UNIVERSITAT POLITÈCNICA DE VALÈNCIA

DSIC DEPARTAMENT DE SISTEMES INFORMÀTICS I COMPUTACIÓ

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Dpto. de Sistemas Informáticos y Computación

Optimización de los procesos de triaje en servicios de urgencias mediante transformadores de visión sobre radiografías de tórax

Trabajo Fin de Máster

Máster Universitario en Inteligencia Artificial, Reconocimiento de Formas e Imagen Digital

AUTOR/A: Soler Guiral, Ferran

Tutor/a: Paredes Palacios, Roberto

Cotutor/a externo: DOLZ ZARAGOZA, MANUEL FRANCISCO

CURSO ACADÉMICO: 2022/2023

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

DSIC
DEPARTAMENTO DE SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

Máster Universitario en Inteligencia Artificial, Reconocimiento de Formas e Imagen Digital

Trabajo Final de Máster

Optimización de los procesos de triaje en servicios de urgencias mediante transformadores de visión sobre radiografías de tórax

Autor:
Ferran Soler Guiral

Tutor:
Roberto Paredes Palacios
Cotutores:
Manuel F. Dolz Zaragozá

Departamento de Sistemas Informáticos y Computación

Curso académico 2022/2023

Resumen

La sencillez de adquisición de las radiografías simples de tórax, así como su gran utilidad en la detección de diversas patologías, las convierten en una de las pruebas más solicitadas en los servicios de urgencias hospitalarias. Sin embargo, la alta demanda de los servicios radiológicos hace inviable que todas ellas puedan ser revisadas e informadas a tiempo, reduciendo así la calidad de la atención al paciente.

En respuesta a esta problemática, este trabajo propone el desarrollo de una herramienta de apoyo al radiodiagnóstico, basada en redes neuronales, para optimizar los procesos de triaje y priorización de los pacientes más urgentes. Para desarrollar esta herramienta, se entrena y analiza la precisión de un conjunto de transformadores de visión, unos nuevos modelos de red neuronal originalmente utilizados en tareas de procesamiento de lenguaje natural capaces de superar las prestaciones de las redes convolucionales clásicas. En el trabajo, se describen las técnicas empleadas para *i)* el entrenamiento de los modelos; *ii)* la transferencia de aprendizaje mediante modelos preentrenados; *iii)* el uso del paradigma profesor-estudiante; *iv)* el aumento de datos realizado para mejorar la generalización; y *v)* la optimización de hiperparámetros. El entrenamiento de los modelos se ha llevado a cabo utilizando las bases de datos PadChest y ChestX-ray14, previamente analizadas y procesadas para entrenar los transformadores de visión mediante un nuevo sistema de agrupación por zonas patológicas y nivel de urgencia. Los resultados obtenidos sobre los datos de validación superan el 90 % de AUC, mejorando así la precisión de las redes convolucionales, tradicionalmente empleadas en tareas de visión por computador.

Palabras clave

Imagen médica, Transformers visión, Ayuda al diagnóstico

Índice general

Capítulo 1

Introducción

En este capítulo se exponen brevemente las bases del proyecto, así como sus objetivos, explicando tanto el contexto como la motivación de los mismos. Además, se detalla la estructura de esta memoria.

1.1. Contexto y motivación del trabajo

Los Servicios de Urgencias Hospitalarios (SUH) se han convertido en uno de los puntos más importantes del Sistema Nacional de Salud (SNS), siendo la radiografía simple uno de los estudios más solicitados para los pacientes [1]. A continuación, se evalúa el estado actual de los servicios radiológicos, y en concreto, el de la radiografía simple, en base a diversos factores:

- La tendencia al alza en la utilización los SUH desde el 2010, tal y como indica el informe anual del SNS de 2021[1], ha producido un incremento en el número de estudios radiológicos que deben ser informados de forma casi inmediata con un alto nivel técnico [2].
- El diseño organizativo subóptimo de los servicios de radiológicos y la falta de adecuación de las plantillas del personal a la carga de trabajo [3].
- La implantación de técnicas que requieren de un mayor tiempo de análisis como ecografías, tomografías o resonancias magnéticas, dejan en un segundo plano a la radiografía simple [3].

[1] https://www.sanidad.gob.es/estadEstudios/estadisticas/sisInfSanSNS/tablasEstadisticas/InfAnSNS.htm

- La elevada carga de trabajo en otras áreas, como la docencia de residentes, la organización de los pacientes o la investigación [4].

En base a estos factores, se comprende la sobresaturación del servicio, así como la imposibilidad del personal radiológico para informar todas las radiografías simples, algunas de atención inmediata. Esto repercute negativamente en la calidad de las imágenes, los informes y el tiempo de espera de los pacientes.

El proyecto RELIANCE[2] (Radiología intELIgente en procesos AsisteNCiales urgEntes), desarrollado por el grupo de investigación HPC&A[3] de la Universitat Jaume I e iniciado en marzo de 2022, busca desarrollar una herramienta, basada en aprendizaje profundo y redes neuronales, que permita diferenciar las radiografías simples de tórax entre urgentes, patológicas y sanas, optimizando el proceso de triaje en los SUH. Con el objetivo de desarrollar una herramienta con la mejor precisión posible se investigan los avances más recientes en el campo de la visión por computador, entre los cuales se encuentran los transformadores de visión.

El proyecto RELIANCE parte de un estado previo desarrollado también por el grupo HPC&A bajo el proyecto RADIANT (RADiología Inteligente de precisión en procesos Asistenciales urgeNTes), dónde los investigadores proponen el entrenamiento de redes neuronales convolucionales mediante transferencia de aprendizaje, utilizando el conjunto de datos de PadChest [5]. En concreto, utilizando una agrupación por zonas anatómicas, estas redes neuronales alcanzan un AUC[4] de 0,861.

Durante ambos proyectos, el grupo HPC&A colabora con la empresa ActualTec S.L.[5] para la obtención estudios de validación a través de un conjunto de datos no etiquetado.

1.2. Objetivos del trabajo

El objetivo del proyecto RELIANCE es adaptar la herramienta desarrollada bajo el proyecto RADIANT a las necesidades del entorno de urgencias, con la intención final de obtener el marcado CE para productos sanitarios. Para ello, el proyecto plantea diversos objetivos específicos, que quedan alineados con los objetivos de este trabajo de fin de máster. Estos objetivos son:

1. Ampliar la clasificación por zonas anatómicas añadiendo un nivel de urgencia para cada una de ellas, con el objetivo de optimizar el proceso de triaje priorizando a los pacientes

[2] https://www.uji.es/serveis/ocit/base/empresa/patents/reliance
[3] www.hpca.uji.es
[4] *Area Under the ROC*: Métrica utilizada en tareas de clasificación multietiqueta
[5] http://www.actualmed.com/

urgentes.

2. Ampliar la clasificación para incluir patologías concretas de importancia en los SUH.

3. Aumentar la cantidad de datos de entrenamiento con el objetivo de mejorar la precisión y generalización de la herramienta.

4. Analizar diversas arquitecturas de transformadores de visión, y comparar su rendimiento con las redes neuronal convolucionales tradicionales.

1.3. Estructura de la memoria

La memoria de este trabajo describe cómo han sido alcanzados los diferentes objetivos, así como los resultados obtenidos, organizándolos en los siguientes capítulos:

- En el Capítulo 2 se describen los conceptos básicos necesarios para comprender esta memoria, como el almacenamiento de radiografías, la transferencia de aprendizaje, el paradigma estudiante-profesor o las técnicas de regularización. Además, se estudia el funcionamiento básico de los transformadores y la arquitectura BERT, conceptos necesario para describir el funcionamiento de un transformador de visión.

- En el Capítulo 3 se describe el funcionamiento de los transformadores de visión, explicando diferentes arquitecturas y seleccionando diferentes modelos para ser explorados en el análisis de radiografías.

- En el Capítulo 4 se describen los avances en la literatura desde 2017 hasta la actualidad, pasando por el uso de las redes neuronales convolucionales sobre diferentes bases de datos hasta la aplicación de transformadores de visión.

- En el Capítulo 5 se detalla el procesamiento de datos necesario para las bases de datos utilizadas durante el trabajo; PadChest [5], ChestX-ray14 [6] y el conjunto de datos ofrecido por la empresa ActualTec S.L.

- En el Capítulo 6 se explican en profundidad algunos conceptos del estado previo del sistema, como los hiperparámetros y la metodología de entrenamiento. Se describen los detalles y optimizaciones del entrenamiento de las diferentes redes neuronales en PyTorch[6] con multi-GPU. También se realizan las primeras comparativas de resultados entre las redes neuronales convolucionales y los transformadores de visión.

[6] https://pytorch.org/

- En el Capítulo 7 se describe la búsqueda de hiperparámetros realizada, y se comentan los resultados y conclusiones más importantes de la misma en cuanto a AUC y eficiencia. En base a dichas conclusiones, se entrenan diversos modelos que sirven para establecer los resultados finales del trabajo.
- En el Capítulo 8 se concluye el trabajo realizado, recapitulando los resultados más destacables y mostrando nuevas ramas de investigación a futuro.

Capítulo 2

Conceptos previos

En este capítulo de la memoria se describen los conceptos básicos necesarios para la comprensión del desarrollo del trabajo. En primer lugar, se describe el formato DICOM de almacenamiento de radiografías e imagen médica. Posteriormente, se describen diferentes conceptos de la redes neuronal como la transferencia de aprendizaje, las redes convolucionales, las técnicas de regularización o el paradigma estudiante profesor. En última instancia, se describe las redes de tipo transformador aplicadas al procesamiento de lenguaje natural.

2.1. Almacenamiento de radiografías y el formato DICOM

El protocolo *Digital Imaging and Communications in Medicine* (DICOM) es el estándar de comunicaciones y almacenamiento de imágenes médicas más extendido del mundo. No solo se encarga de almacenar radiografías simples, resonancias magnéticas o tomografías computerizadas, sino también de definir protocolos de visualización de imágenes, tanto 2D como 3D, compartición de imágenes de manera segura o impresión. A continuación se describen algunos atributos importantes almacenados en una radiografía en formato DICOM:

- La propia imagen médica.
- Varios identificadores únicos, como el identificador de paciente o de estudio.
- La localización anatómica del estudio. Por ejemplo, en las radiografías de tórax se buscan valores como CHEST, TORAX, THORAX, PECHO, entre otros. No hay un estándar claro en su escritura ya que se trata de un campo opcional.

- La vista con la que la zona anatómica ha sido analizada. En el caso de radiografías de tórax se tienen: PA (Posterior-Anterior), AP (Anterior-Posterior), lateral y costal. Además, la proyección AP tiene dos subtipos diferenciados, AP vertical, que se toma con el paciente de pie o acostado hacia un lado y AP horizontal, que se toma con el paciente acostado hacia arriba. Mientras que las proyecciones AP y PA se consideran similares y se diferencian principalmente en la orientación y amplitud del corazón, la proyección Lateral es diferente, y se utiliza cuando la radiografía en proyección AP o PA no es concluyente. Por otro lado, la proyección Costal se centra en la observación de las costillas.

- La profundidad de bits de la imagen. Los valores habituales son 8, 12 y 16 bits.

- El esquema de color utilizado al almacenar la imagen, que puede contener la información *inverse* o *identity*. En el primer caso se tiene una imagen donde el blanco queda identificado como 0 en lugar del valor máximo habitual y debe ser convertida a *identity* para trabajar con la herramienta de manera adecuada.

2.2. Problema de clasificación

Un problema de clasificación es aquel en el que se desea conocer la clase o etiqueta para una cierta entrada no etiquetada. Algunos ejemplos podrían ser la detección de *spam* en un correo, o la clasificación de un dígito manuscrito. Las **redes neuronales** resuelven los problemas de clasificación calculando la probabilidad de que la entrada pertenezca a cada una de las clases o etiquetas en una última capa lineal de clasificación, con tantas neuronas como clases o etiquetas. La configuración de esta última capa lineal, así como la función de pérdida utilizada para corregir la red neuronal, dependen del tipo de problema de clasificación que se tenga:

- **Problema de clasificación binario**: Clasifica una entrada entre dos posibles clases. Algunos ejemplos podrían ser la detección de *spam*, fraude, o pacientes patológicos. En esta situación, la última capa lineal queda formada por una única neurona, con una activación de tipo *sigmoid*, de forma que se obtiene un valor en el intervalo (0,1), visto como una probabilidad. Se aplica la función de perdida entropía cruzada binaria (*binary cross entropy* o BCE).

- **Problema de clasificación multiclase**: Clasifica una entrada en una única clase entre más de 2 clases. Algunos ejemplos podrían ser la detección de dígitos manuscritos o tipos de plantas. En esta situación, la última capa lineal queda formada por tantas neuronas como clases y se emplea una activación de tipo *softmax*, de forma que se obtiene una función de probabilidad en donde la salida de todas las neuronas suman uno. Se aplica la función de pérdida entropía cruzada categórica (*categorical cross entropy* o CCE).

- **Problema de clasificación multietiqueta**: Clasifica una entrada en una o varias etiquetas al mismo tiempo. Un ejemplo podría ser la detección de diferentes enfermedades en una radiografía, puesto que un paciente puede padecer varias enfermedades a la vez. En esta situación, la última capa lineal queda formada por tantas neuronas como etiquetas y con una activación de tipo *sigmoid*, de forma que se obtiene una probabilidad para cada etiqueta. Se aplica la función de pérdida BCE tratando cada neurona y su respectiva etiqueta como un problema binario independiente. Una métrica muy utilizada en la comparativa de la literatura para este tipo de problema es la métrica AUC o AUROC (*Area Under the ROC*), que evalúa el balance entre TPR (*True Positive Rate*) y FPR (*False Positive Rate*) para diferentes umbrales. Se trata de un valor entre 0 y 1, dónde 1 es el clasificador perfecto, 0,5 un clasificador aleatorio y 0 un clasificador inverso perfecto, dónde todas las muestras son etiquetadas al contrario de como se debería. Aunque este último valor es teóricamente posible, en la práctica, dicho valor no se alcanza, oscilando en valores alrededor de 0,5 durante los primeros lotes del entrenamiento.

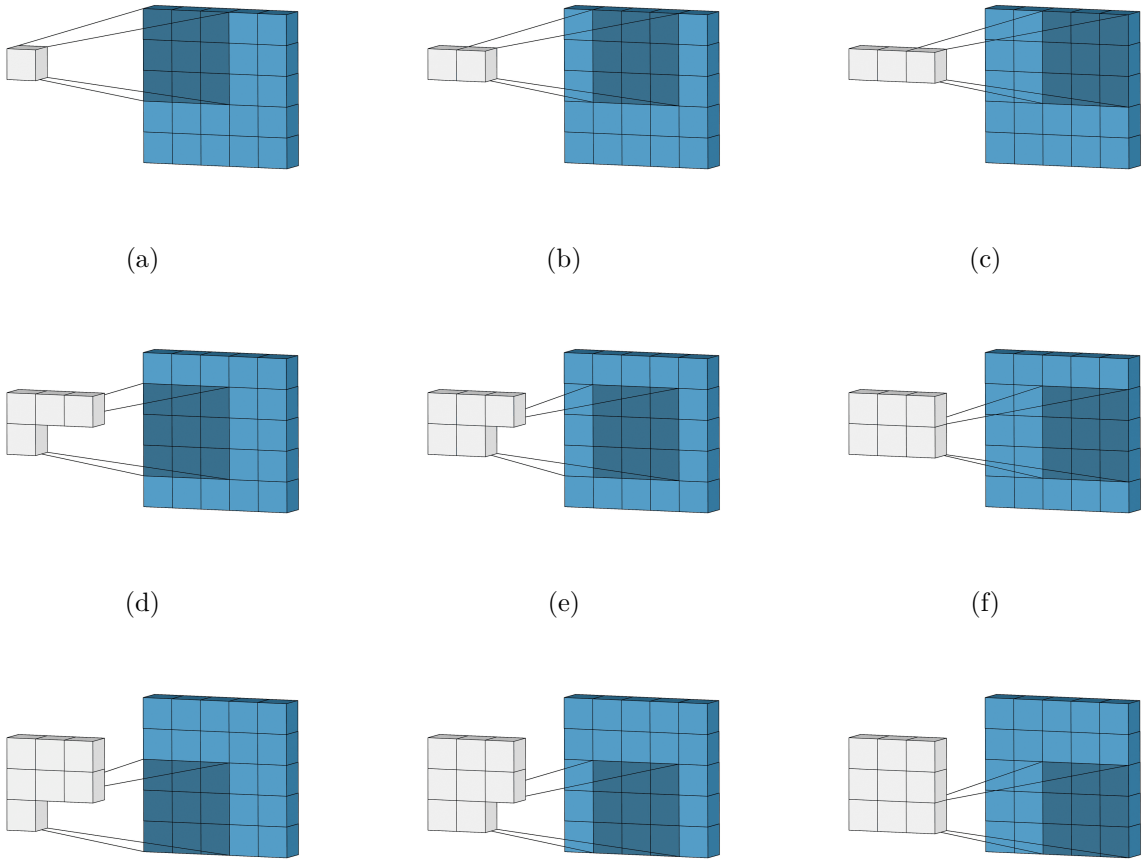
2.3. Redes neuronales convolucionales

Las **redes neuronales convolucionales** (*Convolutional Neural Networks* o CNN) se basan en la aplicación de una **operación de convolución** de forma repetida a través de diversas capas. Esta operación aplica diversos productos escalares matriciales, dónde una de las matrices es un parámetro entrenable denominado *kernel* y la otra matriz es una parcela restringida de la entrada. En el caso de la operación de convolución 2D, dicha entrada queda formada por una serie de canales (C) y las dimensiones anchura $\times$ altura ($W \times H$).

El funcionamiento del producto escalar de matrices entre el *kernel*, de tamaño $K_W \times K_H$, y un canal de la entrada queda ilustrado en la Figura 2.1. Se observa que la resolución de entrada queda reducida a 3×3 en la salida. En caso de querer mantener la resolución de entrada se debe añadir relleno (*padding* o P) alrededor del canal de entrada. En el caso de que se desee reducir significativamente la resolución de entrada, se debe aplicar un *stride* (S) superior a 1, lo que permite desplazar la aplicación del *kernel* en un número de píxeles concreto en lugar del siguiente píxel. Tanto el relleno como el *stride* pueden aplicarse en distinta medida en anchura y en altura. Con todos estos conceptos, se define a continuación la fórmula utilizada para calcular la dimensión de salida, por ejemplo, en la anchura:

$$W_{out} = \frac{W_{in} - K_W + 2P_W}{S_W} + 1$$

Este procedimiento se repite para cada uno de los C canales de entrada utilizando un *kernel* diferente, para posteriormente sumar las C matrices resultantes en un único canal. De esta



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figura 2.1: Aplicación del *kernel* sobre un canal de entrada mediante producto escalar.

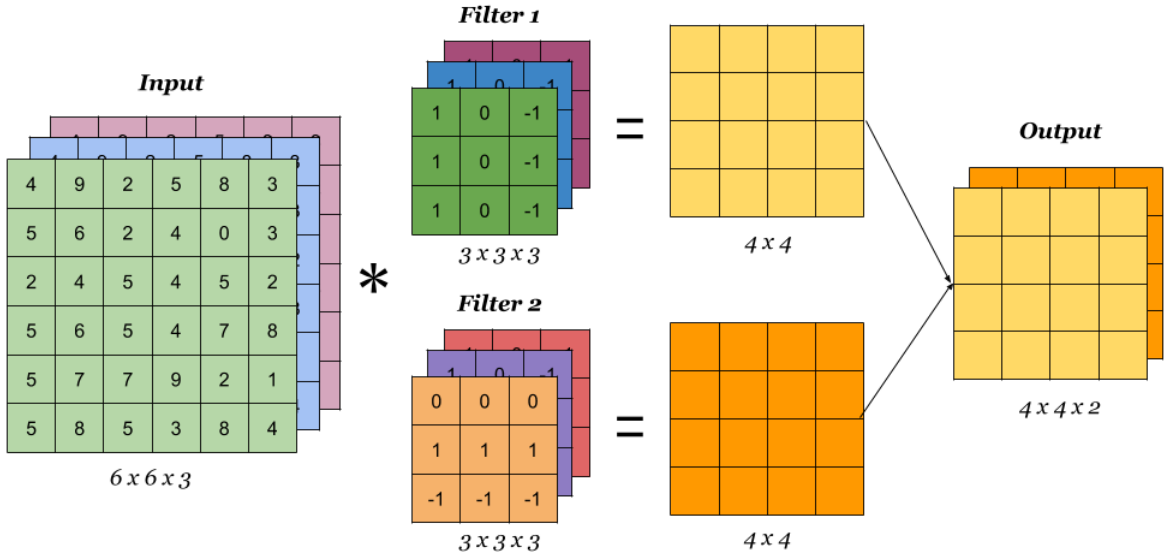


Figura 2.2: Funcionamiento de una operación de convolución al completo[7].

forma, para aplicar una operación de convolución se requiere del mismo número de *kernels* que de canales de entrada. El conjunto de los diferentes *kernels* se denomina filtro. Finalmente, el número de filtros utilizados en una capa convolucional determina el número de canales de salida, dado que cada filtro obtiene un único canal como resultado. En la Figura 2.2 se ilustra un ejemplo con tres canales de entrada, de forma que cada filtro está formado por 3 *kernels* y 2 canales de salida, de forma que se tiene una capa convolucional con 2 filtros. Se tienen *kernels* de tamaño 3×3 con *stride* 1 y sin relleno, por lo que la resolución de entrada de 6×6 se convierte a 4×4 en la salida.

2.4. Transformadores

Los **transformadores** (*transformers*) son unas redes neuronales especialmente diseñadas para el procesamiento de lenguaje natural, con aplicaciones en diferentes áreas como clasificación de texto, traducción, predicción de siguiente palabra o frase e incluso escritura de resúmenes. En esta sección se estudia BERT [8], una de las arquitecturas más utilizadas y sobre la que se han inspirado la gran mayoría de transformadores.

El texto de entrada se convierte a una serie de *tokens* únicos, pertenecientes a un diccionario, entre los cuales se producen interacciones mediante mecanismos de autoatención (*self-attention*) dentro de la red neuronal. La combinación de las diferentes interacciones entre todos los *tokens* forman la salida del modelo. Además de los *tokens* provenientes del texto, se añaden otros *tokens*

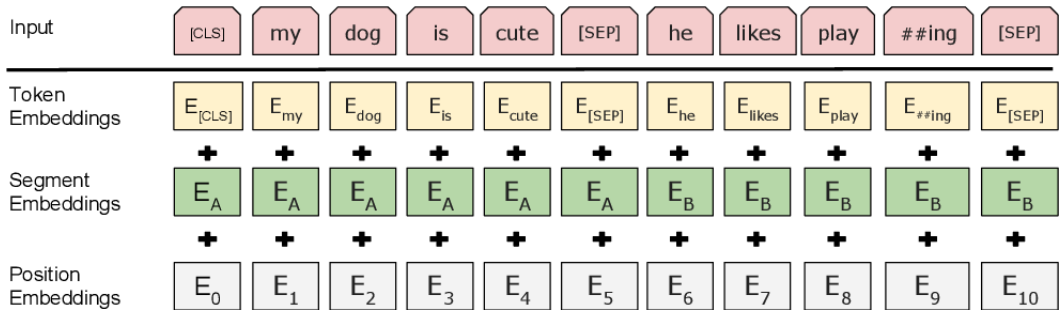


Figura 2.3: Embeddings de entrada en el modelo BERT [8].

especiales necesarios para el modelo:

- ***Token* de clasificación** (`[CLS]`): Se utiliza como representación del texto para tareas de clasificación. Se incluye justo al principio de la secuencia de entrada al modelo.
- ***Token* de separación** (`[SEP]`): Se emplea principalmente para delimitar pares de frases de entrada en tareas como predicción de siguiente frase o pregunta-respuesta.
- ***Token* de relleno** (`[PAD]`): Dado que el modelo requiere de un tamaño entrada constante, denominado longitud de secuencia (*sequence length*), este *token* se utiliza para rellenar el texto hasta llegar a dicho tamaño.
- ***Token* de máscara** (`[MASK]`): Se utiliza para representar que uno de los *tokens* originales del texto se encuentra enmascarado.

Cada *token* del vocabulario se encuentra representado por *embeddings* con un determinado *hidden size* fijado por la arquitectura de transformador concreta. Tal y como se observa en la Figura 2.3, el transformador trabaja sobre los *token embeddings*, junto con otros dos tipos de *embeddings*:

- ***Embeddings* de segmento**: Se encargan de indicar la frase a la que pertenece el *token*. Se utiliza en conjunto con el *token* especial `[SEP]` en tareas con dos frases.
- ***Embeddings* de posición**: Dado que el modelo no procesa cada uno de los *tokens* de forma secuencial, como si ocurre con modelos recurrentes, no se tiene el orden con el que los *tokens* forman la secuencia de entrada. Los *embeddings* de posición solucionan este problema añadiendo dicho orden.

De esta forma, la entrada del modelo está compuesta por los embeddings de *token*, segmento y posición. Como salida, se obtiene un estado oculto del con el tamaño de *hidden size* por cada

token de entrada y por cada capa del modelo. Estos estados ocultos se utilizan de distintas formas según la tarea deseada.

2.5. Transferencia de aprendizaje

En general, tanto las redes neuronales de tipo CNN como las de tipo transformador se están volviendo cada vez más complejas, requiriendo de una mayor cantidad de datos de entrenamiento, que en algunas ocasiones no se encuentra disponible. La **transferencia de aprendizaje** (*transfer learning*) propone solucionar este problema preentrenando la red neuronal sobre una tarea diferente y sobre la que se tiene una mayor cantidad de datos. En la aplicación de esta técnica, la red neuronal se divide en dos partes bien diferenciadas:

- **Extracción de características**: Se encarga de obtener una representación compacta a partir de la entrada, de forma que puede ser trabajada con mayor sencillez. En este procedimiento se obtienen diferentes características informativas y no redundantes que representan la entrada.
- **Clasificador**: Se encarga de clasificar la entrada en las diferentes clases o etiquetas partiendo de las características extraídas. Puede tratarse de una única capa lineal de clasificación, o estructuras más complejas, como perceptrones multicapa o redes neuronales recurrentes, que en cualquier caso deben finalizar en una capa lineal de clasificación adecuada a la tarea.

El preentrenamiento de una red neuronal es computacionalmente costoso, por lo que se descarga directamente con una estructura predefinida y optimizada ya preentrenada. En general, cuando se desarrolla una nueva red neuronal, se publica ya preentrenada por los propios desarrolladores. La transferencia de aprendizaje consiste en mantener los pesos preentrenados de la extracción de características y sustituir el clasificador para adaptarse a la tarea deseada.

2.5.1. Transferencia de aprendizaje en redes CNN

Las redes de tipo CNN, mediante el uso de *kernels* capaces de analizar grupos de píxeles de una imagen, se encuentran especialmente diseñados para tareas de visión por computador. Habitualmente, el preentrenamiento de las CNNs se basa en una tarea de clasificación supervisada mediante la base de datos ImageNet [9], en sus dos versiones:

- **ImageNet-21k** es un base de datos formada por 14 197 122 imágenes clasificadas en

21 841 etiquetas no exclusivas [10]. Esto es debido a que las diferentes etiquetas forman jerarquías. Por ejemplo, una imagen de una silla puede ser clasificada como silla y como mueble.

- **ImageNet-1k**, simplemente referida como ImageNet, es un subconjunto de ImageNet-21k formado por 1,2 millones de imágenes clasificadas en 1 000 clases exclusivas [9]. El hecho de que las clases no formen jerarquías, así como la reducida cantidad de datos, convierte ImageNet-1k en la base de datos mas utilizada para evaluar prestaciones y preentrenar redes neuronales.

Con esta base de datos se obtiene una red neuronal preentrenada en la tarea de clasificación de ImageNet. De esta forma, mientras que el clasificador se ajusta a dicha tarea, y debe ser sustituido para una nueva tarea, la extracción de características puede ser reutilizada en otras tareas. Además del clasificador, se deben ajustar los datos de entrenamiento de la tarea deseada al formato de los datos con el que el modelo ha sido preentrenado. Esto incluye:

- **Resolución de entrada**: Mientras que algunos modelos pueden funcionar con cualquier resolución de entrada, otros modelos requieren de redimensionar la entrada para poder ser utilizados. También pueden requerir de una resolución mínima o máxima para ser funcionales.
- **Número de canales**: Los modelos preentrenados requieren una imagen de entrada de 3 canales, puesto que han sido preentrenados en conjuntos de imágenes convencionales como ImageNet. Dado que las radiografías son imágenes en escala de grises se debe triplicar el canal para poder utilizar estos modelos.
- **Normalización**: Durante el preentrenamiento es posible que se normalicen las imágenes siguiendo una media y desviación concreta en cada canal. De esta forma, si la red ha sido preentrenada en ImageNet y las imágenes han sido normalizadas según la media y desviación de ImageNet, es habitual normalizar las imágenes de la tarea deseada con la misma media y desviación.

2.5.2. Transferencia de aprendizaje en redes transformador

Las redes de tipo transformador aplicadas al procesamiento de lenguaje natural recurren con frecuencia a grandes conjuntos de texto no etiquetados. En concreto, durante el preentrenamiento del modelo BERT se han empleados las siguientes tareas auto supervisadas:

- ***Masked Language Modeling***: Consiste en enmascarar de forma aleatoria un porcentaje de los *tokens* de entrada, sustituyéndolos por el *token* [`MASK`]. El modelo debe predecir las

palabras que han sido enmascaradas, entrenando los pesos del modelo y los diferentes embeddings.

- **Predicción de siguiente frase**: Consiste en introducir pares de frases, A y B, de forma que en el 50 % de los casos A y B son frases consecutivas, mientras que el otro 50 % son frases aleatorias. El modelo debe predecir si A y B son en realidad frases consecutivas, entrenando la capacidad del modelo para entender la relación entre dos frases.

Los modelos preentrenados con estas u otras técnicas, se utilizan en la transferencia de aprendizaje en diferentes áreas. Una de las más importantes es la tarea de clasificación, dónde habitualmente se recurre al estado oculto del *token* `[CLS]` de algunas de las capas.

2.5.3. Metodología de entrenamiento

El entrenamiento mediante transferencia de aprendizaje ajusta únicamente los pesos del clasificador, congelando los parámetros del resto de la red neuronal. Sin embargo, puede resultar conveniente ajustar los pesos del resto del modelo con un ratio de aprendizaje (*learning rate* o LR) menor, lo que se denomina *fine-tuning*. Durante el trabajo se ha utilizado una metodología de entrenamiento por doble fase:

1. Se sustituye el clasificador del modelo preentrenado, y se entrena con un LR elevado durante unas pocas épocas.

2. Se entrena el modelo al completo durante algunas épocas más con un LR menor, al que se aplican reducciones cuando la precisión del modelo permanece constante durante varias épocas. Esta técnica se denomina *reduce on plateau*.

Es importante remarcar que, no efectuar la primera fase, entrenando directamente todo el modelo, puede producir una perdida del conocimiento adquirido durante el preentrenamiento, debido a la inicialización aleatoria del clasificador en su sustitución.

Otra práctica muy habitual en la transferencia de aprendizaje, y en general, en los algoritmos de aprendizaje automático, es el uso de diversos modelos de predicción a la vez en una estructura denominada ***ensemble***. Para ello, se entrenan modelos con diferentes arquitecturas y metodologías en una tarea concreta y mediante un sistema de votación se forma una predicción final en base a la predicción de cada uno de los diferentes modelos.

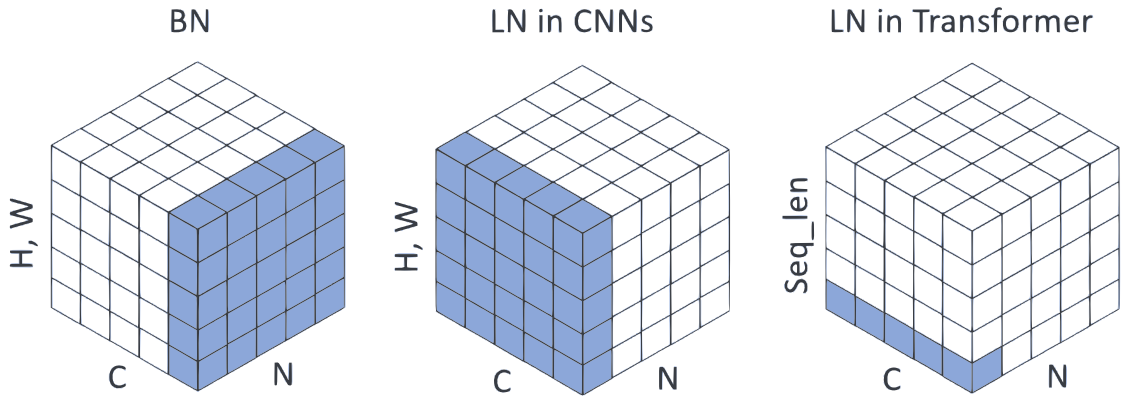


Figura 2.4: Comparativa entre BN y LN [11].

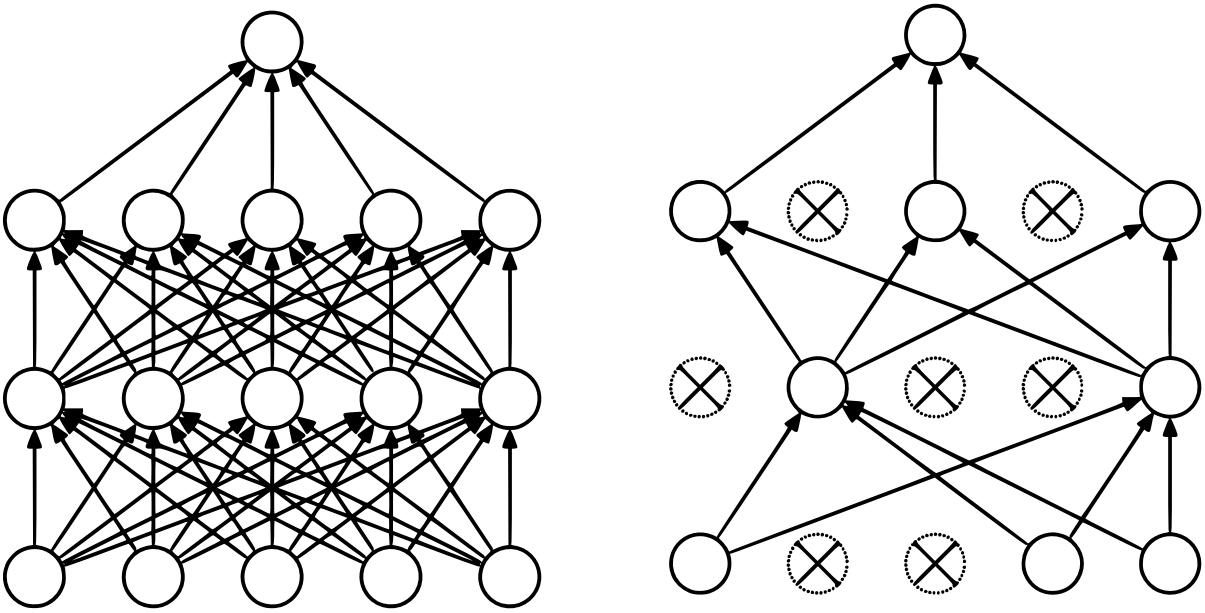
2.6. Técnicas de regularización

En esta sección se explican diversas técnicas de regularización de redes neuronales utilizadas durante el trabajo.

Por un lado, la capa de **normalización de lote** (*Batch Normalization* o BN) se introduce entre otras dos capas de la red neuronal, recibiendo como entrada la salida de la capa anterior y pasando la salida a la siguiente capa. La capa BN se encarga de normalizar el lote a media 0 y varianza 1, para luego desplazarlas mediante dos parámetros entrenables, que se ajustan a la tarea y los datos durante el entrenamiento. También almacena la media móvil exponencial (*Exponential Moving Average* o EMA) de la media y varianza obtenida durante el entrenamiento, y que sirven para normalizar durante la inferencia.

Además de la capa BN existen otros tipos de normalización como la **normalización de capa** (*Layer Normalization* o LN) que en lugar de normalizar a lo largo del lote, normaliza a lo largo de las características de la para cada entrada de forma individual, sin necesidad de tener en cuenta el resto del lote. Se utiliza habitualmente en arquitecturas de transformadores y entrenamiento multi-GPU, dónde un lote se divide en varias GPUs, obligando a su sincronización en caso de usar la capa BN. En la Figura 2.4 se puede observar la diferencia de funcionamiento entre las capas BN y LN. En las CNNs, la capa BN se aplica en las dimensiones de lote (N) y canal (C), mientras que la capa LN se aplica a nivel de canal y resolución (H,W). También se muestra que LN se aplica únicamente sobre la dimensión oculta en una red de tipo transformador.

Por otro lado, la capa ***dropout*** se introduce entre dos capas de la red, recibiendo como entrada la salida de la capa anterior y pasando la salida a la siguiente capa. Durante el entrenamiento, esta capa se encarga de ignorar un conjunto de neuronas de la capa anterior de forma aleatoria, tal y como se observa en la Figura 2.5 para minimizar el sobreaprendizaje y mejorar la



(a) Funcionamiento estándar.

(b) Después de aplicar *dropout*.

Figura 2.5: Efecto producido por el *dropout* en las diferentes capas de un perceptrón multicapa [12].

generalización. Durante la fase de inferencia se deshabilita y no produce ningún efecto. Durante el trabajo se utilizan dos tipos de *dropout*:

- **Tradicional**: Se trata de la versión más básica y se encarga de ignorar un conjunto de neuronas de la capa anterior en su totalidad. La probabilidad con la que ignora cada una de las neuronas depende de un ratio que es pasado como parámetro de la capa. En la Figura 2.5 se puede observar el efecto de la *dropout* entre las diferentes capas ocultas de un perceptrón multicapa.
- **Alpha**: Funciona de forma similar al *dropout* tradicional, pero manteniendo los datos normalizados a media cero y desviación uno. También recibe un hiperparámetro de ratio.

Por último, las técnicas de **aumento de datos** modifican la imagen de entrada, de forma que la misma imagen nunca se entrena dos veces, mejorando la generalización. Estas transformaciones no pueden ser totalmente aleatorias, sino que deben respetar la naturaleza de la imagen para mantener las etiquetas originales. En este trabajo, dada la tarea de análisis de radiografías, se aplican las siguientes técnicas de aumento de datos:

- **Volteo horizontal**: La imagen se gira en torno al eje de simetría vertical con un 50 % de

probabilidad. De esta forma, una patología que se encuentra en el lado izquierdo se entrena como si estuviese en el lado derecho el 50 % de las veces y viceversa. Además, cambia la orientación del corazón, de forma que las imágenes en proyección PA pueden entrenarse como si fueran proyección AP y viceversa. Su efecto se observa en la Figura 2.6b.

- **Translación**: La imagen se desplaza en sentido horizontal y vertical, cortando parte de la radiografía y rellenando con píxeles en negro en el lado contrario. Su efecto se detalla en la Figura 2.6c.
- ***Shear***: La imagen se distorsiona a lo largo de varios ejes, de forma que se crean nuevos ángulos de percepción de la imagen. Su efecto puede verse en la Figura 2.6d.
- **Rotación**: La imagen se rota en sentido horario o antihorario en un cierto ángulo limitado. Su efecto puede observarse en la Figura 2.6e.
- **Escalado**: La imagen se amplia, cortando parte de la radiografía (Figura 2.6f) o se reduce, añadiendo píxeles en negro en los laterales (Figura 2.6g).

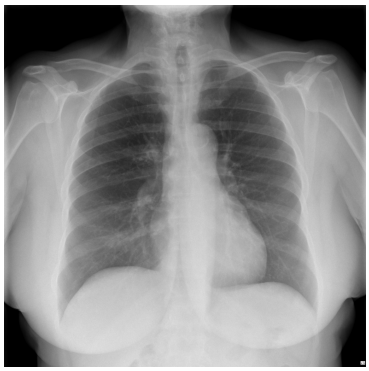
2.7. Paradigma estudiante-profesor

La **destilación de conocimiento** (*Knowledge Distillation* o KD) es una técnica que permite transferir el conocimiento de una red neuronal a otra completamente diferente. Una de las variantes más extendidas es el paradigma estudiante-profesor (*Student-Teacher* o S-T), que en general, funciona de la siguiente forma:

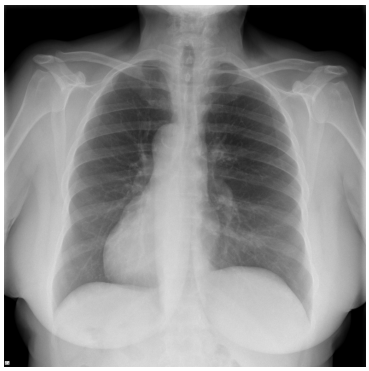
- Se tiene un modelo o varios en forma de *ensemble*, denominado profesor.
- El profesor precalcula y almacena las predicciones de todas las entradas que se utilizan durante el entrenamiento.
- El estudiante entrena a partir de la predicciones dadas por el profesor en lugar de las etiquetas reales de la entrada.

Hay diversas formas de trabajar con el paradigma S-T [13, 14]. A continuación se detallan aquellas relacionadas con tareas de clasificación y que son necesarias para la comprensión del trabajo.

- **Etiquetas débiles**: Durante el entrenamiento se aplica una función de pérdida de dos componentes, a las que se les puede asignar diferentes pesos. Una de las componentes

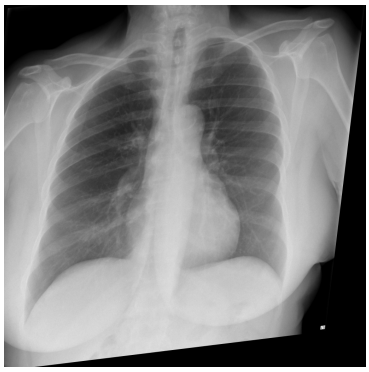


(a) Original.



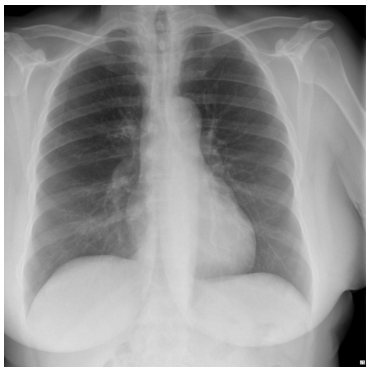
(b) Volteo horizontal.

(c) Translación.



(d) Shear.

(e) Rotación.



(f) Escalado: Ampliación.

(g) Escalado: Reducción.

Figura 2.6: Ejemplos de las diferentes técnicas de aumentos de datos sobre una radiografía de tórax.

aplica la función de pérdida adecuada el problema de clasificación en cuestión entre las predicciones del estudiante y la etiqueta real de una entrada, mientras que la otra componente calcula la divergencia de Kullback-Leibler entre las predicciones del estudiante y las del profesor para una misma entrada.

- **Etiquetas fuertes**: Durante el entrenamiento se aplica una función de pérdida de dos componentes, a las que se les puede asignar diferentes pesos, que aplican la función de pérdida adecuada al problema de clasificación en cuestión. Una componente calcula la función de pérdida entre la predicción del estudiante y la etiqueta real, mientras que la otra compara dicha predicción con la ofrecida por el profesor. La técnica de etiqueta fuerte convierte la predicción del profesor, una función de probabilidad que suma 1 entre las diferentes activaciones de la capa de clasificación, en una etiqueta ficticia mediante la función *argmax*. Por ejemplo, convierte la predicción {0,2, 0,2, 0,6} en {0, 0, 1}.
- **Entrenamiento no etiquetado**: Durante el entrenamiento se pueden utilizar datos no etiquetados, de forma que el modelo estudiante aprende las predicciones dadas por el modelo profesor a pesar de no tener etiquetas reales. De esta forma no se utiliza la componente de la etiqueta real de la función de pérdida de las dos metodologías descritas.
- **Entrenamiento con ruido**: Se pueden precalcular las salidas de la red neuronal profesor sobre los datos de entrada no aumentado para reducir el coste computacional. Al mismo tiempo, el modelo estudiante trabaja sobre entradas con aumento de datos, de forma que se le entrena a predecir igual que el profesor, que ha trabajado en datos limpios. En consecuencia, se produce una mejora de la regularización y precisión del modelo estudiante.

En este capítulo se han explicado los diferentes conceptos necesarios para la comprensión de este trabajo. Por un lado, entender el formato DICOM es fundamental en el procesamiento de la base de datos de ActualTec en el Capítulo 5. Por otro lado, diferentes conceptos de redes neuronales como la transferencia de aprendizaje, las técnicas de regularización o el paradigma S-T se utilizan a lo largo de la memoria para entrenar los diferentes modelos de análisis de radiografías. También se describe el funcionamiento de la operación de convolución, utilizada en los Capítulos 6 y 7. En última instancia, comprender el funcionamiento de las redes de tipo transformador aplicadas al lenguaje natural es necesario para entender el funcionamiento de los transformadores de visión descritos en el próximo capítulo.

Capítulo 3

Transformadores de visión

Las redes de tipo transformador se han adaptado a las tareas de visión por computador típicamente reservadas a las CNN, produciendo los **transformadores de visión** (*Vision Transformers* o VT). En este capítulo se describe el funcionamiento básico de los VT, seleccionando diversas arquitecturas en base a la precisión de la mismas. Para cada arquitectura, se detalla el preentrenamiento realizado, combinando técnicas de entrenamiento supervisado habitualmente utilizadas en el preentrenamiento de arquitecturas CNN y técnicas de entrenamiento auto supervisado empleadas en el preentrenamiento de arquitecturas de tipo transformador. Por otro lado, se describe también la normalización y clasificador recomendada, así como la resolución de entrada del modelo.

3.1. Arquitectura VT y criterios se selección

El gran éxito cosechado por los transformadores en el procesamiento de lenguaje natural, ha llevado a la aplicación de los mismos a diversas áreas para las que no estaban inicialmente diseñados. Una de estas áreas es la visión por computador, produciendo las arquitecturas VT. Se introducen por primera vez en 2020 en la publicación de la arquitectura ViT [15], momento a partir del cual se comienzan a investigar nuevas arquitecturas VT.

Estas arquitecturas se basan principalmente en dividir la imagen en parcelas (*patches*), de forma que cada una de ellas simboliza un *token*. Dichas parcelas son aplanadas y proyectadas, obteniendo *patch embeddings* de un determinado *hidden size*. De igual forma que en los transformadores, se añade un *token* de clasificación `[CLS]` y *embeddings* de posición, pero se eliminan los *embeddings* de segmento, tal y como se puede observar en la Figura 3.1. Los propios desarrolladores preentrenan y publican los diferentes modelos VT en conjuntos de datos como

ImageNet, para su uso en transferencia de aprendizaje y *fine-tuning*.

Se seleccionan diversas arquitecturas VT disponibles en el entorno Huggingface[1] (HF) para estudiar su funcionamiento y precisión en el análisis de radiografías. Se seleccionan diversas arquitecturas presentes en la taxonomía de la Figura 3.2. Dentro de cada arquitectura se estudian modelos con distintos tamaños y resoluciones de entrada. Dichas resoluciones son cuadradas, de forma que se abrevian a un único número. Por ejemplo, la resolución 512×512 se representa simplemente como 512.

En particular, se comparan los diferentes modelos de una misma arquitectura en base a los resultados mostrados por sus propias publicaciones. A partir de estos resultados se obtienen diversas conclusiones y se seleccionan los modelos con mayor potencial. De esta forma, se reduce el número de modelos a entrenar en la tarea de clasificación de imagen médica propuesta en este trabajo en el Capítulo 5.

3.2. ViT

La arquitectura ViT [15, 17] es la primera arquitectura VT publicada, y ha servido como inspiración para la creación y publicación de nuevas arquitecturas VT. El funcionamiento de ViT es análogo al explicado en la sección anterior. Los modelos publicados siguen la siguiente metodología preentrenamiento:

1. **Preentrenamiento inicial**: El modelo se preentrena en amplios conjuntos de datos como ImageNet-21k, JFT-300M y JFT-3B en resolución 224 de forma supervisada. Los conjuntos JFT son conjuntos privados de Google que contienen 300 M y 3000 M de imágenes respectivamente. Los modelos preentrenados en dichos conjuntos no se encuentran disponibles al ser privados.
2. ***Fine-tuning***: Se ajusta el modelo al conjunto ImageNet en resoluciones 224, 384 y 512.

Además de los diferentes conjuntos de datos de preentrenamiento y resolución de *fine-tuning*, un modelo ViT queda definido por dos hiperparámetros:

- **Tamaño del modelo**: Hay tres tamaños de ViT; *Base* (B), *Large* (L) y *Huge* (H), formados por 86 M, 307 M y 632 M parámetros y un *hidden size* de 768, 1024, 1280 respectivamente.

[1] https://huggingface.co/

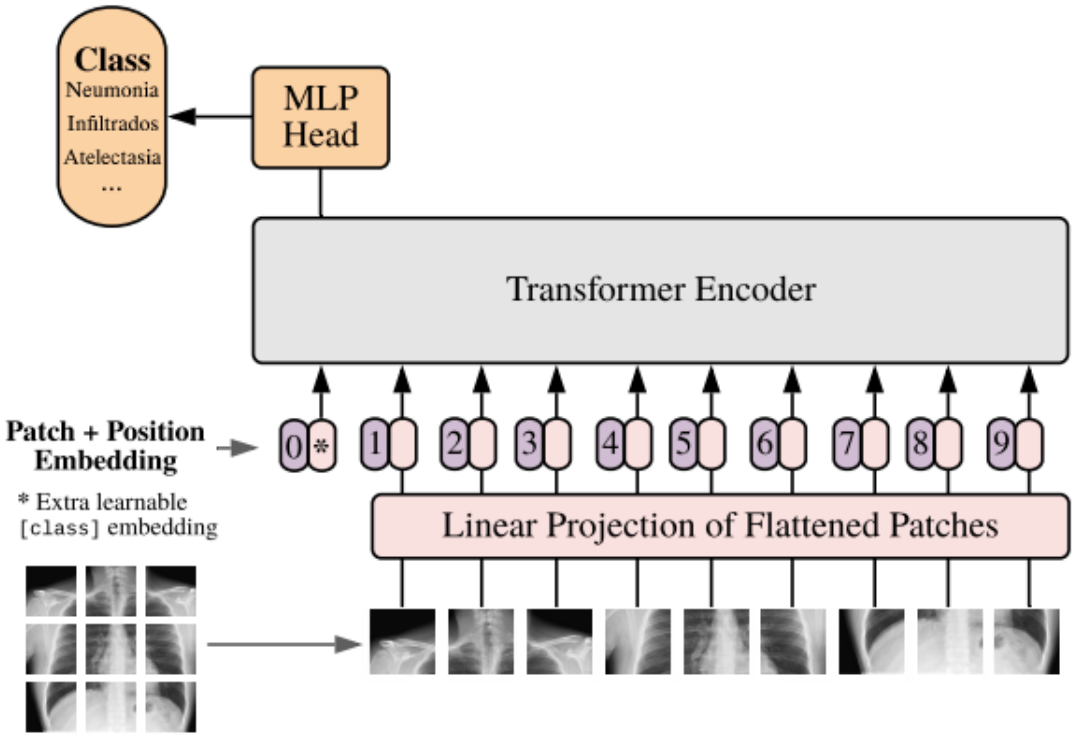


Figura 3.1: Esquema básico de funcionamiento de una arquitectura VT.

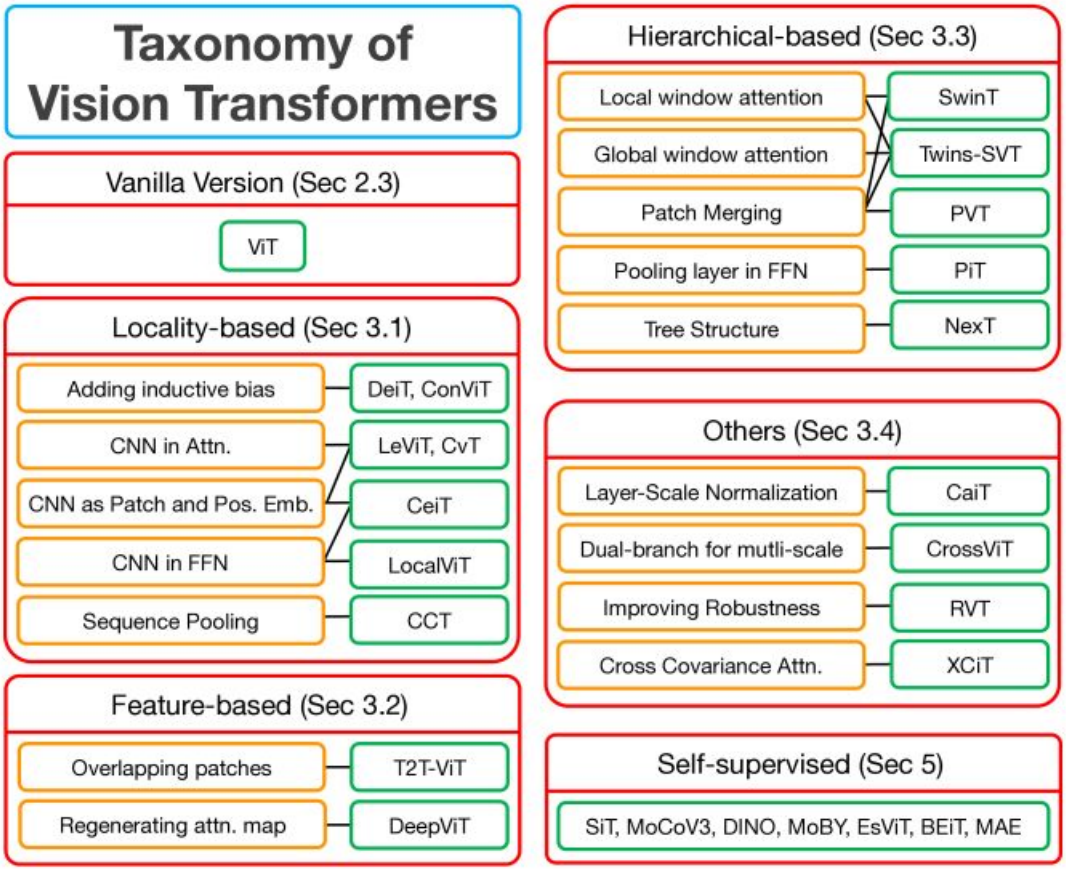


Figura 3.2: Taxonomía de los VT [16].

- **Tamaño de parcela**: Hay diferentes tamaños de parcela disponibles, que son referidas como un único número al ser cuadradas. En el caso de las configuraciones *Base* y *Large* se prueba con tamaños de 16 y 32, mientras que en el caso de *Huge* se utiliza únicamente un tamaño de parcela de 14.

3.2.1. Análisis y selección de modelos

De esta forma, un modelo ViT queda identificado como ViT-modelo/parcela. Los resultados de precisión de las diferentes combinaciones entre las configuraciones de preentrenamiento y modelo sobre ImageNet pueden observarse en la Tabla 3.1 y justifican las siguientes conclusiones:

- Los modelos con parcela 32 tienen una precisión significativamente inferior que los modelos con parcela 16, por lo que son descartados, siguiendo además la recomendación de los desarrolladores. De esta forma, se reducen el número de modelos a probar durante la búsqueda de hiperparámetros.
- Los modelos *Large* funcionan mejor que los modelos *Base* en los grandes conjuntos de datos, mientras que obtienen un peor rendimiento en ImageNet.
- El modelo *Huge* funciona mejor que *Large* con preentrenamiento en JFT-300M, pero obtiene un resultado similar en ImageNet-21k. El modelo *Huge* preentrenado en JFT-300M no se encuentra disponible en HF, por lo que su uso queda descartado.
- A mayor cantidad de datos de preentrenamiento, mayor precisión. Sin embargo, los modelos preentrenados en JFT no se encuentran disponibles en HF.
- El modelo ViT-L/16 con resolución 512 funciona mejor que con resolución 384, pero no se encuentra disponible en HF.
- Los resultados de los modelos con resolución 224 no han sido publicados a pesar de que los modelos sí se encuentran disponibles. Únicamente se destaca un peor rendimiento respecto a resolución 384.
- **Selección**: Se seleccionan los modelos **ViT-B/16** y **ViT-L/16** en resolución 384 para ser investigados en el análisis de radiografías, estableciendo así un *baseline* para las arquitecturas VT.

3.2.2. Normalización y clasificador

La arquitectura ViT requiere de normalizar las imágenes con una media de 0,5 y una desviación estándar de 0,5 en cada canal en lugar de la media y la desviación estándar de ImageNet.

Preentrenamiento	Modelo	Resolución	Disponible	Precisión
ImageNet	ViT-B/16	384	✗	77,90
	ViT-L/16	384	✗	76,50
ImageNet-21k	ViT-B/16	384	✓	**83,97**
	ViT-B/32	384	✓	81,28
	ViT-L/16	384	✓	**85,15**
	ViT-L/16	512	✗	85,30
	ViT-L/32	384	✓	80,99
	ViT-H/14	384	✗	85,13
JFT-300M	ViT-B/16	384	✗	84,15
	ViT-B/32	384	✗	80,73
	ViT-L/16	384	✗	87,12
	ViT-L/16	512	✗	87,76
	ViT-L/32	384	✗	84,37
	ViT-H/14	384	✗	88,04
	ViT-H/14	512	✗	88,55
JFT-3B	ViT-B/16	384	✗	86,60
	ViT-L/16	384	✗	88,50

Tabla 3.1: Resultados de precisión sobre ImageNet con diferentes ViT [15, 17, 18, 14].

En cuanto al clasificador recomendado, se parte del estado oculto del *token* `[CLS]` en la última capa del modelo, que queda representado por un vector de dimensión *hidden size*. El clasificador está formado por una única capa lineal con activación de tipo *sigmoid* con tantas salidas como etiquetas tenga la tarea deseada. En este trabajo se ha probado la aplicación de capas de normalización y una capa *dropout* con diferentes valores de p antes de dicha capa de clasificación.

3.3. BEiT

La arquitectura BEiT (*Bidirectional Encoder representation from Image Transformers*) [18] parte de los modelos *Base* y *Large* de ViT con parcelas de tamaño 16, y añade algunas modificaciones como *LayerScale* y *Relative Position Bias*. Modifica el preentrenamiento de la siguiente forma:

1. **Preentrenamiento inicial**: El modelo se preentrena de forma auto supervisada en conjuntos de datos como ImageNet, ImageNet-21k y ImageNet-70k, un conjunto privado, en resolución 224.
2. ***Fine-tuning* intermedio**: El modelo se preentrena de nuevo sobre ImageNet-21k en resolución 224, pero de forma supervisada.
3. ***Fine-tuning* final**: Se ajusta el modelo al conjunto ImageNet de forma supervisada en diferentes resoluciones como 224, 384 y 512.

3.3.1. Análisis y selección de modelos

Se denota BEiT para los modelos que siguen la metodología de preentrenamiento auto supervisada y mantienen exactamente la misma arquitectura que ViT, mientras que se denota BEiT$^+$ a aquellos que añaden *LayerScale* y *Relative Position Bias*. Dado que BEiT únicamente utiliza de modelos *Base* y *Large* con tamaño de parcela 16, no es necesario denotar dicha característica como si es necesario con ViT

Los resultados de precisión sobre ImageNet de ambos modelos sobre diferentes resoluciones en el *fine-tuning* final se detallan en la Tabla 3.2. A través de ellos se obtienen las siguientes conclusiones:

- Los modelos *Large* obtienen una mejor precisión tanto en ImageNet como ImageNet-21k, demostrando que no requieren de un gran volumen de datos de entrenamiento.

Preentrenamiento	Modelo	Resolución	Disponible	Precisión
ImageNet	BEiT-B	224	✗	83,20
	BEiT-B	384	✗	84,60
	BEiT-L	224	✗	85,20
	BEiT-L	384	✗	86,30
ImageNet-21k	BEiT-B$^+$	224	✓	**85,20**
	BEiT-B$^+$	384	✓	**86,80**
	BEiT-L$^+$	384	✓	**88,40**
	BEiT-L$^+$	512	✓	**88,60**
ImageNet-70k	BEiT-L$^+$	384	✗	89,30
	BEiT-L$^+$	512	✗	89,50

Tabla 3.2: Resultados de precisión sobre ImageNet con diferentes BEiT [18].

- Los modelos funcionan mejor a mayor resolución.

- **Selección**: Los cuatro modelos BEiT disponibles en HF; **BEiT$_{224}$-B$^+$**, **BEiT$_{384}$-B$^+$**, **BEiT$_{384}$-L$^+$** y **BEiT$_{512}$-L$^+$**, tienen una gran precisión, por lo que son seleccionados para el trabajo. Para simplificar la notación se omite el uso de $^+$ en el resto de la memoria dado que solo se usan modelos de este tipo.

3.3.2. Normalización y clasificador

El modelo BEiT normaliza las imágenes utilizando una media de 0,5 y una desviación estándar de 0,5 en cada canal, en lugar de utilizar la media y la desviación estándar de ImageNet, de la misma forma que ViT.

En cuanto al clasificador, se calcula la media del estado oculto de todos los *tokens*, excepto el *token* `[CLS]`, en la última capa del modelo. De esta forma, se tiene un vector de dimensionalidad *hidden size*; 768 para el modelo *Base* y 1024 para el modelo *Large*. El clasificador queda formado por una única capa lineal con activación de tipo *sigmoid* con tantas salidas como etiquetas tenga la tarea deseada. Durante el trabajo se prueba la aplicación de capas de normalización y una capa *dropout* con diferentes valores de p antes de dicha capa de clasificación.

3.4. DeiT

La arquitectura DeiT (*Data Efficient image Transformers*) [14] parte de la arquitectura ViT y añade regularizaciones, aumentos de datos y diferentes técnicas que le permiten obtener buenos resultados con una menor cantidad de datos de entrenamiento. En particular, ce centra en mejorar el entrenamiento del modelo ViT-B sobre ImageNet sin datos externos como ImageNet-21k o JFT-300M. También ofrecen dos nuevos modelos, *Tiny* (Ti) y *Small* (S), que reducen la cantidad de parámetros a 5 M y 22 M, respectivamente.

El trabajo propone un nuevo tipo de paradigma S-T diferente de las etiquetas fuertes y débiles explicadas en la Sección 2.7 . Se añade un *token* especial de destilación además del *token* de clasificación habitual. Sobre ambos *tokens* se añade un clasificador, calculando así dos predicciones diferentes. Sobre la predicción obtenida mediante el clasificador del *token* destilado se calcula la función de pérdida BCE con las predicciones obtenidas por el profesor, mientras que las predicciones del clasificador del *token* de clase se calcula dicha función de pérdida con las etiquetas reales de la entrada. Cada función de pérdida se utiliza para ajustar el modelo. Para determinar la predicción del modelo completo durante el entrenamiento y sobretodo, inferencia, se calcula la media de las predicciones de ambos clasificadores. Los modelos entrenados con esta metodología se denotan mediante el símbolo del alambique ⚗ y requieren parámetros adicionales. De esta forma, DeiT-T⚗ y DeiT-B⚗ contienen 6 M y 87 M de parámetros, respectivamente. El número de parámetros de DeiT-S⚗ también se ve incrementado, pero sin llegar a los 23 M.

El entrenamiento de los diferentes modelos, tanto para el entrenamiento tradicional como para el destilado (⚗), requiere dos pasos:

1. **Preentrenamiento**: Se preentrena el modelo sobre ImageNet en resolución 224 de forma supervisada utilizando diferentes técnicas de aumento de datos y regularización no aplicadas en el entrenamiento del ViT original.
2. **Fine-tuning**: Se ajusta el modelo sobre ImageNet en resolución 224 y 384 utilizando las mismas técnicas de aumento de datos y regularización.

3.4.1. Análisis y selección de modelos

Los resultados de precisión sobre ImageNet de los diferentes modelos DeiT, se observan en la Tabla 3.3, a partir de los cuales se obtienen las siguientes conclusiones:

- Los modelos destilados obtienen mejores resultados de precisión en los diferentes tamaños y resoluciones utilizados.

Modelo	Resolución	Disponible	Precisión
DeiT-Ti	224	✓	72,20
DeiT-S	224	✓	79,80
DeiT-B	224	✓	**81,80**
DeiT-B	384	✓	**83,10**
DeiT-Ti⚗	224	✓	76,60
DeiT-S⚗	224	✓	82,60
DeiT-B⚗	224	✓	**84,20**
DeiT-B⚗	384	✓	**85,20**

Tabla 3.3: Resultados de precisión sobre ImageNet con diferentes DeiT [14].

- La precisión del modelo *Base* mejora el aumentar la resolución a 384.

- **Selección**: Se seleccionan los modelos **DeiT-B⚗** en resoluciones 224 y 384, debido al potencial del preentrenamiento no etiquetado dado por el paradigma S-T mostrado en la Sección 2.7. Se seleccionan además los modelos **DeiT-B** en resoluciones 224 y 384 con el objetivo de establecer una comparativa entre entrenamiento tradicional y S-T.

3.4.2. Normalización y clasificador

Las arquitecturas DeiT normalizan las imágenes con una media de 0,5 y una desviación estándar de 0,5 por cada canal, de la misma forma que la arquitectura ViT. Por otro lado, las arquitecturas DeiT⚗ normalizan según la media y desviación estándar de ImageNet, manteniendo así la misma normalización que la CNN profesor.

Para los diferentes modelos el clasificador parte del estado oculto del *token* `[CLS]` en la última capa. De esta forma se tiene un vector de dimensionalidad *hidden size*; 192 para el modelo *Tiny*, 384 para el modelo *Small* y 768 para el modelo *Base*. El clasificador queda formado por una única capa lineal con activación de tipo *sigmoid* con tantas salidas como etiquetas tenga la tarea deseada. Durante el trabajo se experimentan con el uso de de capas de normalización y *dropout* con diferentes valores de p antes de dicha capa de clasificación.

En el caso de los modelos DeiT⚗ se aplica el mismo clasificador sobre el *token* de destilación, y se toma la media de ambos clasificadores para formar la predicción final del modelo.

3.5. SwinT

La arquitectura SwinT [19, 20] es un tipo de VT jerárquico, que reduce el tamaño de la parcela a 4 y aplica atención por ventanas cuadradas en lugar de global, reduciendo el coste computacional. Además, aplica unión de parcelas dentro del modelo, de forma que el tamaño de la parcela va en aumento dentro del mismo. Estos cambios le permiten procesar imágenes con mayor resolución, así como realizar tareas de segmentación de imágenes, dónde se debe determinar la clase a la que pertenece cada píxel.

Esta memoria se centra en los modelos SwinTV2 [20] en sus cuatro variantes; *Tiny*, *Small*, *Base* y *Large*, formados por 28 M, 50 M, 88 M y 197 M de parámetros, respectivamente. Los diferentes modelos siguen dos metodologías de entrenamiento:

- **Preentrenamiento ImageNet**: Los modelos *Tiny*, *Small* y *Base* se preentrenan usando ImageNet en resolución 256 y con un tamaño de ventana de 16. No se aplica una segunda fase de *fine-tuning*.
- **Preentrenamiento ImageNet-21k**: Los modelos *Base* y *Large* se preentrenan con ImageNet-21k en resolución 192 y un tamaño de ventana de 12. Posteriormente, se aplica *fine-tuning* en resolución 256 con tamaño de ventana 16 y resolución 384 con tamaño de ventana 24.

3.5.1. Análisis y selección de modelos

Los resultados de precisión sobre ImageNet de los diferentes modelos SwinTV2 se detallan en la Tabla 3.4, a partir de los cuales se obtienen las siguientes conclusiones:

- Los modelos con mayor número de parámetros obtienen mejores resultados sin importar el preentrenamiento.
- El mismo modelo obtiene mejores resultados en sus versiones a mayor resolución de entrada.
- **Selección**: Se seleccionan los 4 modelos preentrenados en ImageNet-21k debido a su elevada precisión y potencial.

Preentrenamiento	Modelo	Resolución	Tamaño de ventana	Disponible	Precisión
ImageNet	SwinTV2-T	256	16	✓	82,80
	SwinTV2-S	256	16	✓	84,10
	SwinTV2-B	256	16	✓	85,60
ImageNet-21k	SwinTV2-B	256	16	✓	**86,20**
	SwinTV2-B	384	24	✓	**87,10**
	SwinTV2-L	256	16	✓	**86,90**
	SwinTV2-L	384	24	✓	**87,60**

Tabla 3.4: Resultados de precisión sobre ImageNet de diferentes modelos SwinTV2 [20]

3.5.2. Normalización y clasificador

Las imágenes se normalización utilizando la media y desviación estándar de ImageNet. En cuanto al clasificador, se calcula la media del estado oculto de todos los *tokens*, obteniendo un vector de dimensionalidad 768 para los modelos *Tiny* y *Small*, 1024 para el modelo *Base* y 1536 para el modelo *Large*. El clasificador queda formado por una única capa lineal con activación de tipo *sigmoid* con tantas salidas como etiquetas tenga la tarea deseada. Durante el trabajo se prueba la aplicación de capas de normalización y una capa *dropout* con diferentes valores de p antes de dicha capa de clasificación.

3.6. Comparativa de VT

Como conclusión a este capítulo, se compara la precisión de los diferentes modelos VT seleccionados durante el mismo, tal y como se observa en la Tabla 3.5. De esta forma, se establece el potencial de los diferentes modelos en su aplicación en el análisis de radiografías.

Se observa que los modelos SwinTV2 se sitúan por debajo de los modelos BEiT en cuanto a precisión, mientras que consiguen mejor precisión que ViT, DeiT y DeiT⚗. Por otro lado, las arquitecturas ViT y DeiT⚗ tienen resultados de precisión similares, pero se espera que el paradigma S-T y el uso del entrenamiento no etiquetado sobre DeiT⚗, produzca una mejoría de precisión en el análisis de radiografías. Por último, los modelos DeiT tienen la menor precisión de entre todos los modelos. Sin embargo, el modelo DeiT-B en resolución 384 no se encuentra muy alejado de su análogo ViT-B/16 en la misma resolución, sobretodo teniendo en cuenta la menor cantidad de datos de entrenamiento utilizados.

Modelo	Datos de preentrenamiento	Número de parámetros	Resolución	Precisión
ViT-B/16	14 M	86 M	384	83,97
ViT-L/16	14 M	307 M	384	85,15
DeiT-B	1,2 M	86 M	224	81,80
DeiT-B	1,2 M	86 M	384	83,10
DeiT-B⚗	1,2 M	87 M	224	84,20
DeiT-B⚗	1,2 M	87 M	384	85,20
BEiT-B	14 M	86 M	224	85,20
BEiT-B	14 M	86 M	384	86,80
BEiT-L	14 M	307 M	384	88,40
BEiT-L	14 M	307 M	512	**88,60**
SwinTV2-B	14 M	88 M	256	86,20
SwinTV2-B	14 M	88 M	384	87,10
SwinTV2-L	14 M	197 M	256	86,90
SwinTV2-L	14 M	197 M	384	87,60

Tabla 3.5: Resultados de precisión sobre ImageNet de los modelos ViT, BEiT, DeiT y SwinTV2 seleccionados.

Capítulo 4

Estado de la cuestión

La sobresaturación de los servicios radiológicos, así como la necesidad de optimizar los procesos de triaje, no solo se produce a nivel nacional, sino también internacional. De este modo, la aplicación de aprendizaje automático a la imagen médica es una cuestión recurrente. Este capítulo se centra en explicar los avances en esta cuestión desde 2017 hasta la actualidad, con la llegada de las arquitecturas VT.

4.1. Bases de datos

Antes de examinar las diferentes técnicas aplicadas en el análisis de radiografías mediante redes neuronales, es necesario definir las diferentes bases de datos sobre las que se entrenan dichas redes. En esta sección se describen diferentes bases de datos, cuyas características principales pueden observarse en la Tabla 4.1. Estas bases de datos son las siguientes:

- **ChestX-ray14** [6] es una base de datos desarrollada por el NIH (*National Institutes of Health*). Se trata de la primera gran base de datos de radiografías de tórax, convirtiéndose así en uno de los conjuntos de datos más utilizados en la comparativa del estado de la cuestión. Las diferentes radiografías son asignadas en 14 etiquetas patológicas: Atelectasia, Cardiomegalia, Derrame, Infiltrado, Masa, Nódulo, Neumonía, Neumotórax, Consolidación, Edema, Enfisema, Fibrosis, Engrosamiento pleural y Hernia. Además, incluye una etiqueta para diferenciar los pacientes sanos.

- **PadChest** [5] es una base de datos desarrollada por la Universidad de Alicante, la fundación FISABIO y el Hospital San Juan. Sus principales puntos de ventaja respecto al

	ChestX-Ray14	CheXpert	MIMIC-CXR	PadChest	BIMCV
Imágenes	112 120	224 316	371 920	160 868	23 527
Pacientes	30 805	65 240	65 383	67 625	✗
Imágenes por paciente	3,3	3,4	5,6	2,3	✗
Etiquetas	14	14	14	193	✗
Incertidumbre	No	Si	Si	No	Si
Año publicación	2017	2019	2019	2019	2020

Tabla 4.1: Estadísticas de diferentes bases de datos de radiografías de tórax.

resto de bases de datos son la baja media de imágenes por paciente, y el elevado número de etiquetas patológicas disponibles, establecidas en una jerarquía.

- **CheXpert** [21] es una base de datos desarrollada por Stanford en colaboración con el Hospital Stanford. Su principal ventaja, además de un elevado número de imágenes, es la creación de un etiquetador de informes automático que puede detectar la incertidumbre, lo que permite determinar si el radiólogo no estaba seguro de si un paciente padece cierta enfermedad. Además, utiliza una estructura de etiquetas jerárquica, observable en la Figura 4.1. es una base de datos desarrollada por el MIT, en colaboración con Stanford, Harvard y el Beth Israel Deaconess Medical Center. Tomando el etiquetador de CheXpert para obtener etiquetas de incertidumbre, sus principales puntos de ventaja son la elevada cantidad de imágenes, y la presencia de *bounding boxes* de las diferentes patologías en algunas de las imágenes.
- **BIMCV COVID-19+** es una base de datos desarrollada por los investigadores de la base de datos de PadChest en colaboración con la universidad Miguel Hernández y el Centro de Investigación Príncipe Felipe, cuya ventaja principal es la presencia de radiografías con COVID-19.

4.2. Arquitecturas CNN en el análisis de radiografías

Se describe el estado de la cuestión desde 2017, con la publicación de la base de datos ChestX-ray14, sobre las que se publican diferentes herramientas basadas en arquitecturas CNN:

- El trabajo realizado por Wang et al. [6], además de publicar el conjunto de datos por primera vez, prueba diversos modelos de CNNs preentrenados en ImageNet, como AlexNet, GoogleLeNet, VGG-16 y ResNet-50, obteniendo los mejores resultados con esta última. El clasificador se encuentra formado por una capa de pooling Log-Sum-Exp (LSE) y una de la capa de clasificación. Estos modelos trabajan con las imágenes en resolución 1024.

Trabajo	AUC Medio
Wang et al.	0,738
Yao et al.	0,803
CheXNet	**0,841**
Gündel et al.	0,807

Tabla 4.2: AUC medio de diferentes trabajos sobre ChestX-ray14.

- El trabajo realizado por Yao et al. [22], utiliza bloques DenseNet en una resolución menor de 512 y sin preentrenamiento. Como clasificador utiliza una LSTM (*Long Short-Term Memory*), un tipo de red neuronal recurrente, buscando explotar la relación entre las etiquetas. Además, introduce diferentes aumentos de datos aleatorios, como translación, rotación o escalado.
- El trabajo realizado por CheXNet [23] prueba una DenseNet-121 preentrenada en ImageNet, sobre imágenes en resolución 224 y normalizadas por la media y desviación de ImageNet. Como aumento de datos aplican únicamente volteo horizontal aleatorio.
- El trabajo realizado por Gündel et al. [24] utiliza una DenseNet-121 preeentrenada en ImageNet, y normaliza las imágenes por la media y desviación típica de ImageNet, de la misma forma que CheXNet. Sin embargo, utiliza las imágenes en resolución 1024, que son procesadas por dos capas convolucionales con *stride* 2, reduciendo la resolución a 256, antes de ser procesadas por la DenseNet121.

En esta revisión se comparan las diferentes herramientas mediante la métrica AUC, dado que se trata de un problema multi-etiqueta, a través de las publicaciones de las propias herramientas. Este AUC global se calcula a través de la media no ponderada del AUC individual de cada etiqueta. Sobre esta métrica, el trabajo realizado por CheXNet obtiene los mejores resultados, tal y como se puede observar en la Tabla 4.2.

En 2019 se publican CheXpert, MIMIC-CXR y PadChest, que sustituyen a ChestX-ray14 como comparativa del estado de la cuestión, sobre las que se desarrollan también diferentes herramientas:

- Junto con la publicación del conjunto de datos PadChest [5] se desarrolla una red neuronal basada en la arquitectura ResNet-18 sobre imágenes en resolución 1500, únicamente la tarea binaria de la enfermedad efusión.
- Junto con publicación del conjunto de datos CheXpert [21], se prueban diversas arquitecturas de CNN con resolución 320, como ResNet-152, Inception-v4 y DenseNet-121, obteniéndose los mejores resultados con esta última.

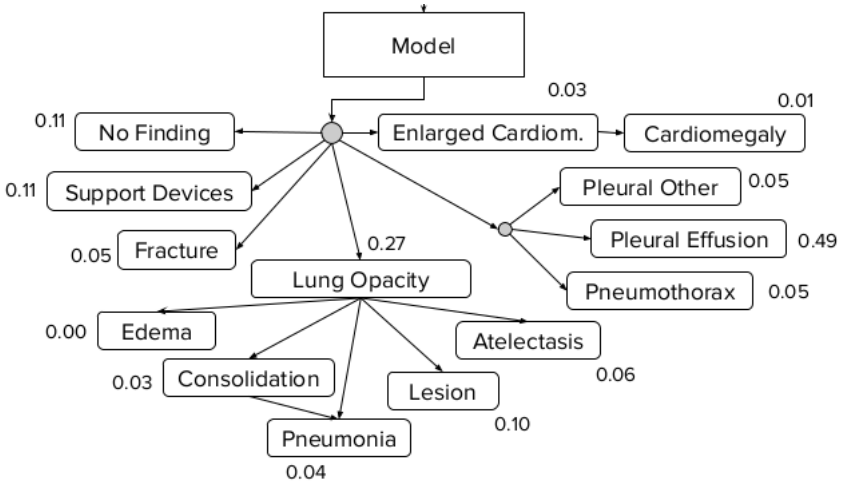


Figura 4.1: Jerarquía de etiquetas en Chexpert y MIMIC-CXR [21].

- El primer puesto de la tabla de lideres de CheXpert, con un AUC de 0,93, lo ocupa DeepAUC-v1 [25]. Se utiliza un *ensemble* de 5 modelos de CNN preentrenados en ImageNet; DenseNet121, DenseNet169, DenseNet201, DenseNet161 e Inception-ResNet-v2. Utiliza las imágenes en resolución 320, normalizadas por la media y desviación de ImageNet, y aplican rotación, translación y escalado como aumentos de datos.
- El segundo puesto de la tabla de líderes de Chexpert, también con un 0,93 de AUC, lo ocupa Hierarchical-Learning-V1 [26]. Se utiliza un *ensemble* de 6 modelos de CNNs tradicionales preentrenados: DenseNet-121, DenseNet-169, DenseNet-201, Inception-ResNet-v2, Xception y NASNetLarge, trabajando a resolución 224 y con normalización de ImageNet. En estos, se aplican diversos aumentos de datos como volteo horizontal aleatorio, rotación, *shearing* y escalado. Su mayor punto de diferencia con DeepAUC es un entrenamiento que tiene en cuenta la jerarquía establecida entre las diversas etiquetas, mostrada en la Figura 4.1, al establecer la predicción final para una radiografía.

Por otro lado, en 2020 se publica la base de datos BIMCV COVID-19+ [27], la base de datos de imagen médica de COVID-19 más amplia en el momento de su publicación. En plena pandemia, y con el objetivo de agilizar la detección de COVID-19 mediante la posible sustitución de la prueba PCR por la radiografía simple, se desarrollan diversas herramientas de visión por computador basadas en redes CNN:

- El trabajo realizado por Arias-Garzón et al. [28] propone el uso de las arquitecturas VGG-16 y VGG-19 preentrenadas sobre unas radiografías que han sido preprocesadas, mediante redes de segmentación como U-Net, para utilizar únicamente información relativa a los pulmones, eliminando el bias en el conjunto de datos. Se utilizan imágenes en resolución

224 y normalizadas por la media y normalización del propio conjunto de datos. El trabajo se centra en la tarea de clasificación binaria para la COVID-19 obteniendo una precisión de detección del 97 %.

- El trabajo realizado por Nishio et al. [29] propone el uso de la arquitectura EfficientNet-B5, basada en el paradigma S-T. El trabajo plantea una tarea de clasificación más avanzada, detectando neumonía producida por COVID-19, neumonía no producida por COVID-19, y pacientes sanos, obteniendo una precisión del 86 %. Se utilizan las bases de datos BIMCV COVID-19+ y PadChest, en conjunto con una base de datos privada.

Para acabar esta sección, cabe destacar que el foco de investigación del análisis de radiografías está cambiando de la tarea de clasificación a la redacción automática de informes mediante técnicas de *image captioning*. En esta línea se han publicado diversos estudios [30, 31] que utilizan una red CNN como codificador y un decodificador de lenguaje. La revisión detallada de estos trabajos, no obstante, queda fuera del ámbito de este trabajo.

4.3. Arquitecturas VT

Las altas prestaciones ofrecidas por las arquitecturas VT, las han convertido en un punto de referencia en diferentes ámbitos de la visión por computador. Una de estas áreas es la imagen médica, sobre las se han publicado diversos trabajos:

- El trabajo realizado en COVID-Transformer [32] se basa en un modelo ViT [15] preentrenado, al que añade un clasificador. Utiliza una resolución 224 y un aumento de datos que aplica translación, rotación y volteo horizontal. Plantea una tarea de clasificación para diferenciar COVID-19, neumonía no producida por COVID y pacientes sanos sobre diferentes bases de datos. Este trabajo obtiene un AUC de 0,98 y mejorando los resultados obtenidos en redes CNN tradicionales como DenseNet-121, Xception o ResNet-v2, entre otras.
- El trabajo realizado por Park et al. [33] utiliza una DenseNet-121 para extraer las características de las imágenes en resolución 512, y que luego son introducidas en el modelo ViT. Plantea una tarea de clasificación para diferenciar pacientes con COVID-19, sanos y con otras patologías, obteniendo un AUC de 0,94, 0,90 y 0,91 en 3 conjuntos externos de test. Para el entrenamiento se utilizan conjuntos de datos públicos como BIMCV COVID-19+ para entrenamiento.
- El trabajo realizado por Matsoukas et al. [34] compara la precisión de los modelo ResNet-50 y DeiT-S sobre diferentes tareas de clasificación de imagen médica, obteniendo resultados ligeramente superiores con el modelo DeiT-S. También muestran un bajo rendimiento del modelo DeiT-S si no se usan pesos preentrenados de ImageNet.

- El trabajo realizado por Shamshad et al. [35] recopila diferentes estudios de uso de arquitecturas VT en imagen médica, destacando una tendencia al alza en el uso de las arquitecturas VT en dicho tipo de imágenes en diferentes áreas como segmentación, clasificación, síntesis, restauración o generación de borradores de informes. En concreto, se tienen únicamente 5 estudios centrados en la clasificación de radiografías simples, basados en arquitecturas SwinT y ViT. En este estudio se observa una mayor cantidad de estudios en otras áreas, de mayor complejidad, y dónde el uso de arquitecturas de transformadores pueden resultar más beneficiosas.

Durante esta revisión del estado de la cuestión se han analizado las prácticas más habituales en el análisis de radiografías, observando una elevada tendencia al uso de modelos DenseNet preentrenados sobre el conjunto ImageNet. Se utilizan resoluciones bajas de entre 224 y 512, y se aplican técnicas de aumento de datos como volteo horizontal, translación, rotación o escalado. También se ha observado la tendencia al alza en el uso de arquitecturas VT en la imagen médica, aunque sin demasiado énfasis en la clasificación de radiografías simples, dónde se han empleado pocas arquitecturas. El objetivo de este trabajo es analizar diferentes arquitecturas e hiperparámetros sobre el problema de clasificación explicado en el próximo capítulo.

Capítulo 5

Procesamiento de datos

En este capítulo se describen las diferentes bases de datos utilizadas durante este trabajo, y el procesamiento necesario realizado en cada una de ellas. En particular, se muestran las diferentes agrupaciones realizadas, así como la metodología de particionado en entrenamiento, validación y test utilizada.

5.1. Procesamiento del conjunto de datos PadChest

La base de datos de PadChest [5] contiene 160 868 radiografías de tórax, pertenecientes a 67 625 pacientes, en proyecciones AP (vertical), AP horizontal, PA, Lateral y Costal, junto con los informes radiológicos, y etiquetados en 174 hallazgos radiológicos y 19 diagnósticos diferenciales. Este etiquetado ha sido realizado mediante una herramienta automática, partiendo de un 27 % de de la base de datos etiquetada por radiólogos. Se incluyen además 4 etiquetas especiales:

- La etiqueta *Normal* es una etiqueta especial que indica que el paciente no padece ninguna enfermedad. Se observan varios casos con errores, donde para un mismo estudio se han activado la etiqueta *Normal* y al menos una etiqueta patológica. Dichos estudios son revisados, concluyendo que son patológicos.
- La etiqueta *Unchanged* es una etiqueta radiológica que determina que el paciente no sufre cambios para una patología en evolución. En algunos casos, no se nombra dicha patología en el informe, siendo *Unchanged* la única etiqueta que se puede ser extraída. Esto impide su uso durante el entrenamiento.

Proyección	Número de muestras
PA	91 255
AP	4 509
AP horizontal	14 279
Costal	621
Lateral	49 298
Desconocida	8
Excluir	11

Tabla 5.1: Número de radiografías en las diferentes proyecciones.

- La etiqueta *Exclude* es una etiqueta especial que indica que, por diferentes motivos, el estudio no debe ser analizado o utilizado.

- La etiqueta *Suboptimal Study* es una etiqueta especial que indica que la radiografía no ha sido tomada de manera adecuada, dificultando su análisis.

Por otro lado, cabe destacar que los diagnósticos no siempre se encuentran presentes en el etiquetado de un estudio patológico. Esto ocurre cuando se tienen hallazgos patológicos cuyo diagnóstico es desconocido o solo puede determinarse a partir de otros síntomas del paciente. Esto lo ilustra PadChest con un ejemplo:

> *For example, an alveolar pattern is a radiographic finding that would prompt a very long list of differential diagnoses (including both infectius and non-infectious diseases such as lung edema, respiratory distress, etc.), but wheneverr it is present in a patient with fever, cough, leukocytosis, high CRP levels and crackles in the localization of the alteration, then it will almost certainly be pneumonia*

5.1.1. Estadísticas de la base de datos

A pesar de que la publicación respectiva a PadChest indica que se tienen 160 868 imágenes, se tienen en realidad 160 861 una vez descargada. Además, 880 estudios deben ser eliminados al no contener ninguna etiqueta, dejando el total en 159 981 estudios. En la Tabla 5.1 se observa que la proyección PA es la más documentada, seguida de la proyección Lateral, mientras que las proyecciones AP, AP horizontal y sobretodo costal, están menos representadas. También se tienen algunos estudios en los que la proyección es desconocida o está marcada para exclusión.

5.1.2. Procesamiento de las imágenes de PadChest

Antes de entrenar las redes neuronales, se realiza el siguiente procesamiento de imágenes:

1. **Cambio de resolución**: Las radiografías de la base de datos han sido tomadas en distintos años y máquinas, dando lugar a resoluciones muy diferentes comprendidas entre 2000 y 5000 píxeles. Para estandarizar la base de datos y reducir los tiempos de carga, se redimensionan todas las imágenes a 1024 píxeles.

2. **Reducción de la profundidad de bit**: Las radiografías de la base de datos se encuentran originalmente en escala de grises con 16 bits de profundidad. Teniendo en cuenta que cada píxel se traslada al rango $[0, 1]$ durante el entrenamiento, no se espera una diferencia de precisión significativa usando 8 bits de profundidad, reduciendo el coste en almacenamiento y tiempos de carga.

3. **Detección de imágenes corruptas**: Durante los 2 pasos anteriores se han detectado un total de 39 imágenes corruptas que no han podido ser procesadas y no pueden ser utilizadas durante el entrenamiento. De esta forma, se reduce el conjunto de datos a 159 942 radiografías.

5.2. Agrupación de etiquetas y criterios de exclusión

Se muestran a continuación las agrupaciones de PadChest utilizadas en la memoria. En primer lugar se analiza la agrupación por zonas anatómicas utilizada en el proyecto RADIANT, y sobre la que se realizan las primeras comparativas entre VT y CNN. En segundo lugar, se analiza la agrupación por zonas anatómicas y nivel de urgencia utilizada en el proyecto RELIANCE. Por último, se muestra la ampliación realizada mediante ChestX-ray14, que se emplea en la búsqueda de hiperparámetros y en la obtención de los resultados finales del trabajo.

5.2.1. Agrupación en zonas anatómicas

Debido al elevado número de etiquetas con una baja presencia, se realiza una agrupación de las 193 etiquetas en 7 zonas anatómicas, además de una etiqueta general, Patológica, que agrupa todas. Dado que un paciente puede sufrir diversas enfermedades en comorbilidad en diferentes zonas, se tiene un problema multietiqueta con 8 etiquetas. Para detectar los pacientes sanos la red neuronal debe predecir una probabilidad del 0 % en todas las neuronas de la capa de clasificación. Esta ausencia de activación en todas las etiquetas indica que el paciente no

Zona anatómica	Número de muestras
Patológica	53 240
Pulmón	32 403
Calcificación	4 861
Cuerpos extraños	6 432
Mediastino hila	17 814
Pleura	9 240
Diafragma	2 650
Pared torácica	13 135

Tabla 5.2: Número de instancias por zona anatómica sobre PadChest.

padece ninguna patología. Esta agrupación contiene un total de 87 604 imágenes, después de aplicar los siguientes criterios de exclusión:

- 24 de las 193 etiquetas de PadChest no han podido ser asignadas a ninguno de los grupos, resultando en la pérdida de 1690 estudios que se encontraban únicamente etiquetados por algunas de estas 24 etiquetas.
- Se trabaja únicamente sobre la proyección PA.
- Se eliminan aquellos estudios etiquetados como *Exclude* o únicamente como *Unchanged*.

En Tabla 5.2 se observan las diferentes zonas anatómicas definidas, junto con el número de instancias disponibles de cada una de ellas. En cuanto al número de estudios sanos, no mostrados en dicha tabla al no definirse como etiqueta, se tienen un total de 36 098.

5.2.2. Agrupación en zonas anatómicas urgentes

Previo a marzo de 2022, en el proyecto RELIANCE se trabaja con la agrupación anterior mediante una prueba de concepto, obteniendo buenos resultados de 0,861 y 0,855 de AUC sobre las particiones de test y validación respectivamente. Con el objetivo de desarrollar y validar la herramienta en un entorno de urgencias real, se desarrolla una nueva agrupación con las siguientes características y criterios de exclusión:

- Se amplía la agrupación para trabajar con la proyección AP y AP horizontal, excluyendo únicamente las proyecciones costal y lateral.
- Se añade una etiqueta de urgencia en las diferentes zonas patológicas.

- Se añaden etiquetas concretas para diversas patologías, como derrame pleural o masas, consideradas importantes en el entorno de urgencias.
- Se juntan las zonas Pleura, Diafragma y Pared torácica en una única zona patológica con el objetivo de suplir la baja presencia de enfermedades con pronóstico urgente en las zonas de diafragma y pared torácica. Dicha zona se denota como PDPT para el resto de la memoria.
- Se añade una etiqueta para detectar estudios subóptimos, de manera que dichas radiografías puedan ser repetidas o analizadas con mayor cuidado y para que se tenga en cuenta en la valoración de la red neuronal para el resto de etiquetas en dicha radiografía.
- Se añade una etiqueta para detectar estudios sanos. De esta forma se puede utilizar la ausencia de activación de todas las etiquetas para detectar radiografías fuera del dominio introducidas a la red neuronal de forma errónea, como se explica en [36].
- Se define la agrupación como una jerarquía, de forma similar al trabajo realizado por CheXpert [21]. Dicha jerarquía, definida como parcial y no exclusiva, se observa en la Figura 5.1. De esta forma, se pueden tener etiquetas asignadas a un grupo padre sin pertenecer a ninguno de los hijos. Es importante remarcar que las etiquetas para pacientes sanos y estudios subóptimos quedan fuera de la jerarquía.
- 11 etiquetas de las 193 no han podido ser asignadas a ninguno de los grupos, resultando en la pérdida de 114 estudios que se encontraban únicamente etiquetados por algunas de estas 11 etiquetas.
- Se excluyen los estudios etiquetados como *Exclude*, únicamente etiquetados como *Unchanged* y aquellos dónde el paciente sea menor de 16 años.

Los criterios de exclusión dejan el total de imágenes disponibles en 101 598 clasificadas en un total de 31 etiquetas. Resaltado en rojo, en la Tabla 5.3, se advierte una deficiencia de estudios en un grupo de etiquetas. A excepción de hernia de hiato, el resto de deficiencias se deben subsanar al tratarse de enfermedades de pronóstico urgente.

5.2.3. Ampliación con ChestX-Ray14

Dado que ChestX-ray14 contiene estudios en proyecciones PA y AP con las etiquetas que se encuentran poco presentes en PadChest, se decide combinar ambas bases de datos para subsanar la falta de muestras. Se asignan una o varias de las 31 etiquetas definidas en la sección anterior a cada una de las 14 etiquetas de ChestX-ray14, tal y como se observa en la Tabla 5.4. Se destaca que, dado que no hay una deficiencia de pacientes sanos, se eliminan de la base de datos de ChestX-ray14, reduciendo la cantidad de datos a entrenar con el objetivo de reducir los tiempos

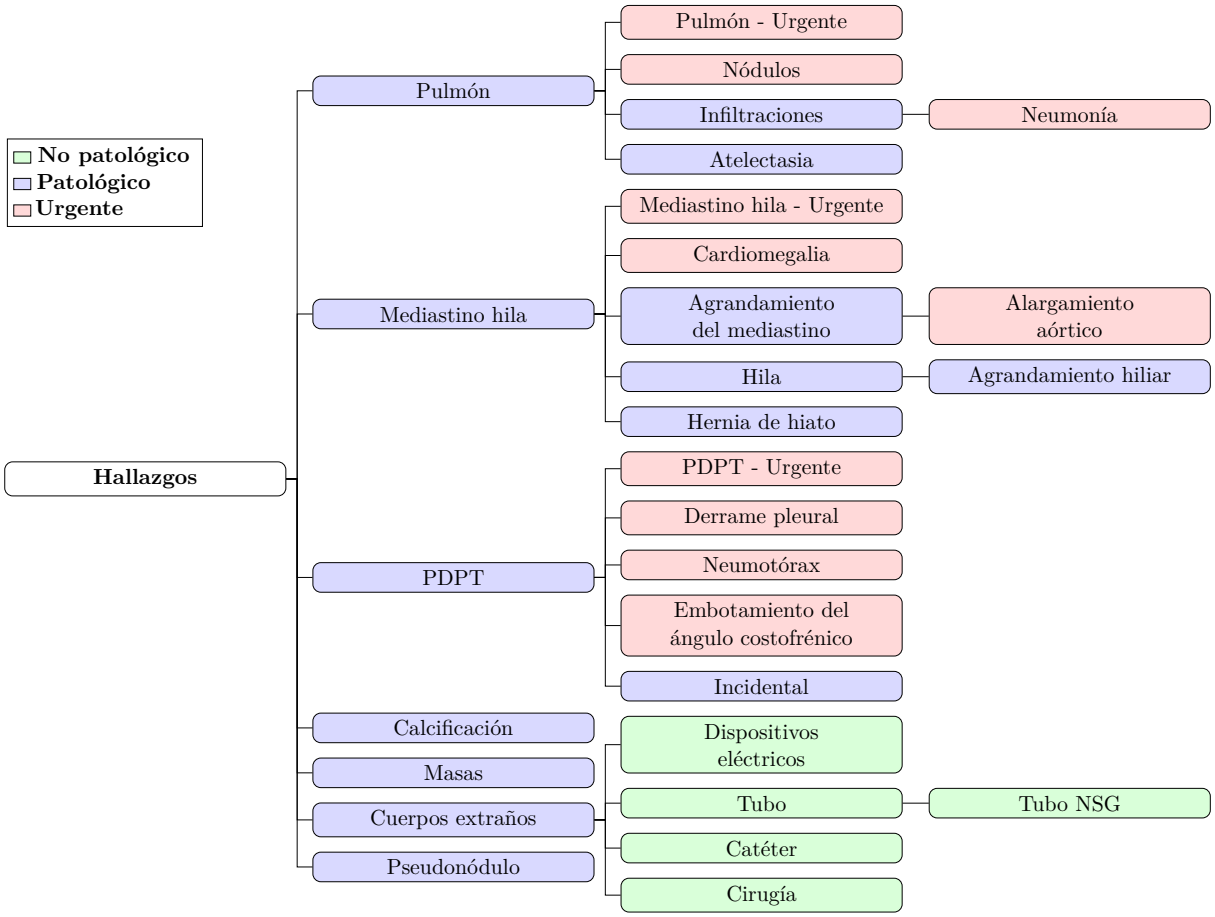


Figura 5.1: Jerarquía de la clasificación.

Etiqueta	Número de muestras
Pulmón	43 833
Urgente	16 513
Atelectasia	5 979
Nódulos	2 543
Infiltraciones	14 416
Neumonía	4 264
Mediastino hila	21 330
Urgente	18 874
Cardiomegalia	9 721
Agrandamiento del mediastino	9 302
Alargamiento aórtico	8 377
Hila	5 580
Agrandamiento hiliar	4 740
Hernia de hiato	1 598
PDPT	28 290
Urgente	11 805
Incidental	5 854
Derrame pleural	6 953
Neumotórax	339
Embotamiento del ángulo costofrénico	4 199
Calcificación	5 314
Masas	890
Cuerpos extraños	15 170
Dispositivos eléctricos	2 454
Tubo	6 731
Tubo NSG	4 898
Catéter	5 364
Cirugía	5 301
Pseudonódulo	1 876
Subóptimo	1 365
Normal	33 891

Tabla 5.3: Número de instancias por etiqueta sobre PadChest.

Etiqueta de ChestX-Ray14	Etiqueta agrupada de PadChest
Atelectasia	Atelectasia, Pulmón
Cardiomegalia	Cardiomegalia, Mediastino hila - urgente, Mediastino hila
Derrame	Derrame pleural, PDPT - urgente, PDPT
Infiltrado	Infiltraciones, Pulmón
Masa	Masa
Nódulos	Nódulos, Pulmón - urgente, Pulmón
Neumonía	Neumonía, Infiltraciones, Pulmón - urgente, Pulmón
Neumotórax	Neumotórax, PDPT - urgente, PDPT
Consolidación	Infiltraciones, Pulmón
Edema	Pulmón - urgente, Pulmón
Enfisema	Pulmón
Fibrosis	Pulmón - urgente, Pulmón
Engrosamiento pleural	PDPT
Hernia	Hernia de hiato, Mediastino hila

Tabla 5.4: Correspondencias entre las clases de ChestX-ray14 y las clases agrupadas de PadChest.

de entrenamiento. En la Tabla 5.5 se detalla la distribución final de las etiquetas de la base de datos utilizada durante el trabajo. Por otro lado, en la Tabla 5.6 se muestra el desbalanceo entre las diferentes proyecciones, mientras que la Tabla 5.7 contempla una proporción por sexo adecuada.

Las imágenes del conjunto de datos ChestX-ray14 no requieren de procesamiento adicional, dado que ya se encuentran en una resolución fija de 1024 píxeles y 8 bits de profundidad.

5.3. Particionado de los datos

Una vez definido el conjunto de datos utilizado en el trabajo, se deben determinar las particiones de entrenamiento, validación y test, de forma que las diferentes pruebas realizadas con diferentes parámetros se encuentren en igualdad de condiciones.

A pesar de que la metodología ideal es probar las diferentes combinaciones en un *k-fold cross-validation*, se ha optado por una partición fija de 70 % de entrenamiento, 10 % de validación y 20 % de test para reducir los tiempos de ejecución de cada prueba. Para lograr dicha partición se han tenido en cuenta dos factores:

Etiqueta	Número de muestras
Total	151 678
Pulmón	81 455
Urgente	30 988
Atelectasia	17 265
Nódulos	8 754
Infiltraciones	37 431
Neumonía	5 609
Mediastino hila	24 191
Urgente	21 515
Cardiomegalia	12 362
Agrandamiento del mediastino	9 302
Alargamiento aórtico	8 377
Hila	5 580
Agrandamiento hiliar	4 740
Hernia de hiato	1 825
PDPT	47 691
Urgente	28 936
Incidental	5 854
Derrame pleural	19 941
Neumotórax	5 431
Embotamiento del ángulo costofrénico	4 199
Calcificación	5 314
Masas	6528
Cuerpos extraños	15 170
Dispositivos eléctricos	2 454
Tubo	6 731
Tubo NSG	4 898
Catéter	5 364
Cirugía	5 301
Pseudonódulo	1 876
Subóptimo	1 365
Normal	33 891

Tabla 5.5: Número de instancias por etiqueta sobre PadChest + ChestX-ray14.

Proyección	Número de muestras
PA	113 129
AP	25 828
AP horizontal	12 721

Tabla 5.6: Número de muestras por proyección sobre PadChest + ChestX-ray14.

Sexo	Número de muestras
Femenino	72 327
Masculino	79 334
No conocido	17

Tabla 5.7: Número de muestras por sexo sobre PadChest + ChestX-ray14.

1. Para evitar la contaminación de los resultados de validación y test, se mantienen todos los estudios de un mismo paciente en una única partición. Esto se debe a que las mayoría de los estudios de un mismo paciente, analizan la progresión de una o varias enfermedades, de forma que si se ha entrenado con alguna radiografía de dicha progresión, se puede predecir más fácilmente el resto de radiografías del paciente.

2. Para reducir el efecto del desbalanceo de datos, se busca estratificar las particiones para que tengan la misma proporcionalidad de representación para las diferentes etiquetas. Para ello, se han realizado varias particiones aleatorias manteniendo el primer requisito, almacenando la que consigue mejores proporciones.

De esta forma, el conjunto de datos de 151 678 imágenes se divide en entrenamiento, validación y test tal y como indica la Tabla 5.8.

Partición	Número de muestras
Entrenamiento	106 294
Validación	15 027
Test	30 357

Tabla 5.8: Número de imágenes disponibles por partición.

5.4. Procesamiento del conjunto de datos ActualTec

ActualTec es una base de datos proporcionada por la empresa ActualTec[1] para el proyecto RELIANCE que ofrece 44 566 estudios médicos no etiquetados en diferentes proyecciones y localizaciones del cuerpo. El procesamiento de esta base de datos deja el total de imágenes disponibles en 27 685 después aplicar los siguientes pasos:

1. Se mantienen los DICOM con localización tórax.
2. Se mantienen las proyecciones PA y AP.
3. Se extrae la profundidad de bit de la radiografía.
4. Se extrae el esquema de color de la radiografía.
5. Se extraen y procesan los píxeles de la radiografía para obtener una resolución de 1024, 8 bits de profundidad y esquema de color *identity*.

En este capítulo se ha explicado el preprocesamiento necesario en las diferentes bases de datos utilizadas durante el trabajo. Por un lado, se detalla la agrupación por zonas anatómicas utilizada el entrenamiento de los modelos CNN bajo el proyecto RADIANT, cuyos resultados se muestran en el Capítulo 6. En dicho capítulo se muestran también diferentes resultados de la arquitectura ViT sobre esta misma agrupación. Por otro lado, se explica la agrupación por zonas anatómicas con nivel de urgencia ampliada con ChestX-ray14, empleada en la búsqueda de hiperparámetros del Capítulo 7. Sobre esta agrupación se detalla el particionado de los datos en entrenamiento, validación y test, necesario para entrenar y comparar los diferentes modelos. En última instancia, se muestra el procesamiento aplicado sobre el conjunto de datos no etiquetado ActualTec, cuyo propósito se explica en detalle en el Capítulo 7.

[1] http://www.actualmed.com/

Capítulo 6

Estado previo e implementación

En este capítulo se describe el estado de la herramienta previo dado por el proyecto RADIANT que trabaja sobre la agrupación de 8 etiquetas. Por un lado, se define la arquitectura e hiperparámetros utilizados, junto con el AUC de validación y test obtenidos para los mismos. Por otro lado, se entrenan diversos modelos ViT sobre esta misma agrupación, con el objetivo de establecer su potencial con respecto al uso de CNN tradicionales. Además, se muestra la optimización de carga de datos aplicada para reducir los tiempos de entrenamiento en el resto del trabajo.

6.1. Estado previo de la herramienta

La herramienta comienza su desarrollo en mediados de 2020 bajo el proyecto RADIANT, basándose en modelos CNN de forma parecida al trabajo realizado por [23, 24]. Trabaja sobre la agrupación en zonas patológicas sobre el conjunto de datos de PadChest. Se tiene la siguiente estructura para la extracción de características:

1. El modelo parte de radiografías en resolución 1024 en un solo canal y normalizadas por la media y desviación de ImageNet.
2. Aplicación de diferentes aumentos de datos aleatorios en GPU.
3. Una capa convolucional que se encarga de triplicar el canal de entrada, así como de reducir la resolución de la imagen a 512. Para ello, se utiliza tamaño de kernel de 3×3, *stride* de 2×2, relleno de 1×1 y tres filtros. Opcionalmente, se añade una segunda capa convolucional con las mismas características para utilizar modelos con resolución de entrada de 256, de

forma análoga al trabajo realizado por [24]. Se obtienen mejores resultados utilizando una única capa convolucional y resolución 512.

4. Un modelo DenseNet-169 preentrenado en ImageNet, obteniendo una matriz de $16 \times 16 \times 1664$. Se prueba también el modelo DenseNet-121, obteniendo peores resultados.
5. Se aplica una capa *global average pooling* obteniendo un vector de 1664 componentes.

En cuanto al clasificador, se tiene la siguiente arquitectura:

1. Capa BN.
2. Capa dropout con ratio de 0,5.
3. Capa lineal intermedia de 512 neuronas con activación de tipo ReLU.
4. Capa BN.
5. Capa dropout con ratio de 0,5
6. Capa lineal de clasificación de 8 neuronas, una por cada etiqueta del problema de clasificación.
7. Una activación de tipo *sigmoid*.

Se aplican los siguientes aumentos de datos durante el entrenamiento:

- Rotación horizontal aleatoria con una probabilidad del 50 %.
- Translación aleatoria en dirección horizontal y vertical con un ratio del 5 %. Esto implica que la imagen puede ser trasladada entre 0 y $1024 \times 5\,\% = 51$ píxeles.
- Rotación aleatoria con un ratio del 5 %. Esto implica que la imagen puede ser rotada en sentido horario o antihorario entre 0 y $360 \times 5\,\% = 18$ grados.
- Escalado aleatorio con un ratio del 5 %, de forma que la imagen puede verse aumentada o disminuida en dicho porcentaje.

En cuanto a los hiperparámetros de entrenamiento se utiliza el optimizador Adam con un tamaño de lote de 32 sobre una metodología de entrenamiento por doble fase:

1. Se entrena el clasificador con un LR de 10^{-3} durante dos épocas.

2. Se entrena el modelo al completo con un LR de 10^{-4} durante 20 épocas, aplicando *reduce on plateau* sobre la función de pérdida de validación con paciencia 2 y factor 0,1. De esta forma, cuando la pérdida de validación no mejore durante 2 épocas se reduce el LR multiplicando por el factor 0,1. Este hecho es ignorado cuando el LR llega a 10^{-8}. Se añade también *early stop*, de forma que el entrenamiento se detiene cuando el valor de la función pérdida sobre el conjunto de validación no mejora durante 3 épocas.

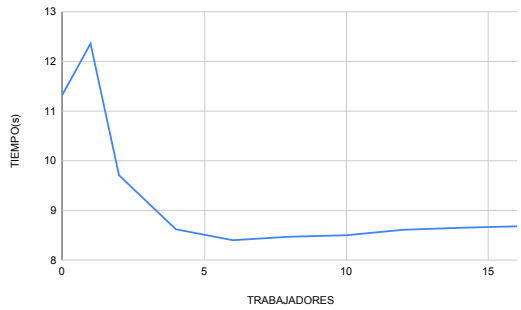
3. Para mitigar el efecto del desbalanceo de clases, se aplican **pesos de clase** (*class weights*). Estos pesos se encargan de sobreaprender muestras de clases poco representadas e infra-aprender muestras de clases bien representadas. Para ello, se multiplica el resultado de la función de pérdida BCE por dichos pesos, necesitando por lo tanto de un peso por clase. Dicho peso es calculado como $W_i = \frac{N_i}{c \times N}$, dónde N_i es el número de muestras de la clase i en el conjunto de entrenamiento, c el número de clases de la tarea y N el número de muestras del conjunto de entrenamiento. Esta técnica es aplicada a en todas las pruebas a lo largo de toda la memoria.

Con las características de arquitectura, entrenamiento y aumento de datos descrito se obtiene un AUC medio de 0,861 y 0,855 sobre las particiones de test y validación respectivamente

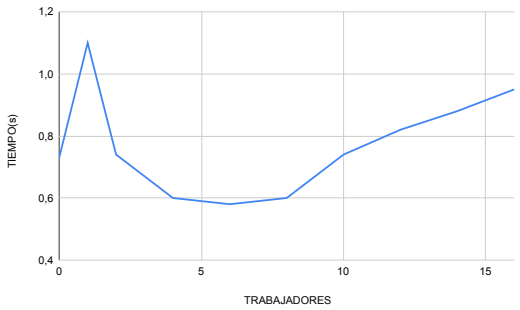
6.2. Optimización de la carga de datos

Antes de entrenar los primeros modelos se decide optimizar los tiempos de entrenamiento en la medida de lo posible. Una parte clave en la optimización de este tiempo de entrenamiento es la optimización de los tiempos de carga de los archivos utilizados durante el mismo. Por ello se proponen y evalúan dos técnicas diferentes para cargar dichos datos. Por un lado, se prueba a cargar los archivos en formato PNG comprimido, reduciendo el espacio en disco y por lo tanto el tiempo de carga, pero requiriendo de decodificado. Por otro lado, se prueba a cargar las imágenes descomprimidas en formato PT (soportado por PyTorch), de forma que el tiempo de carga es mayor pero no requiere de decodificado.

Se crean 500 imágenes en escala de grises de 8 bit de forma aleatoria, que ocupan 24 KiB y 1024 KiB en formato PNG y PT respectivamente, notando que este último ocupa significativamente más. Se prueban ambas metodologías con diferentes cantidades de subprocesos de carga. En el caso de cero subprocesos, el proceso principal se encarga de cargar las imágenes una a una. En el resto de casos, los subprocesos cargan imágenes de forma simultanea. En la Tabla 6.1 se advierte un tiempo de carga significativamente menor con el formato PT, para cualquier número de subprocesos. Para ambos tipos de carga se tiene una tendencia similar con respecto al número de subprocesos, con un incremento del tiempo de carga con uno solo y un decremento constante hasta 8. A partir de ese momento los tiempos de carga aumentan, especialmente para



(a) Formato PNG.



(b) Formato PT.

Figura 6.1: Tiempos de carga de 500 imágenes sobre PNG y PT.

el formato PT. Las diferentes pruebas se han realizado sobre un CPU Intel(R) Xeon(R) Silver 4210R CPU a 2.40GHz de 10 núcleos.

De esta forma, la metodología de carga para el resto de carga se basa en el uso del formato PT, por lo que todas las imágenes de la agrupación de 31 etiquetas se descomprimen en formato PT. Por otro lado, a pesar de que se observa una reducción del tiempo de carga usando entre 2 y 8 subprocesos, la propia documentación de PyTorch recomienda mantener dicho valor a 0 en entrenamientos multi-GPU. Siendo esta nuestra situación, se emplea una metodología de carga de datos en formato PT con cero subprocesos para el resto del trabajo.

6.3. Resultados iniciales de VT

Con el objetivo de establecer un *baseline* de las arquitecturas VT respecto a las CNN, se entrenan diversos modelos ViT-B/16 y ViT-L/16 en resolución 384 utilizando los siguientes hiperparámetros de entrenamiento:

- Se mantiene la metodología de entrenamiento por doble fase con los con tamaño de lote de 80 para reducir el tiempo de entrenamiento. Por otro lado, el elevado número de parámetros de entrenamiento de los modelos ViT pueden requerir de un LR menor, por lo que también se prueba un valor de 10^{-5}.
- Las imágenes se normalizan con una media y desviación típica de 0,5 en cada canal, ajustándose al preentrenamiento de los modelos ViT.
- Se mantienen los mismos aumentos de datos.

- Se redimensiona la imagen a una resolución de 384, dado que esta no puede ser obtenida a mediante de capas convolucionales partiendo de una resolución de 1024.

En cuanto al clasificador, se definen las siguientes posibles configuraciones:

- **Clasificador A**: Se mantiene el mismo clasificador usado para la DenseNet
  1. Capa BN.
  2. Capa *dropout* con ratio de 0,5.
  3. Capa lineal intermedia de 512 neuronas con activación de tipo ReLU.
  4. Capa BN.
  5. Capa *dropout* con ratio de 0,5.
  6. Capa lineal de clasificación de 8 neuronas, una por cada etiqueta del problema de clasificación.
  7. Una activación de tipo *sigmoid*.
- **Clasificador B**: Se elimina la capa oculta de 512 neuronas y las capas BN y de *dropout* previas a la misma, pero manteniendo las previas a la capa de clasificación.
  1. Capa BN.
  2. Capa dropout con ratio de 0,5.
  3. Capa lineal de clasificación de 8 neuronas, una por cada etiqueta del problema de clasificación.
  4. Una activación de tipo *sigmoid*.
- **Clasificador C**: Se eliminan las capas BN y *dropout* previas a la capa de clasificación pues no se utilizan en el clasificador recomendado descrito en el Capítulo 3. Se añade una única capa lineal de 8 neuronas, una por cada etiqueta del problema de clasificación y una activación de tipo *sigmoid*.

En cualquier caso se debe aplicar la función de pérdida BCE al tratarse de un problema multi-etiqueta.

Los diferentes tamaños de lote y valores de LR, así como los diferentes clasificadores, se prueban también sobre las DenseNet-169. Por otro lado, se entrenan modelos DenseNet-161 debido su mejora de precisión en ImageNet respecto a los primeros, con el objetivo de establecer una mejor comparativa con diversos modelos de CNNs. Para reducir los tiempos de entrenamiento se entrenan los modelos durante una época de primera fase y otra época de segunda fase. Los resultados, mostrados en la Tabla 6.1, generan las siguiente conclusiones:

Modelo	Resolución	Ratio de aprendizaje	Clasificador	*Dropout*	AUC
DenseNet-169	512	10^{-4}	A	0.5	0,820
DenseNet-161	512	10^{-4}	A	0.5	0,825
ViT-B/16	384	10^{-4}	A	0.5	0,809
ViT-L/16	384	10^{-4}	A	0.5	0,807
DenseNet-169	512	10^{-5}	A	0.5	0,781
DenseNet-161	512	10^{-5}	A	0.5	0,790
ViT-B/16	384	10^{-5}	A	0.5	0,794
ViT-L/16	384	10^{-5}	A	0.5	0,811
DenseNet-169	512	10^{-4}	B	0.5	0,815
DenseNet-161	512	10^{-4}	B	0.5	0,822
ViT-B/16	384	10^{-4}	B	0.5	0,800
ViT-L/16	384	10^{-5}	B	0.5	0,809
DenseNet-169	512	10^{-4}	C	✗	0,830
DenseNet-161	512	10^{-4}	C	✗	**0,838**
ViT-B/16	384	10^{-4}	C	✗	0,816
ViT-L/16	384	10^{-5}	C	✗	0,822

Tabla 6.1: Resultados sobre AUC de validación de diferentes configuraciones ViT y DenseNet.

- La DenseNet-161 obtiene una pequeña ventaja de AUC con respecto a la DenseNet-169 en todas las configuraciones.
- El modelo ViT-L/16 alcanza resultados ligeramente superiores al modelo ViT-B/16 en las diferentes configuraciones.
- Los modelos DenseNet y ViT-B/16 consiguen resultados significativamente mejores con un LR de 10^{-4}, mientras que el modelo ViT-L/16 logra un rendimiento similar con ambos ratios de aprendizaje.
- En general, los modelos DenseNet superan en precisión a los modelos ViT.
- El clasificador C, que queda formado por una única capa lineal de clasificación, obtiene resultados superiores respecto al resto de clasificadores en todos los modelos.

Estas primera pruebas parecen dejar la arquitectura ViT por detrás de los modelos DenseNet, arquitecturas CNN más tradicionales y conceptualmente simples. Para comparar la convergencia de los modelos DenseNet-169 y DenseNet-161 y los modelos ViT-B/16 y ViT-L/16 a largo plazo, se entrena la mejor configuración para cada modelo durante 2 épocas de primera fase y 20 épocas de segunda fase aplicando *reduce on plateau*. Los resultados, mostrados en la Tabla 6.2, demuestran una convergencia superior de los modelos DenseNet. Por otro lado, no se observan

Modelo	Resolución	Tamaño de lote	LR	Clasificador	*Dropout*	AUC
Estado previo de la herramienta						
DenseNet-169	512	32	10^{-4}	A	0,5	0,855
DenseNet-169	512	80	10^{-4}	C	✗	**0,869**
DenseNet-161	512	80	10^{-4}	C	✗	**0,868**
ViT-B/16	384	80	10^{-4}	C	✗	0,847
ViT-L/16	384	80	10^{-5}	C	✗	0,850

Tabla 6.2: Resultados sobre AUC de validación de diferentes configuraciones ViT y DenseNet.

diferencias de rendimiento significativas entre los dos modelos DenseNet probados, así como los modelos ViT.

En esta sección se han probado diversas configuraciones ViT y DenseNet con el objetivo de establecer unos resultados iniciales de las arquitecturas VT. Los diferentes experimentos mejoran la precisión obtenida en el estado previo de la herramienta, desarrollado bajo el proyecto RADIANT. Dichas mejoras se obtienen de la simplificación el clasificador y no del uso de arquitecturas ViT que obtienen un peor resultado. De esta forma, se advierte la necesidad de probar diversas configuraciones con las diferentes arquitecturas CNN y VT. Sin embargo, probar todas las posibles combinaciones resulta computacionalmente inviable, por lo que se recurre a una búsqueda bayesiana de hiperparámetros en el próximo capítulo.

Capítulo 7

Búsqueda de hiperparámetros y resultados

En este capítulo se entrenan diversos modelos sobre la partición de entrenamiento de la agrupación de 31 etiquetas. Dado que probar todas las combinaciones es computacionalmente inviable, se opta por una búsqueda bayesiana de hiperparámetros en la que se realizan 630 entrenamientos. Estos algoritmos detectan y descartan combinaciones poco prometedoras en base a resultados previos obtenidos sobre combinaciones parecidas, de forma que se la búsqueda acaba convergiendo a una serie de hiperparámetros óptimos.

7.1. Configuración

Se prueban hiperparámetros pertenecientes a diferentes áreas de una red neuronal. A continuación se muestran, de forma ordenada, las diferentes etapas de la arquitectura utilizada, junto con los hiperparámetros que la definen. También se describe la configuración del optimizador con la que la red neuronal se ha entrenado. Por último, se describe el sistema sobre el que se han ejecutado las diferentes pruebas.

7.1.1. Aumento de datos

Se parte de una imagen en resolución 1024 y un único canal, sobre la que se aplican diferentes aumentos de datos. Con el objetivo de experimentar con aumentos más y menos agresivos se prueban diferentes ratios en cada una de las técnicas:

- **Rotación horizontal**: 50 %
- **Translación**: Ratios de 0 hasta 0,1 con saltos de 0,01.
- **Rotación**: Ángulos de 0 hasta 20 con saltos de 2.
- **Escalado**: Ratios de 0 hasta 0,1 con saltos de 0,01.
- ***Shear***: Ángulos de 0 hasta 5 con saltos de 0,5.

7.1.2. Preprocesamiento

Una vez se aplican los diferentes aumentos de datos, se debe ajustar la imagen aumentada a la entrada esperada por el modelo. Se prueban los siguientes tres procedimientos:

- **Redimensionado normal**: Se aplica una simple capa de *resize* de PyTorch. Posteriormente, se aplica una capa convolucional con 3 filtros, kernel y *stride* 1×1 y sin relleno, con el objetivo de triplicar el canal. Los pesos y bias, inicializados 1 y 0 respectivamente, son congelados durante todo el entrenamiento. Esta metodología funciona para cualquier resolución de entrada exigida por los modelos.
- **Redimensionado inteligente**: Se aplican una o dos capas convolucionales con el objetivo de reducir la resolución de la imagen, además de triplicarla, tal y como se describe en el estado previo de la herramienta. Esta metodología funciona correctamente con resoluciones de 256 y 512 pero no puede aplicarse en el resto de resoluciones. En esos casos, se aplica una capa *resize* y posteriormente se añade una capa con 3 filtros, kernel 3×3 y stride y relleno 1×1, con el objetivo de triplicar el canal de entrada de forma inteligente, permitiendo su entrenamiento.
- **Redimensionado inteligente fijo**: Se aplica la misma metodología anterior pero fijando unos pesos iniciales a las capas convolucionales de $1/9$ para los pesos y 0 para el bias, permitiendo de igual forma su entrenamiento.

Con estas tres metodologías se pone a prueba si el uso de capas convolucionales para reducir la resolución supone una ventaja con respecto a aplicar un redimensionado normal. Además, se investiga si la inicialización aleatoria de dichas capas convolucionales puede influir negativamente en la precisión de la red neuronal.

Una vez la imagen se ha aumentado y reducido, se normaliza siguiendo la media y desviación típica indicada para cada modelo en las diferentes publicaciones, y que son descritas en la próxima sección.

7.1.3. Extracción de características

Se prueban los modelos VT seleccionados en el Capítulo 3, así como diversos modelos DenseNet para establecer una comparativa entre ambos. Con el objetivo de analizar el comportamiento arquitecturas CNN recientes, se prueban diferentes modelos ConvNeXt [37], una arquitectura que toma inspiración de varias decisiones de diseño de las arquitecturas VT manteniendo la operaciones convolucionales.

Los diferentes detalles como la normalización requerida o la precisión sobre ImageNet de los diferentes modelos utilizados se describen en la Tabla 7.1. En dicha tabla se observa como cada arquitectura tiene dos posibles especificaciones, dónde la que se compone de un mayor número de parámetros se define como la especificación de tamaño grande. El uso de una especificación grande o pequeña se convierte en un hiperparámetro de búsqueda. Por otro lado, se observa que todas las arquitecturas excepto ViT tienen dos posibles resoluciones de diferente tamaño, lo que se convierte en otro hiperparámetro. Destacar que la precisión de los modelos DenseNet es la obtenida mediante modelos en resolución 224, dado que las precisiones obtenidas con otras resoluciones no están publicadas.

El modelo Deit⚗ utiliza las metodologías de entrenamiento no etiquetado y con ruido descritas en la Sección 2.7, así como el entrenamiento por *token* de destilación descrito en la Sección 3.4. De esta forma, se entrena un modelo DenseNet-161 profesor de forma completa con los hiperparámetros utilizados para dicho modelo en la Tabla 6.2 del capítulo anterior, obteniendo un AUC de 0,9084 sobre validación. Estos hiperparámetros son:

- **Aumento de datos**: Se aplica rotación horizontal con probabilidad del 50 %, translación y escalado aleatorio con un ratio del 0,05 y rotación aleatoria de 18 grados.
- **Preprocesamiento**: Se aplica un preprocesamiento de redimensionado inteligente no fijo.
- **Clasificador**: No se utiliza ninguna capa oculta, ni *dropout* ni capa de normalización de ningún tipo.
- **Optimizador**: Se utiliza un tamaño de lote de 80 con el optimizador Adam. La primera fase consiste en 2 épocas de entrenamiento con un LR de 10^{-3}. La segunda fase consiste en 20 épocas más con un LR de 10^{-4} y aplicando *reduce on plateau*.

Una vez se tiene el modelo profesor, se precomputan las predicciones del conjunto de datos de 31 etiquetas y el conjunto no etiquetado ActualTec, sin aplicar técnicas de aumento de datos. Posteriormente, el modelo DeiT se entrena de forma no etiquetada sobre ActualTec mediante las predicciones del profesor DenseNet-161 utilizando el *token* de destilación y aplicando aumento de datos. Finalmente se reentrena el modelo de forma supervisada sobre el conjunto de 31 etiquetas

utilizando de nuevo las predicciones del profesor para calcular una pérdida de destilación y las etiquetas reales para calcular la pérdida habitual. Para obtener la predicción final del modelo, y así poder medir el AUC de validación, se calcula la media de los clasificadores de los *tokens* de destilación y de clase.

7.1.4. Clasificador

En cuanto al clasificador, se prueban los siguientes hiperparámetros:

- **Capas ocultas**: Añadir una capa oculta con activación de tipo ReLU antes de la capa de clasificación o no añadir ninguna.
- **Neuronas ocultas**: 128, 256, 384 y 512 neuronas en caso de añadir la capa oculta.
- **Capa de normalización**: Añadir BN, LN o ninguna antes de la capa oculta, si se añade, y antes de la capa de clasificación.
- **Tipo de *dropout***: Tradicional y alpha.
- **Ratio de *dropout***: Ratios de 0 hasta 0,5 con saltos de 0,1.

7.1.5. Configuración del optimizador

Respecto al optimizador, se prueban los siguiente hiperparámetros:

- **Optimizador**: Adam y AdamW.
- **Tamaño de lote**: 64 y 80. No se prueban tamaños de lote superiores a 80 dado que algunos modelos sobresaturan la GPU, mientras que no se prueban tamaños inferiores a 64 ya que incrementan el tiempo de entrenamiento.
- **Primera fase**: Se entrena el modelo con un LR de 10^{-3} durante una sola época. En el caso del modelo DeiT⚗ se entrena sobre el conjunto no etiquetado ActualTec utilizando las etiquetas precalculadas del modelo DenseNet-161 profesor.
- **Segunda fase**: Se entrena el modelo al completo durante una época más con diferentes valores de LR: 10^{-6}, 5×10^{-6}, 10^{-5}, 5×10^{-5}, 10^{-4} y 5×10^{-4}. En el caso del modelo DeiT⚗ se entrena el modelo al completo durante una época sobre ActualTec y una época más sobre el conjunto de 31 etiquetas.

Arquitectura	Especificación	Parámetros	Resolución	Normalización	Precisión
Arquitecturas VT					
ViT	*Base*	87 M	384	0,5	83,97
ViT	*Large*	307 M	384	0,5	85,15
BEiT	*Base*	87 M	224	0,5	85,20
BEiT	*Large*	307 M	384	0,5	88,40
BEiT	*Base*	87 M	384	0,5	86,80
BEiT	*Large*	307 M	512	0,5	**88,60**
DeiT	*Base*	87 M	224	0,5	81,80
DeiT	⚗ *Base*	88 M	224	ImageNet	84,20
DeiT	*Base*	87 M	384	0,5	83,10
DeiT	⚗ *Base*	88 M	384	ImageNet	85,20
SwinTV2	*Base*	88 M	256	0,5	86,20
SwinTV2	*Large*	197 M	256	0,5	86,90
SwinTV2	*Base*	88 M	384	0,5	87,10
SwinTV2	*Large*	197 M	384	0,5	87,60
Arquitecturas CNN					
DenseNet	169	14 M	256	ImageNet	75,60[a]
DenseNet	161	28 M	256	ImageNet	77,13[a]
DenseNet	169	14 M	512	ImageNet	75,60[a]
DenseNet	161	28 M	512	ImageNet	77,13[a]
ConvNeXt	*Large*	198 M	224	ImageNet	86,6
ConvNeXt	*XLarge*	350 M	224	ImageNet	87,0
ConvNeXt	*Large*	198 M	384	ImageNet	87,5
ConvNeXt	*XLarge*	350 M	384	ImageNet	87,8

[a]Resultados obtenidos en resolución 224.

Tabla 7.1: Las diferentes arquitecturas VT y CNN utilizadas en la búsqueda de hiperparámetros con su respectiva precisión sobre ImageNet obtenida a través de las diferentes publicaciones.

Configuración de GPU	
GPU	NVIDIA A100-SXM
VRAM	80 GB
Número de GPUs	8
Conexión GPU a GPU	
NVLink	Tercera generación
NVSwitch	Segunda generación
Ancho de banda	600 GB/s
Configuración de CPU	
Procesador	AMD EPYC 7513
Número de núcleos	32
Número de hilos	64
Frecuencia base	2,6 GHz
Frecuencia máxima	3,65 GHz
Número de procesadores	2
Configuración de RAM	
Memoria RAM	2 TB
Tipo de memoria	DDR4

Tabla 7.2: Características del nodo utilizado durante la búsqueda de hiperparámetros.

7.1.6. Configuración del sistema

Para las diferentes pruebas de la búsqueda de hiperparámetros se ha utilizado el clúster shirka, ofrecido por la Universitat Jaume I a los investigadores para ejecutar trabajos de alto coste computacional. Para utilizar dicho clúster, se deben ejecutar los trabajos mediante el uso del sistema de colas SLURM (*Simple Linux Utility for Resource Management*) sobre uno o varios de los diferentes nodos disponibles. En concreto, en este trabajo se recurre a un nodo multi-GPU cuyas características se describen en la Tabla 7.2.

7.2. Resultados

Tras realizar la búsqueda bayesiana de hiperparámetros, las 10 mejores pruebas coinciden en los siguientes hiperparámetros:

Tipo de preprocesamiento	AUC medio	AUC máximo
Redimensionado normal	**0,896**	**0,903**
Redimensionado inteligente	0,875	0,899
Redimensionado inteligente fijo	0,886	0,899

Tabla 7.3: AUC medio y máximo obtenido en las diferentes pruebas de la búsqueda de hiperparámetros para cada tipo de preprocesado.

- El modelo DeiT⚗-B en resolución 384.
- Un clasificador simple sin capa oculta intermedia ni capas de normalización.
- El redimensionado normal en lugar de inteligente.
- Un LR de 5×10^{-5}.

Otros hiperparámetros como el optimizador, el tamaño de lote, los aumentos de datos o el *dropout* no quedan completamente definidos. Por ello, se opta por analizar el comportamiento de dichos hiperparámetros, junto con los ya definidos, en todas las pruebas en lugar de centrarse únicamente en las 10 mejores. Para mejorar la legibilidad, se muestran agrupados en diversas secciones, ordenadas según su efecto en la precisión final del modelo. Cada sección fija diferentes hiperparámetros para el entrenamiento final, así como para el análisis del resto de hiperparámetros en secciones consecutivas.

7.2.1. Preprocesamiento

La Tabla 7.3 muestra el AUC medio y máximo obtenido en todas las pruebas en función del preprocesamiento utilizado. Se puede observar una clara ventaja del redimensionado normal respecto al inteligente en sus dos versiones, especialmente en el AUC medio. Se advierte que ambos tipos de redimensionado inteligente obtienen un AUC máximo parecido, pero con un AUC medio superior en el caso del redimensionado inteligente fijo. Esto indica que con la misma estructura de capas convolucionales de redimensionado inteligente, la inicialización aleatoria de dichas capas puede perjudicar la precisión del modelo, empeorando su estabilidad y reproducibilidad. Esta misma conclusión, también se obtiene de la Tabla 7.4, que muestra la desviación estándar del AUC para cada tipo de preprocesado, siendo más elevada en el redimensionamiento inteligente. Esta tabla muestra que en general un mayor estabilidad del redimensionamiento normal frente al redimensionamiento inteligente fijo y no fijo.

Además de obtener una mejor precisión, el redimensionado normal permite almacenar las imágenes en la resolución final de entrada del modelo, en lugar de redimensionar a una resolución

Tipo de preprocesamiento	Desviación estándar del AUC
Redimensionado normal	**0,0099**
Redimensionado inteligente	0,0341
Redimensionado inteligente fijo	0,0152

Tabla 7.4: Desviación estándar del AUC entre las diferentes pruebas de la búsqueda de hiperparámetros para cada tipo de preprocesado.

Arquitectura	AUC medio	AUC máximo
DeiT	**0,898**	**0,903**
ConvNeXt	0,896	0,900
SwinTV2	0,888	0,890
BEiT	0,8872	0,898
ViT	0,884	0,888
DenseNet	0,881	0,890

Tabla 7.5: AUC medio y máximo obtenido en las diferentes pruebas de la búsqueda de hiperparámetros para cada arquitectura.

de 1024 píxeles para luego reducirlos mediante capas convolucionales durante el entrenamiento. Esto reduce los tiempos de carga y entrenamiento, así como el almacenamiento fijo necesario. Por estos motivos, el redimensionado normal se deja fijo en el entrenamiento final del modelo, así como para el análisis del resto de hiperparámetros en las próximas secciones.

7.2.2. Extracción de características

La Tabla 7.5 muestra el AUC medio y máximo obtenido entre todas las pruebas en función de la arquitectura base utilizada. A partir de esta tabla, se obtienen las siguientes conclusiones:

- La arquitectura DeiT obtiene el mejor AUC medio y máximo.
- El resto de arquitecturas VT tienen un AUC medio y máximo por debajo de DeiT y ConvNeXt.
- La arquitectura ConvNeXt mejora a la arquitectura DenseNet de forma significativa tanto en AUC medio como máximo, a pesar de que ambas son de tipo CNN.

Arquitectura	Especificación	AUC medio	AUC máximo
DeiT	*Base*	0,881	0,886
	⚗ *Base*	**0,898**	**0,903**
ConvNeXT	*Large*	0,896	0,900
BEiT	*Base*	0,867	0,870
	Large	**0,889**	**0,898**
SwinTV2	*Base*	0,886	0,886
	Large	0,887	0,890
ViT	*Base*	0,856	0,856
	Large	**0,886**	**0,888**
DenseNet	169	**0,890**	**0,890**
	161	0,880	0,886

Tabla 7.6: AUC medio y máximo obtenido en las diferentes pruebas de la búsqueda de hiperparámetros para cada arquitectura y tamaño.

Por otro lado, en dicha tabla se aprecia que la arquitectura BEiT obtiene un AUC medio inferior la arquitectura SwinTV2, pero un AUC máximo significativamente mejor. Estas disparidades entre AUC máximo y medio se repiten al comparar otras arquitecturas. La Tabla 7.6 muestra el AUC medio y máximo en función de la especificación para cada arquitectura, con el objetivo de analizar si el AUC máximo proviene de una especificación de la arquitectura, mientras que la otra baja la media de la arquitectura. Se obtienen las siguientes conclusiones:

- Las especificaciones de mayor tamaño obtienen un mayor AUC medio y máximo respecto a las especificaciones de menor tamaño, a excepción de la DenseNet-169 que supera en precisión a DenseNet-161 y la red ConvNeXt-L dado que no se ha probado la especificación de mayor tamaño *XLarge*.
- La red DenseNet-169 supera en AUC medio y máximo a ambas especificaciones de ViT, mientras que obtiene resultados similares a SwinTV2. La red DenseNet-161 produce una bajada de la media global de la arquitectura DenseNet, lo que se refleja en la Tabla 7.5.
- La red BEiT-L mejora el AUC máximo respecto SwinTV2 en cualquiera de sus dos especificaciones. El AUC medio global de la arquitectura BEiT es menor tal y como se ha observado en la Tabla 7.5 debido al bajo rendimiento de BEiT-B.

En última instancia, la Tabla 7.7 analiza el efecto de la resolución de entrada en la precisión para cada arquitectura, únicamente sobre su mejor especificación. Se observa mejor rendimiento

Arquitectura	Especificación	Resolución	AUC medio	AUC máximo
DeiT	⚗ *Base*	224	0,880	0,891
		384	**0,899**	**0,903**
ConvNeXT	*Large*	384	0,896	0,900
BEiT	Large	384	0,866	0,866
		512	**0,891**	**0,898**
SwinTV2	*Large*	384	0,887	0,890
DenseNet	169	512	0,890	0,890

Tabla 7.7: AUC medio y máximo obtenido en las diferentes pruebas de la búsqueda de hiperparámetros para diferentes resoluciones sobre cada arquitectura en su mejor especificación.

Arquitectura	Especificación	Resolución	Tiempo medio por época (s)
DeiT	⚗ *Base*	384	**418**
ConvNeXT	*Large*	384	685
BEiT	*Large*	512	2427
SwinTV2	*Large*	384	1133
ViT	*Large*	384	1156
DenseNet	169	512	**383**

Tabla 7.8: Tiempo de entrenamiento medio por época para las diferentes arquitecturas con la mejor especificación y resolución en base a precisión.

a mayor resolución en los modelos DeiT y BEiT. Para los modelos ConvNeXt, SwinTV2 y DenseNet no se tienen pruebas con redimensionado normal y las especificaciones indicadas en cada arquitectura con una resolución de tamaño pequeño. Por otro lado, se recuerda que el modelo ViT solo se ha probado en resolución 384 a lo largo de la búsqueda de hiperparámetros, por lo que no se muestra en la tabla.

De esta forma se selecciona la arquitectura DeiT con especificación ⚗ *Base* y resolución de 384 píxeles para el entrenamiento final, así como para el análisis del resto de hiperparámetros en las próximas secciones, dado su mayor precisión. En la Tabla 7.8 se muestra el tiempo medio de ejecución de una época, incluyendo la fase de entrenamiento y la de validación, para cada arquitectura en su especificación y resolución con mayor AUC. Dado que el tamaño de lote influye en el tiempo de entrenamiento, se fija el tamaño de lote a 64. Se destaca que DeiT⚗-B no solo es el mejor modelo en cuanto a precisión, sino también el segundo con menor tiempo de entrenamiento necesario.

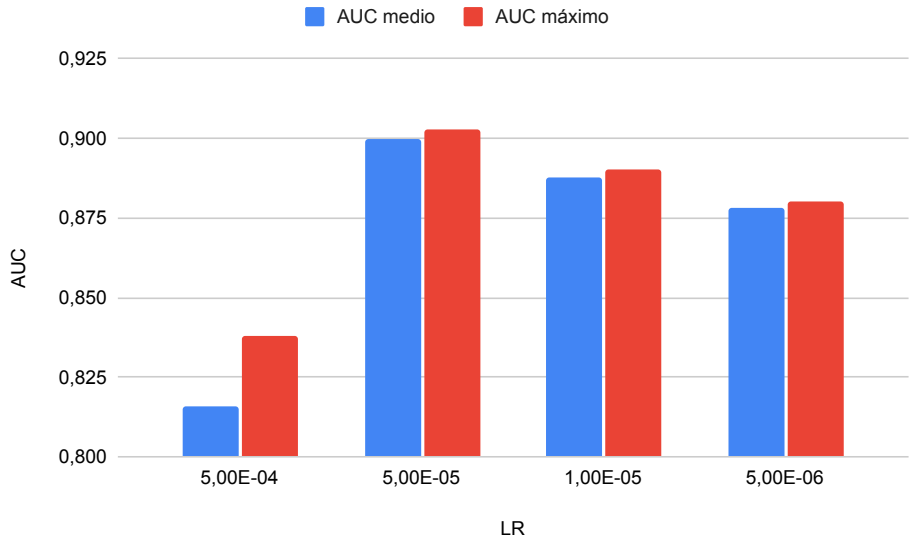


Figura 7.1: AUC medio y máximo en función del LR para la configuración fijada.

7.2.3. Configuración del optimizador

En la Figura 7.1 se muestra el efecto del LR en el entrenamiento sobre una configuración fija de modelo DeiT⚗-B con resolución de 384 píxeles y redimensionado normal como preprocesamiento. Se observa un rendimiento significativamente mejor tanto en el AUC medio como máximo con $LR = 5 \times 10^{-5}$. Algunos valores de LR introducidos en la búsqueda no se encuentran en dicha figura al no haber sido probados con la configuración fijada. Este valor del LR se fija para el entrenamiento final del modelo, así como para el análisis del resto de hiperparámetros.

Las Tablas 7.9 y 7.10 muestran el AUC medio y máximo entre todas las pruebas que utilizan la red DeiT⚗-B en resolución 384 y con redimensionado normal en función del tamaño de lote y del optimizador utilizado respectivamente. Se observa que dichos hiperparámetros no afectan no afectan de forma significativa a los resultados. De esta forma, se fija un tamaño de lote de 64 y el optimizador Adam para el entrenamiento final del modelo debido a un máximo ligeramente superior. Sin embargo, sí que se tienen en cuenta las dos opciones para ambos hiperparámetros en el análisis del resto debido a la reducida diferencia de precisión en posteriores secciones.

Tamaño de lote	AUC medio	AUC máximo
64	0,900	0,903
80	0,900	0,902

Tabla 7.9: AUC medio y máximo en función del tamaño de lote.

Optimizador	AUC medio	AUC máximo
Adam	0,900	0,903
Adamw	0,900	0,902

Tabla 7.10: AUC medio y máximo en función del tamaño de lote.

7.2.4. Clasificador

En cuanto al clasificador hay que analizar y definir diversos hiperparámetros:

- La Tabla 7.11 muestra el AUC medio y máximo en función del número de capas ocultas utilizadas en el clasificador. Se observa un mejor rendimiento, sobretodo en cuanto a AUC medio, al no utilizar ninguna capa oculta. De esta forma, dicho hiperparámetro se fija para el entrenamiento final del modelo, así como para el análisis del resto de hiperparámetros.

- La Tabla 7.12 muestra el AUC medio y máximo en función de la capa de normalización utilizada en el clasificador. Señala una mayor precisión al no utilizar ninguna capa de normalización, mientras que en caso de aplicar alguna capa de normalización expone mejores resultados con la capa LN. De esta forma, este hiperparámetro queda fijado para el entrenamiento final y el análisis del resto de hiperparámetros.

- La Figura 7.2 muestra el AUC medio y máximo en función del ratio de *dropout* aplicado y exhibe una tendencia negativa con el incremento del mismo. Se observan valores cercanos de AUC para ratios de 0,0, 0,1 y 0,2, por lo que se prueban los tres en el entrenamiento final. También se fijan estos valores para el análisis del resto de hiperparámetros.

- La Tabla 7.13 muestra el AUC medio y máximo en función del tipo de *dropout* aplicado. A pesar de exponer únicamente una ligera ventaja en en el uso del *dropout* tradicional, se fija para el entrenamiento final pero no para el análisis del resto de hiperparámetros.

Capas ocultas	AUC medio	AUC máximo
0	**0,900**	**0,903**
1	0,897	0,900

Tabla 7.11: AUC medio y máximo en función del número de capas ocultas.

Capa de normalización	AUC medio	AUC máximo
Ninguna	**0,900**	**0,903**
LN	0,898	0,900
BN	0,896	0,899

Tabla 7.12: AUC medio y máximo en función de la capa de normalización.

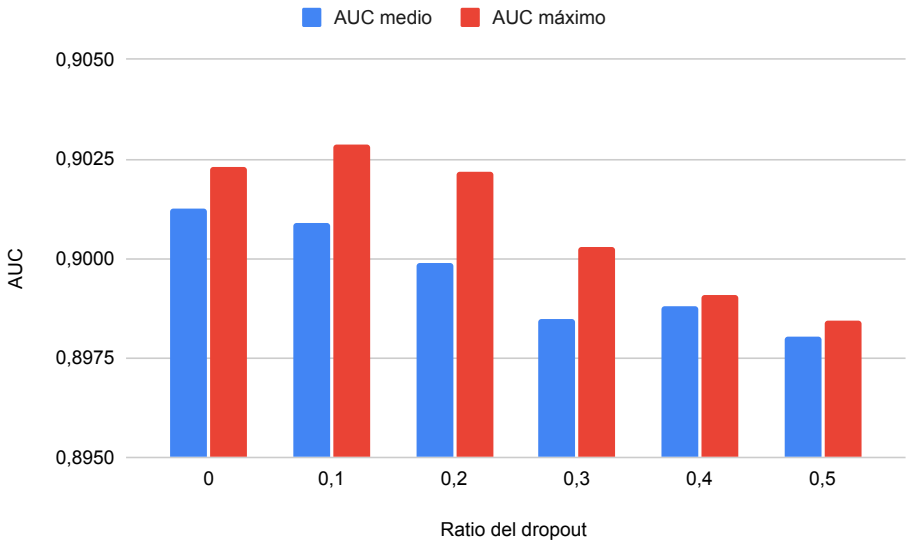
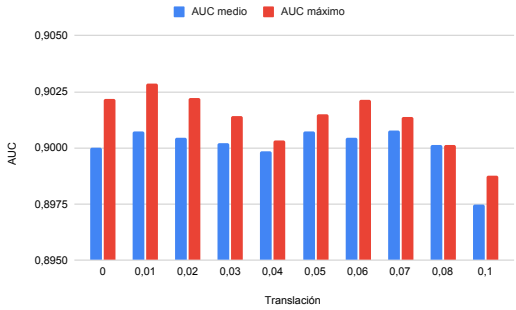


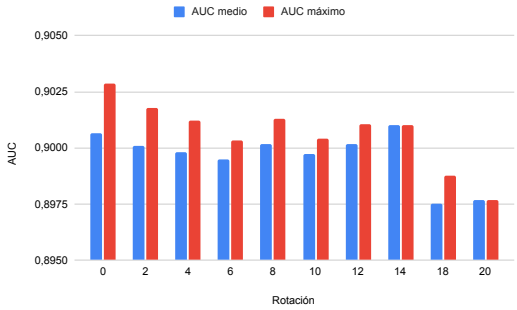
Figura 7.2: AUC medio y máximo en función del ratio de *dropout*.

Tipo de *Dropout*	AUC medio	AUC máximo
Tradicional	**0,900**	**0,903**
Alpha	0,899	0,902

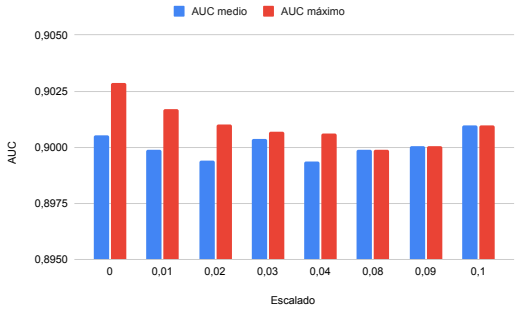
Tabla 7.13: AUC medio y máximo en función del tipo de *dropout*.



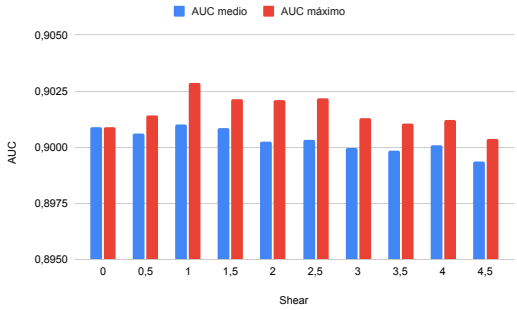
(a) Translación.



(b) Rotación.



(c) Escalado.



(d) *Shear*.

Figura 7.3: AUC medio y máximo en función del ratio o ángulo aplicado en los diferentes aumentos de datos.

7.2.5. Aumento de datos

En última instancia, la Figura 7.3 muestra el efecto de diferentes ratios en los aumentos de datos sobre el AUC medio y máximo, sin observar una tendencia clara. En el caso de la rotación y escalado se obtienen los mejores resultados cuando no se aplican, mientras que para la translación y el *shear* se consiguen los mejores resultados con valores de 0,01 y 1, respectivamente. Sin embargo, el efecto de técnicas de regularización, como el aumento de datos, se suele observar en un mayor número de épocas de entrenamiento. De esta forma, ninguna configuración debería ser descartada en base a las pocas épocas de entrenamiento realizadas en la búsqueda de hiperparámetros.

Nivel	Translación	Rotación	Escalado	*Shear*
Ninguno	0.0	0.0	0.0	0.0
Bajo	0.02	4.0	0.02	1.0
Medio	0.06	8.0	0.06	2.0
Alto	0.1	14.0	0.1	4.0

Tabla 7.14: Los diferentes niveles de aumentos de datos probados.

7.3. Entrenamiento final del modelo

Durante el análisis se han fijado para el entrenamiento final del modelo los siguientes hiperparámetros:

- **Preprocesamiento**: Redimensionado normal.
- **Modelo**: Arquitectura DeiT⚗-B con resolución de 384 píxeles:
- **Optimizador**: Optimizador Adam con un LR de 5×10^{-5} y 64 de tamaño de lote.
- **Clasificador**: No se añade ninguna capa oculta o de normalización. Se utiliza *dropout* tradicional.

Quedan por definir los ratios del *dropout* y de los aumentos de datos, que no han convergido durante la búsqueda de hiperparámetros. Por ello, se prueban todas las posibles combinaciones de las dos siguientes configuraciones:

- Ratios de *dropout* de 0,0, 0,1 y 0,2.
- 4 niveles de aumentos de datos, cuyos ratios quedan indicados en la Tabla 7.14.

Tratándose de un modelo DeiT⚗, cada configuración se entrena una primera época sobre el conjunto de datos no etiquetado de ActualTec congelando el modelo de transferencia de aprendizaje. Posteriormente, se entrena una época más descongelando todo el modelo. Ambas épocas utilizan un LR de 10^{-3} y el optimizador Adam con un tamaño de lote de 64. Finalmente, se entrena el modelo al completo durante 20 épocas sobre la agrupación de 31 etiquetas dada por la combinación de las bases de datos de PadChest y ChestX-ray14. Como optimizador se utiliza la configuración indicada al principio de esta sección y se aplica *reduce on plateau* con paciencia 2 y factor 0.1 y sin *early stop* para poder analizar el sobreaprendizaje. Destacar que se guardan los pesos del modelo únicamente al alcanzar un nuevo AUC de validación máximo

y no después de cada época. Esta forma de entrenar permite almacenar el modelo con mejor generalización. Para cada configuración, en la Tabla 7.15, se recopilan los siguientes valores :

- El AUC de validación y entrenamiento obtenido en la primera de las 20 épocas de entrenamiento sobre la agrupación de 31 etiquetas. De esta forma, se puede analizar el efecto de los aumentos de datos y *dropout* al principio del entrenamiento.
- El número de época en la que se obtiene el mejor AUC de validación.
- El AUC de validación y entrenamiento obtenido en dicha época. Este AUC de validación es el utilizado para elegir la mejor configuración.
- El AUC de validación y entrenamiento obtenido en la última de las 20 épocas de entrenamiento, con el objetivo de estudiar el sobreaprendizaje de las diferentes configuraciones.
- Los diferentes AUC de entrenamiento y validación se indican en las columnas tr y vl.

A partir de dicha tabla se obtienen las siguientes conclusiones:

- La no aplicación de aumento de datos produce, para cualquier ratio de *dropout*, un elevado sobreaprendizaje en las últimas épocas en el conjunto de entrenamiento, obteniendo un AUC cercano a 1.
- El AUC de validación máximo oscila alrededor de 0,911 en todas las combinaciones. De esta forma, se determina que el nivel del aumento de datos y el ratio del *dropout* no afectan de forma significativa en la precisión final del modelo.
- En la primera época se observa un rendimiento ligeramente inferior al utilizar aumento de datos. Estos resultados se corresponden con los obtenidos durante la búsqueda de hiperparámetros, que sin converger del todo, se decanta por aumentos de datos bajos o directamente sin aplicar.
- Como modelo final para este trabajo se selecciona el modelo entrenado con ratio 0,1 y nivel de aumento de datos bajo, dado que tiene el mayor AUC de validación, empatado con la configuración de ratio 0,0 y aumento de datos bajo, pero con un mayor AUC de entrenamiento.

Ratio de *dropout*	Nivel de aumento	Primera época		Mejor época			Última época	
		tr	vl	Número	tr	**vl**	tr	vl
0,0	Ninguno	0,888	0,902	4	0,935	0,910	0,995	0,902
	Bajo	0,887	0,900	4	0,924	**0,912**	0,978	0,908
	Medio	0,885	0,901	6	0,935	0,911	0,983	0,906
	Alto	0,882	0,899	8	0,955	0,911	0,970	0,909
0,1	Ninguno	0,887	0,900	4	0,935	0,909	0,995	0,902
	Bajo	0,886	0,900	5	0,930	**0,912**	0,984	0,906
	Medio	0,885	0,900	4	0,922	0,911	0,973	0,909
	Alto	0,882	0,900	5	0,925	0,911	0,973	0,908
0,2	Ninguno	0,888	0,902	4	0,935	0,909	0,995	0,900
	Bajo	0,886	0,900	5	0,930	0,911	0,982	0,906
	Medio	0,885	0,899	5	0,928	0,911	0,979	0,907
	Alto	0,883	0,898	5	0,925	0,911	0,972	0,908

Tabla 7.15: Resultados de los entrenamientos completos con diferentes configuraciones de aumento de datos y ratios de dropout.

7.4. Resultados finales

Para concluir este trabajo se analiza el rendimiento del modelo sobre la partición de test en base a diversos factores:

- Se obtiene un AUC de **0,912** sobre el conjunto de test. Este es el mismo AUC obtenido sobre la partición de validación, mostrando la capacidad de generalización del sistema en muestras no analizadas.
- En la Tabla 7.16 se muestra el AUC obtenido por etiqueta con el objetivo de detectar diferentes puntos de mejora. Se observan, resaltadas en rojo, diversas etiquetas de elevada importancia como pulmón, pulmón urgente, neumonía o PDPT con un rendimiento inferior a la media del sistema. Se resaltan en negro diversas etiquetas importantes con un funcionamiento por encima de la media como cardiomegalia, derrame pleural, neumotórax o normal. Por último, se remarca que el elevado AUC obtenido en las etiquetas de cuerpos extraños, consideradas sencillas de detectar, produce una mejora artificial de la media de AUC global del sistema, teniendo en cuenta además que se cuentan dos veces en el cálculo del AUC medio del sistema.
- En la Tabla 7.17 se muestra rendimiento del sistema en las diferentes proyecciones, observándose una ventaja notable en las proyecciones AP y AP horizontal, a pesar del desbalanceo mostrado en el Capítulo 5.

- En la Tabla 7.18 se muestra el rendimiento del sistema para ambos sexos, con una ligera diferencia en favor del sexo masculino.

En este capítulo se ha mostrado la configuración de la búsqueda de hiperparámetros, así como sus resultados y conclusiones más importantes. Posteriormente, se han entrenado diversos modelos de forma completa para analizar el efecto del aumento de datos y *dropout* en la precisión de los modelos. En última instancia, se ha seleccionado y analizado uno de estos modelos sobre la partición de test, obteniendo un AUC comparable al obtenido sobre validación, demostrando así la capacidad de generalización del sistema. Además, se ha mostrado que la herramienta obtiene una precisión similar sin importar la proyección o el sexo.

Etiqueta	AUC
Global	0,912
Pulmón	0,846
Urgente	0,832
Atelectasia	0,876
Nódulos	0,854
Infiltraciones	0,877
Neumonía	0,834
Mediastino hila	0,893
Urgente	0,898
Cardiomegalia	0,935
Agrandamiento del mediastino	0,947
Alargamiento aórtico	0,936
Hila	0,872
Agrandamiento hiliar	0,862
Hernia de hiato	0,955
PDPT	0,818
Urgente	0,908
Incidental	0,865
Derrame pleural	0,934
Neumotórax	0,953
Embotamiento del ángulo costofrénico	0,936
Calcificación	0,885
Masas	0,915
Cuerpos extraños	0,978
Dispositivos eléctricos	0,997
Tubo	0,993
Tubo NSG	0,992
Catéter	0,991
Cirugía	0,956
Pseudonódulo	0,866
Subóptimo	0,932
Normal	**0,937**

Tabla 7.16: AUC por etiqueta en la agrupación de 31 etiquetas.

Proyección	AUC
PA	0,911
AP	0,917
AP horizontal	0,917

Tabla 7.17: AUC medio obtenido por proyección en la agrupación de 31 etiquetas.

Sexo	AUC
Femenino	0,912
Masculino	0,913

Tabla 7.18: AUC medio obtenido por sexo en la agrupación de 31 etiquetas.

Capítulo 8

Conclusiones

8.1. Contribuciones y resultados del trabajo

En este trabajo se han estudiado diversas técnicas de redes neuronales, desde un nivel básico, como las redes de tipo CNN o la transferencia de aprendizaje, hasta un nivel avanzado, como el paradigma S-T, los transformadores o las arquitecturas VT. Se ha desarrollado una red neuronal que obtiene un AUC de 0,912 en las particiones de validación y test sobre una tarea de clasificación de 31 etiquetas, cumpliendo los objetivos propuestos para este trabajo:

1. Se ha ampliado la clasificación por zonas anatómicas añadiendo nivel de urgencia.
2. Se ha ampliado la clasificación para detectar patologías concretas de importancia para los SUH.
3. Se ha aumentado la cantidad de datos de entrenamiento mediante el uso de las bases de datos ChestX-ray14 y ActualTec, no utilizadas en el estado previo del sistema dado por el proyecto RADIANT.
4. Se han analizado y comparado diversas arquitecturas VT, junto con redes neuronales convolucionales como DenseNet y ConvNeXt a través de una búsqueda de hiperparámetros que ha tenido en cuenta otros factores como el preprocesado, el clasificador y el optimizador.

Los resultados dados por la búsqueda de hiperparámetros, junto con el entrenamiento completo de diversos modelos han permitido desarrollar el mejor modelo posible, obteniendo diversas conclusiones importantes:

- El uso del redimensionado normal en el preprocesamiento en lugar del redimensionado inteligente planteado en el proyecto RADIANT produce una mejora de precisión significativa, especialmente en su estabilidad.
- La metodología de entrenamiento de los VT es crucial en la precisión final de la red. La arquitectura BEiT se basa en la arquitectura ViT, pero obtienen mejores resultados al aplicar entrenamiento auto supervisado incluso sobre una menor cantidad de datos. La arquitectura DeiT⚗ mejora posteriormente a BEiT incluso con menos datos y un modelo con menos parámetros gracias al uso del paradigma S-T.
- DeiT⚗-B en resolución 384 es la red idónea para la tarea planteada.
- Las redes convolucionales tradicionales obtienen resultados de AUC comparables estas nuevas arquitecturas. Por ejemplo, la red DenseNet-169 mejora a ViT *Base* y *Large* tanto en la agrupación de 8 etiquetas como en la de 31, con un menor tiempo de entrenamiento por época. Por otro lado iguala el rendimiento de SwinTV2 *Base* y *Large*.
- La red convolucional ConvNeXt-Large, mediante la aplicación de diversas técnicas propias de los transformadores y las arquitecturas VT, mejora a la red DenseNet-169 de forma significativa. De hecho, se convierte en la segunda mejor arquitectura acorde a la búsqueda de hiperparámetros, ligeramente por detrás de DeiT⚗-B.
- En la tarea en cuestión planteada, el aumento de datos o el ratio de *dropout* utilizado no afectan al mejor AUC de validación obtenido durante el entrenamiento, afectando únicamente al sobreaprendizaje del modelo durante las últimas épocas.

La contribución más importante de este trabajo final de máster ha sido la de analizar el rendimiento de diversas arquitecturas VT, comparándolos directamente a dos arquitecturas de tipo convolucional, sobre problemas de clasificación de radiografía simple. De esta forma se ha ampliado la investigación de los VT en este ámbito, en el que se han realizado pocas publicaciones dado el mayor interés generado en tareas más complejas como la segmentación de imágenes o *image captioning*.

8.2. Trabajo futuro

En base al trabajo realizado se abren diversas lineas de investigación a futuro:

- Conseguir la explicabilidad de los modelos en lugar de usarlos como una caja negra, lo que puede resultar crucial en un entorno de urgencias. Para ello, hay diversas técnicas que se dedican a extraer mapas de activación de las diferentes etiquetas en una imagen

de entrada. Aplicar alguna de estas técnicas, como Grad-CAM, es necesario no solo para conseguir explicabilidad, sino también para analizar si una red neuronal se fija en la zona de imagen adecuada al realizar las predicciones. Diversos trabajos [38, 39] muestran como la red neuronal aprende un cierto bias del conjunto de datos para obtener buenas predicciones, en lugar de analizar la propias zonas anatómicas de una radiografía.

- Utilizar las arquitecturas VT en tareas de visión por computador más complejas como *image captioning*, que permitiría la redacción automática de borradores de informes. De esta forma, se pueden utilizar conjuntos de datos no etiquetados puesto que solo se necesita la radiografía y el informe. Otro posible campo de investigación es la segmentación de radiografías, que permitiría localizar directamente las diferentes patologías de una radiografía sin necesidad de utilizar de mapas de activación.

- Realizar un estudio de ablación para analizar el efecto en la precisión de validación con diferentes cantidades de datos de entrenamiento. En concreto, se podría analizar también el efecto de la cantidad de datos de ActualTec utilizado durante el entrenamiento no etiquetado de los modelos DeiT⚗.

- Analizar los porcentajes de falsos positivos y falsos negativos producidos por la red neuronal sobre la partición de validación y test. Así como la el cálculo del límite a partir del cual se considera que la etiqueta es positiva. Este análisis es de elevada importancia especialmente en una aplicación médica, que debe reducir lo máximo posible la clasificación de pacientes patológicos como sanos.

- Estudiar diferentes redes como profesores de la arquitectura DeiT⚗, dada la mejor precisión obtenida por las redes ConvNeXt-L y DenseNet-169 respecto a la DenseNet-161 utilizada como profesor durante el trabajo. Se podría analizar también el uso de un *ensemble* como profesor.

- Preentrenar un modelo DeiT⚗ propio sobre un ImageNet en escala de grises en lugar de en formato RGB. De esta forma, el modelo se encuentra diseñado para trabajar en dicho formato, mejorando las prestaciones, tal y como se muestra en [40].

- Estudiar nuevas métricas que tengan en cuenta la jerarquía de las etiquetas para evitar que se cuenten varias veces en el cálculo medio final de todas las etiquetas.

Estos puntos tienen el potencial de desarrollar herramientas más útiles, ofreciendo explicabilidad y borradores de informes, y más precisas mediante el estudio de diferentes configuraciones de entrenamiento. En conjunto, estas mejoras pueden ser cruciales en el entorno de urgencias, optimizando los procesos de triaje de forma significativa más allá del trabajo realizado en esta memoria.

Bibliografía

[1] Pablo Valdés Solís, Ángel Morales Santos, Isabel Gonzalez Álvarez, and Carmen Martínez Serrano. El informe de la radiología simple. algo más que un imperativo legal. *Radiología (Madr., Ed. impr.)*, 55(4):279–282, 2013.

[2] Ángel Morales Santos and José Maria Artigas Martín. Organización y gestión de la radiología urgente. *Radiología*, 53:7–15, 2011.

[3] Ángel Morales Santos. El informe en radiología de urgencias ¿podemos excluir la radiografía simple?, 2013.

[4] Pablo Valdés Solís, Carmen Martínez Serrano, Mariana Rovira Cañellas, Antonio Fernando Fernández Alarza, Luis Concepcion Aramendia, Alfonsa Friera Reyes, Lola Esteba Bech De Careda, Juan Arrazola García, and Ángeles Franco López. Las cargas de trabajo en radiología, 2020.

[5] Aurelia Bustos, Antonio Pertusa, Jose-Maria Salinas, and Maria de la Iglesia-Vayá. Padchest: A large chest x-ray image dataset with multi-label annotated reports. *Medical Image Analysis*, 66:101797, 2020.

[6] Xiaosong Wang, Yifan Peng, Le Lu, Zhiyong Lu, Mohammadhadi Bagheri, and Ronald M. Summers. ChestX-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, jul 2017.

[7] Student notes: Convolutional neural networks (cnn) introduction. `https://indoml.com/2018/03/07/student-notes-convolutional-neural-networks-cnn-introduction`.

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805, 2019.

[9] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, Dec 2015.

[10] Tal Ridnik, Emanuel Ben-Baruch, Asaf Noy, and Lihi Zelnik-Manor. Imagenet-21k pre-training for the masses, 2021.

[11] Zhuliang Yao, Yue Cao, Yutong Lin, Ze Liu, Zheng Zhang, and Han Hu. Leveraging batch normalization for vision transformers. In *2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pages 413–422, 2021.

[12] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, jan 2014.

[13] Lin Wang and Kuk-Jin Yoon. Knowledge distillation and student-teacher learning for visual intelligence: A review and new outlooks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(6):3048–3068, 2022.

[14] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers & distillation through attention. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 10347–10357. PMLR, 18–24 Jul 2021.

[15] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[16] Bo-Kai Ruan, Hong-Han Shuai, and Wen-Huang Cheng. Vision transformers: State of the art and research challenges, 2022.

[17] Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers, 2021.

[18] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. Beit: Bert pre-training of image transformers, 2021.

[19] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows, 2021.

[20] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, Furu Wei, and Baining Guo. Swin transformer v2: Scaling up capacity and resolution, 2021.

[21] Jeremy Irvin, Pranav Rajpurkar, Michael Ko, Yifan Yu, Silviana Ciurea-Ilcus, Chris Chute, Henrik Marklund, Behzad Haghgoo, Robyn Ball, Katie Shpanskaya, Jayne Seekins, David A. Mong, Safwan S. Halabi, Jesse K. Sandberg, Ricky Jones, David B. Larson, Curtis P. Langlotz, Bhavik N. Patel, Matthew P. Lungren, and Andrew Y. Ng. Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison, 2019.

[22] Li Yao, Eric Poblenz, Dmitry Dagunts, Ben Covington, Devon Bernard, and Kevin Lyman. Learning to diagnose from scratch by exploiting dependencies among labels, 2017.

[23] Pranav Rajpurkar, Jeremy Irvin, Kaylie Zhu, Brandon Yang, Hershel Mehta, Tony Duan, Daisy Ding, Aarti Bagul, Curtis Langlotz, Katie Shpanskaya, Matthew P. Lungren, and Andrew Y. Ng. Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning, 2017.

[24] Sebastian Guendel, Sasa Grbic, Bogdan Georgescu, Kevin Zhou, Ludwig Ritschl, Andreas Meier, and Dorin Comaniciu. Learning to recognize abnormalities in chest x-rays with location-aware dense networks, 2018.

[25] Zhuoning Yuan, Yan Yan, Milan Sonka, and Tianbao Yang. Large-scale robust deep auc maximization: A new surrogate loss and empirical studies on medical image classification, 2020.

[26] Hieu H. Pham, Tung T. Le, Dat Q. Tran, Dat T. Ngo, and Ha Q. Nguyen. Interpreting chest x-rays via cnns that exploit hierarchical disease dependencies and uncertainty labels, 2019.

[27] Maria de la Iglesia Vayá, Jose Manuel Saborit, Joaquim Angel Montell, Antonio Pertusa, Aurelia Bustos, Miguel Cazorla, Joaquin Galant, Xavier Barber, Domingo Orozco-Beltrán, Francisco García-García, Marisa Caparrós, Germán González, and Jose María Salinas. Bimcv covid-19+: a large annotated dataset of rx and ct images from covid-19 patients, 2020.

[28] Daniel Arias-Garzón, Jesús Alejandro Alzate-Grisales, Simon Orozco-Arias, Harold Brayan Arteaga-Arteaga, Mario Alejandro Bravo-Ortiz, Alejandro Mora-Rubio, Jose Manuel Saborit-Torres, Joaquim Ángel Montell Serrano, Maria de la Iglesia Vayá, Oscar Cardona-Morales, and Reinel Tabares-Soto. Covid-19 detection in x-ray images using convolutional neural networks. *Machine Learning with Applications*, 6:100138, 2021.

[29] Mizuho Nishio, Daigo Kobayashi, Eiko Nishioka, Hidetoshi Matsuo, Yasuyo Urase, Koji Onoue, Reiichi Ishikura, Yuri Kitamura, Eiro Sakai, Masaru Tomita, Akihiro Hamanaka, and Takamichi Murakami. Deep learning model for the automatic classification of covid-19 pneumonia, non-covid-19 pneumonia, and the healthy: a multi-center retrospective study. *Scientific Reports*, 12(1):8214, May 2022.

[30] Alexander Selivanov, Oleg Y. Rogov, Daniil Chesakov, Artem Shelmanov, Irina Fedulova, and Dmitry V. Dylov. Medical image captioning via generative pretrained transformers, 2022.

[31] Edward Vendrow and Ethan Schonfeld. Understanding transfer learning for chest radiograph clinical report generation with modified transformer architectures, 2022.

[32] Debaditya Shome, T Kar, Sachi Nandan Mohanty, Prayag Tiwari, Khan Muhammad, Abdullah AlTameem, Yazhou Zhang, and Abdul Khader Jilani Saudagar. COVID-transformer: Interpretable COVID-19 detection using vision transformer for healthcare. *Int. J. Environ. Res. Public Health*, 18(21):11086, October 2021.

[33] Sangjoon Park, Gwanghyun Kim, Yujin Oh, Joon Beom Seo, Sang Min Lee, Jin Hwan Kim, Sungjun Moon, Jae-Kwang Lim, and Jong Chul Ye. Vision transformer for covid-19 cxr diagnosis using chest x-ray feature corpus, 2021.

[34] Christos Matsoukas, Johan Fredin Haslum, Magnus Söderberg, and Kevin Smith. Is it time to replace cnns with transformers for medical images?, 2021.

[35] Fahad Shamshad, Salman Khan, Syed Waqas Zamir, Muhammad Haris Khan, Munawar Hayat, Fahad Shahbaz Khan, and Huazhu Fu. Transformers in medical imaging: A survey, 2022.

[36] Sagar Vaze, Kai Han, Andrea Vedaldi, and Andrew Zisserman. Open-set recognition: a good closed-set classifier is all you need?, 2021.

[37] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s, 2022.

[38] Alex J. DeGrave, Joseph D. Janizek, and Su-In Lee. Ai for radiographic covid-19 detection selects shortcuts over signal. *Nature Machine Intelligence*, 3(7):610–619, Jul 2021.

[39] Jennifer Dhont, Cecile Wolfs, and Frank Verhaegen. Automatic coronavirus disease 2019 diagnosis based on chest radiography and deep learning – success story or dataset bias? *Medical Physics*, 49(2):978–987, 2022.

[40] Yiting Xie and David Richmond. Pre-training on grayscale imagenet improves medical image classification. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, September 2018.