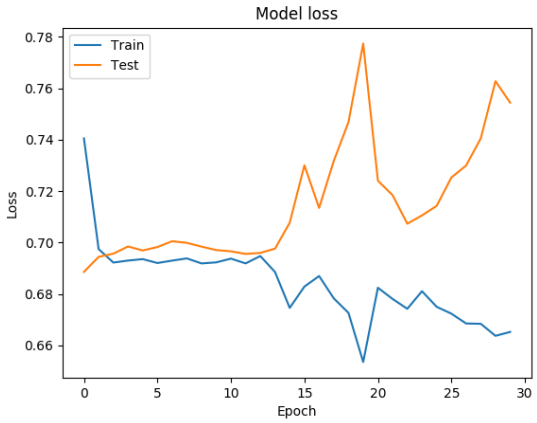
21: Entropía cruzada



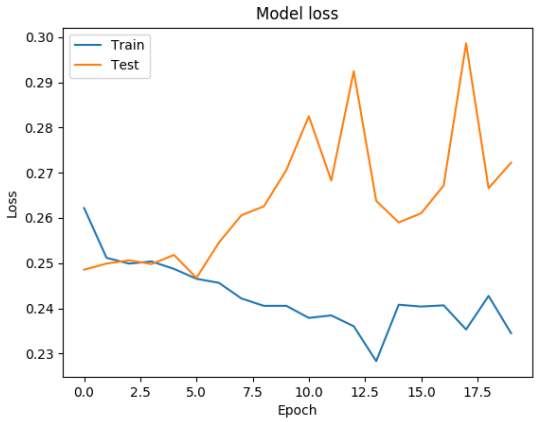
22: Entropía cruzada

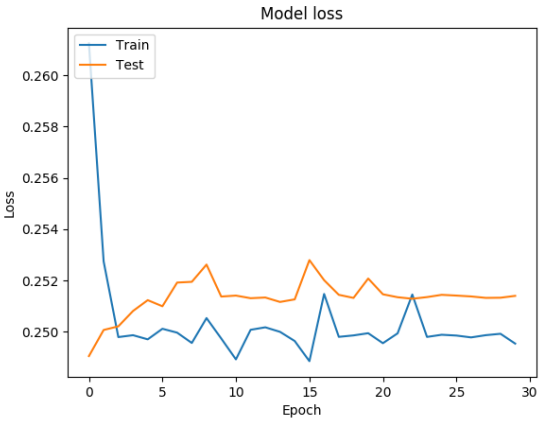


23: Error cuadrático



24: Error cuadrático





RED NEURONAL 1

```
CNN accuracy: 0.85       [0.96 0.86 1.   0.95 0.95 0.91 0.91 0.86 0.57 0.52]
```

RED NEURONAL 2

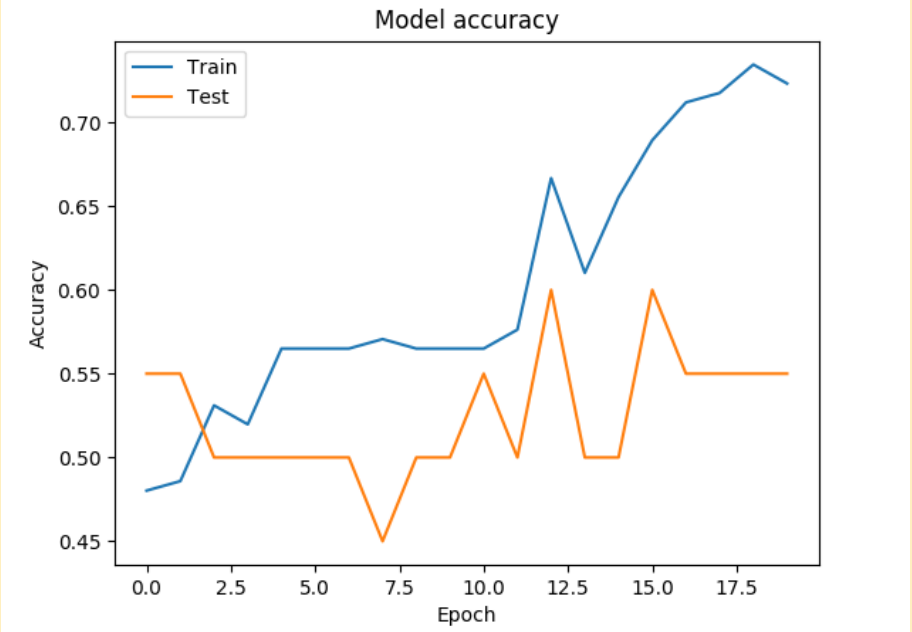
```
CNN accuracy: 0.85       [0.96 0.86 1.   0.95 0.95 0.91 0.91 0.86 0.57 0.52]
```

4.1.21 Prueba 7: DL 25%

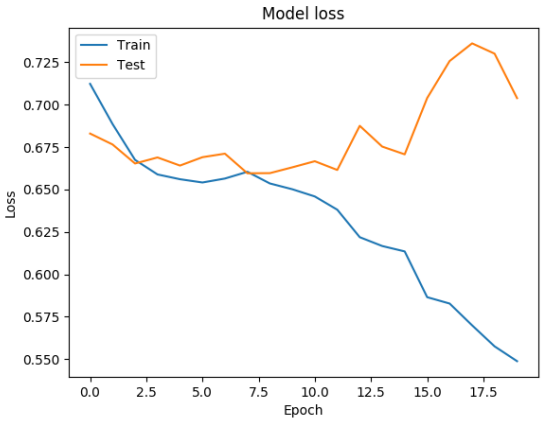
4.1.21.1 Primer intervalo

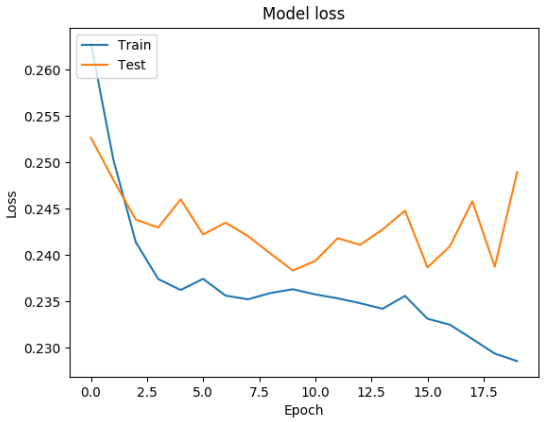
RED NEURONAL 1

20 épocas



25: Entropía cruzada *26: Error cuadrático*





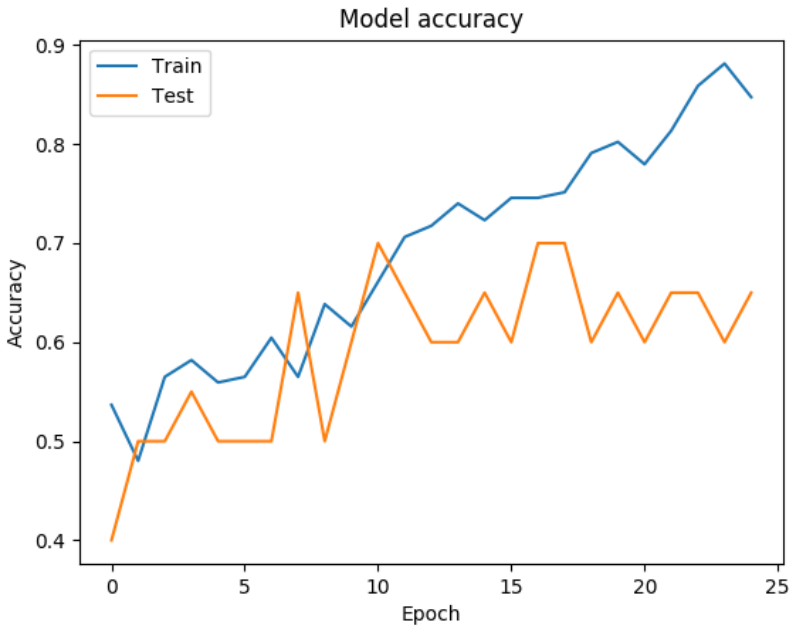
RED NEURONAL 1

```
CNN accuracy: 0.90      [1. | 1.   1.   1.   1.   1.   1.   0.95 0.67 0.43]
```

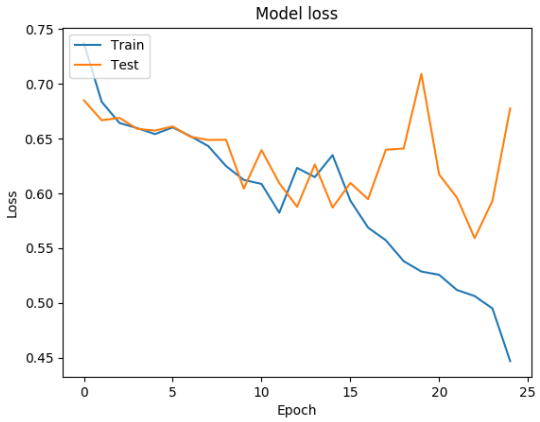
4.1.21.2 Segundo intervalo

RED NEURONAL 1

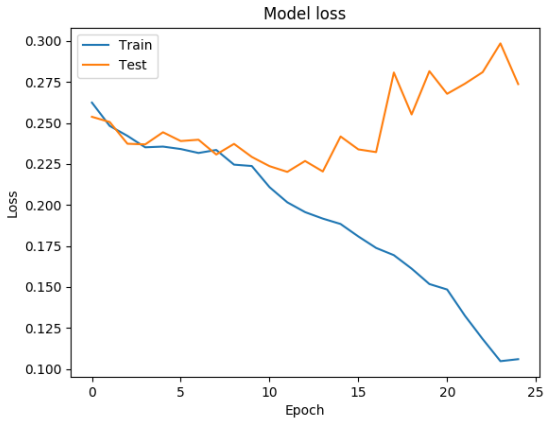
25 épocas



27: Entropía cruzada



28: Error cuadrático



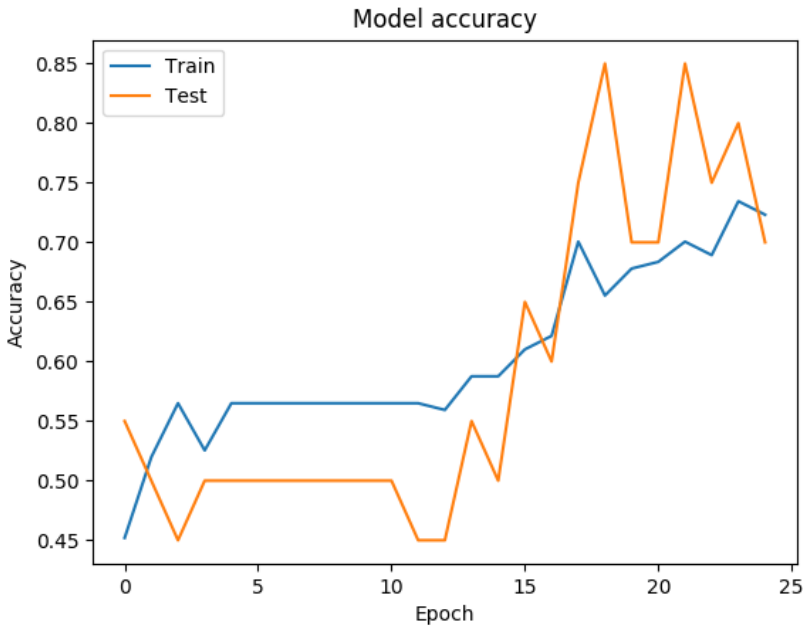
RED NEURONAL 1

```
CNN accuracy: 0.92      [1.   1.   1.   1.   1.   1.   1.   0.95 0.57 0.71]
```

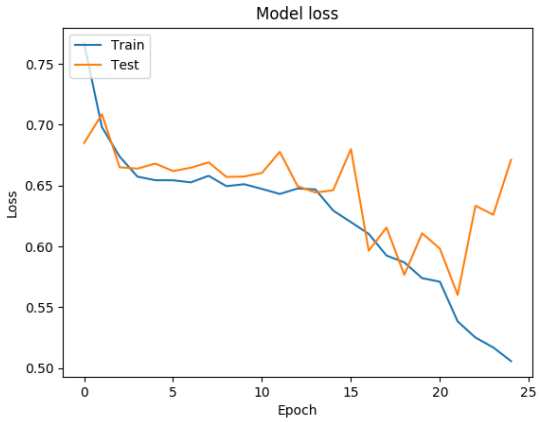
4.1.21.3 Tercer intervalo

RED NEURONAL 1

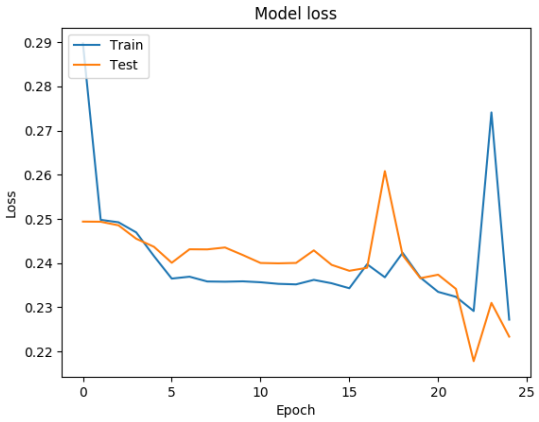
25 épocas



29: Entropía cruzada



30: Error cuadrático



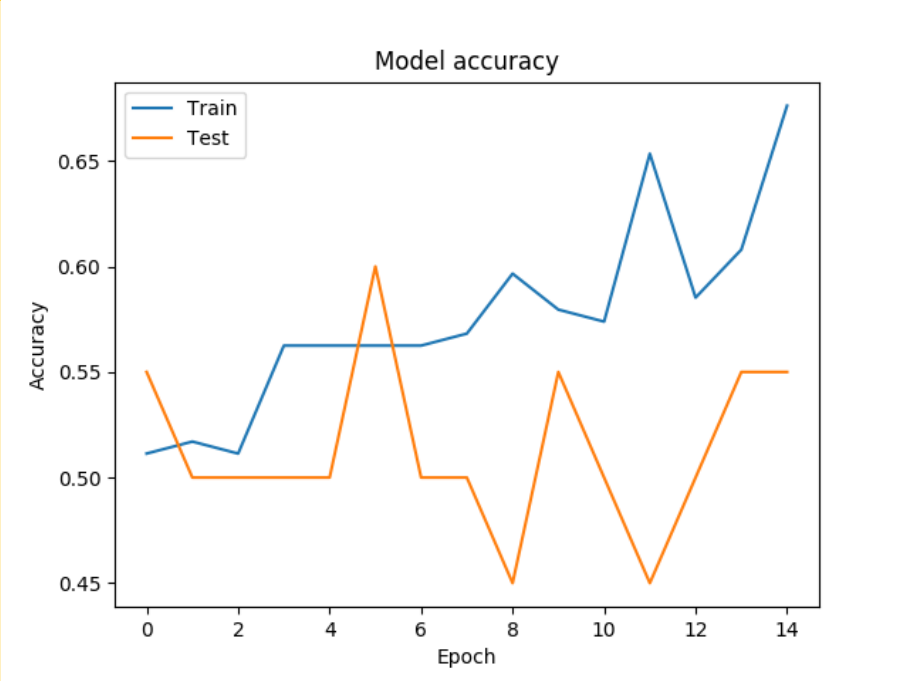
RED NEURONAL 1

```
CNN accuracy: 0.91      [1.   1.   1.   1.   1.   1.   1.   0.95 0.57 0.57]
```

4.1.21.4 Cuarto intervalo

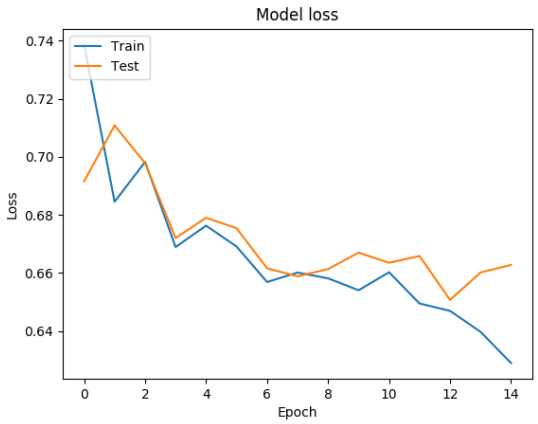
RED NEURONAL 1

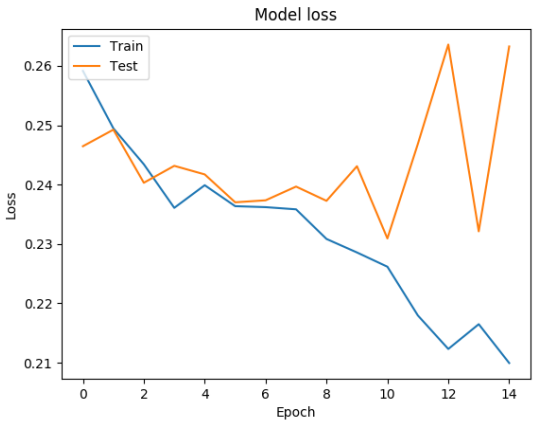
15 épocas



*31: **Entropía cruzada***

32: Error cuadrático





RED NEURONAL 1

```
CNN accuracy: 0.93      [1.   1.   1.   1.   1.   1.   1.   1.   0.71 0.57]
```

Los resultados de *accuracy* fueron exactamente los mismos a los obtenidos anteriormente con un entrenamiento de dos épocas. Esto significaría que a partir de las épocas elegidas en esta última prueba el aprendizaje no mejora, por tanto el resultado al aumentar el número de épocas se mantendría igual.

Para evaluar mejor la función de pérdida, he utilizado dos funciones diferentes: *entropía cruzada* y *error cuadrático*. La primera de ellas se suele utilizar en problemas de clasificación, como sería este caso, y la segunda en problemas de regresión. Si comparamos ambas gráficas se puede comprobar que son bastante similares en la mayoría de los casos, esto nos hace pensar que el hecho de que las curvas de aprendizaje y entrenamiento se deba a que la red escogida no sea la más adecuada para este tipo de problema.

4.2 Optical Flow y ADV

Los resultados oficiales obtenidos en el CLEF 2019 fueron bastante prometedores ya que se acercaron al sistema con mayor acierto presentado. Dichos resultados se pueden ver en la siguiente figura:

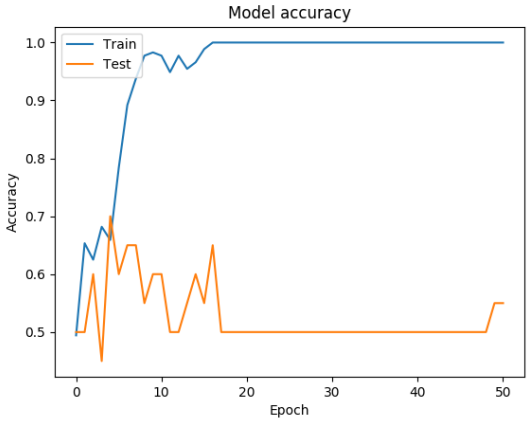
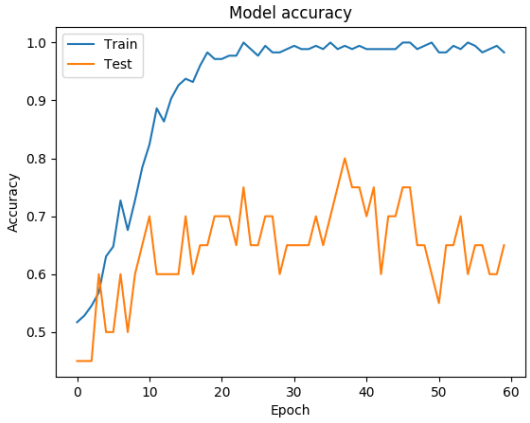
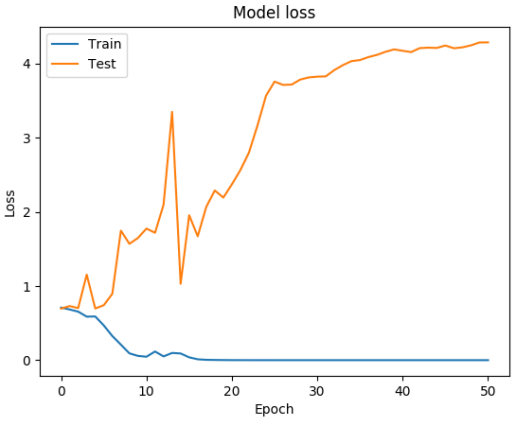
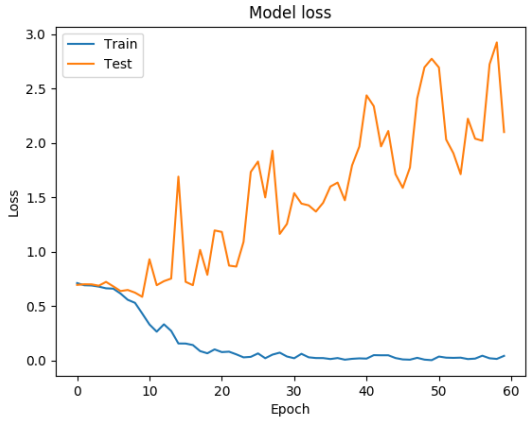
Run	AUC	ACC	Rank
UIIP_BioMed	0.7877	0.7179	1
SVR-SVM-axis-mode-4.txt	0.7013	0.7009	12
SVR-MC	0.7003	0.7009	14
SVR-LDA-axis-mode-4.txt	0.6842	0.6838	18
SVR-SVM-axis-svm-4.txt	0.6761	0.6752	20
SVR-LDA-axis-svm-4.txt	0.6499	0.6496	23

Figura 76: Tabla con los resultados obtenidos VS el mejor modelo presentado

Para el segundo modelo he intentado la clasificación con las 3 redes neuronales mencionadas anteriormente.

4.2.1 Red Neuronal 1 y 2

4.2.1.1 LRF normalizado

RED NEURONAL 1	RED NEURONAL 2
	
	

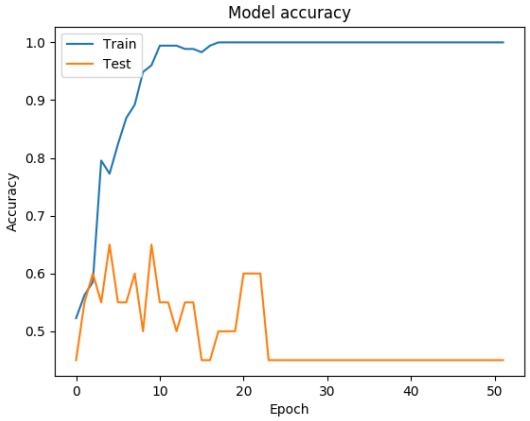
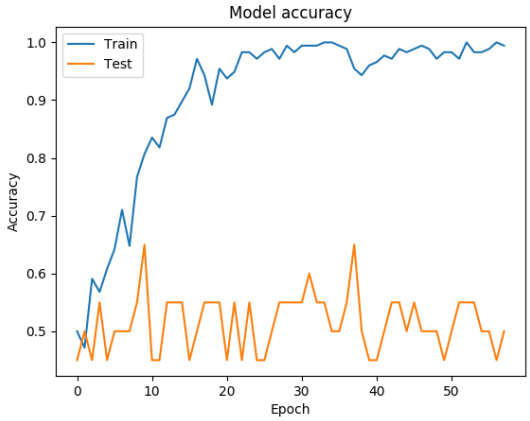
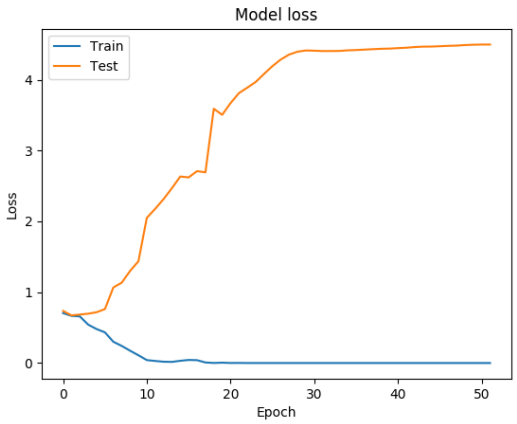
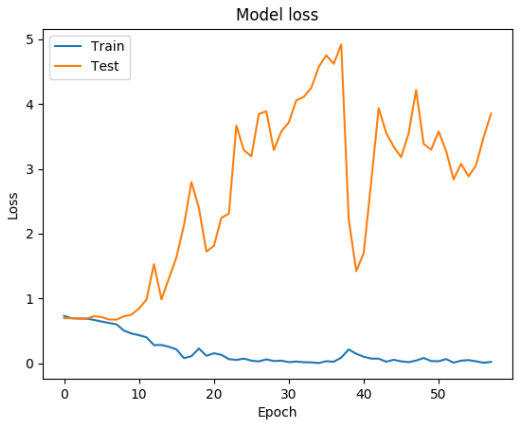
RED NEURONAL 1

```
CNN accuracy: 0.58      [0.52 0.77 0.55 0.59 0.59 0.55 0.64 0.48 0.48 0.67]
```

RED NEURONAL 2

```
CNN accuracy: 0.55      [0.57 0.73 0.55 0.32 0.64 0.64 0.5  0.52 0.48 0.57]
```

4.2.1.2 UDF normalizado

RED NEURONAL 1	RED NEURONAL 2
	
	

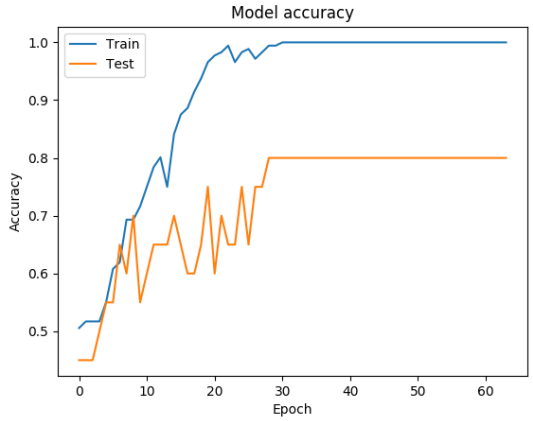
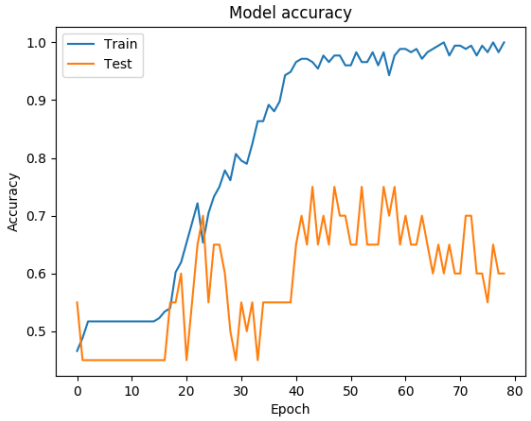
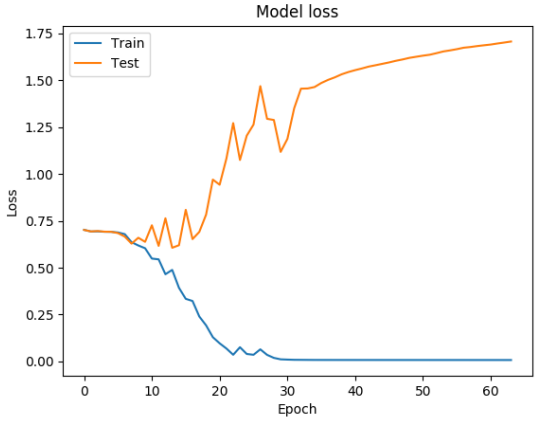
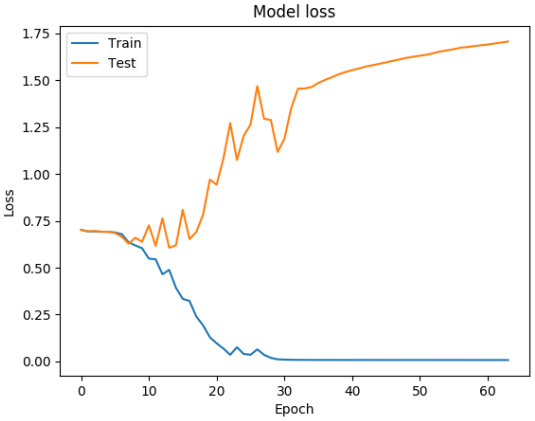
RED NEURONAL 1

```
CNN accuracy: 0.49      [0.39 0.45 0.45 0.41 0.5  0.5  0.55 0.57 0.57 0.48]
```

RED NEURONAL 2

```
CNN accuracy: 0.53      [0.7  0.45 0.45 0.36 0.64 0.68 0.5  0.48 0.57 0.48]
```

4.2.1.3 LRF

RED NEURONAL 1	RED NEURONAL 2
	
	

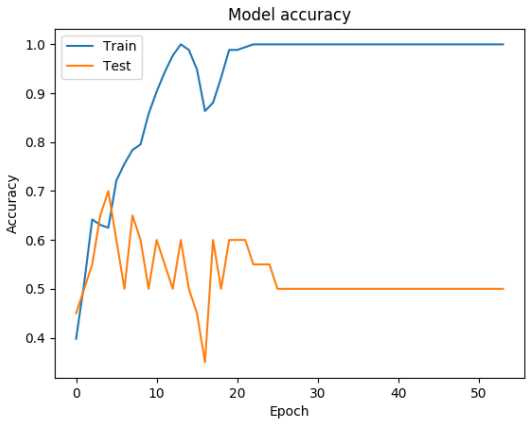
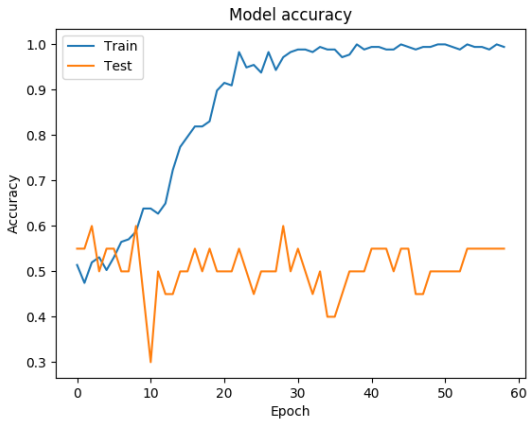
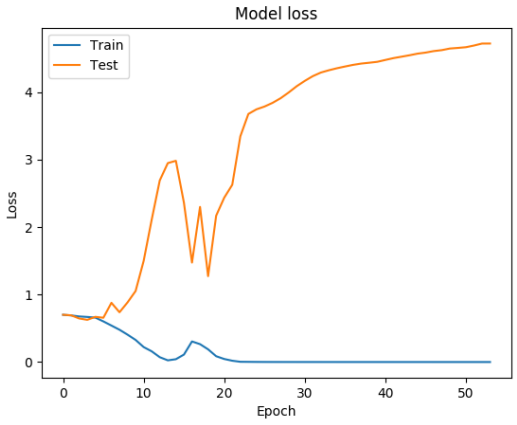
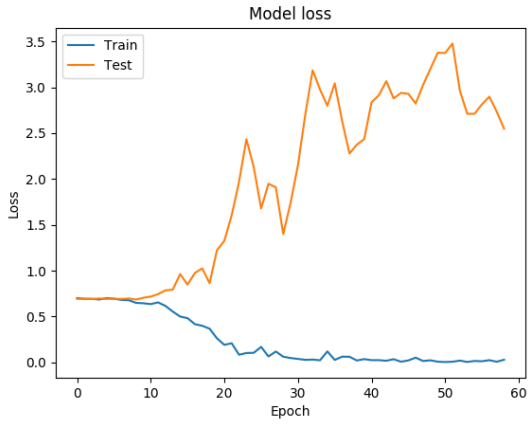
RED NEURONAL 1

```
CNN accuracy: 0.57      [0.39 0.59 0.41 0.64 0.5  0.68 0.5  0.62 0.67 0.67]
```

RED NEURONAL 2

```
CNN accuracy: 0.53      [0.48 0.5  0.36 0.68 0.5  0.5  0.5  0.62 0.57 0.62]
```

4.2.1.4 UDF

RED NEURONAL 1	RED NEURONAL 2
	
	

RED NEURONAL 1

```
CNN accuracy: 0.45      [0.48 0.32 0.27 0.41 0.55 0.5  0.45 0.48 0.52 0.52]
```

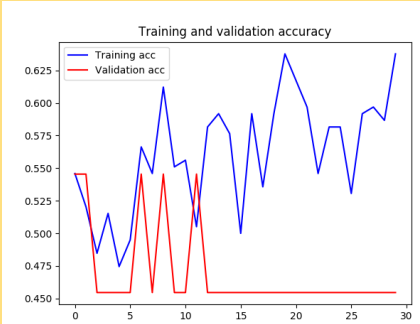
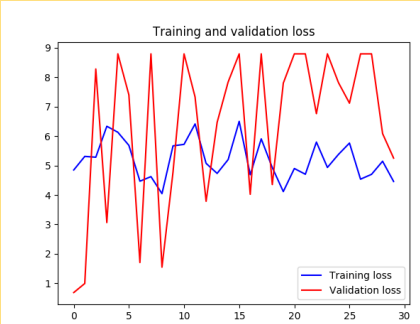
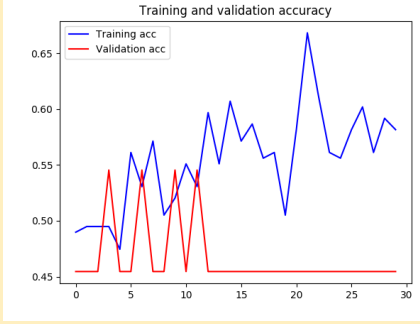
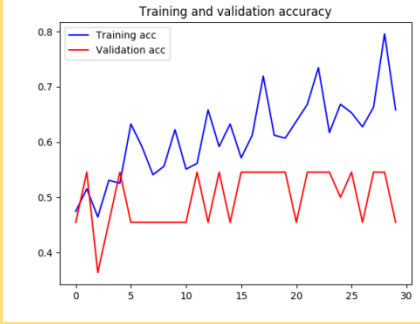
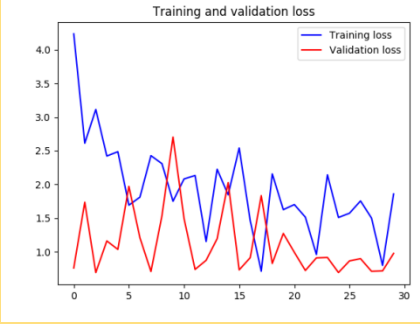
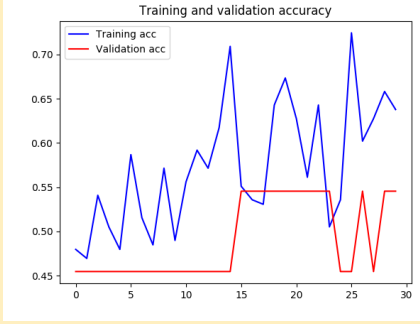
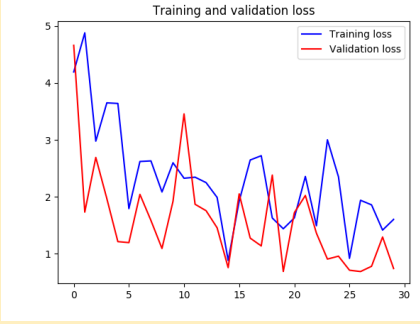
RED NEURONAL 2

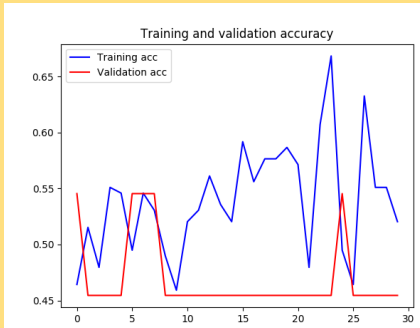
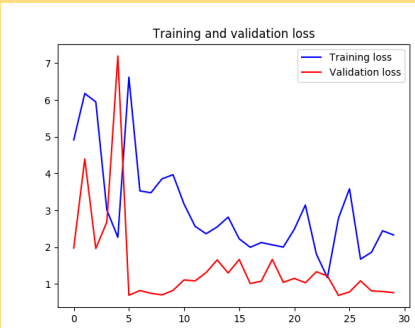
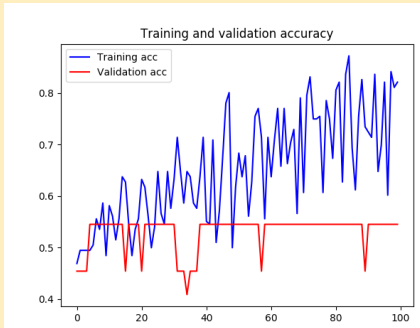
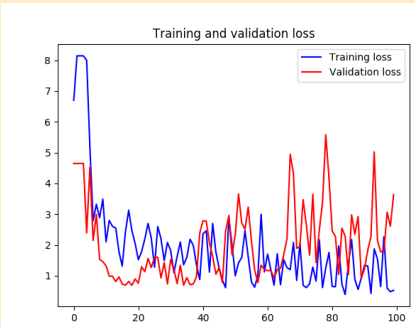
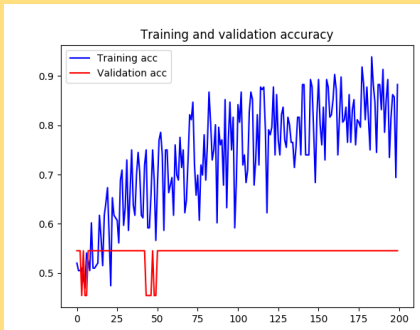
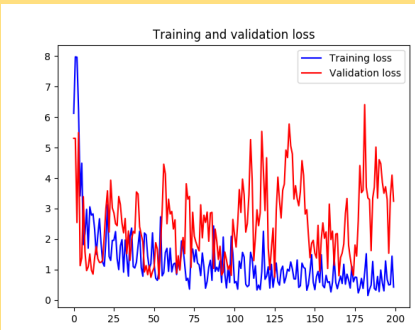
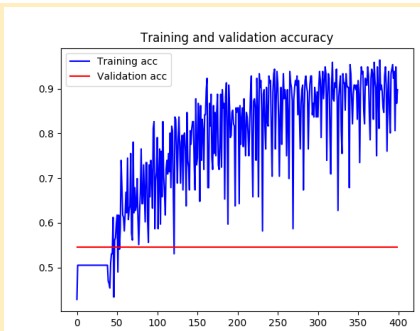
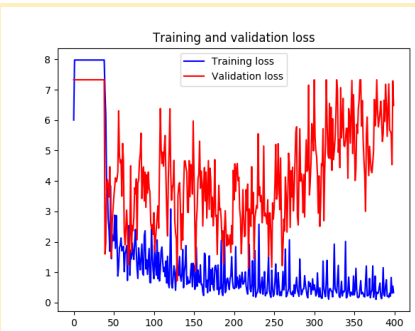
```
CNN accuracy: 0.47      [0.65 0.5  0.41 0.41 0.5  0.55 0.45 0.38 0.48 0.33]
```

Como se puede apreciar, a la hora de asignarle un grado de severidad a la tuberculosis de un determinado paciente es más eficiente usar la matriz LRF que UDF. El hecho de si dicha matriz está normalizada o no, parece no afectar notablemente a la clasificación.

4.2.2 ResNet50

4.2.2.1 LRF normalizado

	Accuracy	Loss
A **Imagen: 512x512** **Bach size training: 10** **Batch size test: 5** **Épocas: 30**		
B **Imagen: 512x512** **Bach size training: 30** **Batch size test: 5** **Épocas: 30**		
C **Imagen: 260x260** **Bach size training: 30** **Batch size test: 5** **Épocas: 30**		
D **Imagen: 260x260** **Bach size training: 50** **Batch size test: 10** **Épocas: 30**		

E **Imagen: 260x260** **Bach size training: 100** **Batch size test: 10** **Épocas: 30**		
F **Imagen: 260x260** **Bach size training: 50** **Batch size test: 10** **Épocas: 100**		
G **Imagen: 260x260** **Bach size training: 50** **Batch size test: 10** **Épocas: 200**		
H **Imagen: 260x260** **Bach size training: 50** **Batch size test: 10** **Épocas: 400**		

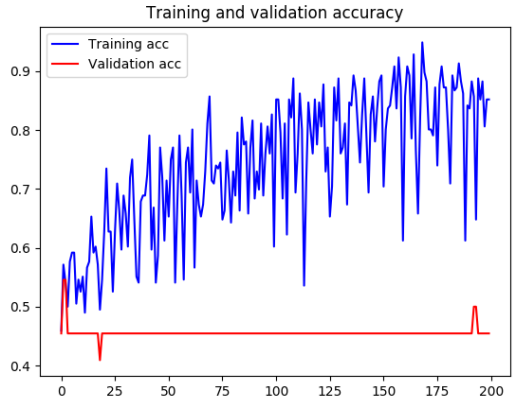
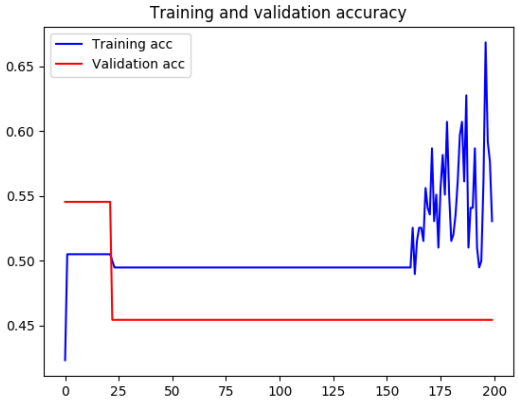
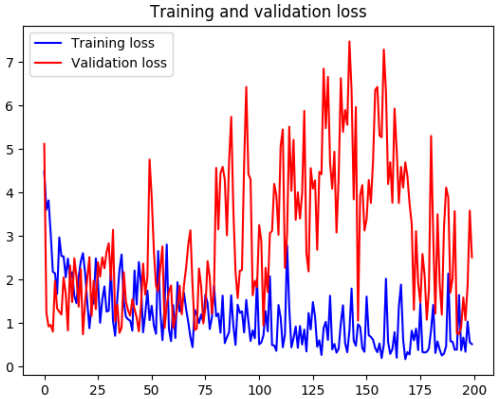
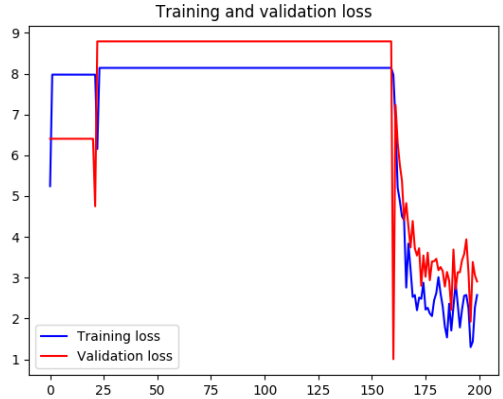
Analizando los resultados obtenidos se puede comprobar que cuanto menor es el batch_size escogido peor resulta la clasificación, pero aumentar demasiado el valor de este parámetro genera problemas de ejecución por falta de memoria y además tampoco permite obtener buenos resultados.

En cambio, la reducción de la imagen de entrada a la mitad, parEce no afectar al entrenamiento, permitiendo aumentar el batch_size sin que aparezcan errores de memoria.

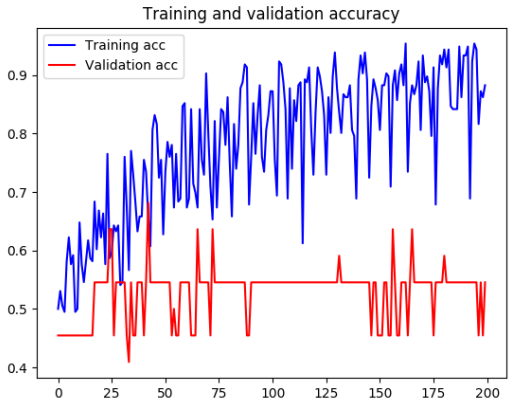
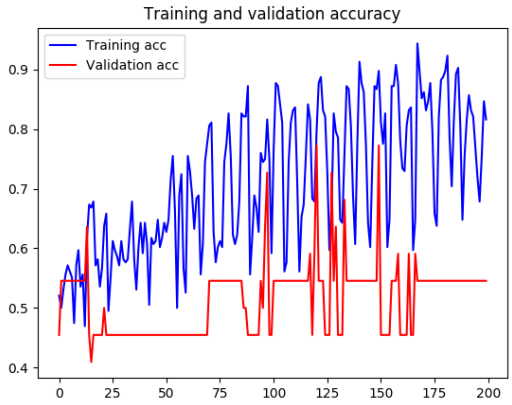
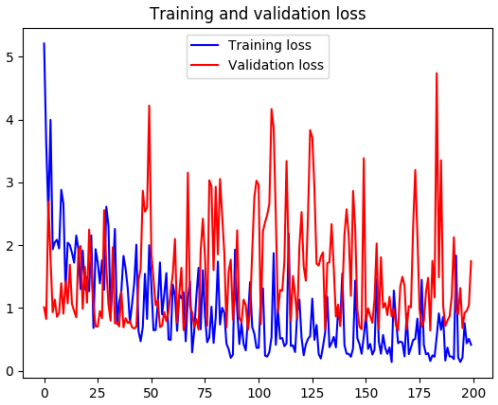
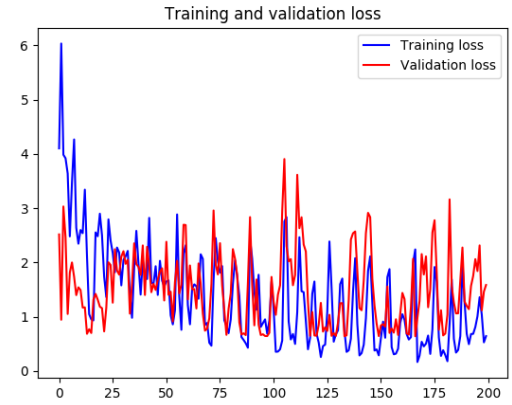
Por último, cabe añadir que a partir de 100 épocas, el valor de accuracy no mejora por mucho que se aumente el número de estas.

4.2.2.2 LRF

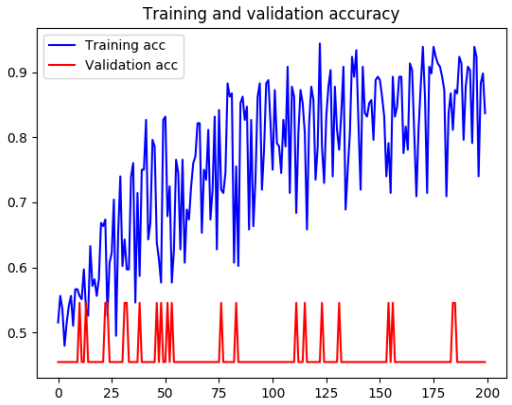
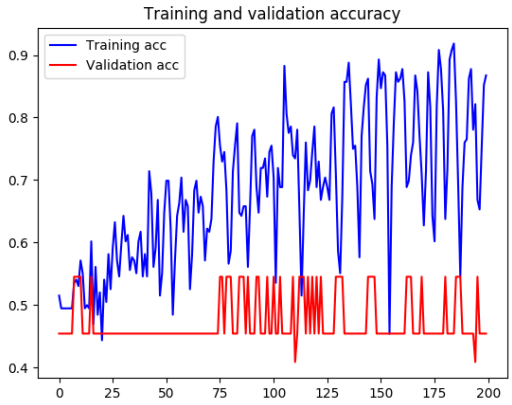
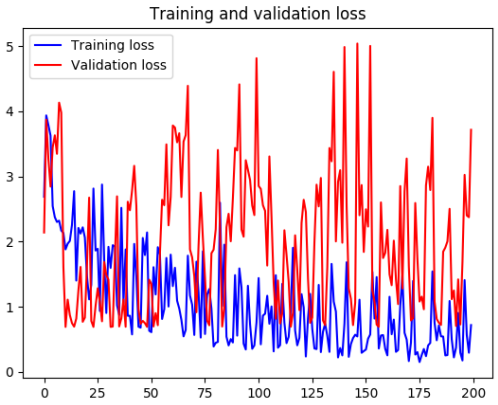
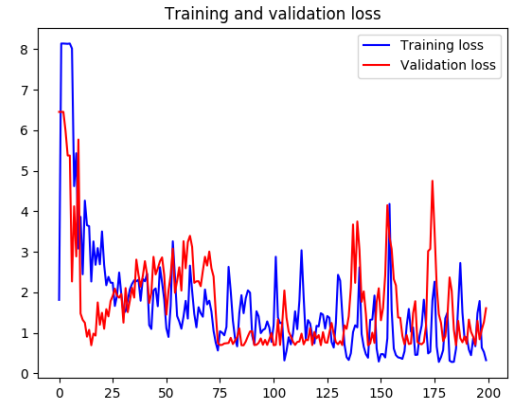
Basándome en los resultados anteriores realicé las pruebas con los parámetros con los que se obtenía las mejores clasificaciones, imagen 260x260 y 200 épocas.

G	I
Batch size entrenamiento : 50	**Batch size entrenamiento :100**
Batch size test: 10	**Batch size test: 10**
	
	

4.2.2.3 UDF normalizado

G	I
Batch size entrenamiento : 50 **Batch size test: 10**	**Batch size entrenamiento :100** **Batch size test: 10**
	
	

4.2.2.4 UDF

G	I
Batch size entrenamiento : 50	**Batch size entrenamiento :100**
Batch size test: 10	**Batch size test: 10**
	
	

A diferencia de las pruebas realizadas con la Red Neuronal 1 y 2, en este caso el hecho de que las matrices estén normalizadas sí que varía el resultado obtenido. Además, utilizando la ResNet50, se obtienen resultados similares con ambas imágenes, incluso llegando a ser mejores con la matriz UDF, todo lo contrario a lo ocurrido con las otras redes neuronales utilizadas.

4.3 Comparativa de los resultados obtenidos

A continuación, se presenta una tabla comparativa de los resultados obtenidos con las distintas pruebas, tanto con las redes neuronales como con el cálculo del ADV, todos ellos en función del *accuracy*.

	Nº de épocas hasta la cual la pérdida no se reduce tras 50 épocas		200 épocas		Nª de épocas hasta la cual las curvas de entrenamiento y validación se separan	
	Red nuronal 1	Red nuronal 2	Red nuronal 1	Red nuronal 2	Red nuronal 1	Red nuronal 2
DL 19	0.52	0.49	0.90	0.90	0.90	0.90
DL conjunto 400 duplicadas	0.50	0.51	0.89	0.89	0.89	0.89
DL conjunto 400 negro	0.58	0.56	0.92	0.92	0.92	0.92
DL 400	0.48	0.5	0.89	0.89	0.89	0.89
DL 121 conjunto duplicadas	0.59	-	0.94	0.94	0.94	0.94
DL conjunto 121 negro	0.57	0.52	0.85	0.85	0.85	0.85
DL 25% primer intervalo	0.53	0.61	0.90	-	0.90	-
DL 25% segundo intervalo	0.57	0.57	0.92	-	0.92	-
DL 25% tercero intervalo	0.52	0.60	0.91	-	0.91	-
DL 25% cuarto intervalo	0.55	0.56	0.93	-	0.93	-

		LRF Normalizado	LRF	UDF Normalizado	UDF
Red Neuronal 1		0.58	0.57	0.49	0.45
Red Neuronal 2		0.55	0.53	0.53	0.47
ResNet50	A	0.45	-	-	-
	B	0.45	-	-	-
	C	0.45	-	-	-
	D	0.50	-	-	-
	E	0.45	-	-	-
	F	0.50	-	-	-
	G	0.55	0.45	0.55	0.45
	H	0.55	-	-	-
	I	-	0.45	0.57	0.47

	ACC
SVR-SVM-axix-mode	0.7009
SVR-MC	0.7009
SVR-LDA-axis-mode	0.6838
SVR-SVM-axis-svm	0.6752
SVR-LDA-axis-svm	0.6496

5. CONCLUSIÓN

3.2. Conclusión

Sin duda alguna la salud es uno de los temas que más preocupan a nuestra sociedad. A los avances médicos en la lucha contra las enfermedades se ha incorporado con mucha fuerza el uso de elementos tecnológicos que sirvan de apoyo.

La tuberculosis sigue siendo una enfermedad que causa millones de muertes todavía en pleno siglo XXI. Un diagnóstico precoz y más eficaz sin duda alguna permitiría reducir estos números terribles. Esta línea ha sido la conductora de este trabajo final de grado donde se ha tratado de aplicar diferentes modelos de análisis a imágenes tomográficas de pacientes con tuberculosis para determinar el tipo de la misma que sufren dichos pacientes. Se ha trabajado en dos líneas, por un lado en el preprocesado y composición de las imágenes que forman la tomografía para su posterior análisis en redes de aprendizaje profundo. Por otro lado he aplicado técnicas de obtención de matrices de ADV a partir de las imágenes tomográficas proporcionadas para su posterior clasificación. A pesar de ser una investigación inicial los resultados son interesantes dada la alta complejidad de la tarea tal como se indica en las conferencias CLEF, encargadas de evaluar este tipo de sistemas y que nos acerca a resultados de grupos de investigación más consolidados.

Aunque dichos resultados no sean los esperados, el uso de la tecnología en el campo de la medicina tiene un gran futuro y un enorme margen de mejora.

Creo que herramientas de éste tipo deberían introducirse tanto en hospitales como en pequeñas clínicas. No para sustituir al médico, sino para que éste puede utilizarlo como apoyo en diagnósticos, toma de decisiones a la hora de recetar tratamientos o realizar operaciones. Este hecho supondría un progreso y mejoría en la calidad de vida de las personas, ya que, por ejemplo, se podrían evitar pruebas innecesarias y dolorosas para el paciente.

3.3. Trabajos Futuros

Hay muchos elementos de estudio abiertos en este campo, muchos grupos de investigación implicados y en concreto dentro del campo de la tuberculosis mucho todavía por mejorar tal como se indican en las conferencias CLEF. Para mí se abren dos líneas de trabajo en continuación de los desarrollos iniciados con este trabajo final de grado, en primer lugar, seguir mejorando los modelos que hemos venido aplicando, y en otra línea estudiar si cabe aplicar estos modelos a la detección de otros tipos de enfermedades.

3.4. Conclusión Personal

La realización de este Trabajo de Fin de Grado me ha permitido descubrir la gravedad de una enfermedad cómo es la Tuberculosis, la cual no tenía en mente a la hora de pensar en afecciones que causan tantas muertes actualmente. Para mí, poder comprobar cómo se pueden desarrollar aplicaciones en un ordenador que analizan y pueden ayudar a diagnosticar correctamente esta enfermedad ha sido muy gratificante. Es una línea de trabajo que me gustaría poder seguir en el futuro ya que estoy convencida que hay mucho camino por recorrer en este ámbito y grandes beneficios que se pueden conseguir.

Por otro lado, he ampliado y afianzado mis conocimientos adquiridos durante la carrera sobre el uso de las redes neuronales al utilizarlas en un caso práctico, el cual considero realmente importante. Todo el desarrollo del Trabajo Final de Grado me ha permitido conocer el modelo de trabajo en una aplicación mucho más compleja de las que había desarrollado, además estudiar tareas que realizar y planificarlas con su tiempo. También he podido aprender de algunos errores así como superar algunas frustraciones cuando las pruebas planeadas no salían como suponía. Todo ello me ha dado la posibilidad de aprender a cómo plantear y desarrollar esas pruebas para minimizar errores y pérdidas de tiempo.

Por otro lado, he tenido la posibilidad de estudiar en profundidad un lenguaje de programación como es el Python que parece tener un futuro muy prometedor. Adicionalmente, he podido evaluar varios entornos de trabajo para mi desarrollo, además de contrastarlos y utilizar finalmente el que mejor rendimiento me ofrecía.

6. BIBLIOGRAFÍA

Nyein Naing, W. Y., & Z. Htike, Z. (2015). Advances in Automatic Tuberculosis Detection in Chest X-Ray Images. *Signal & Image Processing : An International Journal*, *5*(6), 41–53. https://doi.org/10.5121/sipij.2014.5604

Qué es overfitting y underfitting y cómo solucionarlo | Aprende Machine Learning. (n.d.). Retrieved July 4, 2019, from https://www.aprendemachinelearning.com/que-es-overfitting-y-underfitting-y-como-solucionarlo/

Rojas, R. (n.d.). *Lucas-Kanade in a Nutshell*. Retrieved from http://www.inf.fu-berlin.de/inst/ag-ki/rojas_home/documents/tutorials/Lucas-Kanade2.pdf

Hermann, S., & Werner, R. (n.d.). *LNCS 8333 - High Accuracy Optical Flow for 3D Medical Image Registration Using the Census Cost Function*. Retrieved from https://link.springer.com/content/pdf/10.1007%2F978-3-642-53842-1_3.pdf

Azorin-Lopez, J., Saval-Calvo, M., Fuster-Guillo, A., Garcia-Rodriguez, J., & Mora-Mora, H. (2017). Constrained self-organizing feature map to preserve feature extraction topology. *Neural Computing and Applications*, *28*(S1), 439–459. https://doi.org/10.1007/s00521-016-2346-0

Mart??nez, S. M. R., Vidotti, S. A. B. G., & Jorente, M. J. V. (2016). Representaci??n conceptual de im??genes m??dicas digitales: Integraci??n de contexto y contenido visual. *Revista General de Informacion y Documentacion*, *26*(2), 651–672.

Bomanji, J. B., Gupta, N., Gulati, P., & Das, C. J. (2015). Imaging in tuberculosis. *Cold Spring Harbor Perspectives in Medicine*, *5*(6). https://doi.org/10.1101/cshperspect.a017814

Sociedad Española de Patología Digestiva., L. A. (2004). Revista española de enfermedades digestivas. In *Revista Española de Enfermedades Digestivas* (Vol. 100). Retrieved from http://scielo.isciii.es/scielo.php?script=sci_arttext&pid=S1130-01082008001100012

Tomografía Computarizada (TC) | National Institute of Biomedical Imaging and Bioengineering. (n.d.). Retrieved June 22, 2019, from https://www.nibib.nih.gov/espanol/temas-cientificos/tomografía-computarizada-tc

Imagen por Resonancia Magnética (IRM) | National Institute of Biomedical Imaging and Bioengineering. (n.d.). Retrieved June 22, 2019, from https://www.nibib.nih.gov/espanol/temas-cientificos/imagen-por-resonancia-magnética-irm

Cid, Y. D., & Müller, H. (2018). Texture-based graph model of the lungs for drug resistance detection, tuberculosis type classification, and severity scoring: Participation in the ImageCLEF 2018 tuberculosis task. *CEUR Workshop Proceedings*, *2125*.

Gentili, A. (2018). ImageCLEF2018: Transfer learning for deep learning with CNN for tuberculosis classification. *CEUR Workshop Proceedings*, *2125*, 6–12.

Ahmed, M. S., Obaidullah, S. M., Jayatilake, M., Gonçalves, T., & Rato, L. (2018). Texture analysis from {3D} model and individual slice extraction for tuberculosis {MDR} detection, type classification and severity scoring. *CLEF2018 Working Notes*.

Dicente Cid, Y., Liauchuk, V., Kovalev, V., & Müller, H. (2018). Overview of ImageCLEFtuberculosis 2018 - Detecting multi-drug resistance, classifying tuberculosis types and assessing severity scores. *CEUR Workshop Proceedings*, *2125*(July).

An Introduction to Biomedical Image Analysis with TensorFlow and DLTK - Medium. (n.d.). Retrieved June 22, 2019, from https://medium.com/tensorflow/an-introduction-to-biomedical-image-analysis-with-tensorflow-and-dltk-2c25304e7c13

5 Redes Neuronales Multicapa

_____ *5.1 La ADALINA*. (n.d.). Retrieved from http://www.lcc.uma.es/~jmortiz/archivos/Tema5.pdf

Patil, V., & Rudrakshi, S. (n.d.). *Enhancement Of Medical Images Using Image Processing In Matlab*. Retrieved from www.ijert.org

CS231n Convolutional Neural Networks for Visual Recognition. (n.d.). Retrieved June 7, 2019, from http://cs231n.github.io/convolutional-networks/

Srinivas, S., Sarvadevabhatla, R. K., Mopuri, K. R., Prabhu, N., Kruthiventi, S. S. S., & Babu, R. V. (2016). A Taxonomy of Deep Convolutional Neural Nets for Computer Vision. *Frontiers in Robotics and AI*, *2*. https://doi.org/10.3389/frobt.2015.00036

Dumoulin, V., & Visin, F. (2016). *A guide to convolution arithmetic for deep learning*. Retrieved from http://arxiv.org/abs/1603.07285

Ren, S., He, K., Girshick, R., & Sun, J. (2015). *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*. Retrieved from http://arxiv.org/abs/1506.01497

Redes neuronales convolucionales con TensorFlow. (n.d.). Retrieved June 7, 2019, from https://relopezbriega.github.io/blog/2016/08/02/redes-neuronales-convolucionales-con-tensorflow/

Redes Neuronales - Funcionamiento básico. (n.d.). Retrieved June 7, 2019, from http://perso.wanadoo.es/alimanya/funcion.htm

Keras Tutorial: Deep Learning in Python (article) - DataCamp. (n.d.). Retrieved June 7, 2019, from https://www.datacamp.com/community/tutorials/deep-learning-python

Función de activación - Redes neuronales - Diego Calvo. (n.d.). Retrieved June 3, 2019, from http://www.diegocalvo.es/funcion-de-activacion-redes-neuronales/

Activation Functions in Neural Networks – Towards Data Science. (n.d.). Retrieved June 3, 2019, from https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6

Keras Tutorial: Deep Learning in Python (article) - DataCamp. (n.d.). Retrieved June 3, 2019, from https://www.datacamp.com/community/tutorials/deep-learning-python

ImageCLEFmed Tuberculosis | ImageCLEF / LifeCLEF - Multimedia Retrieval in CLEF. (n.d.). Retrieved May 30, 2019, from https://www.imageclef.org/2019/medical/tuberculosis

ImageCLEFtuberculosis | ImageCLEF / LifeCLEF - Multimedia Retrieval in CLEF. (n.d.). Retrieved May 30, 2019, from https://www.imageclef.org/2018/tuberculosis

Understanding AUC - ROC Curve – Towards Data Science. (n.d.). Retrieved May 30, 2019, from https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5

Liauchuk, V., Tarasau, A., Snezhko, E., & Kovalev, V. (2018). ImageCLEF 2018: Lesion-based TB-descriptor for CT image analysis. *CEUR Workshop Proceedings*, *2125*(September).

Llopis, F., Fuster-Guilló, A., Ramón Rico-Juan, J., Azorín-López, J., & Llopis, I. (n.d.). *Tuberculosis detection using optical flow and the activity description vector*. Retrieved from http://www.ua.es