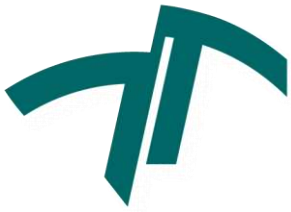


Universidad Nacional de Mar Del Plata
Facultad de Ingeniería
Departamento de Ingeniería Electrónica y Computación

REDES BASADAS EN APRENDIZAJE PROFUNDO PARA LA DETECCIÓN DE ANOMALÍAS EN RADIOGRAFÍAS DE TÓRAX

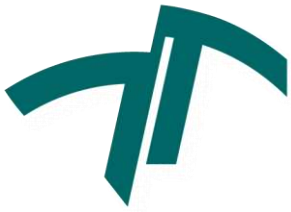
Proyecto Final

Ingeniería en Computación

Luciana Simón González

Director: Dr. Ing. Diego Sebastián Comas

Codirectora: Dra. Ing. Virginia Laura Ballarin



Universidad Nacional de Mar Del Plata
Facultad de Ingeniería
Departamento de Ingeniería Electrónica y Computación

REDES BASADAS EN APRENDIZAJE PROFUNDO PARA LA DETECCIÓN DE ANOMALÍAS EN RADIOGRAFÍAS DE TÓRAX

Proyecto Final

Ingeniería en Computación

Luciana Simón González

Director: Dr. Ing. Diego Sebastián Comas

Codirectora: Dra. Ing. Virginia Laura Ballarin

Resumen

Las imágenes de rayos X asisten a los expertos médicos en el diagnóstico y en la detección de patologías resultando esenciales, por ejemplo, para el diagnóstico de neumonía, la detección de masas y, recientemente, para la detección de afecciones características de COVID-19. Debido a que es uno de los exámenes radiológicos más accesibles, la radiografía de tórax es una de las primeras pruebas de imagen que se realizan cuando se sospecha de alguna patología.

Las redes neuronales basadas en el aprendizaje profundo, en particular las redes neuronales convolucionales, han tomado impulso en los últimos años y se han convertido en herramientas fundamentales para la clasificación de imágenes. Particularmente, los enfoques de *transfer-learning* han permitido utilizar conocimiento de redes previamente entrenadas, superando la necesidad de grandes conjuntos de datos y disminuyendo el gran costo computacional asociado a este tipo de redes.

En el presente Proyecto Final se aborda el estudio de redes neuronales basadas en aprendizaje profundo para la detección de anomalías en radiografías de tórax. Se estudian distintos enfoques basados en redes convolucionales, considerando la base de datos ChestX-ray14, con más de 100.000 imágenes de rayos-X con etiquetas relacionadas con 14 posibles patologías y evaluando diferentes objetivos de clasificación. Se implementan redes basadas en *transfer-learning*, a partir de las redes pre-entrenadas VGG19, ResNet50, e Inceptionv3, considerando diferentes esquemas para la etapa de clasificación y aumento de datos. Asimismo, se plantea y evalúa una arquitectura *ad-hoc*, sin *transfer-learning,* para el objetivo de clasificación con mayor cantidad de ejemplos.

Los resultados indican un rendimiento aceptable en la mayoría de los casos ensayados basados en el *transfer-learning*, lo que demuestra que es un camino inicial válido para el uso de redes profundas en casos en los que no hay suficiente cantidad de imágenes etiquetadas (*Gold Standard*), problema muy común al trabajar con imágenes médicas. Por su parte, la red *ad-hoc* mostró una buena generalización utilizando aumento de datos y un valor de exactitud aceptable. Los resultados obtenidos indican que las redes neuronales convolucionales con y sin *transfer-learning* constituyen un buen enfoque para el diseño de clasificadores para detectar patologías en radiografías de tórax.

Agradecimientos

Años atrás, durante mi adolescencia, leí el libro “Yo, robot” de Isaac Asimov. Aquel libro despertó en mí una gran curiosidad por el futuro, por la inteligencia de las máquinas y su autonomía para tomar decisiones. Dicha lectura, determinó que mis pasos a seguir serían estudiar Ingeniería. Por ello, quiero expresar mi agradecimiento a la universidad pública que me dio la oportunidad de formarme como ingeniera, que fue mi segundo hogar durante todos estos años y que me permitió conocer a personas extraordinarias.

Quiero agradecer a mi director Dr. Ing. Diego Comas y a mi co-directora Dra. Ing. Virginia Ballarin por su acompañamiento constante en este último tramo de mi carrera. Muchas gracias por sus enseñanzas y consejos, los cuales me van a seguir acompañando durante mi desarrollo como profesional.

Por último, pero no por eso menos importante, gracias a mis padres por motivarme a estudiar una carrera que me haga feliz, por apoyarme y acompañarme durante todos estos años hasta la finalización de la misma. Gracias a toda mi familia, a mis amigas y amigos, y a mi novio, por ser mi contención y mi sostén durante todo este recorrido, sin su apoyo no hubiese llegado hasta acá. Gracias a todos y todas por estar.

Índice

1 Introducción

1.1 Planteo del problema y estado de la cuestión

Los avances en la tecnología de adquisición, almacenamiento y procesamiento de datos han hecho posible la obtención de enormes cantidades de ellos a un bajo costo y han aumentado sustancialmente la capacidad de procesarlos para convertirlos en información útil y contribuir al desarrollo de conocimiento [1]. En el campo de las imágenes médicas esto provocó un aumento sustancial de la información disponible, dejando atrás los días en los que los datos relacionados con el ámbito de la salud eran escasos [2]; lo que supone un gran desafío a la hora de generar herramientas adecuadas para su análisis e interpretación que asistan al especialista en la toma de decisiones [3]. En particular, un gran volumen de estudios de radiografías de tórax acompañados de informes radiológicos se acumulan y almacenan en los sistemas informáticos de muchos hospitales modernos [4].

El Procesamiento Digital de Imágenes (PDI) ha sido fuente de algoritmos y herramientas exitosas en imágenes médicas permitiendo realizar tanto segmentación como clasificación. En este contexto, la segmentación define una partición tal que las regiones obtenidas correspondan a estructuras anatómicas, procesos o regiones de especial interés y sus resultados se utilizan para comparar volúmenes, morfologías y características con otros estudios u otras regiones de la misma imagen; estudiar la distribución de los tejidos, detectar lesiones, comprender la anatomía, planificar cirugías, planear terapias de radiación y detectar tejidos anormales, entre otras tareas. La clasificación requiere un análisis global de la imagen y se utiliza habitualmente como soporte a decisiones sobre diagnóstico y tratamiento.

Las redes neuronales basadas en aprendizaje profundo (DNN, del inglés *Deep Neural Network*), específicamente las convolucionales (CNN, del inglés *Convolutional Neural Networks*), han recibido un enorme auge durante los últimos años, debido principalmente al aumento de la capacidad y disponibilidad de unidades específicas para procesamiento gráfico (GPU, del inglés *Graphics Processing Unit*), la reducción significativa del costo del hardware y los recientes avances en aprendizaje automático (conocido como *machine learning* en inglés) [5]. En especial, en imágenes médicas, el número de aplicaciones exitosas de DNN es creciente, mostrando excelentes resultados en múltiples modalidades y con múltiples objetivos [2], [6], [7]; incluyendo: segmentación

de órganos y subestructuras, detección de tumores, clasificación de muestras (imágenes completas) y registración.

Dentro de las aplicaciones de las redes de aprendizaje profundo en imágenes médicas, la detección automática de anomalías en radiografías de tórax es un problema de enorme relevancia que se está estudiando actualmente y que, en el último año, ha tomado gran impulso en el mundo académico debido a la posibilidad de detectar afecciones características del COVID-19 [7], [8].

1.2 Objetivos

En este Proyecto Final se aborda la detección automática de anomalías en radiografías de tórax por medio de CNN, partiendo de la base de datos ChestX-ray14 [4], que contiene más de 100.000 imágenes con la presencia de 14 diferentes anomalías relacionadas con posibles patologías. El trabajo comprende un estudio profundo tanto de la problemática como de las CNN y el desarrollo de programas específicos para el diseño, entrenamiento y evaluación de las redes utilizando la API *Keras* sobre *Python* y procesamiento sobre GPU.

El objetivo general es estudiar, proponer, implementar y validar DNN para detección de anomalías en radiografías de tórax. A partir de éste, se definen los siguientes objetivos específicos:

- Realizar un relevamiento de DNN aplicadas a imágenes médicas.
- Estudiar trabajos existentes que aborden específicamente el procesamiento de radiografías de tórax.
- Desarrollar los algoritmos y realizar los programas requeridos para utilizar y validar DNN para detección de anomalías en radiografías de tórax.

1.3 Estructura del informe

A modo de guía para el lector, se detalla la estructura del informe y el contenido de cada uno de sus capítulos.

En el Capítulo 2 se introducen los conceptos más importantes para poner en contexto al lector sobre las DNN y las técnicas y enfoques empleados durante el desarrollo de este Proyecto Final.

En el Capítulo 3 se detallan los distintos modelos implementados, las diversas pruebas realizadas con los mismos, así como las tecnologías y el soporte físico sobre el cuál se llevó a término el desarrollo.

En el Capítulo 4 se presentan y analizan los resultados obtenidos a partir de los experimentos realizados con los modelos implementados en el Proyecto.

En el Capítulo 5 se presentan las conclusiones obtenidas a partir del análisis de los resultados y el desarrollo de este Proyecto Final.

Se incluye un capítulo con apéndices (Capítulo 7) en el que se encuentran los documentos de Plan de Proyecto y de Especificación de requerimientos que dieron soporte al desarrollo del Proyecto Final de grado.

2 Redes basadas en aprendizaje profundo

2.1 Introducción

La historia de la inteligencia artificial tuvo sus inicios a finales de los años 40, cuando algunos pioneros del campo de la computación comenzaron a cuestionarse si se podría lograr que las computadoras fueran inteligentes, o bien, que las máquinas “pensaran” [9]. En la actualidad, más de 70 años después, el campo de investigación ha crecido enormemente, abarcando desde el *machine-learning*, que comprende el aprendizaje profundo, hasta muchos otros enfoques que no implican aprendizaje en absoluto.

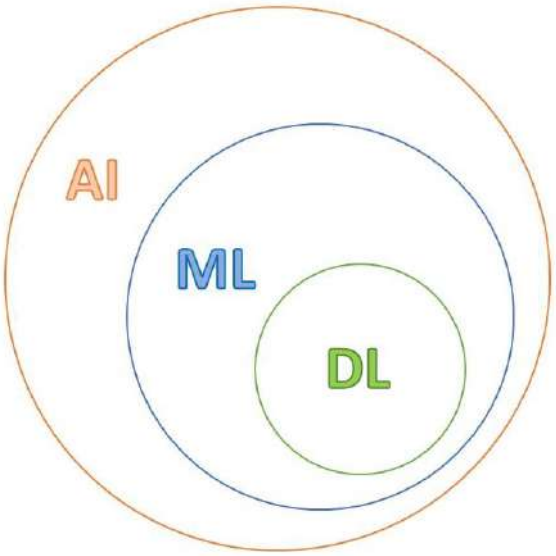


Figura 2.1 *- Relación entre la inteligencia artificial (AI, Artificial Intelligence), el aprendizaje automático (ML, Machine-Learning) y el aprendizaje profundo (DL, Deep-Learning).*

En particular, el aprendizaje profundo ha mejorado drásticamente el estado del arte, creando algoritmos de aprendizaje y arquitecturas capaces de resolver problemas complejos como el reconocimiento de imágenes, la detección y la segmentación de objetos, la traducción de idiomas y las tareas de comprensión del lenguaje natural, entre otros [10].

La motivación del presente capítulo es introducir los conceptos más importantes para poner en contexto al lector sobre las DNN y las técnicas y enfoques empleados durante el desarrollo del Proyecto Final.

2.2 Redes neuronales artificiales

El término red neuronal proviene de la neurobiología y hace referencia a un modelo de razonamiento basado en el cerebro humano, conformado por un conjunto de neuronas conectadas entre sí mediante conexiones sinápticas, las cuales pueden variar su

intensidad en respuesta a estímulos externos dando lugar al proceso de aprendizaje [11]. Las **redes neuronales artificiales** inspiran su funcionamiento en este mecanismo biológico, sin embargo, no son realmente modelos del cerebro sino un marco matemático para aprender representaciones a partir de datos [9].

El **perceptrón simple** (Figura 2.2) es la forma más sencilla de red neuronal. Consiste en una sola neurona artificial, pesos sinápticos $w_1, w_2, \dots, w_n$; y una función de activación ϕ que es definida como la función signo. Tiene como objetivo clasificar las entradas $x_1, x_2, \dots, x_n$, es decir, los estímulos externos, en una de las dos posibles salidas de la red, también llamadas clases. Por lo general, se hace uso de un umbral θ para desplazar el límite de decisión entre una clase u otra. La salida del perceptrón se obtiene a partir de la especialización de la función de activación en el valor que resulta de la suma de cada una de las entradas $x_1, x_2, \dots, x_n$ pesadas por los pesos $w_1, w_2, \dots, w_n$, menos el umbral θ, es decir, formalmente, la salida Y se obtiene como:

$$Y = \phi(v_i) = \phi\left(\sum_{i=1}^{n} x_i w_i - \theta\right) \qquad (2.1)$$

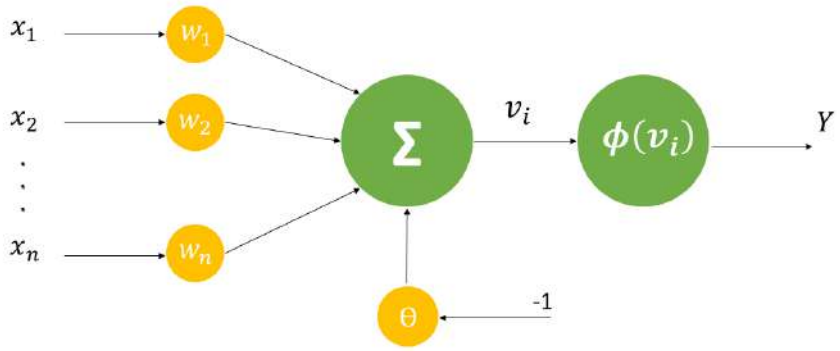


Figura 2.2 - *Ilustración de un perceptrón simple. La salida Y corresponde a la función de activación ϕ especializada en el valor resultante de la suma pesada de cada entrada menos el factor de umbral θ.*

En esencia, el perceptrón simple es un clasificador de patrones linealmente separables, donde el espacio n-dimensional está particionado por un hiperplano en dos regiones de decisión, cada una perteneciente a una de las salidas. Para el caso particular de dos entradas, x_1 y x_2, se genera una recta de separación dada por la ecuación:

$$x_1 w_1 + x_2 w_2 - \theta = 0 \qquad (2.2)$$

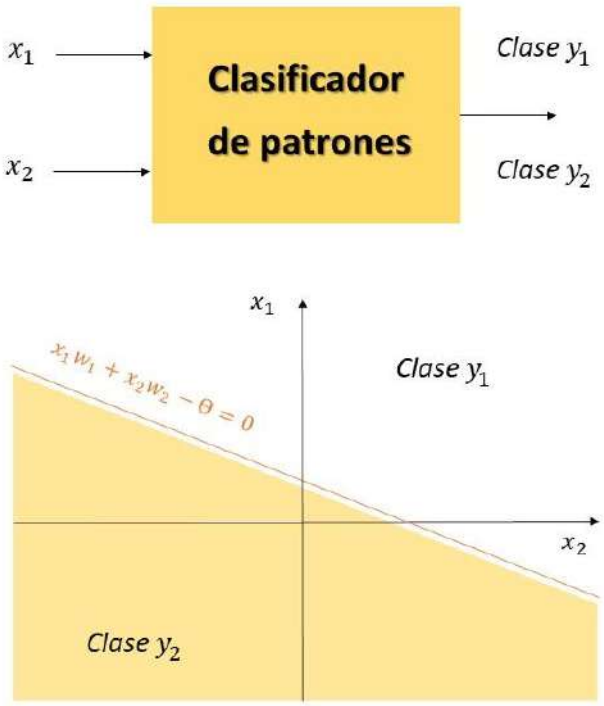


Figura 2.3 *- Representación del perceptrón simple y la partición generada para un ejemplo con datos de dos dimensiones.*

Tanto los pesos sinápticos como el umbral de activación (conocidos como parámetros internos) son inicializados de manera aleatoria y deben ser ajustados para cada problemática en particular, de manera tal que el hiperplano definido sea el adecuado para el problema. El proceso de adaptación de los parámetros internos es iterativo y se conoce como entrenamiento.

Durante el entrenamiento, se presentan a la red datos de entrada y etiquetas deseadas (definidas en el *Gold-Standard*). En cada iteración, los parámetros internos son ajustados acercando la respuesta de la red a la deseada. La velocidad de cambio de los parámetros libres (conocida como constante de aprendizaje) disminuye a medida que el entrenamiento avanza. Una vez finalizado el entrenamiento, la red puede utilizarse para clasificar nuevos datos, proceso que se conoce como consulta. La capacidad que tenga la red para predecir correctamente la clase que corresponde a datos que no fueron utilizados durante el entrenamiento se conoce como capacidad de generalización. El error de generalización, una estima de la probabilidad de clasificar correctamente un nuevo dato, es la principal medida de calidad de una red neuronal.

El hecho de que el perceptrón simple sólo pueda resolver problemas sobre datos linealmente separables supone una gran limitación, principalmente en el caso del PDI, donde se requiere que la función de entrada-salida de la red neuronal sea insensible a

las variaciones en la posición, la orientación, la iluminación de un objeto, entre otras [10]. No obstante, la interpretación del perceptrón como unidad computacional básica, permite juntar múltiples unidades para crear modelos mucho más complejos, cuya arquitectura se denomina **perceptrón multicapa** (Figura 2.4).

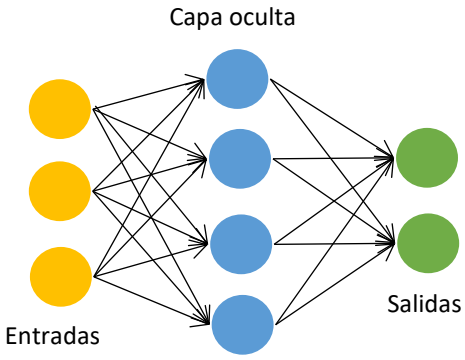


Figura 2.4 *- Diagrama de perceptrón multicapa con una única capa oculta.*

Cada neurona del perceptrón multicapa tiene los mismos elementos que el perceptrón simple, es decir, un conjunto de entradas, pesos sinápticos, un valor de umbral y una función de activación, que en este caso no corresponde a la función signo, sino que se trata de una función suave y derivable con el fin de favorecer el proceso de aprendizaje y optimización. Para este tipo de redes la selección del algoritmo de entrenamiento constituye uno de los parámetros de diseño junto con su arquitectura (números de capas ocultas y neuronas) y funciones de activación.

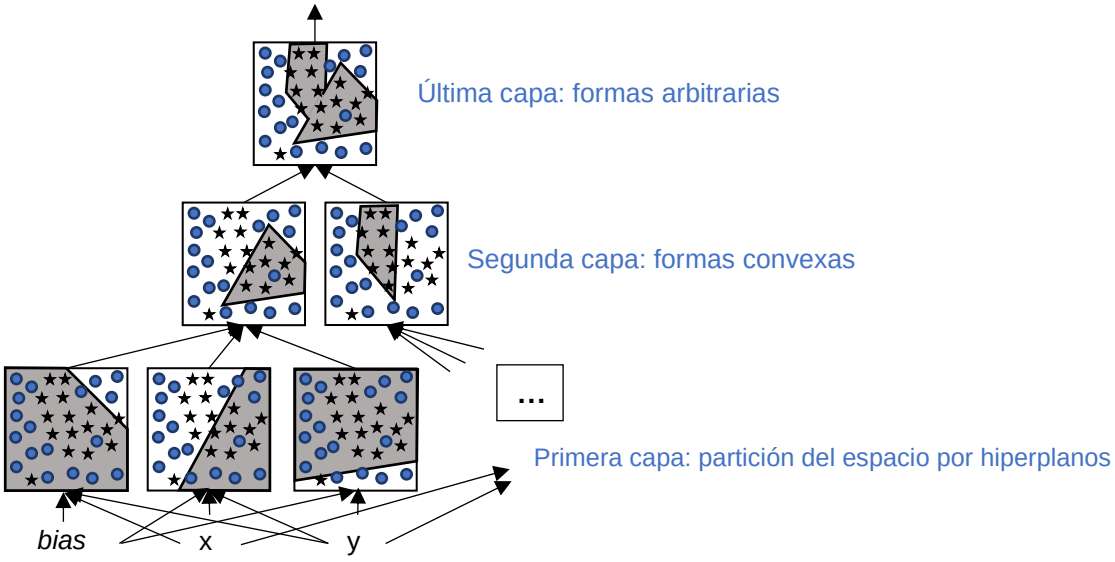


Figura 2.5 *- Superficies de separación de un perceptrón multicapa con dos capas ocultas y funciones de activación lineales. La forma de las superficies está relacionada con las funciones de activación utilizadas.*

Un perceptrón multicapa puede verse como un gráfico computacional conformado por unidades elementales, generando regiones de decisión cada vez más complejas a medida que aumentan la cantidad de capas y de neuronas (Figura 2.5).

2.3 Aprendizaje profundo

El **aprendizaje profundo** utiliza arquitecturas jerárquicas y hace hincapié en el aprendizaje de representaciones o características cada vez más significativas (y complejas) a través de las capas de la red [9].

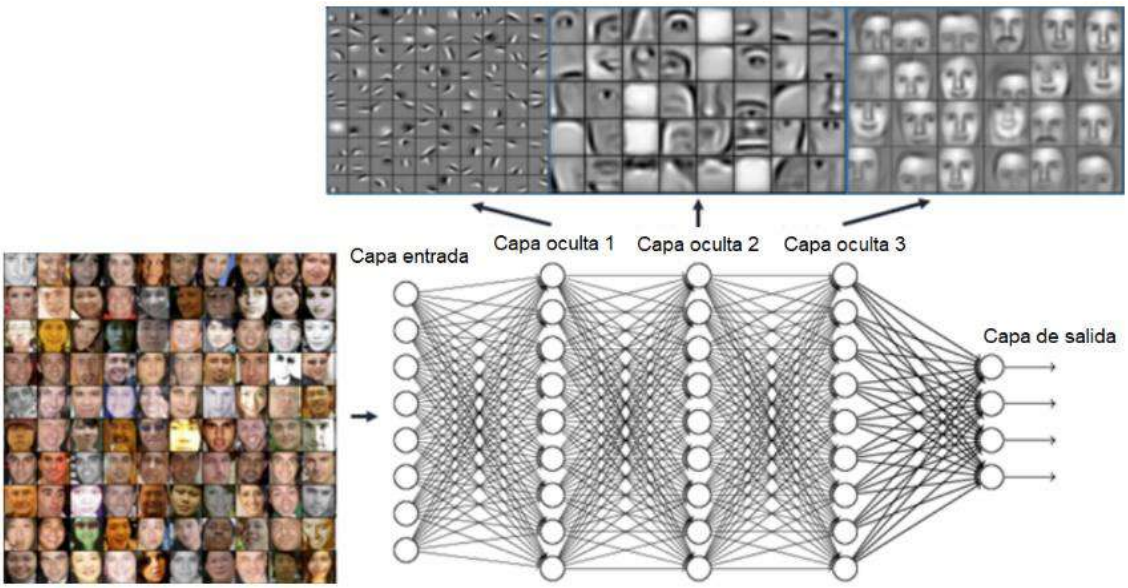


Figura 2.6 *- Ejemplo de una red neuronal profunda con tres capas ocultas donde se puede visualizar el aprendizaje jerárquico: al avanzar sobre las capas, las características aprendidas son más complejas [12].*

En la actualidad, el término profundo o *deep* implica decenas, cientos o incluso miles de capas constituyentes. Cada capa tiene un conjunto de parámetros internos que deben ser ajustados durante el entrenamiento, el cual se realiza de la misma manera que el caso de los perceptrones, lo que hace que el número de ejemplos requeridos para tener una buena capacidad de generalización sea alto. Sin embargo, con esta enorme cantidad de neuronas y capas, una DNN puede aprender una gran cantidad de características, siempre que el entrenamiento sea exitoso.

La disponibilidad de grandes bases de datos y el aumento de la capacidad de procesamiento ha hecho posible el uso de este tipo de redes con buena capacidad de generalización, principalmente en el caso de imágenes [2]. Asimismo, las DNN han demostrado ventajas importantes frente a métodos tradicionales dentro del aprendizaje automático, como por ejemplo los perceptrones [13]:

1. Las neuronas pueden descubrir directamente representaciones de los datos de entrenamiento, pudiendo compensar e incluso superar el poder de discriminación de los métodos convencionales de extracción de características.
2. La interacción y la jerarquía de las representaciones pueden explotarse conjuntamente dentro de la red.
3. La extracción y selección de características y la clasificación se realizan dentro de la misma red, por lo que puede optimizarse durante el entrenamiento.

2.4 Redes neuronales convolucionales y su implementación práctica con *transfer-learning*

Dentro de las DNN, las CNN son un tipo de red comúnmente utilizada para el reconocimiento y la clasificación de imágenes. A diferencia de las arquitecturas tradicionales basadas en el perceptrón, las CNN incluyen una o más capas convolucionales que extraen características de las imágenes de entrada a través de operaciones de convolución.

Una imagen digital, en el contexto del PDI, puede ser definida como una función bidimensional o tridimensional discreta. Está compuesta por un número finito de elementos, denominados píxeles, cada uno de los cuales tiene una ubicación y un valor determinados [14]. Las CNN son capaces de capturar con éxito las variaciones espaciales en una imagen mediante la aplicación de filtros de convolución, del mismo tipo que los utilizados en PDI para el filtrado espacial.

La arquitectura de una CNN típica está conformada principalmente por las siguientes capas (ver Figura 2.7):

- **Capa de convolucional**. Esta capa le da nombre a la red y es el elemento central. Las capas convolucionales aplican sistemáticamente filtros espaciales lineales para crear mapas de características que describen las particularidades de las imágenes de entrada. Cada neurona de una capa convolucional aplica un filtro espacial específico, realizando la convolución matricial.

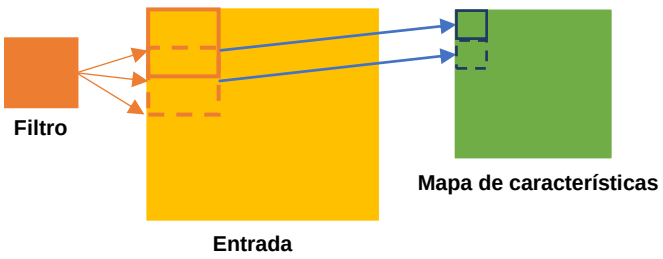


Figura 2.7 *- El filtro y la entrada realizan una operación matricial conocida como convolución, la cual resulta en un mapa de características.*

- **Capa de activación**. Una vez creado el mapa de características, se pasa cada valor del mapa de características a través de una función de activación que aporta no linealidad, de forma muy parecida a lo que se hace con las salidas de una capa totalmente conectada. La función más comúnmente utilizada es la función ReLU (del inglés *Rectified Linear Unit*) que, además de permitir mapeos no lineales, es matemáticamente simple y computacionalmente eficiente.
- **Capa de submuestreo o *pooling***. Una limitación del enfoque convolucional es que las características descubiertas por el filtro dependen fuertemente de la localización espacial de los objetos o texturas presentes. Por lo tanto, la rotación, el desplazamiento u otros cambios menores en la imagen de entrada darán como resultado un mapa de características diferente, cuando es deseable que las características descubiertas sean independientes de estos cambios. Para abordar este problema se utiliza la capa de *pooling*, en la cual los valores resultantes del filtrado son submuestreados, reduciendo el tamaño final de las matrices. En la Figura 2.8 se muestran dos ejemplos de *pooling* con un tamaño 2x2. En *max-pooling* la matriz resultante se obtiene como el máximo de los valores contenidos en regiones de 2 por 2 píxeles. En el caso del *average-pooling* se reemplazan por el promedio. La idea del *pooling* es cubrir una región mayor del espacio, ampliando el campo de "aprendizaje" de la red haciéndolo más general.

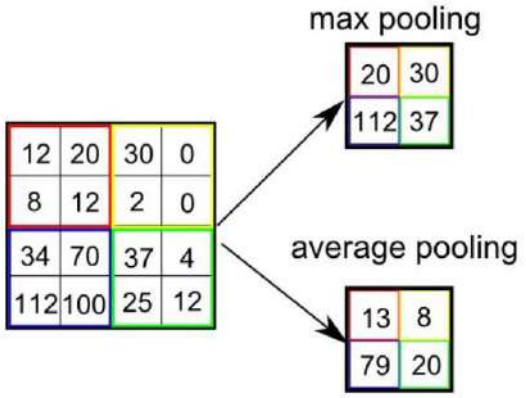


Figura 2.8 *- Ejemplo de dos operación de pooling distintas (max-pooling y average-pooling) aplicada sobre una matriz de entrada [15].*

- **Capa totalmente conectada o *fully connected***. Típicamente se alternan capas de conexión, ya que es una forma de aprender combinaciones no lineales de las características de alto nivel representadas por la salida de la capa convolucional.

Además de las capas anteriores, las CNN pueden incluir capas con perceptrones, con las características descriptas en la sección 2.2.

La estructura general de las CNN puede dividirse en dos fases (ver Figura 2.9): la **fase de extracción de características** y la **fase de clasificación** [16].

- La **fase de extracción de características** alterna varias capas de neuronas convolucionales, capas de activación y capas de submuestreo (*pooling*) que reducen subsecuentemente la cantidad de información (píxeles). Como resultado, la salida de esta fase puede interpretarse como un conjunto de características asociadas a la imagen ingresada. El apilamiento de capas convolucionales en esta fase es lo que permite realizar una descomposición jerárquica de la entrada y que las capas más profundas del modelo aprendan características más abstractas (complejas).
- La **fase de clasificación** es una red neuronal tradicional, del tipo perceptrón multicapa, cuya arquitectura se ajusta según la clasificación a realizar. La cantidad de neuronas de salida iguala la cantidad de clases.

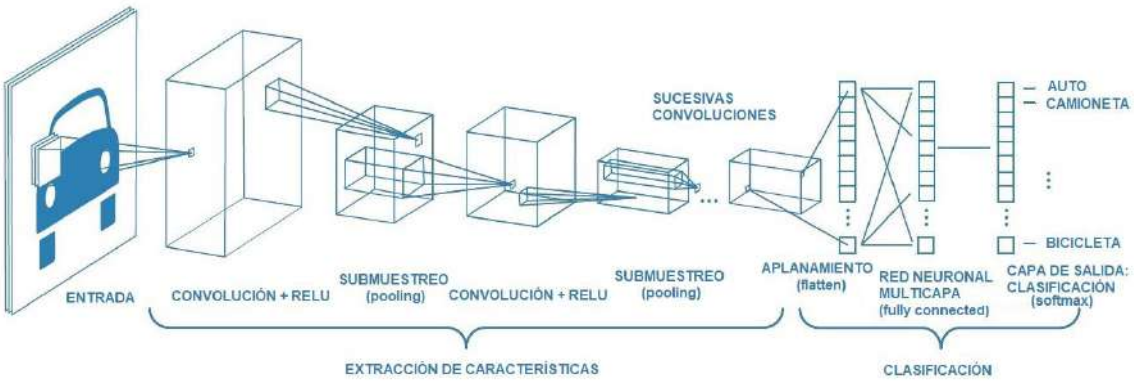


Figura 2.9 *- Estructura de una CNN [15].*

En el contexto del aprendizaje profundo la mayoría de los modelos que resuelven problemas complejos necesitan una gran cantidad de datos con el fin de optimizar la gran cantidad de parámetros que estos modelos poseen. Concretamente, para los modelos supervisados, obtener grandes conjuntos de datos etiquetados puede ser realmente difícil, considerando el tiempo y el esfuerzo que se necesita para el etiquetado [12]. Un ejemplo de ello sería ImageNet [17], [18], una base de datos a gran escala que se compone de millones de imágenes pertenecientes a más de 20.000 categorías diferentes, fuertemente inspirada en la base de la estructura WordNet, y cuya conformación llevó muchos años de etiquetado manual.

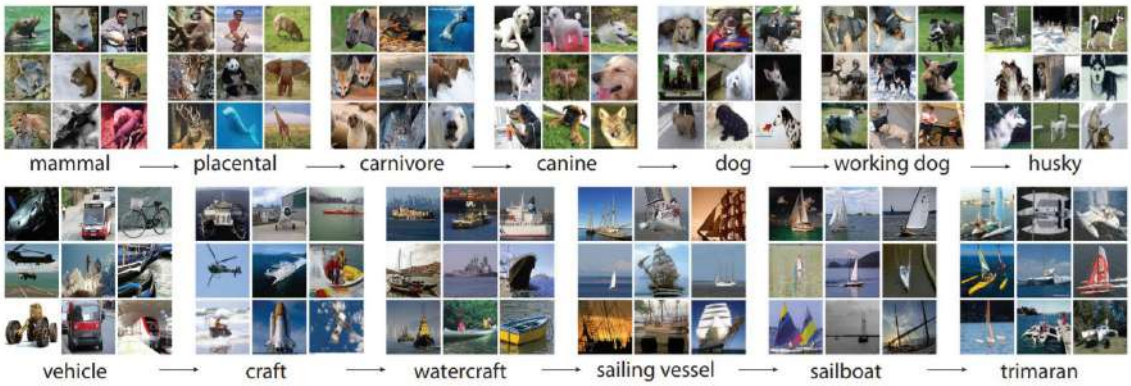


Figura 2.10 *- Una instantánea de dos ramas raíz-hoja de ImageNet: la fila superior es del subárbol de mamíferos; la fila inferior es del subárbol de vehículos. Se presentan imágenes muestreadas al azar [21].*

Las principales arquitecturas de CNN surgieron del *ImageNet Large-Scale Visual Recognition Challenge* (ILSVRC) [17], un concurso de visión computacional desarrollado anualmente desde 2010 y hasta 2017, a partir de subconjuntos de datos obtenidos de ImageNet. Algunas de las arquitecturas pre-entrenadas que surgieron en el marco de este concurso son: AlexNet [19], VGGNet [20], Inception (GoogLeNet) [21], ResNet [22], DenseNet [23], entre otras.

Un enfoque ampliamente utilizado en la implementación práctica de las CNN para resolver el problema de la escasez de datos etiquetados (muy habitual al trabajar con imágenes médicas) es el ***transfer-learning*** [7]. Éste consiste en utilizar una CNN previamente entrenada y reemplazar la fase de clasificación por un nuevo clasificador para reentrenar la red con el conjunto de imágenes de interés, reajustando sólo los parámetros internos de la fase de clasificación; lo que reduce drásticamente el tiempo de entrenamiento y el número de ejemplos necesarios. En la Figura 2.11 se muestra un esquema general del enfoque de *transfer-learning*.

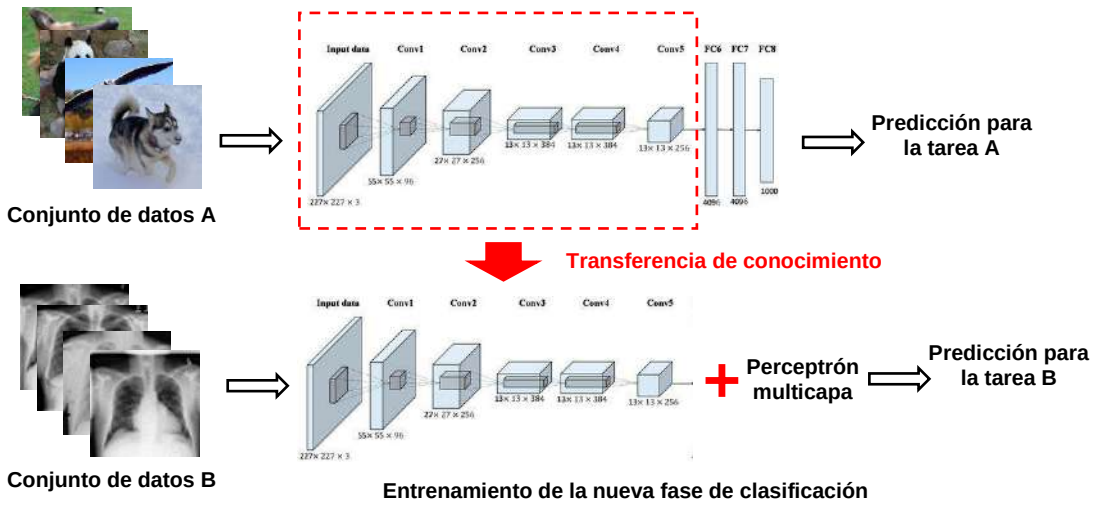


Figura 2.11 *- Ilustración explicativa de la técnica de transfer-learning.*

El uso más común del *transfer-learning* en el contexto del aprendizaje profundo es mediante el siguiente flujo de trabajo [24]:

1. Se toman algunas de las capas de un modelo previamente entrenado.
2. Se congelan las capas del modelo pre-entrenado, para evitar perder la información que contienen en futuras rondas de entrenamiento, es decir, sus parámetros internos se fijan como no “ajustables”.
3. Se añaden nuevas capas entrenables, es decir, con parámetros ajustables, sobre las capas congeladas extraídas en el paso 1). Las nuevas capas aprenderán a convertir las antiguas características en predicciones sobre un nuevo conjunto de datos.
4. Se entrena la red ajustando sólo las nuevas capas con el nuevo conjunto de datos. Las capas del modelo pre-entrenado no se modifican.

Un último paso, opcional, es el ***fine-tuning***, que consiste en descongelar todo el modelo ya entrenado (o parte de él), y volver a entrenarlo con los nuevos datos con una tasa de aprendizaje muy baja. De este modo se pueden conseguir mejoras significativas, adaptando de forma incremental las características pre-entrenadas a los nuevos datos.

Aunque, en general, sólo se toman del modelo pre-entrenado las capas de la fase de clasificación, existen diferentes enfoques de *transfer-learning* en la bibliografía, algunos de los cuales sólo reemplazan y reentrenan la última capa de la fase de clasificación. En este sentido, se entiende como *transfer-learning* todo enfoque en el que se reutilicen capas de una red entrenada previamente.

El flujo de trabajo descrito anteriormente permite modificar dinámicamente los datos de entrada del nuevo modelo durante el entrenamiento, lo que permite utilizar técnicas de *data-augmentation* (ver apartado 2.6.2) para mejorar la generalización de la red.

2.5 CNN utilizadas en el proyecto

Habiéndose introducido los conceptos básicos de las CNN, este subapartado pretende describir brevemente las arquitecturas de las CNN pre-entrenadas empleadas durante el desarrollo del Proyecto Final, la Tabla 2.1 muestra algunas de sus propiedades.

Tabla 2.1 *- Propiedades CNN pre-entrenadas utilizadas en el proyecto.*

Arquitectura	Tamaño imagen de entrada (píxeles)	Profundidad (capas)	Parámetros (millones)
VGG19	224 x 224	19	144
Inception V3	299 x 299	48	23.9
ResNet50	224 x 224	50	25.6

2.5.1 VGGNet

VGGNet [20] fue construida y entrenada por Karen Simonyan y Andrew Zisserman, pertenecientes al Vision Geometry Group (VGG) de la universidad de Oxford, resultando ganadora en la tarea de localización y obteniendo el segundo puesto en la tarea de clasificación del concurso ILSVRC en el año 2014.

Esta red utiliza una arquitectura con filtros de convolución muy pequeños (3x3) logrando una mejora significativa respecto a las configuraciones de CNN anteriores, donde se utilizaron grandes filtros de convolución (9x9 o 11x11 de AlexNet [19]). El filtro de 3x3 el más pequeño que se puede emplear sin perder la noción de izquierda/derecha, arriba/abajo, centro entre los píxeles vecinos.

Existen variantes de la VGG según el número de capas ocultas. En este Proyecto Final se utiliza la VGG19.

2.5.2 Inception (GoogLeNet)

GoogLeNet [21], también conocida como *Inception*, es una CNN desarrollada por investigadores de Google. La arquitectura GoogLeNet presentada en el ILSVRC en el año 2014 obtuvo el primer puesto en la tarea de clasificación de imágenes, superando a VGG. La profundidad de GoogLeNet es mayor a la de VGGNet, sin embargo, la cantidad de parámetros es mucho menor, convirtiéndola una mejor opción para optimizar costos computacionales cuando los recursos disponibles son limitados.

Los datos de entrada a la red son imágenes de dimensión 224x224 píxeles, preprocesadas con un promedio de cero.

2.5.3 ResNet

ResNet introdujo el concepto de aprendizaje residual para abordar el problema del desvanecimiento del gradiente, agregando conexiones directas entre neuronas, sin agregar parámetros adicionales ni complejidad computacional adicional. En el año 2015, fue ganadora en el ILSVRC en las categorías de clasificación, detección y localización de imágenes, así como en MS COCO [25] en las categorías de detección y segmentación.

ResNet introdujo el concepto de bloque residual con la idea de que la información fluyera a través de las conexiones, lo que le permitió construir redes mucho más profundas. El bloque residual consiste en varias capas de la red y una conexión de acceso directo (ver Figura 2.12).

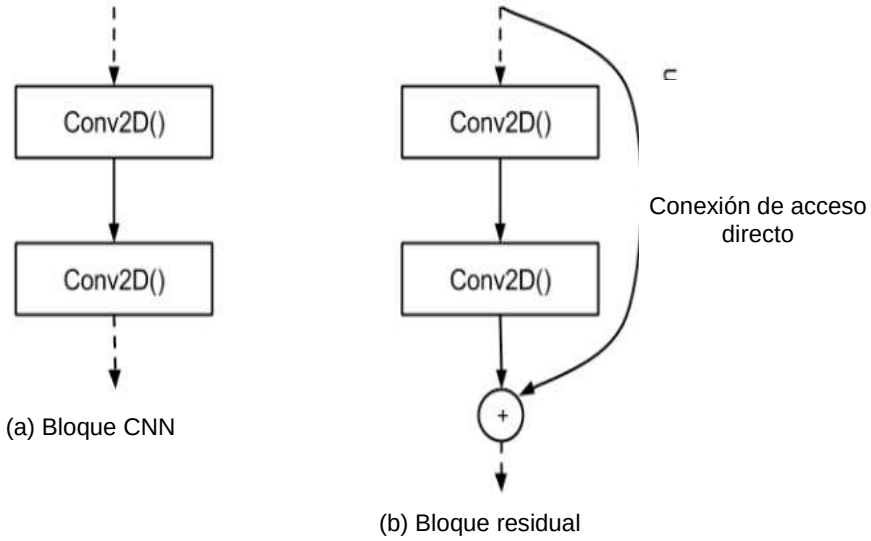


Figura 2.12 *- Comparación entre un bloque de CNN típico (a) y un bloque residual en ResNet (b)* [26].

2.6 Regularización

Al desarrollar DNN es común la aparición del escenario en el cual el modelo parece funcionar bien con los datos de entrenamiento, pero tiene muy baja capacidad de generalización, lo que se conoce como *overfitting* o sobreajuste [27].

Durante el proceso de entrenamiento una red neuronal aprende los patrones latentes en los datos y mejora matemáticamente los pesos del modelo para adaptarse a los patrones que descubre en el proceso de aprendizaje. Este proceso se complica en el momento en que el patrón que descubre resulta ser simplemente ruido en la realidad. En los casos reales, parte de los datos de entrenamiento pueden estar afectados por ruido y estos datos pueden diferir mucho en magnitud respecto a los datos reales. Si una red neuronal se sobreentrena, es decir, sobre aprende los datos de entrenamiento, y éstos contienen ruido, entonces, probablemente los parámetros internos se muevan en su ajuste respecto a los valores que obtendrían en datos no afectados por ruido, lo que hace que la red sea menos robusta a la presencia de ruido y no sea capaz de generalizar bien. En la Figura 2.13 se muestran ejemplos de imágenes de resonancia magnética de cerebro con y sin presencia de ruido. Las imágenes pseudo-color corresponden a resultados de segmentaciones obtenidas a partir de las imágenes en grises en secuencias PD, T1 y T2, que en el ejemplo son las imágenes de entrada.

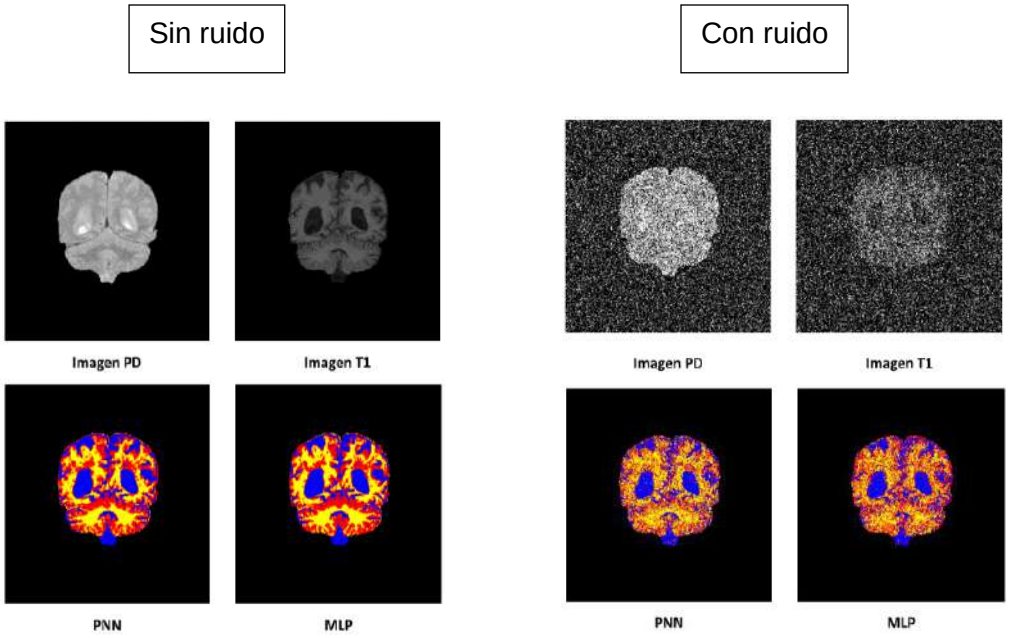


Figura 2.13 *– Derecha: Imágenes con ruido; izquierda: imágenes sin ruido* [28]. PNN: Red neuronal probabilística, MLP: Perceptrón multicapa.

Otra causa común para el *overfitting* se debe a los casos en los que se dispone de muy pocas muestras para aprender, por lo que no se puede entrenar un modelo que pueda generalizar bien. Si los datos fueran infinitos, el modelo estaría expuesto a todos los aspectos posibles de la distribución de datos por lo que nunca podría sobre ajustarse [9].

Se conoce como **regularización** al proceso de reducir el *overfitting*. Existen variedad de métodos de regularización, siendo los más utilizados en DNN: *dropout* y *data-augmentation*.

2.6.1 Dropout

La técnica de ***dropout*** es una de las técnicas más eficaces y utilizadas. Este método, aplicado a una capa, consiste en “desactivar” arbitrariamente (poner en cero) algunas neuronas de una capa durante cada iteración del entrenamiento. Se descarta aleatoriamente un número de características de salida de la capa. La fracción de las características que se reducen a cero es un parámetro a elegir que se conoce como tasa de abandono [27], [29].

Se puede apreciar en la Figura 2.14 como la red regular (izquierda) tiene todas las neuronas y conexiones entre dos capas sucesivas intactas. Mientras que, con el *dropout*, cada iteración induce un cierto grado definido de aleatoriedad al desactivar o descartar arbitrariamente algunas neuronas y sus conexiones asociadas.

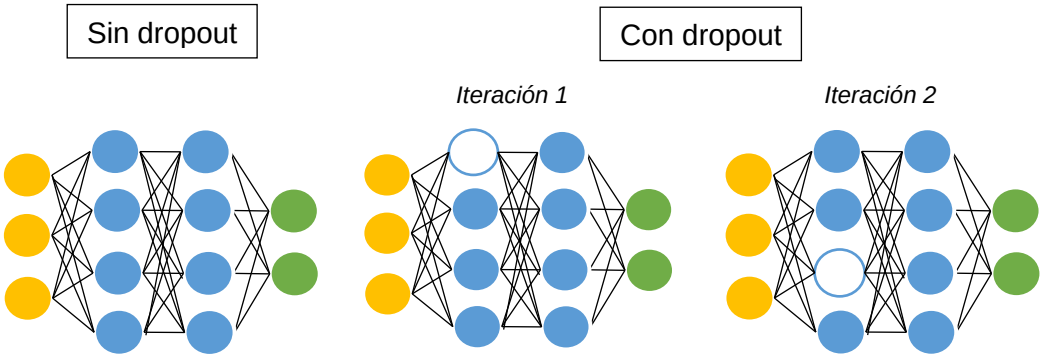


Figura 2.14 *- Figura ilustrativa de la técnica de dropout.*

2.6.2 Data-Augmentation

El *data-augmentation* o aumento de datos consiste en generar más datos de entrenamiento (en cantidad y/o en diversidad) a partir de las muestras de entrenamiento existentes, aumentando las muestras mediante una serie de transformaciones aleatorias que en general consisten en rotaciones, transformaciones afines, traslaciones, escalamiento, entre otras. El objetivo es que, en el momento del entrenamiento, el modelo nunca vea exactamente la misma imagen dos veces y, de esta forma, generalice mejor [9], lo que se conoce como aumento de datos dinámico (ver Figura 2.15).

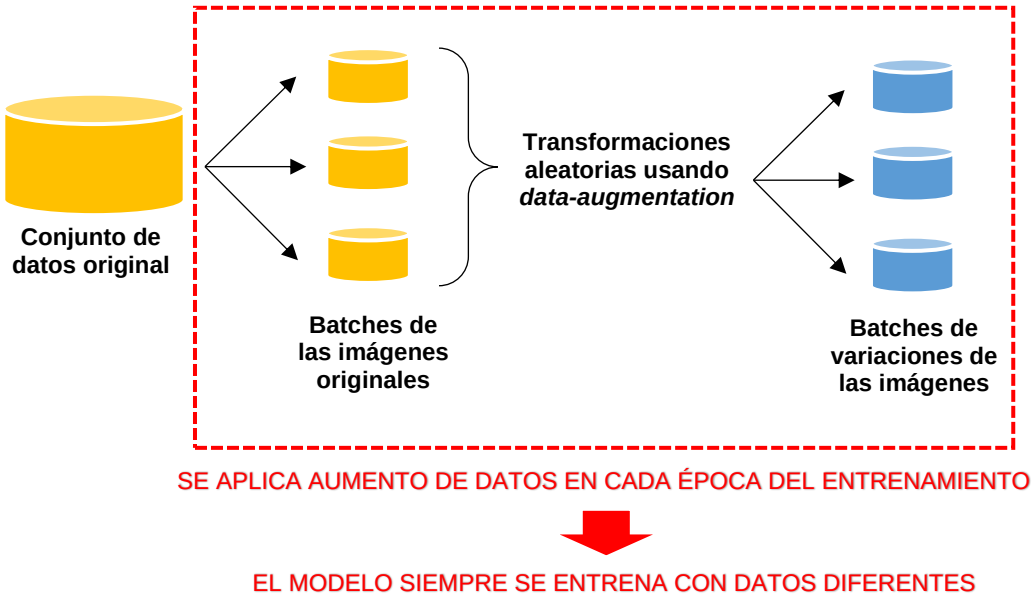


Figura 2.15 *– Aumento de datos dinámico.*

2.7 Validación de modelos

Como se mencionó anteriormente, la capacidad de generalización es la característica esencial con la que se evalúa la calidad de una red neuronal. Debido a esto, se requiere

definir una métrica apropiada y un mecanismo adecuado que permita estimar adecuadamente la capacidad de generalización. Si bien existen muchas métricas y muchos métodos de estimación, en este Proyecto Final se adopta la exactitud como métrica y *hold-out* como método de estimación, ambos de uso habitual al trabajar con DNN.

La **exactitud** (*accuracy* en inglés) del modelo consiste en el número total de predicciones correctas dividido por el número total de predicciones y es una medida global de la capacidad de generalización del modelo.

La estima correcta de la capacidad de generalización requiere evaluar el modelo con datos que no hayan sido presentados durante el entrenamiento. El método de validación *hold-out* se basa en crear dos particiones disjuntas del conjunto de datos original, en forma aleatoria, una de las cuales se utiliza como conjunto de entrenamiento y la otra como conjunto de prueba o conjunto de *test* [11]. La métrica de calidad se evalúa sobre el conjunto de *test*. En la Figura 2.16 se muestra un diagrama del proceso de *hold-out*.

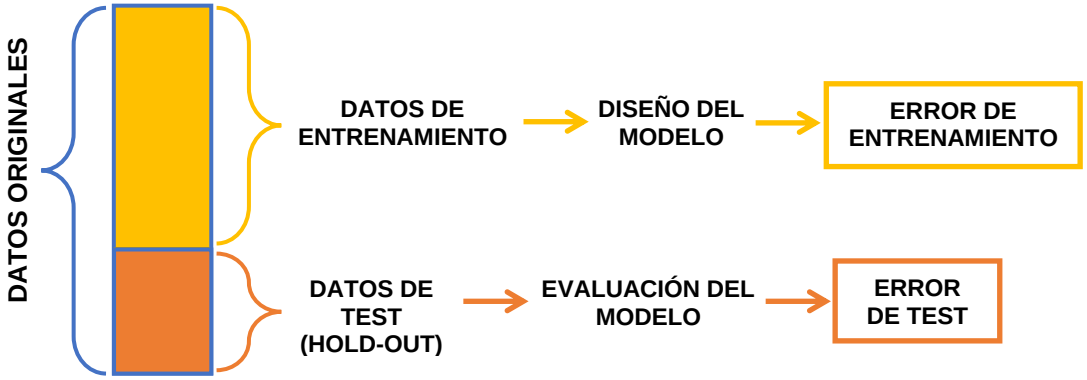


Figura 2.16 *- Partición hold-out simple del conjunto de datos.*

Dado que habitualmente pueden usarse distintos submuestreos del conjunto de datos o inicializaciones aleatorias de los parámetros internos de las DNN, es habitual realizar más de un ciclo de entramiento/test definiendo para cada uno un *hold-out* distinto. En estos casos, la métrica se obtiene como el promedio de los valores obtenidos sobre los distintos ciclos.

3 Desarrollo

3.1 Introducción

Las radiografías de tórax son uno de los exámenes médicos más accesibles para la detección y la determinación de muchas enfermedades. Sin embargo, para un especialista capacitado, el diagnóstico clínico de este tipo de estudio puede ser un gran desafío debido a la complejidad que conlleva.

En la actualidad, la falta de recursos para etiquetar gran cantidad de imágenes es uno de los principales obstáculos para la creación de grandes conjuntos de datos. Particularmente, el gran desafío de entrenar CNN para detectar anomalías de rayos X se presenta ante la escasez de grandes bases de datos de imágenes en este dominio con sus respectivas etiquetas. Hasta ahora, la base de datos pública más grande disponible es ChestX-ray14 [4], sobre la cual se trabajó en este Proyecto Final.

Los modelos de CNN se implementaron a partir de la API *Keras*, basada en *TensorFlow* usando el lenguaje de programación *Python*. Todas las pruebas fueron realizadas sobre el servidor de procesamiento de Laboratorio de Procesamiento de Imágenes del ICyTE que tiene las siguientes características: procesador Intel Core i9-10900, 64 GB de memoria RAM y una GPU NVIDIA Titan V donada por NVIDIA a partir de la NVIDIA GPU Grant.

3.2 Base de datos ChestX-ray14

La base de datos ChestX-ray14 [4] fue recopilada por el NIH (*National Institute of Health*), principal agencia del gobierno de los Estados Unidos responsable de la biomedicina y la salud pública de investigación. Comprende más de 100.000 imágenes de rayos X de tórax con multi-etiquetas de enfermedades comunes de más de 30.000 pacientes anónimos, acumuladas desde el año 1992 hasta el 2015. Los datos fueron extraídos a partir de textos de informes radiológicos mediante técnicas de procesamiento de lenguaje natural (NLP, del inglés *Neurological Language Processing*). En la Figura 3.1 se muestran algunos ejemplos de las imágenes de la base de datos, con sus respectivas etiquetas.

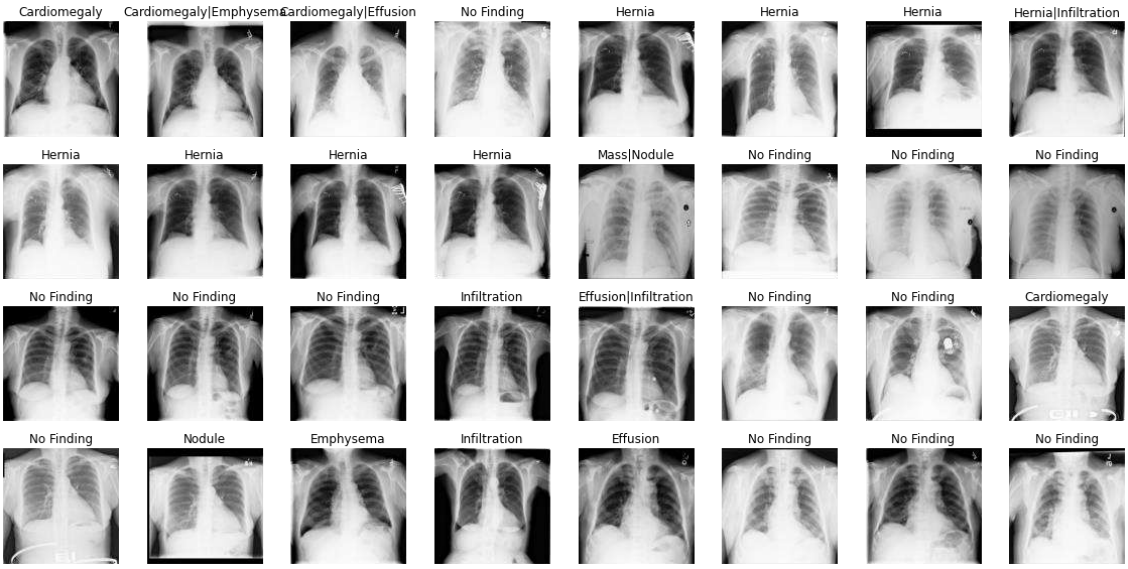


Figura 3.1 *– Imágenes de radiografías de tórax, con sus respectivas etiquetas, pertenecientes al conjunto de datos ChestX-ray14.*

El conjunto de datos contiene las siguientes 14 variedades de anomalías: infiltración, derrame, atelectasia, nódulo, masa, neumotórax, consolidación, hinchazón pleural, cardiomegalia, enfisema, edema, fibrosis, neumonía y hernia. Las radiografías se etiquetan con una o varias palabras clave de patología, resultando en etiquetas simples o en multi-etiquetas respectivamente (ver Figura 3.2). Por su parte, las imágenes no asociadas con ninguna de las 14 clases de patologías se etiquetan como "*No Finding*", lo que no equivale a una imagen de una persona sana, sino que podría contener patrones de enfermedad distintos de los 14 enumerados o hallazgos inciertos dentro de las 14 categorías posibles.

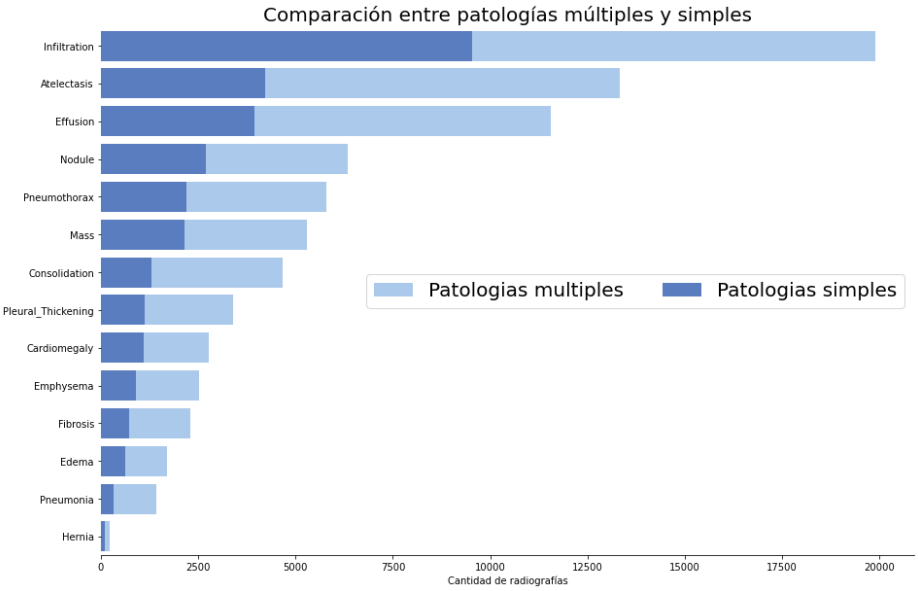


Figura 3.2 *- Frecuencia de aparición de las 14 patologías en las etiquetas simples y en las multi-etiquetas asociadas a las radiografías que componen la base de datos.*

La existencia de más de una patología asociada a ciertas radiografías permite analizar la correlación entre las mismas, la Figura 3.3 revela algunas conexiones entre diferentes patologías, que concuerdan con el conocimiento del dominio de los radiólogos, por ejemplo, que la infiltración se asocia a menudo con la atelectasia y la efusión.

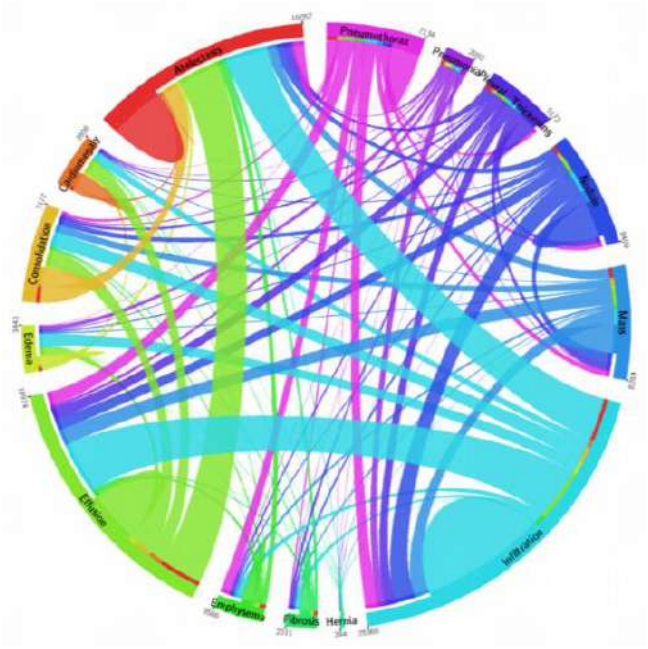


Figura 3.3 *- Correlación existente entre las patologías asociadas a cada una de las radiografías del conjunto de datos [4].*

La Figura 3.4 muestra la frecuencia de aparición de las etiquetas simples, observando una clara diferencia en la frecuencia entre "*No Finding*" (sin hallazgo) y las distintas clases de patología, donde más de 60.000 radiografías no tuvieron ningún hallazgo.

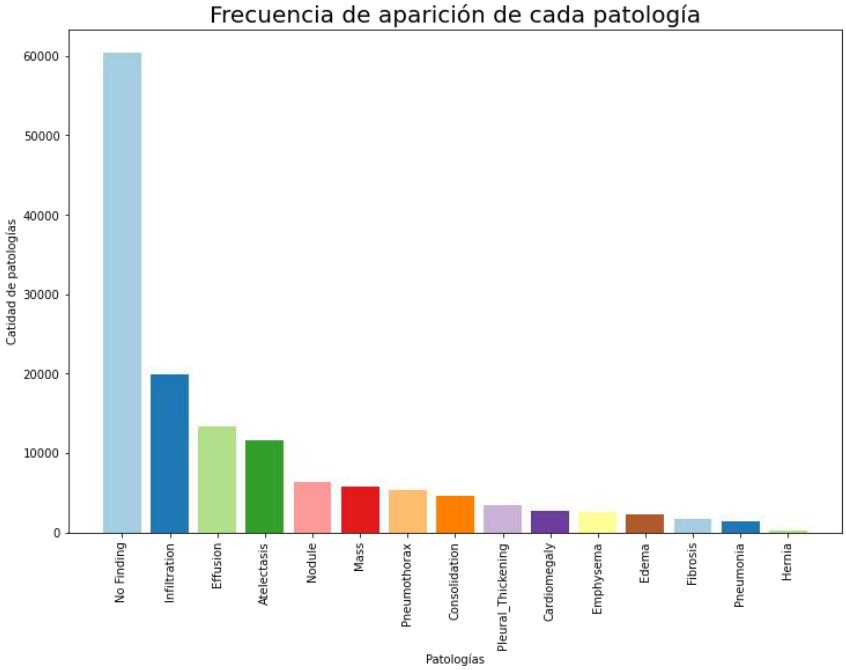


Figura 3.4 *- Gráfico de barras que muestra la frecuencia de aparición de cada etiqueta simple.*

El trabajo con una base de datos como lo es ChestX-ray14 supuso un gran desafío tanto para su estudio como para el desarrollo de DNN por las razones siguientes:

- La resolución de una radiografía de tórax suele ser de 2000×3000 píxeles, mientras que los hallazgos patológicos son a menudo significativamente más pequeños en comparación con la imagen completa y, por lo tanto, son complejos de detectar (Figura 3.5).

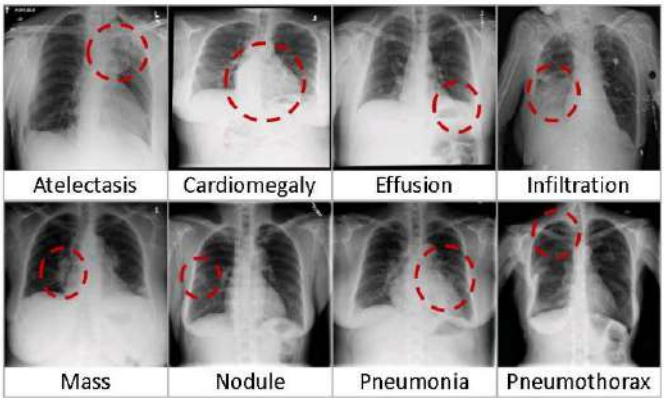


Figura 3.5 *- Imágenes de radiografías de tórax pertenecientes a la base de datos ChestX-ray14 con los respectivos hallazgos patológicos marcados en rojo [4].*

- El conjunto de datos se encuentra desbalanceado, es decir, se cuenta con distinto número de observaciones para las distintas combinaciones de etiquetas, siendo significativamente mayor el número de ejemplo de la clase "*No Finding*" (ver Figura 3.4). Por lo general, los conjuntos desbalanceados tienden a afectar a los algoritmos de aprendizaje automático en su proceso de generalización, desfavoreciendo a las clases menos presentes. Se debe contar con alguna estrategia para manejar la frecuencia de aparición de las etiquetas previo al entrenamiento.
- Al construir la base de datos, el proceso seguido por el equipo del NIH no fue manual, sino que se utilizaron técnicas de minería de textos sobre los informes radiológicos para obtener las etiquetas. Si bien estiman que tienen un 90% de exactitud en comparación con los informes originales, no es posible de verificar debido a que los informes utilizados no se pueden publicar por razones de confidencialidad.
- Los diagnósticos de algunos médicos no siempre son aceptados por otros especialistas. Por lo tanto, al no realizar un etiquetado manual con un grupo de expertos, es probable que algunas de las imágenes del conjunto estén mal etiquetadas.
- Algunas de las radiografías de tórax incluidas en el conjunto se encuentran rotadas 90 grados a la derecha o izquierda, o 180 grados. Así como también

existen algunos estudios ampliados, con bordes negros o blancos. No es posible identificar estos casos aislados a través de la información brindada por el conjunto de datos (no están identificados), por lo que no se pueden filtrar estas imágenes.

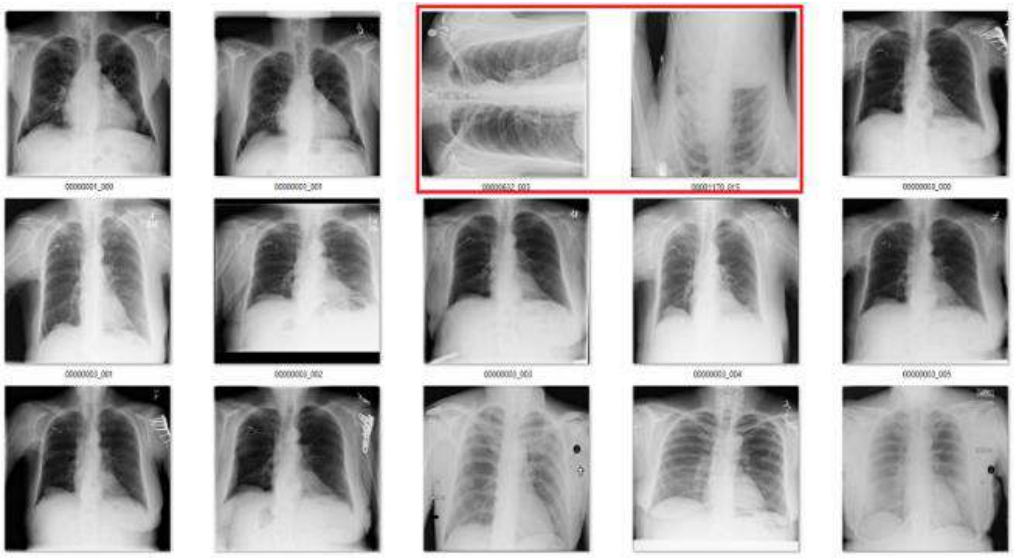


Figura 3.6 *- Radiografías rotadas en el conjunto de datos ChestX-ray14* [30].

- Los radiólogos que interpretaron las imágenes originales tenían alguna otra información para asistirlos, como un historial clínico, resultados anteriores, etc. Por lo que, si los expertos humanos no pueden hacer un diagnóstico sólo a partir de las imágenes, entrenar una DNN para que realice el diagnóstico sólo a partir de imágenes supone un gran desafío.

A pesar de todos los retos que representa la base de datos, ChestX-ray14 continúa siendo hasta la fecha de hoy el repositorio público de radiografías de tórax con mayor cantidad de datos disponibles y, por lo tanto, se convierte en la mejor opción para evaluar la capacidad de detección de patologías a través de radiografías de tórax y *deep-learning*, razón por la cual fue seleccionada en este Proyecto Final.

3.3 Modelos implementados

En esta sección se detallarán las arquitecturas de CNN que fueron implementadas.

3.3.1 Modelos a partir de CNN pre-entrenadas

En una primera etapa del presente Proyecto Final se empleó la técnica de *transfer-learning* a partir de las siguientes CNN pre-entrenadas: VGG19 [20], InceptionV3 [21] y ResNet50 [22], descritos en el apartado 2.5 del capítulo anterior. En cada caso, se utilizó la fase de extracción de características del modelo base, congelando todas las capas

para evitar la pérdida de información, con el propósito de que las salidas de éstas (las características) sean las entradas a un nuevo clasificador. Como fase de clasificación se agregó una red multicapa totalmente conectada. A fin de evaluar la capacidad de detección de patologías en radiografías de tórax, se evaluaron diferentes problemáticas, dando lugar a diferentes arreglos experimentales, los cuales se explican en detalle en las siguientes secciones. Variando las fases de clasificación, se propusieron las siguientes arquitecturas:

- **Esquema #1:** Una capa totalmente conectada (dense) de 512 neuronas, una capa de regularización con *dropout* y finalmente una capa de salida con igual cantidad de neuronas que clases, ajustada según el problema explorado (ver sección 3.4).
- **Esquema #2:** Una capa totalmente conectada (dense) de 512 neuronas seguida de una capa de regularización con *dropout*. A continuación, otra capa totalmente conectada con 50 neuronas seguida de otra capa *dropout*. Finalmente, una capa de salida con igual cantidad de neuronas que clases, ajustada según el problema explorado (ver sección 3.4).

En la Figura 3.7 se muestra un diagrama general de los modelos implementados basados en *transfer-learning*.

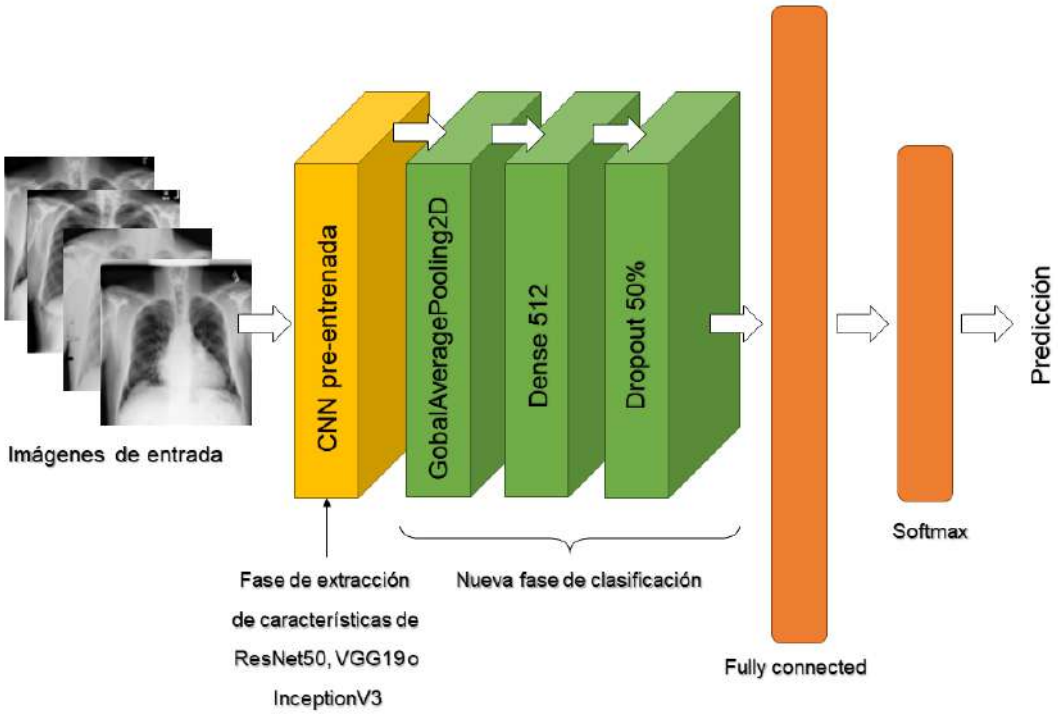


Figura 3.7 *- Arquitectura general del modelo basado en transfer-learning utilizado.*

3.3.2 *Modelo de CNN sin transfer-learning*

Se planteó evaluar el desempeño de una CNN implementada sin *transfer-learning*, como uno de los objetivos del Proyecto Final. Teniendo en cuenta artículos anteriores al presente trabajo en los que se analizan radiografías de tórax [31], se propuso la arquitectura que se muestra en la Figura 3.8.

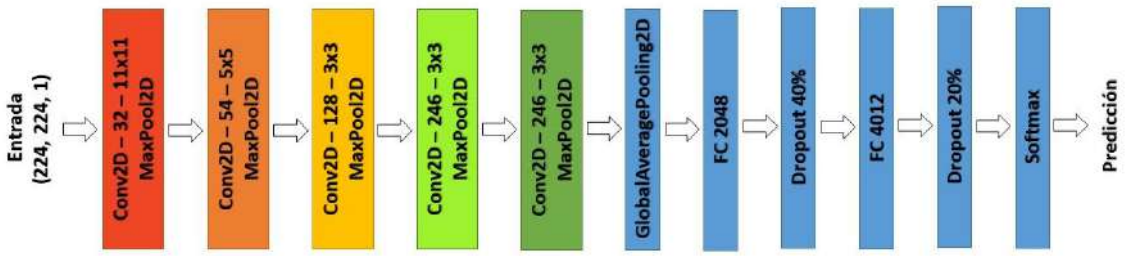


Figura 3.8 *- Arquitectura de CNN propuesta para evaluar la clasificación de radiografías de tórax sin uso de transfer-learning.*

La CNN propuesta cuenta con cinco bloques de convolución que constituyen la fase de extracción de características, donde se utilizan filtros de 11x11, 5x5, y 3x3. Por su parte, la fase de clasificación, cuenta con 2 capas completamente conectadas; dos capas de *dropout*; y una capa de salida con igual cantidad de neuronas que clases, ajustada según el problema explorado. Se decidió reducir el tamaño de las imágenes de entrada a 224x224 píxeles. Finalmente, la red construida cuenta con 10.721,190 parámetros entrenables.

3.4 Experimentos realizados

Con el fin de analizar diferentes aspectos de la capacidad de detección de patologías en radiografías de tórax se definieron diferentes experimentos; utilizando en las evaluaciones los modelos detallados en la sección 3.3.

3.4.1 *Arreglo experimental #1: Evaluación de robustez en la clasificación según orientación*

Se analizaron los cambios en la performance del modelo de clasificación según se utilicen para su ajuste todas las radiografías o se filtren por orientación, con el fin de evaluar la viabilidad del requerimiento de separar las radiografías según su orientación. Se realizó un filtrado de las imágenes entre radiografía anteroposterior (AP) y posteroanterior (PA). Aunque ambas son radiografías frontales, son fundamentalmente diferentes en cuanto al método con el que se obtienen y la imagen de rayos-X resultante. Las radiografías PA se realizan de atrás hacia delante, mientras que las radiografías AP se realizan de adelante hacia atrás. Las radiografías AP generalmente no se prefieren,

excepto en situaciones en las que el paciente demasiado débil o no puede adoptar una posición erguida [32].

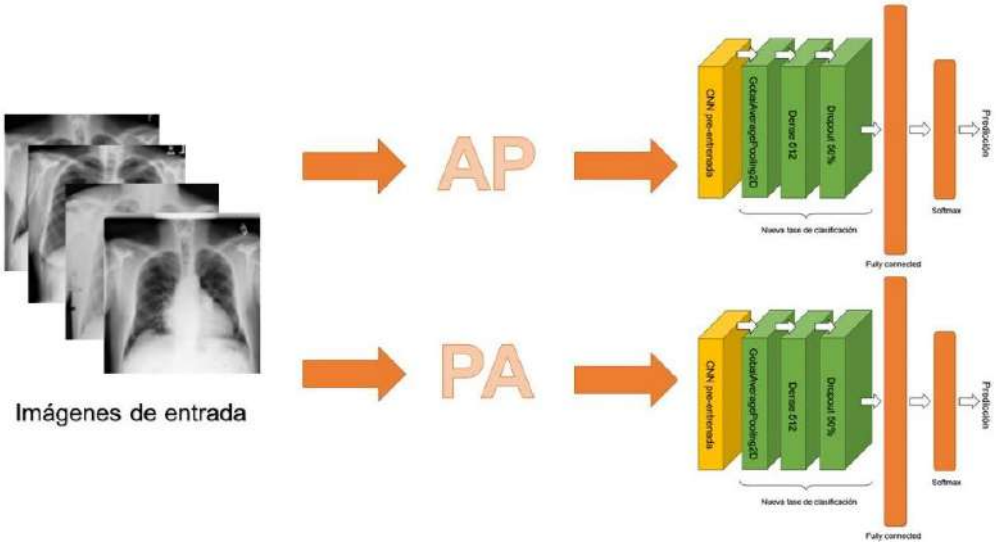


Figura 3.9 *- Filtrado AP y PA.*

3.4.2 Arreglo experimental #2: Clasificación entre sin hallazgo y una patología específica

Con el fin de evaluar la capacidad de las DNN para identificar patologías presentes en radiografías de tórax, se evaluaron modelos para clasificar las imágenes de rayos-X correspondientes a “sin hallazgo” vs una patología específica. Se trabajó sólo con muestras etiquetadas con una única patología (patología pura) y sólo con los 6 casos más frecuentas en la base de datos, debiendo construir subconjuntos de imágenes a partir de la base de datos original. Fueron analizados los casos siguientes:

1. Masa + Sin hallazgo;
2. Nódulo + Sin hallazgo;
3. Infiltración + Sin hallazgo
4. Cardiomegalia + Sin hallazgo;
5. Neumotórax + Sin hallazgo;
6. Atelectasia + Sin hallazgo.

Dado que el número de ejemplos de cada patología pura y el número de sin hallazgo son distintos (son clases desbalanceadas), se programó e implementó un balanceo del conjunto de imágenes, sub-muestreando sin sustitución la clase “sin hallazgo” (mayoritaria) hasta igualar la clase “patológica” (minoritaria).

Para cada uno de los 6 problemas de clasificación, se realizaron 5 ciclos de entrenamiento de 80 épocas cada uno, dividiendo los datos en 80% para entrenamiento

y 20% para *test* y estimando la exactitud como el promedio entre las 5 iteraciones en cada caso. Se repitió el mismo procedimiento utilizando técnicas de *data-augmentation* sobre los conjuntos de datos obtenidos.

3.4.3 Arreglo experimental #3: Clasificación entre "Hallazgo" y "Sin Hallazgo"

Para evaluar la capacidad de las DNN para detectar las patologías incluidas en la base de datos y los casos "sin hallazgo", se construyó un nuevo conjunto de datos conteniendo todos los ejemplos disponibles, etiquetando todos los casos identificados con, al menos, 1 patologías con la etiqueta "hallazgo" (Figura 3.10).

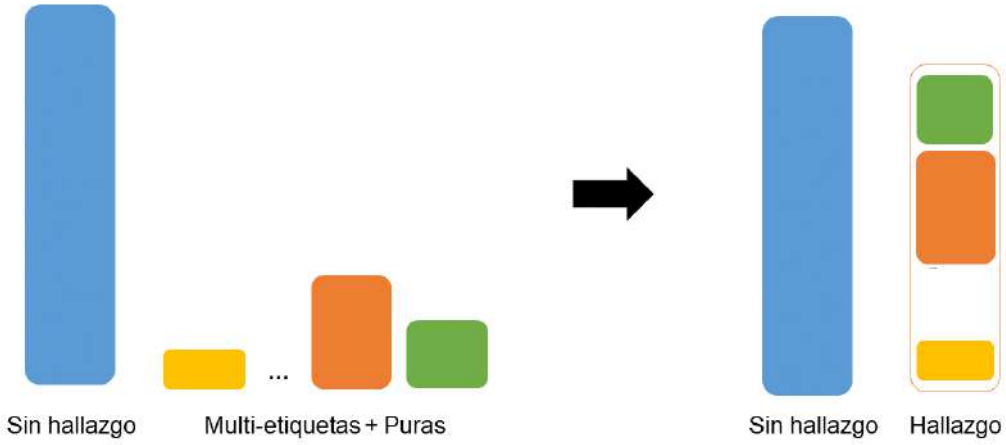


Figura 3.10 *- Arreglo experimental #3: Clasificación entre "Hallazgo" y "Sin Hallazgo".*

Se realizó un único ciclo de entrenamiento de 150 épocas, dividiendo los datos en 80% para entrenamiento y 20% para *test*. Se repitió el mismo procedimiento utilizando técnicas de *data-augmentation* sobre el conjunto de datos.

3.4.4 Arreglo experimental #4: Clasificación entre distintas patologías puras

Con el fin de evaluar la capacidad de los modelos basados en DNN para la clasificación entre distintas patologías, se trabajó con ejemplos puros de cada etiqueta, descartando las no identificadas ("sin hallazgo") y sub-muestreando sin sustitución 500 muestras puras de cada etiqueta. Aquellos casos con menos de 500 muestras se descartaron. Los casos utilizados se muestran en el gráfico de la Figura 3.11.

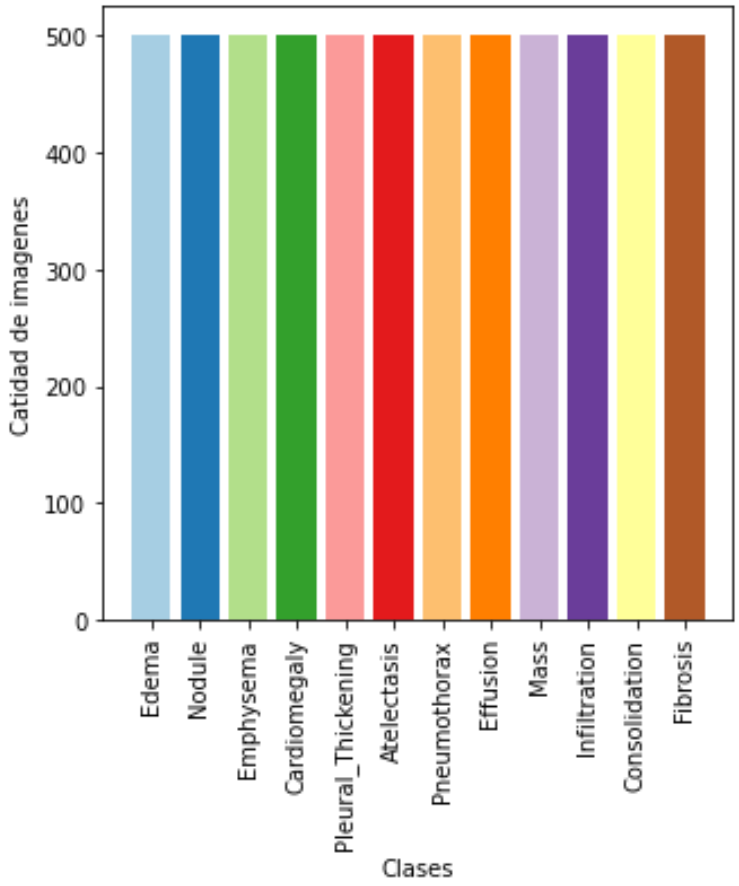


Figura 3.11 *– Gráfico de barras con las patologías puras utilizadas para el arreglo experimental #4.*

Se realizaron 5 ciclos de entrenamiento de 150 épocas cada uno, dividiendo el conjunto de patologías puras en 80% para entrenamiento y 20% para *test*. Se estimó la exactitud como el promedio entre las 5 iteraciones en cada caso. Se repitió el mismo procedimiento utilizando técnicas de *data-augmentation* sobre el conjunto de datos.

3.5 Entorno de desarrollo y programas implementados

Se utilizó el lenguaje de programación *Python* en la versión 3.8.5, valiéndose de la biblioteca de aprendizaje profundo *Keras* en la versión 2.4.3, con *TensorFlow* como *backend*, para la implementación de los modelos de clasificación descritos en el apartado 3.3. La razón fundamental por la que se eligió *Python* para el desarrollo del Proyecto Final es debido a que es un lenguaje que cuenta con una amplia variedad de librerías para análisis de datos y aprendizaje profundo, con extensa documentación y una gran comunidad de programadores que lo respalda.

El código se estructuró en funciones, cada una con una tarea bien definida, buscando crear un código claro y legible. Se definieron funciones para la carga de los datos y para su preprocesamiento; para la obtención de los conjuntos de datos necesarios para cada arreglo experimental propuesto; para la separación en datos de entrenamiento y *test*;

para generar *batches* de imágenes tensoriales con aumento de datos en tiempo real; para la creación de los modelos de clasificación y para el ajuste de los mismos; para graficar los resultados obtenidos y también para su análisis. El código escrito fue pensado para ser reutilizado en cada experimento, con una adecuada documentación, realizando sólo pequeñas variaciones, economizando tiempo y reduciendo la redundancia.

Para llevar a cabo el desarrollo de los códigos se utilizó *Jupyter Notebook*, la cual es una aplicación cliente-servidor desarrollada en el año 2015 por el Proyecto *Jupyter*. Es posible describir una *Jupyter Notebook* como una "narrativa computacional", un documento que permite publicar código y datos con análisis, hipótesis y conjeturas de forma legible y ejecutable. Los archivos de notebooks tienen un formato JSON sencillo y documentado, con la extensión ".ipynb". Se eligió trabajar con esta aplicación pues se accede través de un navegador web, lo que hace que sea posible utilizar la misma interfaz ejecutándose localmente como una aplicación de escritorio, así como en un servidor remoto [33], [34].

Para realizar todas las pruebas se ejecutó *Jupyter Notebook* sobre un servidor remoto perteneciente al Laboratorio de Procesamiento de Imágenes del ICyTE, accediendo a través de un navegador web local y conectándose por medio de una conexión SSH. El protocolo SSH utiliza encriptación para asegurar la conexión entre el cliente y el servidor, donde toda la autenticación del usuario, los comandos, la salida y las transferencias de archivos se encriptan para protegerlos de posibles ataques en la red [35].

Se utilizó *Conda* que es un sistema de gestión de paquetes de código abierto y un sistema de gestión de entornos. Una de las ventajas de *Conda* es que permite configurar entornos virtuales para ejecutar versiones diferentes tanto del lenguaje de programación *Python* como de sus librerías. Se creó un entorno virtual específico para el Proyecto Final, aislándolo de otros proyectos con otros requisitos.

Como al trabajar con *Deep-Learning* se tienen largos tiempos de entrenamiento, para conservar las sesiones activas en el servidor remoto se utilizó *Screen* o *GNU Screen* el cual es un multiplexor de terminales. En otras palabras, es posible iniciar una sesión de pantalla (*screen session*) y luego abrir varias terminales virtuales dentro de la misma sesión, con la posibilidad de desconectarse y que los procesos continúen ejecutándose. Al establecer una nueva conexión se puede volver a acceder a las terminales sin que los procesos en ejecución sean interrumpidos [36].

3.5.1 Obtención de nuevos conjuntos de datos a partir del dataset original ChestX-ray14

Como fue explicado en la sección 3.4, se definieron distintos arreglos experimentales, para los cuales fue necesario crear nuevos conjuntos de datos a partir de la base de datos original. Con el objetivo de trabajar en la obtención de dichos conjuntos se utilizó la librería *open source Numpy* [37] que da soporte para el manejo de vectores y matrices en *Python*, conjuntamente con *Pandas* que es una librería especializada en el manejo y análisis de estructuras de datos.

En esta instancia, se utilizó el archivo CSV incluido en la base de datos ChestX-ray14 que contiene las etiquetas de clase, las rutas de las imágenes, información sobre las radiografías e información de los pacientes para todo el conjunto de datos. En particular, se trabajó con las rutas de las radiografías y las etiquetas asociadas, descartando cualquier otro tipo de dato innecesario para el análisis propuesto. Sólo para el caso particular del arreglo experimental #1 se conservó también la orientación de las radiografías. El principal desafío en esta instancia fue crear funciones que cumplieran con los requerimientos específicos para cada problemática planteada.

En el caso de los arreglos experimentales #2 y #4, se debía cumplir el requerimiento de que las etiquetas fuesen puras, es decir, las radiografías debían presentar una única patología asociada. Para esto, se programaron funciones que permiten el filtrado de la base de datos según categorías, filtrando en este caso según la etiqueta. Esta misma función se utilizó en el caso del arreglo #1 para filtrar las imágenes según la orientación. Por otro lado, para obtener el arreglo #3 se debió re-etiquetar todas las imágenes de rayos-X patológicas (fuesen puras o no) como “hallazgo”, generando un nuevo conjunto de datos con las categorías “sin hallazgo” y “hallazgo”, para lo cual se desarrolló un programa específico.

El notable desbalanceo de los datos fue uno de los grandes desafíos del Proyecto Final, para lo cual se propuso la estrategia mencionada en el apartado 3.4.2 para el arreglo experimental #2. Se programó una función de balanceo de datos, para lo cual fue necesario obtener la cantidad de muestras pertenecientes a la clase patológica pura y luego realizar un submuestreo sin sustitución sobre la clase mayoritaria hasta igualar la cantidad de la clase minoritaria. Reutilizando la misma función para todos los conjuntos compuestos por “patología pura” y “sin hallazgo”. Esta misma lógica se utilizó en el arreglo #4 para obtener un conjunto de datos conformado por imágenes de radiografías etiquetadas con una única patología, submuestreando sin sustitución el conjunto en una cantidad determinada pasada como parámetro.

En todos los casos se requería generar un único conjunto de datos resultante. Sin embargo, en los casos en los que se utilizó filtrado se obtuvieron múltiples conjuntos, cada grupo perteneciente a cada categoría filtrada (las 15 categorías que dispone el conjunto de datos para el arreglo #2 y todas las etiquetas puras tal que su frecuencia sea mayor o igual a 500 para el arreglo #4). Por tanto, para cumplir con el requerimiento, se debieron concatenar los conjuntos obtenidos.

Un contratiempo que aconteció durante el desarrollo fue que la función de concatenación que provee la librería Pandas genera un nuevo conjunto de datos con las etiquetas en el mismo orden en que se encadenaron los conjuntos individuales. Por lo tanto, no puede asegurarse el balanceo en particiones realizadas luego de una concatenación mediante esta librería, lo que perjudica la capacidad de generalización de los modelos y, en consecuencia, la calidad del experimento realizado y las conclusiones que puedan extraerse. Se resolvió programar una función específica para la partición y concatenación de los conjuntos de datos conservando el balanceo entre clases.

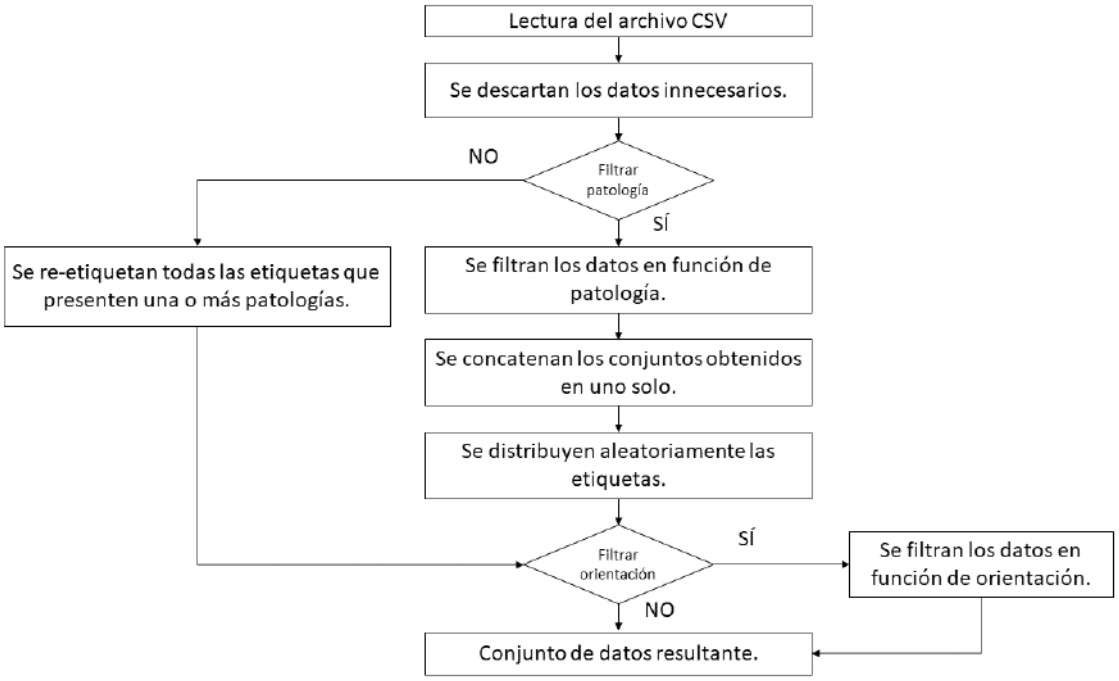


Figura 3.12 *– Diagrama de flujo del trabajo realizado a partir del archivo CSV de la base de datos ChestX-ray14* [30].

3.5.2 Generación de batches de imágenes

La librería *Keras* proporciona utilidades para el preprocesamiento de conjuntos de datos con la API *preprocessing*, la cual dispone de clases y funciones para trabajar con imágenes, que ayudan a pasar de los datos brutos en el disco a un objeto que puede

utilizarse para entrenar un modelo. Fue utilizada para preprocesar los datos de entrada, así como para aplicar *data-augmentation* sobre los mismos.

Contiene a la clase *ImageDataGenerator*, también conocida como generador, que permite establecer parámetros para el preprocesamiento de los datos de entrada a un modelo. Por un lado, para los experimentos realizados con CNN pre-entrenadas, se recurrió a esta clase y se aplicó sobre el conjunto de radiografías el mismo preprocesamiento utilizado en ResNet, VGG e Inception. Por otro lado, para la arquitectura propuesta sin *transfer-learning,* los niveles de gris de las radiografías fueron normalizados al intervalo $[0,1]$. En todos los casos, se programaron funciones específicas.

La mayoría de los conjuntos de datos para problemas de clasificación en *deep-learning* contienen todas las imágenes en directorios separados con los nombres de las respectivas clases (ver Figura 3.13), lo cual permite leer fácilmente las imágenes desde el disco. Sin embargo, este no es el caso de ChestX-ray14, donde todas las imágenes están presentes dentro de un único directorio y sus respectivas etiquetas están mapeadas en un archivo CSV, tal como se mencionó en el subapartado anterior.

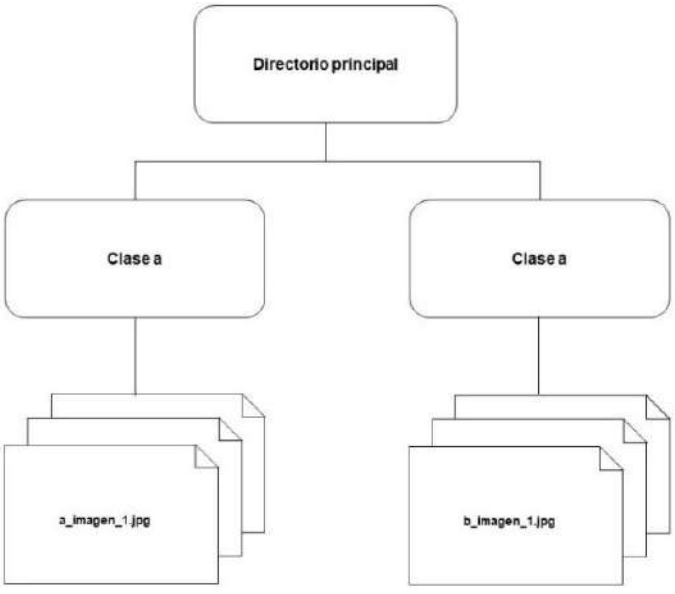


Figura 3.13 *- Estructura de directorio mayoritariamente utilizada para un problema de clasificación binario.*

Para la generación de los *batches* de imágenes, se recurrió al método *flow_from_dataframe*, de la clase *ImageDataGenerator*, el cual permite tomar como parámetros de entrada los *dataframes* generados con las funciones de generación de conjuntos de datos descriptos anteriormente en la sección 3.5.1 [38]. El método retorna un iterador que produce tuplas (x, y) donde “x” es una matriz de lotes o *batches* de

imágenes tensoriales e “y” es un vector con las etiquetas correspondientes. El tamaño de lote o *batch* es el número de imágenes que se utilizan en cada paso del entrenamiento para entrenar un modelo antes de actualizar sus parámetros entrenables. El mismo está limitado por la memoria disponible en la GPU, por lo que se definió un tamaño de *batch* de 128 imágenes y para la validación uno de 256 imágenes. Como resultado de esta etapa, se generaron funciones específicas para la definición de batches para ser utilizadas en el entrenamiento y evaluación de los modelos.

3.5.2.1 Aumento de datos con la clase *ImageDataGenerator*

La clase *ImageDataGenerator* también permite aplicar aumento de datos. El *augmentation* de *Keras* está enfocado en generar diversidad en los datos mostrados al modelo época a época, manteniendo la cantidad original de datos. Al instanciar la clase *ImageDataGenerator* se definen transformaciones y luego, durante el entrenamiento, se arman los *batches* con imágenes transformadas en tiempo real. Busca aprovechar al máximo los ejemplos de entrenamiento de modo que el modelo nunca vea dos veces exactamente la misma imagen, esto ayuda a evitar el sobreajuste y a que el modelo generalice mejor [39].

Para implementar *data-augmentation* se definió una función de manera tal que retornara la instancia a la clase *ImageDataGenerator* configurada para realizar transformaciones sobre los datos. Teniendo en cuenta artículos anteriores que analizaban imágenes de radiografías de tórax [32], se definieron las siguientes transformaciones:

- Una rotación aleatoria dentro del rango de -5/+5 grados.
- Un desplazamiento horizontal y vertical aleatorio del 5%.
- Un rango de aumento aleatorio del 15%.
- Una distorsión aleatoria a lo largo de un eje de 0,1 grados, para crear o rectificar los ángulos de percepción.
- Un *padding* para mantener constante el tamaño de las imágenes de entrada, reflejando los píxeles vecinos para completar los espacios con píxeles faltantes.

La función programada hace uso del método *flow_from_dataframe* para aplicar las transformaciones definidas por el generador y armar los *batches* de imágenes. Se aplicó un aumento de datos dinámico llamando a la función en cada época, durante el entrenamiento del modelo.

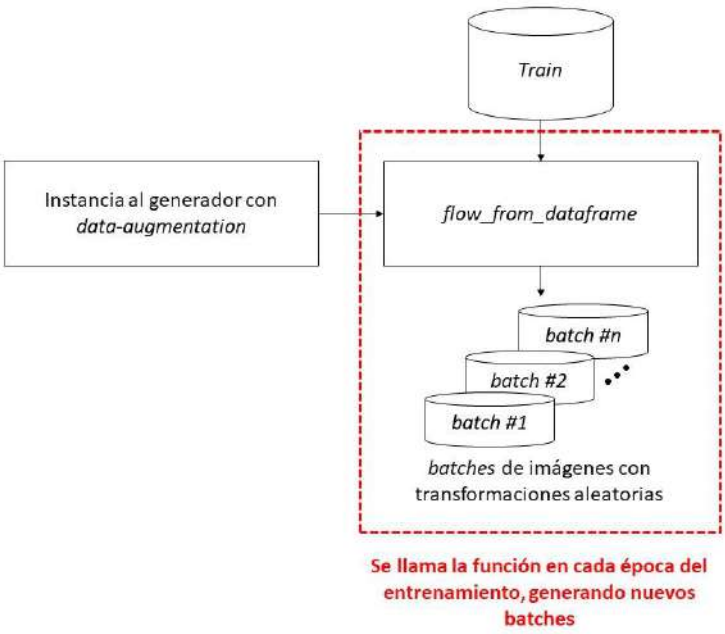


***Figura 3.14** – Flujo de trabajo con la librería preprocessing de Keras para aplicar data-augmentation dinámico al conjunto de datos de entrenamiento.*

Para evaluar correctamente la capacidad de generalización del modelo no se aplicó *data-augmentation* sobre el conjunto de datos de validación, por lo que se debió programar otra función con otro generador sin incluir los parámetros de transformaciones mencionados anteriormente. En este caso, para generar los *batches* de imágenes la implementación utiliza el método *flow_from_dataframe* previamente al entrenamiento del clasificador.

3.5.3 Implementación de los modelos de clasificación utilizando Keras

Los modelos de clasificación fueron creados utilizando la API *functional* [40] de *Keras* que permite manejar modelos con topología no lineal y construir redes más complejas que no pueden ser logradas por el modelo secuencial de *Keras*. Se decidió utilizar esta API debido a que resulta muy útil para implementar *transfer-learning*. Tiene la capacidad para manejar capas compartidas, lo que significa que puede acceder a las activaciones de capas intermedias de un modelo y reutilizarlas en otro lugar, permitiendo hacer uso de sus características para crear nuevos modelos de extracción de características que devuelvan los valores de las activaciones de las capas intermedias.

La librería *Keras* cuenta también con la API *Applications* la cual pone a disposición modelos de aprendizaje profundo junto con los pesos pre-entrenados y se utilizó para descargar las CNN ResNet50, VGG19 e InceptionV3 sobre las cuales se implementaron modelos de *transfer-learning*, tal como se describió en la sección 3.3.1. Se desarrolló un código específico para congelar los pesos de las capas convolucionales (evitando su

ajuste durante el entrenamiento) y también para extraer la fase de extracción de características para utilizarla posteriormente en los modelos basados en *transfer-learning*.

Tanto para implementar la arquitectura propuesta en la sección 3.3.2, como para implementar la fase de clasificación de los modelos obtenidos a partir de CNN pre-entrenadas, se escribieron programas específicos en los que se utiliza la clase *Model* con el objeto de crear el modelo y la clase *Layers* para instanciar las capas necesarias y definir sus parámetros (cantidad de neuronas, funciones de activación, filtros, tamaño de los filtros, tipo de *pooling*, *dropout*, entre otros), ambas clases pertenecen a la librería *Keras*. Por defecto, en *Keras*, las capas se encuentran descongeladas, para ajustar sus pesos durante el entrenamiento de la red.

Para la creación de las arquitecturas propuestas utilizando *transfer-learning* y *fine-tuning* se programó una función que retorna el modelo, a la cual se le pasan como parámetros el modelo base y el esquema de clasificación a adoptar; y la cantidad de clases según el arreglo experimental. Por otro lado, se desarrolló también una función específica para retornar la arquitectura propuesta sin *transfer-learning*.

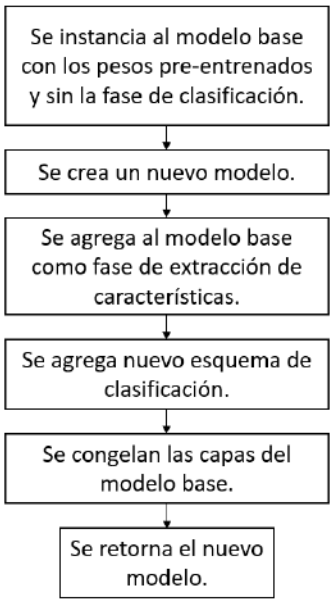


***Figura 3.15** – Flujo de trabajo para la creación de las arquitecturas propuestas utilizando transfer-learning y fine-tuning.*

3.5.4 Validación de los modelos implementados

A partir de la API *Scikit-Learn* se generaron funciones específicas para generar las particiones necesarias para implementar validación mediante *hold-out*. Concretamente se utilizó el método *train_test_split* que divide el conjunto de datos en subconjuntos

aleatorios de entrenamiento y prueba. Se dividieron de manera estratificada, realizando una división de forma tal que la proporción de etiquetas en la muestra producida sea la misma que la proporción de valores proporcionada al parámetro estratificar (las etiquetas de cada arreglo experimental).

Los conjuntos de datos de testeo y de prueba obtenidos fueron almacenados en la memoria del servidor en archivos CSV, con el objeto de poder reproducir los experimentos en el mismo entorno y con los mismos datos en caso de que fuese necesario.

3.5.5 Entrenamiento de la red

Para las distintas variantes definidas por los arreglos experimentales, se programó una función específica, a la cual se le envían como parámetro: el modelo (la CNN a entrenar), la ruta para guardar los resultados, el número de épocas, el número de clases, los *dataframes* de entrenamiento y validación, la función de preprocesamiento de las imágenes de entrada, la función de pérdida y el optimizador y, finalmente, la métrica a utilizar. Esta función incluye el llamado a las funciones previamente descriptas para generar los batches.

3.5.6 Herramientas de visualización

Como herramienta de visualización se recurrió a *Matplotlib*, una biblioteca de visualización de datos multiplataforma construida sobre matrices *NumPy*; y *Seaborn*, que proporciona una API sobre *Matplotlib* que define funciones simples de alto nivel para los tipos de gráficos estadísticos comunes, y se integra con la funcionalidad proporcionada por los *dataframes* de Pandas [41].

Se crearon funciones específicas utilizando las librerías mencionadas. Por un lado, realizar análisis exploratorio sobre los datos, visualizando la frecuencia de aparición de diferentes patologías y también las radiografías. Por otro lado, para representar gráficos de entrenamiento donde es posible visualizar la exactitud de entrenamiento y del *test* en función de la época, así como la pérdida.

4 Resultados y discusión

4.1 Introducción

Este Capítulo tiene el objetivo de detallar los resultados obtenidos para los experimentos realizados en el Proyecto Final, valiéndose de tablas y gráficos; y también de realizar un análisis sobre los mismos.

En primer lugar, se explica la experimentación y el análisis llevado a cabo para la selección del modelo base (CNN pre-entrenada) para las pruebas con *transfer-learning*. En segundo lugar, se muestran los resultados de los arreglos experimentales #1 a #4 para *transfer-learning*. Por último, se evalúa la arquitectura de CNN *ad-hoc* sobre el arreglo experimental que mostró los mejores resultados.

4.2 Selección del modelo base y el esquema de clasificación para *transfer-learning*

En una primera instancia del desarrollo de este Proyecto Final, se realizó un estudio de la capacidad de clasificación de radiografías de tórax de CNN definidas a partir de *transfer-learning*. Partiendo de las CNN pre-entrenadas ResNet50, VGG19 e InceptionV3, descriptas en el capítulo anterior, y considerando las dos estructuras de perceptrones multicapa de la fase de clasificación (esquemas #1 y #2 definido en la sección 3.3.1). En la Tabla 4.1 se muestra un resumen de las arquitecturas estudiadas en la primera etapa del Proyecto.

Tabla 4.1 *- Arquitecturas propuestas a partir de CNN pre-entrenadas.*

Arquitectura propuesta	Modelo base	Esquema Perceptrón multicapa
#1	ResNet50	#1
#2	ResNet50	#2
#3	VGG19	#1
#4	VGG19	#2
#5	InceptionV3	#1
#6	InceptionV3	#2

Es importante remarcar que la experimentación se realizó de manera equitativa para cada una de las arquitecturas propuestas, buscando que los resultados obtenidos sean comparables entre sí. Por este motivo, se utilizaron los mismos hiperparámetros para el entrenamiento en cada una de las experimentaciones, los cuales se pueden ver en la Tabla 4.2.

Tabla 4.2 *– Hiperparámetros de entrenamiento utilizados para los experimentos.*

Hiperparámetro	Valor
Algoritmo de optimización	Descenso estocástico del gradiente
Algoritmo de pérdida	Entropía cruzada
Épocas	150
Tamaño de lote de entrenamiento	128
Tamaño de lote de validación	256

En la búsqueda de seleccionar el mejor modelo base y el mejor esquema de clasificación, se entrenó a los clasificadores ajustando sus parámetros entrenables con los conjuntos de datos obtenidos para el arreglo experimental #2, es decir, la clasificación entre sin hallazgo y una patología específica, los cuales se muestran en la Tabla 4.3.

Tabla 4.3 *– Subconjuntos conformados por patología pura y "sin hallazgo".*

Conjunto de datos	Número total de imágenes
Atelectasia + Sin Hallazgo	8430
Cardiomegalia + Sin Hallazgo	2186
Infiltración + Sin Hallazgo	19094
Masa + Sin Hallazgo	4278
Nódulo + Sin Hallazgo	5410
Neumotórax + Sin Hallazgo	4388

Los mejores resultados de exactitud fueron mostrados para la arquitectura #1, conformada por la CNN pre-entrenada ResNet50 y el primer esquema de clasificación propuesto, los mismos se pueden visualizar en la Tabla 4.4.

Tabla 4.4 *– Resultados de exactitud promedio para la arquitectura #1.*

Conjunto de datos	Exactitud promedio	STD
Atelectasia + Sin Hallazgo	68,5%	0,010
Cardiomegalia + Sin Hallazgo	73,6%	0,012
Infiltración + Sin Hallazgo	62,0%	0,009
Masa + Sin Hallazgo	66,6%	0,013
Nódulo + Sin Hallazgo	64,0%	0,005
Neumotórax + Sin Hallazgo	75,3%	0,012

Las diferencias de exactitud observadas con respecto al resto de arquitecturas pre-entrenadas y esquemas de clasificación propuestos no fueron demasiado significativas, pero sí menores, por lo que se optó por continuar con los experimentos siguientes de *transfer-learning* utilizando la arquitectura #1, es decir, combinando la fase de clasificación de la ResNet50 y el esquema de fase de clasificación #1. Cabe recalcar

que todos los casos presentaron *overfitting*, el cual será analizado con más detalle más adelante en este Capítulo.

4.3 Resultados *transfer-learning*

4.3.1 Resultados del arreglo experimental #1: Evaluación de robustez en la clasificación según orientación

Después de haber evaluado y comparado los diferentes modelos propuestos, se concluyó que ResNet50 junto con el perceptrón multicapa #1 fue el modelo que mejores resultados mostró para la identificación de patologías puras en radiografías de tórax.

Partiendo de ResNet50 y el esquema de fase de clasificación #1, se analizó el problema definido en el arreglo experimental #1, es decir, la necesidad o no de separar las radiografías según su orientación para su posterior clasificación. Tal como se detalló en el capítulo anterior, se realizó un filtrado de las imágenes de la base de datos según orientación anteroposterior (AP) y posteroanterior (PA). Debido a los extensos tiempos de cómputo necesarios para el entrenamiento de la CNN, se tomó la decisión de realizar el análisis sólo para la clasificación entre “sin hallazgo” y “masa”. El resultado de este filtrado dio origen a dos subconjuntos, resumidos en la Tabla 4.5. Tal como se indicó al definir el esquema de validación, se ejecutaron 5 ciclos de entrenamiento/test. Los resultados se presentan en la Tabla 4.6.

Tabla 4.5 *– Conjuntos obtenidos al filtrar por orientación.*

Conjunto de datos	Número total de imágenes
AP	1544
PA	2734

Tabla 4.6 *– Comparación de resultados obtenidos filtrando por orientación y sin filtrar.*

Orientación	Exactitud promedio	STD
Sin filtrar	66,62%	0,013
AP	64,87%	0,009
PA	65,78%	0,016

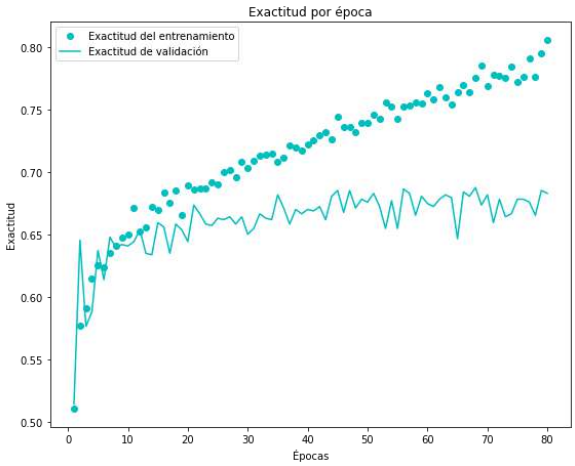


Figura 4.1 *– Gráfico de exactitud por época obtenido para el primer ciclo de entrenamiento/test para el conjunto de "masa" vs "sin hallazgo".*

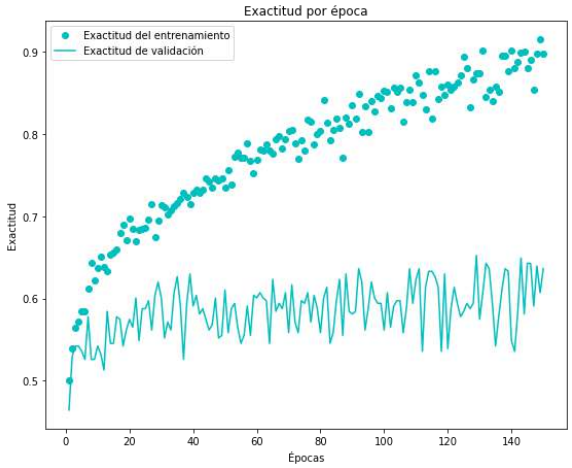


Figura 4.2 *– Gráfico de exactitud por época obtenido para el primer ciclo de entrenamiento/test para el conjunto de "masa" vs "sin hallazgo" filtrado por orientación AP.*

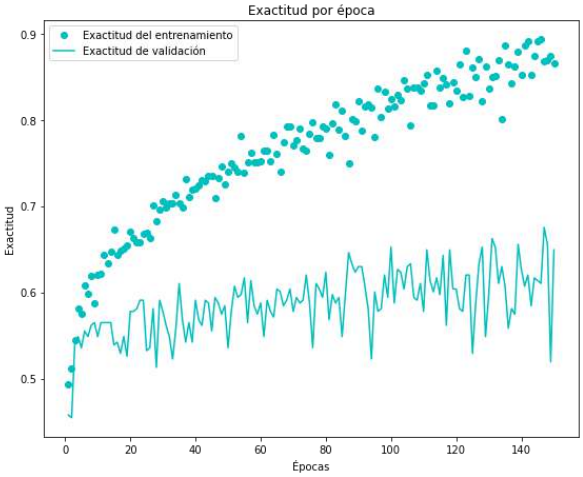


Figura 4.3 *– Gráfico de exactitud por época obtenido para el primer ciclo de entrenamiento/test para el conjunto de "masa" vs "sin hallazgo" filtrado por orientación PA.*

Como se puede apreciar en la Tabla 4.6, no se evidencia ninguna mejora en la exactitud promedio lograda. Por otro lado, las Figuras 4.1, 4.2 y 4.3, muestran como el comportamiento varía mucho con respecto de los resultados obtenidos para los casos evaluados sin el filtrado por orientación, si bien en ambos experimentos existe *overfitting*, el modelo no parece mejorar en la generalización sino al contrario. En base a los resultados obtenidos, se decidió continuar con los experimentos propuestos sin separar los conjuntos por la orientación de las radiografías.

4.3.2 Resultados del arreglo experimental #2: Evaluación de robustez en la clasificación según orientación

Como fue mencionado en el apartado 4.2 se realizaron pruebas con las distintas arquitecturas a partir de CNN pre-entrenadas sobre los conjuntos de datos mostrados en la Tabla 4.1. Los resultados obtenidos en esta etapa del proyecto hicieron evidente la necesidad de trabajar con técnicas de regularización para mejorar la generalización de los modelos.

En la Tabla 4.7 se muestran las exactitudes promedio obtenidas para cada caso de patología pura vs “sin hallazgo” con y sin aumento dinámico de datos, sobre 5 ciclos de entrenamiento/test.

Tabla 4.7 *- Promedio de la exactitud sin y con aumento de datos para los conjuntos de datos conformados por una patología y “sin hallazgo”.*

Conjunto de datos	Sin aumento de datos	Con aumento de datos
Atelectasia + Sin Hallazgo	68,5%	68,9%
Cardiomegalia + Sin Hallazgo	73,6%	68,9%
Infiltración + Sin Hallazgo	62,0%	62,2%
Masa + Sin Hallazgo	66,6%	67,5%
Nódulo + Sin Hallazgo	64,0%	61,8%
Neumotórax + Sin Hallazgo	75,3%	75,6%

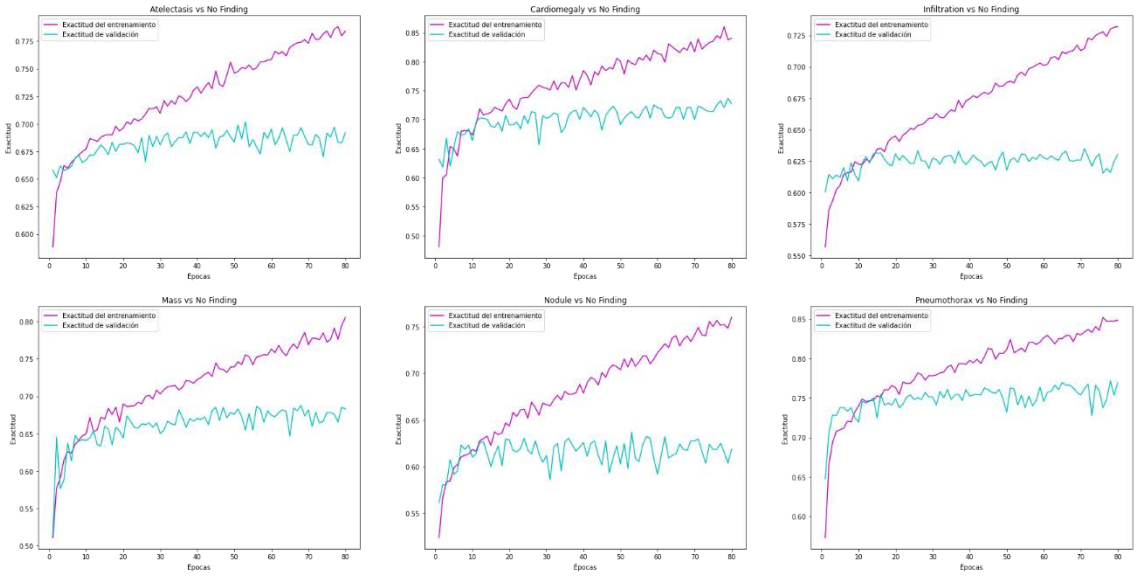


Figura 4.4 *- Gráficos de variación de la exactitud de test en función de la época para el primer ciclo de entrenamiento/test con los conjuntos de datos para patologías puras y "sin hallazgo", sin utilizar aumento de datos dinámico.*

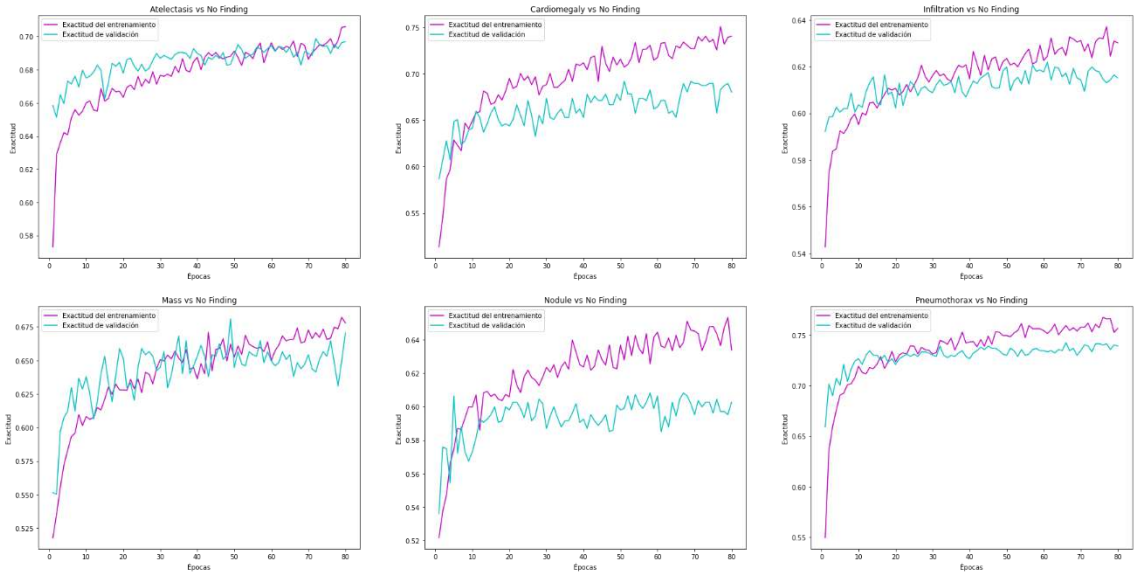


Figura 4.5 *– Gráficos de variación de la exactitud de test en función de la época para el primer ciclo de entrenamiento/test con los conjuntos de datos para patologías puras y "sin hallazgo", utilizando aumento de datos dinámico.*

Al utilizar aumento de datos dinámico se puede apreciar como la exactitud de validación promedio mejora en más de un 5% en términos relativos, menos para la clasificación de "nódulo" vs "sin hallazgo" que muestra una degradación del rendimiento. A partir de los gráficos de las Figuras 4.4 y 4.5, puede observarse que el empleo *de data-augmentation* disminuye el sobreajuste (se observa en el cambio de tendencia entre las curvas de entrenamiento y de validación) que se daba muy marcadamente a partir de la época 10.

4.3.3 Resultados del arreglo experimental #3: Clasificación entre "Hallazgo" y "Sin Hallazgo"

Se construyó un nuevo conjunto de datos conteniendo todos los ejemplos disponibles (ver Tabla 4.8). La clase "hallazgo" incluye a todas las radiografías que presenten una o más patologías, considerando tanto las etiquetas simples como las multi-etiquetas para este arreglo experimental.

Tabla 4.8 *– Conjunto de datos conformado por las imágenes re-etiquetadas como "Hallazgo" y las imágenes etiquetadas como "Sin Hallazgo".*

Conjunto de datos	Número total de imágenes
Hallazgo + Sin Hallazgo	112120

Los resultados obtenidos sin y con aumento dinámico de datos se resumen en la Tabla 4.9. Dado que el conjunto de datos es único (no hay submuestreo) se realizó 1 ciclo de entrenamiento/*test*. Los gráficos con la exactitud de entrenamiento y de test para cada época se muestran en la Figura 4.6 para sin aumento y en la Figura 4.7 para con aumento.

Tabla 4.9 *- Exactitud de validación para el conjunto, sin y con aumento de datos, conformado por "hallazgo" y "sin hallazgo".*

Sin aumento de datos	Con aumento de datos
69,3%	69,5%

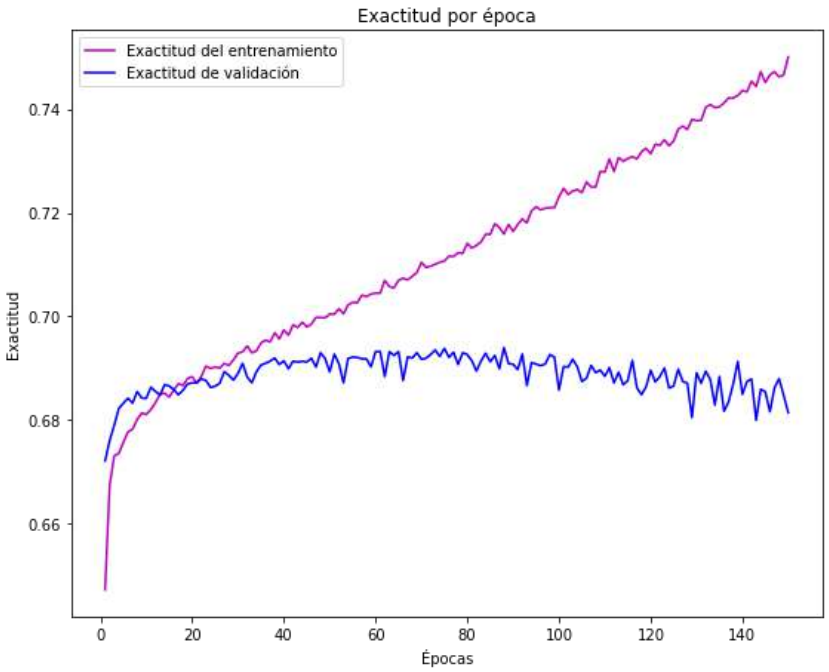


Figura 4.6 *- Gráfico de exactitud por época para el conjunto de datos conformado por las etiquetas "hallazgo" y "sin hallazgo" sin aumento de datos.*

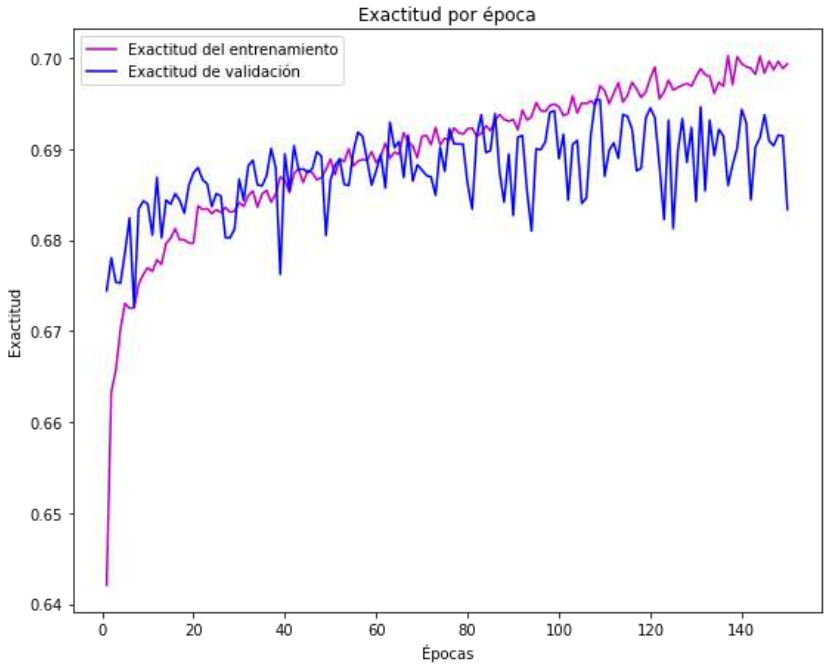


Figura 4.7 *– Gráfico de exactitud por época para el conjunto de datos conformado por las etiquetas "hallazgo" y "sin hallazgo" con aumento de datos.*

En la Tabla 4.9, no se observa una mejora en la exactitud de validación al utilizar *data-augmentation* para la clasificación entre "hallazgo" y "sin hallazgo". Sin embargo, los valores obtenidos para ambos casos son cercanos al 70% de exactitud, lo cual representa un valor muy bueno para la alta complejidad del problema planteado. Por otra parte, al comparar los gráficos de las Figuras 4.6 y 4.7, se ve una mejora importante con respecto al *overfitting* al utilizar aumento de datos dinámico.

4.3.4 Resultados del arreglo experimental #4: Clasificación entre distintas patologías puras

Los resultados obtenidos de exactitud de validación para el promedio de los cinco ciclos de entrenamiento son mostrados en la Tabla 4.10.

Tabla 4.10 *– Promedio de la exactitud sin y con aumento de datos utilizando el conjunto conformado por 12 patologías puras.*

Sin aumento de datos	Con aumento de datos
27,2%	28,1%

Quizás por la complejidad del problema planteado, los resultados obtenidos para esta experimentación no fueron alentadores. La implementación de técnicas de regularización no mostró ninguna mejora en cuanto a la falta de generalización del modelo. Se puede apreciar una mejora del 1% respecto de la exactitud de validación, obtenida del promedio de los cinco ciclos de entrenamiento realizados. Sin embargo, el

valor conseguido para ambos casos se encuentra cerca del 30%, lo que indica que aproximadamente sólo 3 de cada 10 radiografías mostradas al modelo serán clasificadas correctamente.

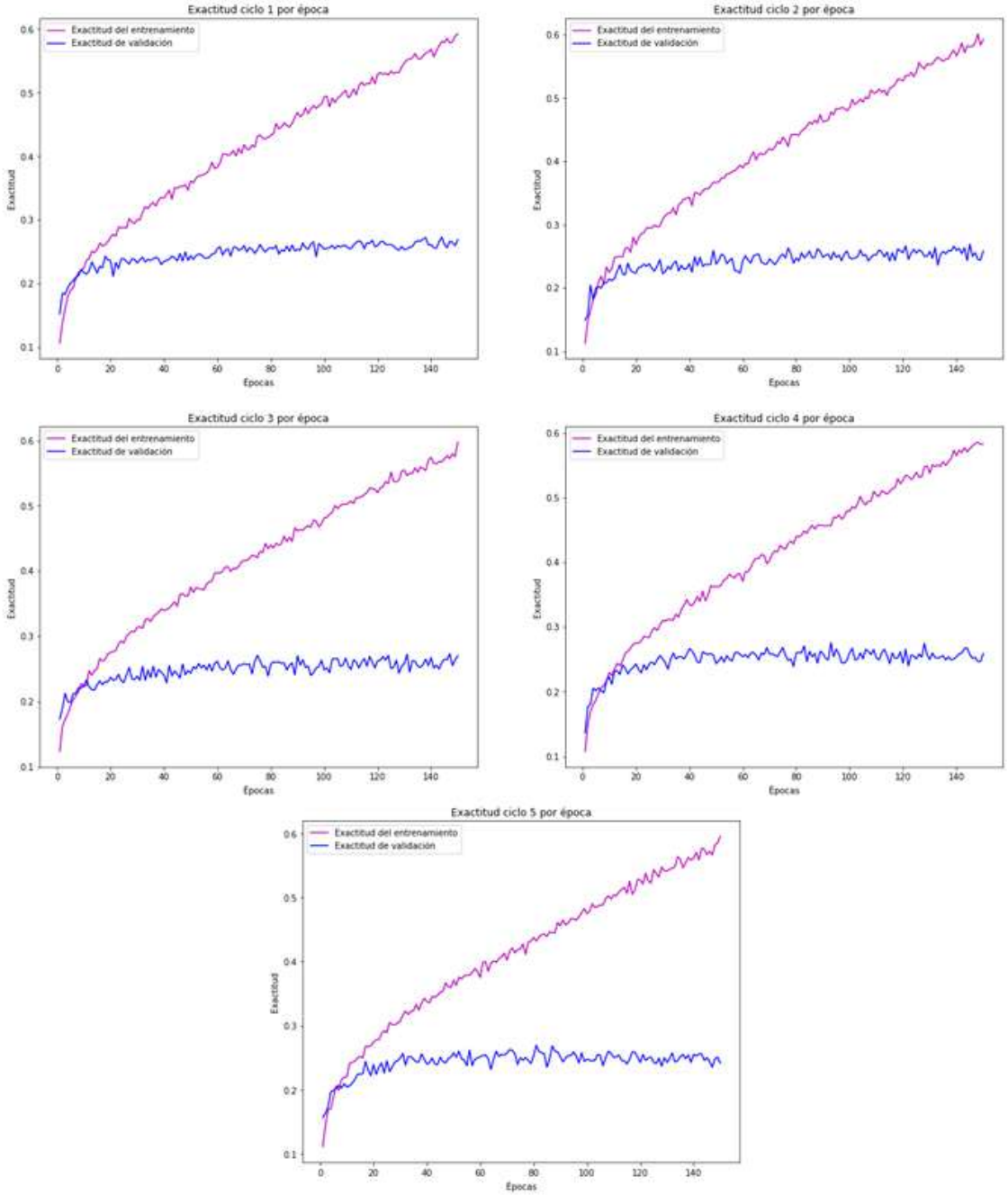


Figura 4.8 *- Gráfico de exactitud por época y por ciclo para el conjunto de datos de patologías puras sin aumento de datos.*

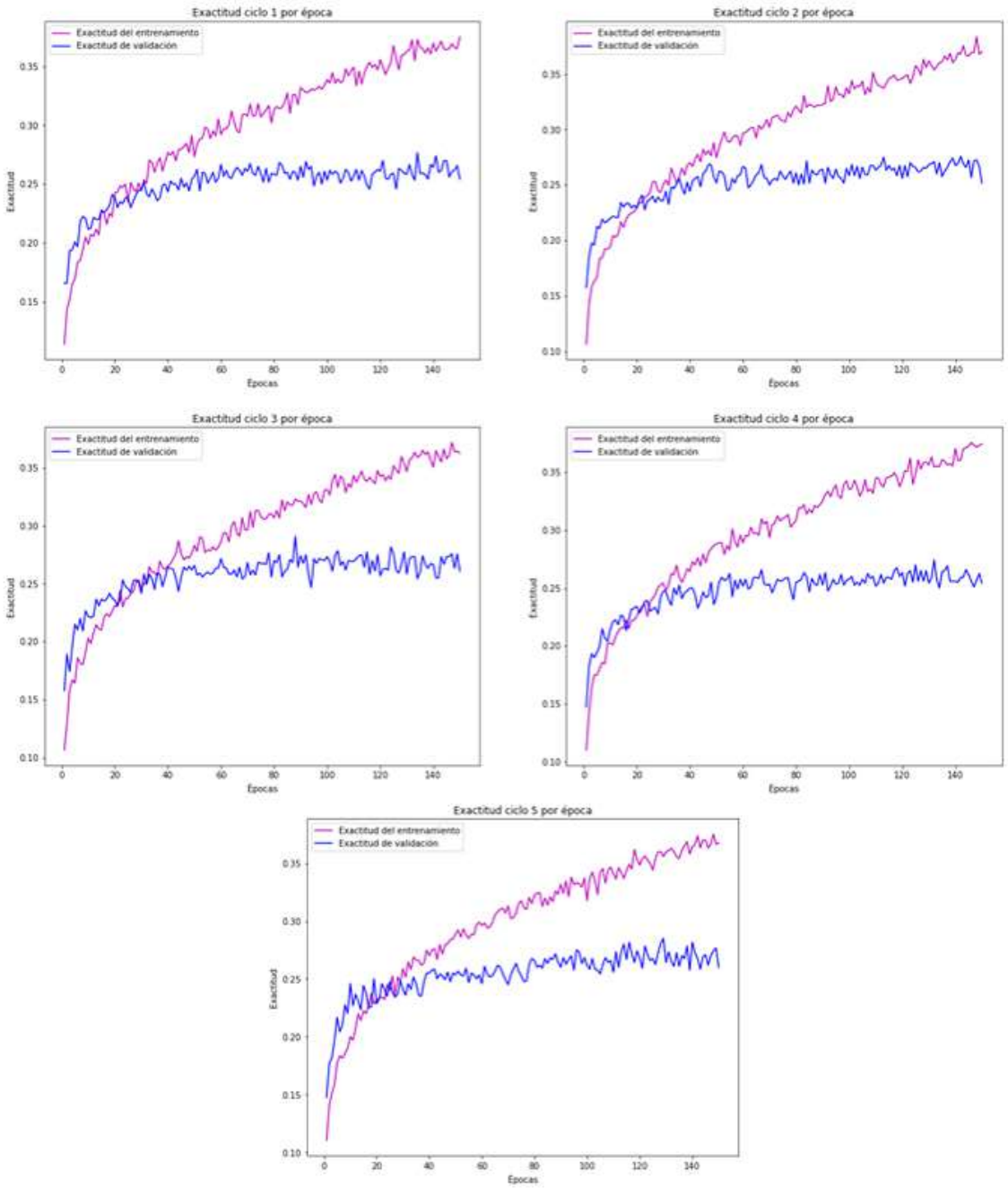


Figura 4.9 *- Gráfico de exactitud por época y por ciclo para el conjunto de datos de patologías puras con aumento de datos.*

Como puede verse en las Figuras 4.8 y 4.9 el aumento de datos dinámicos parece mejorar un poco la capacidad de generalización del modelo, disminuyendo el *overfitting* prematuro. Sin embargo, los resultados obtenidos de exactitud de validación promedio dejan en evidencia que el enfoque propuesto no es viable para este arreglo experimental.

4.4 Resultados de CNN sin *transfer-learning*

A partir de los resultados obtenidos con transfer-learning para los distintos arreglos experimentales, se optó por utilizar el conjunto de datos del del arreglo experimental #3, compuesto por las etiquetas “sin hallazgo” y “hallazgo”, lo que permitió contar con una gran cantidad de imágenes, necesaria para el entrenamiento de una CNN completa.

Se decidió entrenar al modelo durante 300 épocas, utilizando aumento de datos dinámico y los mismos hiperparámetros de entrenamiento que en las experimentaciones con *transfer-learning*. Se realizó *hold-out* con 80% de los datos para entrenamiento y el 20% restante para *test*. Los valores de exactitud de entrenamiento y de *test* para cada época de entrenamiento se muestran en el gráfico de la Figura 4.10.

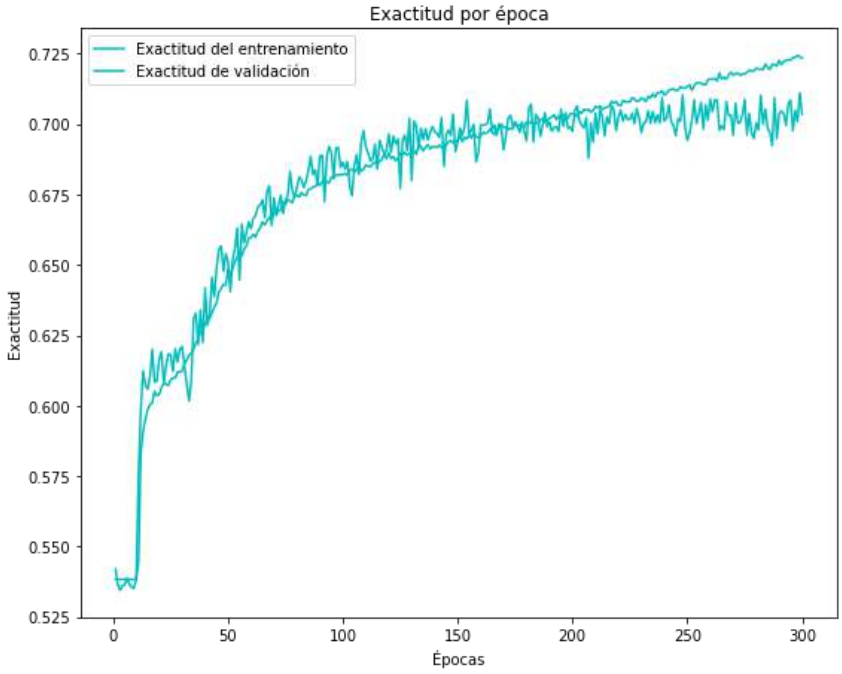


Figura 4.10 *– Exactitud vs época para la CNN sin transfer-learning, entrenada con el conjunto de datos compuesto por las etiquetas “sin hallazgo” y “hallazgo”.*

Como puede verse en la Figura 4.10 el modelo generaliza muy bien, empezando a converger entre las épocas 150 y 200. Si se lo compara con la arquitectura planteada con ResNet50 y el perceptrón multicapa muestra un mejoramiento drástico respecto del sobreajuste. Asimismo, el valor de exactitud alcanzado se mantiene cercano al 70% al igual que con *transfer-learning*.

5 Conclusiones

El presente Proyecto Final de grado tuvo como objetivo principal realizar un estudio de modelos basados en aprendizaje profundo para la identificación de patologías en radiografía de tórax. Uno de los principales desafíos que se presentan al abordar esta problemática, es la escasez de bases de datos de este tipo de imágenes médicas de dominio público, provocando un estado del arte relativamente acotado en la temática. Hasta el día de hoy la base de datos pública con mayor cantidad de radiografías de tórax y sobre la cual se han realizado mayor cantidad de trabajos es ChestX-ray14, motivo por el cuál fue seleccionada para el desarrollo de este proyecto.

El desarrollo del proyecto partió de un fuerte estudio del estado del arte y de las características de la base de datos adoptada, a partir de lo cual se definieron diferentes experimentos con el fin de analizar la capacidad de los modelos estudiados.

Se exploró la detección de patologías mediante enfoques de *transfer-learning* basados en CNN considerando arquitecturas pre-entrenadas como VGG19, ResNet50 e Inceptionv3, siendo ResNet50 la red que mostró la mejor performance. En general, los resultados obtenidos con los enfoques basados en el *transfer-learning* permiten afirmar que esta puede ser una estrategia inicial válida para el uso de CNN en casos en los que no hay suficientes imágenes etiquetadas, muy común en el campo de la medicina, incluso en problemas complejos como lo es la clasificación de radiografías de tórax.

En la mayoría de los casos, los resultados indican una performance aceptable en términos de exactitud en los experimentos abordados, teniendo en cuenta la complejidad de la clasificación de las radiografías de tórax. El aumento de datos dinámico jugó un papel significativo para reducir la tendencia al sobreajuste en todas las pruebas realizadas, sin aportar un aumento significativo del costo computacional y aumentando la estabilidad de los clasificadores.

La arquitectura *ad-hoc* planteada para evaluar el desempeño de una CNN implementada sin *transfer-learning* mostró un rendimiento aceptable en términos de exactitud y una muy buena generalización utilizando *data-augmentation*, lo que indica que es una estrategia válida a seguir cuando se dispone de la suficiente cantidad de imágenes.

Por todo lo dicho, los modelos basados en DNN con y sin transfer-learning mostraron ser un enfoque adecuado para la clasificación de patologías en radiografías de tórax.

Por otro lado, el total de los programas desarrollados en este Proyecto Final para el diseño, entrenamiento y validación de CNN, con su correspondiente documentación, es

una base adecuada para el desarrollo futuro de una librería de funciones que implementen CNN sobre imágenes que se sumarán a otras técnicas actualmente en desarrollo en el Laboratorio.

6 Bibliografía

[1] D. J. Hand, “Principles of data mining,” in *Drug Safety*, 2007, vol. 30, no. 7, doi: 10.2165/00002018-200730070-00010.

[2] M. I. Razzak, S. Naz, and A. Zaib, “Deep Learning for Medical Image Processing: Overview, Challenges and the Future,” Springer, Cham, 2018, pp. 323–350.

[3] G. Wu, D. Shen, and M. R. Sabuncu, *Machine learning and medical imaging*. Academic Press is an imprint of Elsevier, 2016.

[4] X. Wang, Y. Peng, L. Lu, Z. Lu, M. Bagheri, and R. M. Summers, “ChestX-ray8: Hospital-scale chest X-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases,” in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, Nov. 2017, vol. 2017-Janua, pp. 3462–3471, doi: 10.1109/CVPR.2017.369.

[5] L. Deng, “A tutorial survey of architectures, algorithms, and applications for deep learning,” *APSIPA Transactions on Signal and Information Processing*, vol. 3. 2014, doi: 10.1017/ATSIP.2013.99.

[6] G. Litjens *et al.*, “A survey on deep learning in medical image analysis,” *Med. Image Anal.*, vol. 42, pp. 60–88, 2017, doi: 10.1016/J.MEDIA.2017.07.005.

[7] S. Bhattacharya *et al.*, “Deep learning and medical image processing for coronavirus (COVID-19) pandemic: A survey,” *Sustain. Cities Soc.*, vol. 65, p. 102589, Feb. 2021, doi: 10.1016/j.scs.2020.102589.

[8] M. Chakraborty, S. V. Dhavale, and J. Ingole, “Corona-Nidaan: lightweight deep convolutional neural network for chest X-Ray based COVID-19 infection detection,” *Appl. Intell.*, pp. 1–18, Feb. 2021, doi: 10.1007/s10489-020-01978-9.

[9] F. Chollet, *Deep Learning with Python, Manning*. 2018.

[10] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[11] C. C. Aggarwal, *Neural Networks and Deep Learning*. 2018.

[12] D. (DJ) Sarkar, “A Comprehensive Hands-on Guide to Transfer Learning with Real-World Applications in Deep Learning,” 2018. https://towardsdatascience.com/a-comprehensive-hands-on-guide-to-transfer-learning-with-real-world-applications-in-deep-learning-212bf3b2f27a.

[13] J. Z. Cheng *et al.*, “Computer-Aided Diagnosis with Deep Learning Architecture: Applications to Breast Lesions in US Images and Pulmonary Nodules in CT Scans,” *Sci. Rep.*, vol. 6, no. 1, pp. 1–13, 2016, doi: 10.1038/srep24454.

[14] R. C. Gonzales and R. E. Woods, *Digital Image Processing Fourth Edition*, vol. 1, no. 4. 2018.

[15] S. Saha, “A Comprehensive Guide to Convolutional Neural Networks,” 2018. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.

[16] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2014, vol. 8689 LNCS, no. PART 1, doi: 10.1007/978-3-319-10590-1_53.

[17] Jia Deng, Wei Dong, R. Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, “ImageNet: A large-scale hierarchical image database,” 2009, doi: 10.1109/cvprw.2009.5206848.

[18] “ImageNet.” https://www.image-net.org.

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” 2012.

[20] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” 2015.

[21] C. Szegedy *et al.*, “Going deeper with convolutions,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015, vol. 07-12-June-2015, doi: 10.1109/CVPR.2015.7298594.

[22] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016, vol. 2016-December, doi: 10.1109/CVPR.2016.90.

[23] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017, vol. 2017-January, doi: 10.1109/CVPR.2017.243.

[24] F. Chollet, “Transfer learning & fine-tuning.”

https://keras.io/guides/transfer_learning/ (accessed May 17, 2021).

[25] T. Y. Lin *et al.*, “Microsoft COCO: Common objects in context,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2014, vol. 8693 LNCS, no. PART 5, doi: 10.1007/978-3-319-10602-1_48.

[26] R. Atienza, *Advanced Deep Learning with Keras*. 2018.

[27] J. Moolayil, *Learn Keras for Deep Neural Networks*. 2019.

[28] D. S. Comas, “Lógica Difusa Tipo 2 de Intervalos en Segmentación de Imágenes Médicas,” Universidad Nacional de Mar del Plata, 2016.

[29] “Dropout layer.” https://keras.io/api/layers/regularization_layers/dropout/ (accessed Jun. 01, 2021).

[30] Luke Oakden-Rayner, “The unreasonable usefulness of deep learning in medical image datasets.” https://lukeoakdenrayner.wordpress.com/2018/04/30/the-unreasonable-usefulness-of-deep-learning-in-medical-image-datasets/ (accessed Jul. 30, 2021).

[31] A. A. Ortiz-Preciado, J. A. Vega-Fernández, and G. Ochoa-Ruiz, “Clasificación de enfermedades del tórax usando aprendizaje profundo y aumento de datos de calidad en el conjunto de datos ChestX-ray8,” *Res. Comput. Sci.*, vol. 148, no. 8, pp. 41–53, 2019, doi: 10.13053/rcs-148-8-3.

[32] Rohan Bhansali, “Using Machine Learning to Diagnose Chest Xrays and Interpret Patient Symptoms and Medical History,” *Int. J. Eng. Res.*, vol. V9, no. 11, pp. 339–341, 2020, doi: 10.17577/ijertv9is110163.

[33] T. Kluyver *et al.*, “Jupyter Notebooks—a publishing format for reproducible computational workflows,” 2016, doi: 10.3233/978-1-61499-649-1-87.

[34] J. M. Perkel, “Why Jupyter is data scientists’ computational notebook of choice,” *Nature*, vol. 563, no. 7729, 2018, doi: 10.1038/d41586-018-07196-1.

[35] “SSH (Secure Shell) Home Page.” https://www.ssh.com/academy/ssh (accessed Oct. 15, 2021).

[36] “GNU Screen.” https://www.gnu.org/software/screen/ (accessed Jul. 10, 2021).

[37] “Numpy.” https://numpy.org/ (accessed Jun. 20, 2021).

[38] J. Vijayabhaskar, “Tutorial on Keras flow_from_dataframe.”

https://vijayabhaskar96.medium.com/tutorial-on-keras-flow-from-dataframe-1fd4493d237c (accessed Jul. 01, 2021).

[39] F. Chollet, “Building powerful image classification models using very little data,” 2016. https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html (accessed Sep. 12, 2021).

[40] “Functional API guide.” https://keras.io/guides/functional_api/ (accessed Jul. 21, 2021).

[41] J. Vanderplas, *Python Data Science Handbook powered by Jupyter*. 2019.

7 Apéndices

7.1 Plan de proyecto

7.1.1 Introducción

Este documento corresponde a la definición del plan de proyecto para el desarrollo del Proyecto Final de la carrera de Ingeniería en Computación: “Redes basadas en el aprendizaje profundo para la detección de anomalías en radiografías de tórax”, que se desarrollará en el Laboratorio de Procesamiento de Imágenes (LPI), del Instituto de Investigaciones Científicas y Tecnológicas en Electrónica (ICyTE), UNMdP-CONICET.

7.1.2 Objetivos del proyecto

7.1.2.1 Objetivo general

Estudiar, proponer, implementar y validar redes basadas en aprendizaje profundo para detección de anomalías en radiografías de tórax.

7.1.2.2 Objetivos específicos

- Realizar un relevamiento de redes de aprendizaje profundo aplicadas en imágenes médicas.
- Estudiar trabajos existentes que aborden el procesamiento de radiografías de tórax.
- Implementar computacionalmente redes de aprendizaje profundo para detectar anomalías en radiografías de tórax y evaluar su desempeño.

7.1.3 Alcance del proyecto

7.1.3.1 Alcance del trabajo final de grado

La concreción de este Proyecto Final permitirá avanzar en este sentido al estudiar un problema de enorme relevancia en la actualidad como lo es la detección automática de anomalías en radiografías de tórax. Asimismo, el avance sobre CNN en imágenes médicas permitirá estudiar su aplicabilidad ampliando el conocimiento actual que se tiene sobre su uso en este tipo de problemas. Finalmente, la implementación y optimización computacional de las etapas de diseño, entrenamiento y validación de CNN, con su correspondiente documentación, permitirá desarrollar en el futuro una librería de funciones que implementen CNN sobre imágenes que se sumarán a otras técnicas actualmente en desarrollo en el Laboratorio para el análisis de la estructura decisoria de las redes.

7.1.3.2 Gestión del alcance

Para planificar la gestión del alcance se utilizará como herramienta fundamental las reuniones con el equipo de trabajo del laboratorio una vez por semana, tanto para validar el avance del proyecto como la documentación a entregar en las situaciones que correspondan.

Para una mejor distinción de las etapas del proyecto y actividades de trabajo, se estructuró el proyecto en componentes más pequeños y fáciles de manejar creando el plan de tareas y el Gantt que se adjuntan en la sección siguiente.

7.1.4 Desarrollo del proyecto

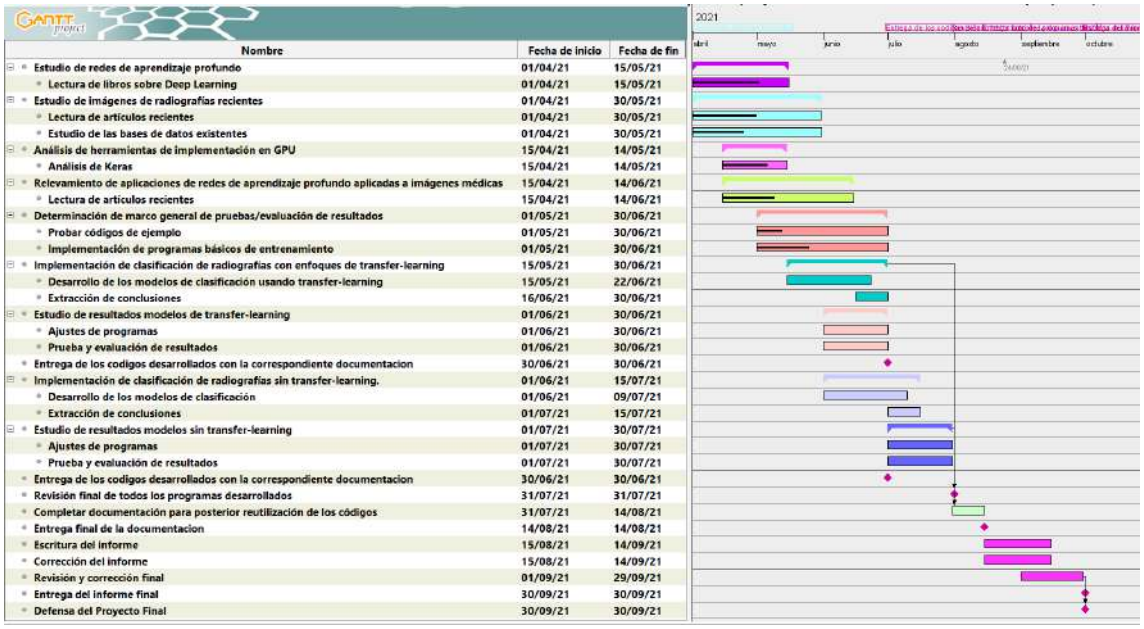
7.1.4.1 Actividades a realizar

En un principio, se prevé una duración del proyecto de 6 meses y las actividades se planifican por bloques de 2 semanas.

	Abril		Mayo		Junio		Julio		Agosto		Septiembre	
Estudio de redes de aprendizaje profundo.	X	X	X									
Análisis de herramientas de implementación en GPU incluyendo Python, Keras.		X	X									
Estudio de imágenes de radiografías de tórax, incluyendo artículos recientes y bases de datos existentes.	X	X	X	X								
Relevamiento de aplicaciones de redes de aprendizaje profundo en imágenes médicas		X	X									
Determinación de marco general de pruebas/evaluación de resultados e implementación de programas básicos de entrenamiento de evaluación			X	X								

Implementación de clasificación de radiografías con enfoques de transfer-learning. Extracción de conclusiones.				X	X	X						
Estudio de resultados y posibles ajustes de programas y nuevo ciclo de prueba/evaluación de resultados.					X	X						
Implementación de clasificación de radiografías sin transfer-learning. Extracción de conclusiones.					X	X	X					
Estudio de resultados y posibles ajustes en los algoritmos implementados.							X	X	X			
Revisión de todos los programas desarrollados, completando su documentación a fines de facilitar su reutilización									X			
Escritura informe										X	X	
Corrección informe										X	X	
Revisión y corrección final											X	X
Defensa del Proyecto Final												X

7.1.4.2 Diagrama de Gantt



Nombre	Fecha de inicio	Fecha de fin
Estudio de redes de aprendizaje profundo	01/04/21	15/05/21
Lectura de libros sobre Deep Learning	01/04/21	15/05/21
Estudio de imágenes de radiografías recientes	01/04/21	30/05/21
Lectura de artículos recientes	01/04/21	30/05/21
Estudio de las bases de datos existentes	01/04/21	30/05/21
Análisis de herramientas de implementación en GPU	15/04/21	14/05/21
Análisis de Keras	15/04/21	14/05/21
Relevamiento de aplicaciones de redes de aprendizaje profundo aplicadas a imágenes médicas	15/04/21	14/06/21
Lectura de artículos recientes	15/04/21	14/06/21
Determinación de marco general de pruebas/evaluación de resultados	01/05/21	30/06/21
Probar códigos de ejemplo	01/05/21	30/06/21
Implementación de programas básicos de entrenamiento	01/05/21	30/06/21
Implementación de clasificación de radiografías con enfoques de transfer-learning	15/05/21	30/06/21
Desarrollo de los modelos de clasificación usando transfer-learning	15/05/21	22/06/21
Extracción de conclusiones	16/06/21	30/06/21
Estudio de resultados modelos de transfer-learning	01/06/21	30/06/21
Ajustes de programas	01/06/21	30/06/21
Prueba y evaluación de resultados	01/06/21	30/06/21
Entrega de los codigos desarrollados con la correspondiente documentacion	30/06/21	30/06/21
Implementación de clasificación de radiografías sin transfer-learning.	01/06/21	15/07/21
Desarrollo de los modelos de clasificación	01/06/21	09/07/21
Extracción de conclusiones	01/07/21	15/07/21
Estudio de resultados modelos sin transfer-learning	01/07/21	30/07/21
Ajustes de programas	01/07/21	30/07/21
Prueba y evaluación de resultados	01/07/21	30/07/21
Entrega de los codigos desarrollados con la correspondiente documentacion	30/06/21	30/06/21
Revisión final de todos los programas desarrollados	31/07/21	31/07/21
Completar documentación para posterior reutilización de los códigos	31/07/21	14/08/21
Entrega final de la documentacion	14/08/21	14/08/21
Escritura del informe	15/08/21	14/09/21
Corrección del informe	15/08/21	14/09/21
Revisión y corrección final	01/09/21	29/09/21
Entrega del informe final	30/09/21	30/09/21
Defensa del Proyecto Final	30/09/21	30/09/21

Por tratarse de un proyecto de investigación, y dado que algunos de los desarrollos involucrados son aportes originales, no existe garantía previa de llegar a un resultado en un lapso de tiempo predeterminado. Es por esto que las fechas mostradas en el Gantt son estimativas y variarán a medida que se avance sobre el proyecto final. Además, se debe tener en cuenta el contexto de pandemia que se está atravesando al momento de realizar el proyecto, que puede afectar también sobre las fechas definidas.

7.1.5 Gestión de los recursos humanos

7.1.5.1 Personas involucradas en el proyecto

Nombre	**Luciana Simón González**
Rol	Estudiante de proyecto final de carrera
Categoría Profesional	Estudiante
Responsabilidad	Desarrollo del proyecto
Información de contacto	lucianasimong@gmail.com

Nombre	**Diego Sebastián Comas**
Rol	Director
Categoría Profesional	Ingeniero electrónico / Dr. en Ingeniería, orientación electrónica
Responsabilidad	Dirección y supervisión del proyecto
Información de contacto	diego.comas@fi.mdp.edu.ar

Nombre	Virginia Laura Ballarin
Rol	Co-directora
Categoría Profesional	Ingeniera electrónica / Dra. en Ciencias Biológicas or. Bioingeniería
Responsabilidad	Dirección y supervisión del proyecto
Información de contacto	vballari@fi.mdp.edu.ar

7.1.5.2 Asignación de responsables

El Proyecto Final de Carrera será desarrollado por Luciana Simón González, quien llevará a cabo una investigación sobre la detección automática de anomalías en radiografías de tórax, desarrollando código de autoría propia y documentando los descubrimientos hallados.

El papel desarrollado por el director y la co-directora del proyecto será el de dirigir, supervisar, acompañar y asesorar a la estudiante en el desarrollo de su Proyecto Final.

7.1.6 Identificación de los riesgos

El Proyecto Final de la carrera a llevar a término tiene diversos riesgos asociados:

- Extensión de la fecha de finalización del proyecto más allá de lo previsto dado que algunos de los desarrollos involucrados son aportes originales y no existe garantía previa de llegar a un resultado en un lapso de tiempo predeterminado.
- Complicaciones a la hora de comprender e implementar de manera correcta los modelos teóricos.
- Incertidumbre relacionada con el contexto actual de pandemia.

El Proyecto de final de Grado es un proceso de aprendizaje continuo en el cual se cuenta con el acompañamiento constante de los directores, quienes ayudarán a disminuir la incertidumbre aportando su conocimiento.

Se procederá a utilizar un proceso iterativo de identificación de riesgos debido a que pueden evolucionar o se pueden ir descubriendo nuevos conforme el proyecto avance a lo largo de su ciclo de vida.

7.2 Especificación de requerimientos

7.2.1 Introducción

Este documento corresponde a la definición del tema de investigación del Proyecto Final de la carrera de Ingeniería en Computación: “Redes basadas en el aprendizaje profundo

para la detección de anomalías en radiografías de tórax", que se desarrollará en el Laboratorio de Procesamiento de Imágenes (LPI), del Instituto de Investigaciones Científicas y Tecnológicas en Electrónica (ICyTE), UNMdP-CONICET.

7.2.2 Definiciones, acrónimos y abreviaturas

Nombre	Descripción
DL	*Deep-Learning*, aprendizaje profundo.
CNN	*Convolutional Neural Networks*, redes neuronales convolucionales.
PDI	Procesamiento Digital de Imágenes.
GPU	*Graphics Processing Unit*, unidad de procesamiento gráfico.

7.2.3 Referencias

[1] D. J. Hand, "Principles of data mining," in *Drug Safety*, 2007, vol. 30, no. 7, doi: 10.2165/00002018-200730070-00010.

[2] M. I. Razzak, S. Naz, and A. Zaib, "Deep Learning for Medical Image Processing: Overview, Challenges and the Future," Springer, Cham, 2018, pp. 323–350.

[3] G. Wu, D. Shen, and M. R. Sabuncu, *Machine learning and medical imaging*. Academic Press is an imprint of Elsevier, 2016.

[4] X. Wang, Y. Peng, L. Lu, Z. Lu, M. Bagheri, and R. M. Summers, "ChestX-ray8: Hospital-scale chest X-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, Nov. 2017, vol. 2017-Janua, pp. 3462–3471, doi: 10.1109/CVPR.2017.369.

[5] L. Deng, "A tutorial survey of architectures, algorithms, and applications for deep learning," *APSIPA Transactions on Signal and Information Processing*, vol. 3. 2014, doi: 10.1017/ATSIP.2013.99.

[6] G. Litjens *et al.*, "A survey on deep learning in medical image analysis," *Med. Image Anal.*, vol. 42, pp. 60–88, 2017, doi: 10.1016/J.MEDIA.2017.07.005.

[7] S. Bhattacharya *et al.*, "Deep learning and medical image processing for coronavirus (COVID-19) pandemic: A survey," *Sustain. Cities Soc.*, vol. 65, p. 102589, Feb. 2021, doi: 10.1016/j.scs.2020.102589.

[8] M. Chakraborty, S. V. Dhavale, and J. Ingole, "Corona-Nidaan: lightweight deep convolutional neural network for chest X-Ray based COVID-19 infection

detection," *Appl. Intell.*, pp. 1–18, Feb. 2021, doi: 10.1007/s10489-020-01978-9.

[9] F. Chollet, *Deep Learning with Python, Manning*. 2018.

[10] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[11] C. C. Aggarwal, *Neural Networks and Deep Learning*. 2018.

[12] D. (DJ) Sarkar, "A Comprehensive Hands-on Guide to Transfer Learning with Real-World Applications in Deep Learning," 2018. https://towardsdatascience.com/a-comprehensive-hands-on-guide-to-transfer-learning-with-real-world-applications-in-deep-learning-212bf3b2f27a.

[13] J. Z. Cheng *et al.*, "Computer-Aided Diagnosis with Deep Learning Architecture: Applications to Breast Lesions in US Images and Pulmonary Nodules in CT Scans," *Sci. Rep.*, vol. 6, no. 1, pp. 1–13, 2016, doi: 10.1038/srep24454.

[14] R. C. Gonzales and R. E. Woods, *Digital Image Processing Fourth Edition*, vol. 1, no. 4. 2018.

[15] S. Saha, "A Comprehensive Guide to Convolutional Neural Networks," 2018. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.

[16] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2014, vol. 8689 LNCS, no. PART 1, doi: 10.1007/978-3-319-10590-1_53.

[17] Jia Deng, Wei Dong, R. Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," 2009, doi: 10.1109/cvprw.2009.5206848.

[18] "ImageNet." https://www.image-net.org.

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," 2012.

[20] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015.

[21] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015, vol. 07-12-June-2015, doi: 10.1109/CVPR.2015.7298594.

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016, vol. 2016-December, doi: 10.1109/CVPR.2016.90.

[23] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017, vol. 2017-January, doi: 10.1109/CVPR.2017.243.

[24] F. Chollet, "Transfer learning & fine-tuning." https://keras.io/guides/transfer_learning/ (accessed May 17, 2021).

[25] T. Y. Lin *et al.*, "Microsoft COCO: Common objects in context," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2014, vol. 8693 LNCS, no. PART 5, doi: 10.1007/978-3-319-10602-1_48.

[26] R. Atienza, *Advanced Deep Learning with Keras*. 2018.

[27] J. Moolayil, *Learn Keras for Deep Neural Networks*. 2019.

[28] D. S. Comas, "Lógica Difusa Tipo 2 de Intervalos en Segmentación de Imágenes Médicas," Universidad Nacional de Mar del Plata, 2016.

[29] "Dropout layer." https://keras.io/api/layers/regularization_layers/dropout/ (accessed Jun. 01, 2021).

[30] Luke Oakden-Rayner, "The unreasonable usefulness of deep learning in medical image datasets." https://lukeoakdenrayner.wordpress.com/2018/04/30/the-unreasonable-usefulness-of-deep-learning-in-medical-image-datasets/ (accessed Jul. 30, 2021).

[31] A. A. Ortiz-Preciado, J. A. Vega-Fernández, and G. Ochoa-Ruiz, "Clasificación de enfermedades del tórax usando aprendizaje profundo y aumento de datos de calidad en el conjunto de datos ChestX-ray8," *Res. Comput. Sci.*, vol. 148, no. 8, pp. 41–53, 2019, doi: 10.13053/rcs-148-8-3.

[32] Rohan Bhansali, "Using Machine Learning to Diagnose Chest Xrays and Interpret Patient Symptoms and Medical History," *Int. J. Eng. Res.*, vol. V9, no. 11, pp. 339–341, 2020, doi: 10.17577/ijertv9is110163.

[33] T. Kluyver *et al.*, "Jupyter Notebooks—a publishing format for reproducible computational workflows," 2016, doi: 10.3233/978-1-61499-649-1-87.

[34] J. M. Perkel, "Why Jupyter is data scientists' computational notebook of choice," *Nature*, vol. 563, no. 7729, 2018, doi: 10.1038/d41586-018-07196-1.

[35] "SSH (Secure Shell) Home Page." https://www.ssh.com/academy/ssh (accessed Oct. 15, 2021).

[36] "GNU Screen." https://www.gnu.org/software/screen/ (accessed Jul. 10, 2021).

[37] "Numpy." https://numpy.org/ (accessed Jun. 20, 2021).

[38] J. Vijayabhaskar, "Tutorial on Keras flow_from_dataframe."

https://vijayabhaskar96.medium.com/tutorial-on-keras-flow-from-dataframe-1fd4493d237c (accessed Jul. 01, 2021).

[39] F. Chollet, “Building powerful image classification models using very little data,” 2016. https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html (accessed Sep. 12, 2021).

[40] “Functional API guide.” https://keras.io/guides/functional_api/ (accessed Jul. 21, 2021).

[41] J. Vanderplas, *Python Data Science Handbook powered by Jupyter*. 2019.

[1] D. J. Hand, “Principles of data mining,” in *Drug Safety*, 2007, vol. 30, no. 7, doi: 10.2165/00002018-200730070-00010.

[2] M. I. Razzak, S. Naz, and A. Zaib, “Deep Learning for Medical Image Processing: Overview, Challenges and the Future,” Springer, Cham, 2018, pp. 323–350.

[3] G. Wu, D. Shen, and M. R. Sabuncu, *Machine learning and medical imaging*. Academic Press is an imprint of Elsevier, 2016.

[4] X. Wang, Y. Peng, L. Lu, Z. Lu, M. Bagheri, and R. M. Summers, “ChestX-ray8: Hospital-scale chest X-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases,” in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, Nov. 2017, vol. 2017-Janua, pp. 3462–3471, doi: 10.1109/CVPR.2017.369.

[5] L. Deng, “A tutorial survey of architectures, algorithms, and applications for deep learning,” *APSIPA Transactions on Signal and Information Processing*, vol. 3. 2014, doi: 10.1017/ATSIP.2013.99.

[6] G. Litjens *et al.*, “A survey on deep learning in medical image analysis,” *Med. Image Anal.*, vol. 42, pp. 60–88, 2017, doi: 10.1016/J.MEDIA.2017.07.005.

[7] S. Bhattacharya *et al.*, “Deep learning and medical image processing for coronavirus (COVID-19) pandemic: A survey,” *Sustain. Cities Soc.*, vol. 65, p. 102589, Feb. 2021, doi: 10.1016/j.scs.2020.102589.

[8] M. Chakraborty, S. V. Dhavale, and J. Ingole, “Corona-Nidaan: lightweight deep convolutional neural network for chest X-Ray based COVID-19 infection detection,” *Appl. Intell.*, pp. 1–18, Feb. 2021, doi: 10.1007/s10489-020-01978-9.

[9] F. Chollet, *Deep Learning with Python, Manning*. 2018.

[10] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[11] C. C. Aggarwal, *Neural Networks and Deep Learning*. 2018.

[12] D. (DJ) Sarkar, “A Comprehensive Hands-on Guide to Transfer Learning with

Real-World Applications in Deep Learning," 2018. https://towardsdatascience.com/a-comprehensive-hands-on-guide-to-transfer-learning-with-real-world-applications-in-deep-learning-212bf3b2f27a.

[13] J. Z. Cheng *et al.*, "Computer-Aided Diagnosis with Deep Learning Architecture: Applications to Breast Lesions in US Images and Pulmonary Nodules in CT Scans," *Sci. Rep.*, vol. 6, no. 1, pp. 1–13, 2016, doi: 10.1038/srep24454.

[14] R. C. Gonzales and R. E. Woods, *Digital Image Processing Fourth Edition*, vol. 1, no. 4. 2018.

[15] S. Saha, "A Comprehensive Guide to Convolutional Neural Networks," 2018. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.

[16] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2014, vol. 8689 LNCS, no. PART 1, doi: 10.1007/978-3-319-10590-1_53.

[17] Jia Deng, Wei Dong, R. Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," 2009, doi: 10.1109/cvprw.2009.5206848.

[18] "ImageNet." https://www.image-net.org.

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," 2012.

[20] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015.

[21] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015, vol. 07-12-June-2015, doi: 10.1109/CVPR.2015.7298594.

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016, vol. 2016-December, doi: 10.1109/CVPR.2016.90.

[23] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017, vol. 2017-January, doi: 10.1109/CVPR.2017.243.

[24] F. Chollet, "Transfer learning & fine-tuning." https://keras.io/guides/transfer_learning/ (accessed May 17, 2021).

[25] T. Y. Lin *et al.*, “Microsoft COCO: Common objects in context,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2014, vol. 8693 LNCS, no. PART 5, doi: 10.1007/978-3-319-10602-1_48.

[26] R. Atienza, *Advanced Deep Learning with Keras*. 2018.

[27] J. Moolayil, *Learn Keras for Deep Neural Networks*. 2019.

[28] D. S. Comas, “Lógica Difusa Tipo 2 de Intervalos en Segmentación de Imágenes Médicas,” Universidad Nacional de Mar del Plata, 2016.

[29] “Dropout layer.” https://keras.io/api/layers/regularization_layers/dropout/ (accessed Jun. 01, 2021).

[30] Luke Oakden-Rayner, “The unreasonable usefulness of deep learning in medical image datasets.” https://lukeoakdenrayner.wordpress.com/2018/04/30/the-unreasonable-usefulness-of-deep-learning-in-medical-image-datasets/ (accessed Jul. 30, 2021).

[31] A. A. Ortiz-Preciado, J. A. Vega-Fernández, and G. Ochoa-Ruiz, “Clasificación de enfermedades del tórax usando aprendizaje profundo y aumento de datos de calidad en el conjunto de datos ChestX-ray8,” *Res. Comput. Sci.*, vol. 148, no. 8, pp. 41–53, 2019, doi: 10.13053/rcs-148-8-3.

[32] Rohan Bhansali, “Using Machine Learning to Diagnose Chest Xrays and Interpret Patient Symptoms and Medical History,” *Int. J. Eng. Res.*, vol. V9, no. 11, pp. 339–341, 2020, doi: 10.17577/ijertv9is110163.

[33] T. Kluyver *et al.*, “Jupyter Notebooks—a publishing format for reproducible computational workflows,” 2016, doi: 10.3233/978-1-61499-649-1-87.

[34] J. M. Perkel, “Why Jupyter is data scientists’ computational notebook of choice,” *Nature*, vol. 563, no. 7729, 2018, doi: 10.1038/d41586-018-07196-1.

[35] “SSH (Secure Shell) Home Page.” https://www.ssh.com/academy/ssh (accessed Oct. 15, 2021).

[36] “GNU Screen.” https://www.gnu.org/software/screen/ (accessed Jul. 10, 2021).

[37] “Numpy.” https://numpy.org/ (accessed Jun. 20, 2021).

[38] J. Vijayabhaskar, “Tutorial on Keras flow_from_dataframe.” https://vijayabhaskar96.medium.com/tutorial-on-keras-flow-from-dataframe-1fd4493d237c (accessed Jul. 01, 2021).

[39] F. Chollet, “Building powerful image classification models using very little data,” 2016. https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html (accessed Sep. 12, 2021).

[40] “Functional API guide.” https://keras.io/guides/functional_api/ (accessed Jul. 21,

2021).

[41] J. Vanderplas, *Python Data Science Handbook powered by Jupyter*. 2019.

1.

7.2.4 Resumen

El avance tecnológico de las últimas décadas en la adquisición de imágenes médicas ha aumentado sustancialmente la información disponible y su accesibilidad; convirtiéndolas en uno de los recursos centrales para la evaluación clínica y diagnóstico. Esto supone un gran desafío a la hora de generar herramientas adecuadas de Procesamiento Digital de Imágenes (PDI) para asistir a los especialistas médicos en su tarea.

En los últimos años, las redes basadas en aprendizaje profundo están siendo ampliamente utilizadas en imágenes médicas, obteniendo excelentes resultados en segmentación de órganos y subestructuras, detección de tumores, clasificación de muestras y registración, entre otras tareas. El Laboratorio de Procesamiento de Imágenes del ICyTE, lugar donde se desarrollará el presente Proyecto Final, viene trabajando activamente con este tipo de redes, abarcando tanto la resolución de problemas como su análisis teórico con el fin de explicar los mecanismos asociados a su estructura decisoria.

Se propone en este proyecto abordar la detección de anomalías en imágenes de radiografías de tórax. El trabajo comprende un estudio profundo tanto de la problemática como de las redes de aprendizaje profundo. Una vez definida una red inicial para la resolución del problema, se desarrollarán programas específicos para el diseño, entrenamiento y evaluación de la red, utilizando la API *Keras* sobre Python; y optimizando todos los programas para la utilización de núcleos TensorCores sobre GPU. El proyecto permitirá avanzar en el PDI sobre imágenes médicas con redes basadas en aprendizaje profundo y ampliar la base de conocimiento actual que se tiene sobre su aplicabilidad en la disciplina.

7.2.5 Problema de investigación

7.2.5.1 Surgimiento de la idea de investigación

Las imágenes médicas son representaciones de los diferentes tejidos, procesos y partes del cuerpo humano, adquiridas mediante diferentes tecnologías, dando lugar a diferentes modalidades de imágenes médicas. Su análisis e interpretación asiste en el

diagnóstico de patologías y en el estudio de la anatomía humana; siendo el especialista quien debe interpretar la información contenida en ellas (Wu, et al. 2016).

El continuo avance tecnológico de las últimas décadas permitió mejorar las tecnologías de adquisición y desarrollar otras, dando lugar a nuevas modalidades; obteniendo, cada vez con mayor impulso, imágenes con altísima resolución y gran nivel de detalle y aumentando sustancialmente la información disponible. Pueden mencionarse entre otras modalidades: Imágenes de Resonancia Magnética (IRM), imágenes de Tomografía Computada (TC), angiografías, radiografías, imágenes de microscopio óptico, ultrasonido 2D y 3D, tomografías por emisión de positrones, tomografías por impedancia eléctrica, entre muchas otras (Litjens, et al. 2017; Wu, et al. 2016). Cada una permite analizar diferentes aspectos fisiológicos o anatómicos con fines específicos, por ejemplo: detección de tejidos patológicos, evaluación preventiva, seguimiento, diagnóstico de patologías, planificación de terapias de radiación y cirugías, screening, entre otros.

Este avance en la tecnología y en la disponibilidad de imágenes médicas supone un gran desafío a la hora de generar herramientas adecuadas que asistan al especialista en su análisis y en la toma de decisiones. El PDI ha sido fuente de algoritmos y herramientas exitosas en imágenes médicas. Como objetivos principales del PDI en imágenes médicas pueden mencionarse la segmentación y la clasificación. La segmentación busca una partición tal que las regiones obtenidas correspondan a estructuras anatómicas, procesos o regiones de especial interés y sus resultados se utilizan para comparar volúmenes, morfologías y características con otros estudios u otras regiones de la misma imagen; estudiar la distribución de los tejidos, detectar lesiones, comprender la anatomía, planificar cirugías, planear terapias de radiación y detectar tejidos anormales, entre otras tareas. La clasificación requiere un análisis global de la imagen y se utiliza habitualmente como soporte a decisiones sobre diagnóstico y tratamiento (Comas, et al. 2020; 2015).

Las redes basadas en aprendizaje profundo son modelos muy potentes para PDI que han tomado enorme impulso en los últimos años, siendo las redes neuronales convolucionales (CNN, del inglés Convolutional Neural Network) las más utilizadas (LeCun, et al. 2015). Como característica esencial frente a modelos tradicionales, no requieren el preprocesamiento de las imágenes, ni su manipulación para extraer características útiles, siendo ellas mismas las que realizan todo el procesamiento.

En el Laboratorio de Procesamiento de Imágenes del ICyTE se viene trabajando activamente en los últimos años en redes de aprendizaje profundo para procesamiento

de imágenes médicas, abarcando tanto su implementación computacional para resolver problemas específicos como también el diseño de experimentos con fines de analizar y comprender los mecanismos asociados a su estructura decisoria; contando actualmente con cuatro investigadores dedicados a la temática.

Dentro de las aplicaciones de las redes de aprendizaje profundo en imágenes médicas, la detección automática de anomalías en radiografías de tórax que se propone abordar en el presente Proyecto Final, es un problema de enorme relevancia que se está estudiando actualmente en el Laboratorio y que, en el último año, ha tomado gran impulso en el mundo académico debido a la posibilidad de detectar afecciones características del COVID-19 (Chakraborty, et al. 2021; Bhattacharya, et al. 2021).

7.2.5.2 Preguntas de investigación

La detección de anomalías en radiografías de tórax por medio de redes basadas en aprendizaje profundo plantea las siguientes inquietudes que guardan relación con los objetivos propuestos y sobre las que se indagará durante el desarrollo del presente Proyecto Final:

- ¿Cuáles son las anomalías que pueden detectarse a partir de radiografías de tórax y qué relación guardan éstas con posibles patologías?
- ¿Con qué grado de certeza puede arribarse a un diagnóstico?
- ¿Cuáles son los mecanismos habituales con los que los especialistas analizan y detectan anomalías en este tipo de estudios?
- ¿Cuáles son los desafíos de procesar automáticamente las radiografías y detectar patologías y qué exactitud tienen los métodos de PDI existentes en la bibliografía?
- ¿Cuáles son las arquitecturas de CNN utilizadas para el procesamiento y clasificación de radiografías? ¿Qué enfoque de diseño/entrenamiento se utiliza comúnmente?
- ¿Cuál es la capacidad de detección de anomalías en radiografías de tórax de las redes basadas en aprendizaje profundo?

El desarrollo del proyecto requerirá realizar diferentes actividades que permitan indagar sobre las cuestiones planteadas para la concreción de los objetivos. Se requerirá:

- Estudiar o comprender las técnicas a implementar/probar (incluyendo un análisis teórico);
- Implementar computacionalmente las técnicas definiendo experimentos para su validación;

- Proceder a etapas de ajuste o replanteo.

7.2.6 Objetivos

7.2.6.1 Objetivo general

- Estudiar, proponer, implementar y validar redes basadas en aprendizaje profundo para detección de anomalías en radiografías de tórax.

7.2.6.2 Objetivos específicos

- Realizar un relevamiento de redes de aprendizaje profundo aplicadas en imágenes médicas.
- Estudiar trabajos existentes que aborden el procesamiento de radiografías de tórax.
- Desarrollar los algoritmos para implementar computacionalmente redes de aprendizaje profundo para detección de anomalías en radiografías de tórax, incluyendo los programas necesarios para evaluar su exactitud.

7.2.7 Justificación y viabilidad de la investigación

En imágenes médicas, el número de aplicaciones exitosas de redes basadas en aprendizaje profundo, específicamente de CNN, es creciente, mostrando excelentes resultados en múltiples modalidades y con múltiples objetivos (Litjens, et al. 2017; Razzak, et al. 2018; Bhattacharya, et al. 2021). Entre las modalidades se destacan: IRM (Shin, et al. 2013; Havaei, et al. 2017), microscopias (Oei, et al. 2019), imágenes de TC (Tajbakhsh, et al. 2015; Saood and Hatem 2021), ultrasonido (Cheng, et al. 2016; Burgos-Artizzu, et al. 2020), rayos X (Liu, et al. 2016; Chakraborty, et al. 2021), mamografías (Wang, et al. 2017a) y angiografías retinales (Mahapatra, et al. 2016; Fu, et al. 2016; Chai, et al. 2018). Dependiendo de la aplicación concreta, los objetivos perseguidos varían en la bibliografía, incluyendo: segmentación de órganos y subestructuras, detección de tumores, clasificación de muestras (imágenes completas) y registración.

La red típica en aprendizaje profundo para PDI es la CNN, que está formada por dos fases: la primera se llama fase de extracción de características y contiene varias capas de filtros convolucionales (filtros especiales lineales) y otras capas que mejoran la calidad de las características (incluyendo submuestreo y max-pooling); y la segunda se llama fase de clasificación y está definida, en general, por una red neuronal tradicional de clasificación. La fase de extracción de características tiene un elevado número parámetros de ajuste, lo que supone un alto costo computacional durante el entramiento,

pero es manejable con el hardware de procesamiento gráfico que se dispone en la actualidad.

Por otro lado, dado el alto número de parámetros, las CNN requieren un gran número de imágenes de entrenamiento (con su Gold-Standard), lo que no siempre es posible, principalmente en problemas de imágenes médicas. En enfoques recientes, se ha trabajado en lo que se conoce como transfer-learning; que consiste en utilizar una CNN previamente entrenada y reemplazar la fase de clasificación por un nuevo clasificador (habitualmente una red neuronal tradicional), para reentrenar la red con el conjunto de imágenes de interés, reajustando sólo los parámetros internos de la fase de clasificación; lo que reduce drásticamente el tiempo de entrenamiento y ha obtenido buenos resultados en imágenes médicas (Akcay, et al. 2016; Lucena, et al. 2017; Comas, et al. 2020; Meschino, et al. 2020).

Se propone en el presente Proyecto Final abordar la detección automática de anomalías en radiografías de tórax por medio de CNN. A partir del objetivo general y los objetivos específicos definidos, se propone la concreción de actividades de investigación, intercalando etapas de estudio de técnicas, de implementación y de análisis de resultados. El trabajo requerirá un estudio profundo de la problemática para comprender su complejidad y definir un abordaje apropiado, y, también, de las CNN (que se realizará con apoyo de los directores y de los investigadores del LPI abocado a la temática) y la posterior implementación computacional de los modelos: incluyendo diseño de las redes, entrenamiento y evaluación de desempeño.

7.2.8 Enfoque

Se iniciará con un estudio teórico de modelos de aprendizaje profundo y, especialmente, de CNN, profundizando en aspectos críticos del diseño tanto de la fase de extracción de características como de la fase de clasificación. Se identificarán funciones específicas de las distintas capas: neuronas convolucionales, neuronas de reducción del muestreo (incluyendo submuestreo y max-pooling), capas recurrentes, etc. En cuanto a la fase de clasificación, se analizarán criterios generales para la definición del tipo y características del clasificador final. Se realizará un relevamiento de arquitecturas definidas con fines específicos en procesamiento de imágenes médicas. Se estudiarán algoritmos de aprendizaje (incluyendo algoritmos de retro-propagación del error y criterios de fin de entrenamiento). Se incluirán enfoques de validación de los resultados y de transfer-learning. Asimismo, se abordará el procesamiento de radiografías de tórax, incluyendo trabajos actuales en la temática, analizando los enfoques elegidos y los resultados obtenidos. Se partirá de una base de datos con más de 100.000 imágenes con la

presencia de diferentes anomalías relacionadas con posibles patologías (Wang, et al. 2017b). De acuerdo con la pertinencia, se incluirán otras bases de datos de radiografías a las que se pueda acceder a través de la bibliografía. En la Figura 1 se muestra un esquema general del tipo de metodología de estudio a seguir en este trabajo. Partiendo de imágenes con anomalías detectadas (Gold-Standard), se propondrá una CNN como solución inicial para posteriormente realizar el entrenamiento, evaluación de los resultados y reajustes. Toda esta etapa la realizará la estudiante a partir de material que brindarán los directores; con un acompañamiento constante.

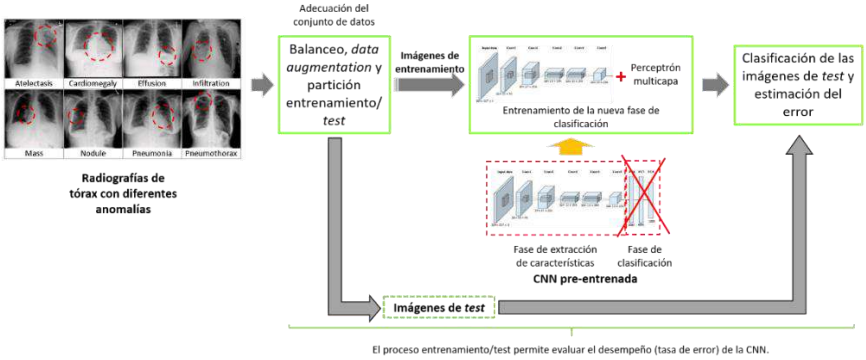


Figura 7.2.1 - Esquema general del tipo de metodologías de entrenamiento/validación a implementar en el Trabajo Final. En el esquema se parte de un enfoque de transfer-learning.

Todos los modelos planteados (incluyendo CNN utilizadas en la bibliografía para procesamiento de imágenes médicas así como otras que sean propuestas como resultado del presente Proyecto) se programarán sobre hardware dedicado con el que cuenta el Laboratorio: un servidor de procesamiento que incluye una GPU NVIDIA Titan V donada por NVIDIA a partir de la *NVIDIA GPU Grant*. Se trabajará sobre *Python*, utilizando la *API Keras* y *TensorFlow*. Esta etapa requerirá un profundo estudio de las funciones básicas de la API y el desarrollo de programas específicos para: la definición la estructura básica de la CNN, la definición y adición de capas, el traspaso de parámetros internos a partir de redes pre-entrenadas (*transfer-learning*), la carga en memoria de la base de datos (incluyendo instrucciones específicas para el manejo de tensores y el aprovechamiento de los núcleos específicos de la GPU), el entrenamiento de la red, y la partición de la base de datos para la validación. En función de la performance obtenida, podrán requerirse etapas de ajuste y optimización de los programas implementados. A los fines de facilitar la reutilización del código, los programas se dividirán por funciones específicas, con una completa documentación.

7.2.9 Alcance

La solución de problemas de imágenes médicas es siempre relevante por su impacto en las tareas de diagnóstico clínico realizadas por los médicos. La concreción de este

Proyecto Final permitirá avanzar en este sentido al estudiar un problema de enorme relevancia en la actualidad como lo es la detección automática de anomalías en radiografías de tórax. Asimismo, el avance sobre CNN en imágenes médicas permitirá estudiar su aplicabilidad ampliando el conocimiento actual que se tiene sobre su uso en este tipo de problemas. Finalmente, el desarrollo de programas específicos que incluyan las etapas de diseño, entrenamiento y validación de CNN, con su correspondiente documentación, permitirá desarrollar en el futuro una librería de funciones (un conjunto de programas con funciones específicas) que implementen CNN sobre imágenes que se sumarán a otras técnicas actualmente en desarrollo en el Laboratorio para el análisis de la estructura decisoria de las redes.