UANL
UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN

FCFM
FACULTAD DE CIENCIAS FÍSICO MATEMÁTICAS

Modelo de visión computacional para Clasificación Multi-Etiqueta para Chest X-Ray Dataset - DenseNet

José Manuel Enríquez Rodríguez 2132982

Carlos Eduardo García Herrera 2133000

Martin Alexis Martínez Andrade 2049334

Daniel Rojas Villareal 2132983

Diego Adrián Moreno Duarte 2132863

Grupo 31 – Inteligencia Artificial

Maestro: Luis Ángel Gutiérrez Rodríguez

San Nicolas de Los Garza Mayo 2025

Tabla de Contenido

Summary

En el presente informe se describe el desarrollo y evaluación de un modelo de aprendizaje profundo orientado a la clasificación multi-etiqueta de imágenes médicas, específicamente radiografías de tórax del dataset NIH Chest X-Ray. Para la extracción de características se utiliza DenseNet121, una arquitectura de red convolucional profunda que ha demostrado un alto rendimiento en tareas de visión por computadora, sirviendo como backbone del modelo.

La arquitectura propuesta está compuesta por dos módulos principales: una **Feature Pyramid Network (FPN)**, encargada de capturar información a múltiples escalas para mejorar la representación espacial de las características extraídas; y un **Classification Wrapper personalizado**, diseñado para adaptar las salidas de la FPN al espacio de etiquetas correspondiente a las patologías presentes en el conjunto de datos.

El informe detalla cada etapa del flujo de trabajo, incluyendo el preprocesamiento de los datos de entrada, la configuración del entorno de entrenamiento en **PyTorch**, la selección de métricas de evaluación apropiadas para problemas de clasificación multi-etiqueta, y el análisis de los resultados obtenidos. Se discuten también las decisiones arquitectónicas tomadas, los hiperparámetros utilizados y las observaciones más relevantes derivadas del desempeño del modelo durante el entrenamiento y la validación.

Los resultados obtenidos evidencian la eficacia de la arquitectura propuesta al abordar el problema de detección simultánea de múltiples condiciones médicas en imágenes radiográficas, destacando el potencial del enfoque empleado para su futura aplicación en entornos clínicos asistidos por inteligencia artificial.

Palabras Clave: Deep Learning, Multi-modal Learning, Feature Pyramid Network (FPN), DenseNet121, Tabular and Image Fusion, Medical Image Classification, Multilabel Classification, Neural Network, Convolutional Neural Network (CNN), Feature Extraction, Late Fusion, Adaptive Pooling, PyTorch, Transfer Learning, Hybrid Model, Clinical Decision Support

Introducción

En las últimas décadas, el campo de la salud se ha beneficiado significativamente de los avances en el cómputo y la inteligencia artificial. Desde la visualización avanzada de imágenes médicas hasta el uso de algoritmos capaces de asistir en diagnósticos complejos, la tecnología ha transformado la manera en que se aborda el cuidado médico. En particular, la inteligencia artificial (IA) aplicada a la visión por computadora ha demostrado un enorme potencial en la detección automática de anomalías en imágenes médicas, ayudando a los profesionales de la salud a mejorar la precisión y eficiencia en sus evaluaciones.

Una de las áreas críticas dentro del diagnóstico médico es la interpretación de radiografías de tórax, que permite detectar una amplia variedad de condiciones como neumonía, edema pulmonar, cardiomegalia, nódulos, entre otras. Debido al gran volumen de radiografías que deben ser revisadas por radiólogos, surge la necesidad de desarrollar sistemas automatizados que puedan asistir en esta tarea de forma rápida y confiable.

En este contexto, las redes neuronales convolucionales (CNNs) han sido ampliamente utilizadas para tareas de clasificación de imágenes médicas. Entre ellas, DenseNet121 se destaca como un backbone robusto y eficiente, capaz de capturar representaciones profundas gracias a su arquitectura densa, donde cada capa recibe como entrada las salidas de todas las capas anteriores. Esta característica permite una mejor propagación de características y gradientes, mejorando la eficiencia del modelo en tareas de clasificación compleja.

El presente proyecto propone un modelo basado en DenseNet121 para la tarea de clasificación multi-etiqueta de enfermedades torácicas a partir de radiografías del conjunto de datos NIH Chest X-Ray. La arquitectura se complementa con una Feature Pyramid Network (FPN) para capturar información en múltiples escalas y un Classification Wrapper personalizado para adaptar la salida del modelo a la estructura de etiquetas múltiples del problema. A diferencia de los modelos de detección de objetos como YOLO, que identifican y localizan objetos específicos en una imagen, este enfoque se enfoca en predecir simultáneamente la presencia de múltiples patologías en una misma radiografía.

Este informe detalla el diseño del modelo, el proceso de preprocesamiento de los datos, el procedimiento de entrenamiento, y la evaluación del desempeño en la tarea de clasificación. Además, se analiza la efectividad del modelo propuesto y su posible aplicabilidad en entornos clínicos asistidos por IA.

Hipótesis

Justificación

El presente proyecto representa una contribución significativa al sector salud, especialmente en el área de diagnóstico médico por imágenes. La incorporación de modelos de inteligencia artificial en la práctica clínica tiene el potencial de transformar la manera en que los profesionales de la salud analizan e interpretan radiografías, al proporcionar una herramienta de apoyo que optimiza tanto la precisión como la eficiencia en la detección de patologías.

En primer lugar, el uso de un modelo de clasificación multi-etiqueta basado en técnicas avanzadas de visión por computadora, como **DenseNet121** y **Feature Pyramid Network**, permite identificar múltiples anomalías en una sola imagen de tórax. Esta capacidad es especialmente útil en un entorno clínico donde un paciente puede presentar varias condiciones simultáneamente. De esta forma, el modelo no solo actúa como un filtro preliminar que destaca posibles áreas de interés, sino que también reduce el riesgo de omisión de enfermedades por parte del personal médico debido a la fatiga o sobrecarga laboral.

En segundo lugar, el sistema propuesto puede servir como una herramienta complementaria para médicos con experiencia, facilitando la priorización de casos urgentes y ayudando en la validación de diagnósticos. Su capacidad para proporcionar predicciones rápidas y confiables mejora el flujo de trabajo en instituciones médicas, permitiendo dedicar más tiempo a la atención directa del paciente. Además, en contextos donde el número de especialistas es limitado, como en hospitales rurales o regiones con baja cobertura médica, esta tecnología puede ser clave para garantizar diagnósticos más oportunos.

Por otro lado, el modelo también tiene un valor formativo. Para médicos en formación o en sus primeros años de práctica profesional, disponer de una herramienta basada en inteligencia artificial puede mejorar significativamente su proceso de aprendizaje. Al contrastar sus propias observaciones con las predicciones del modelo, los estudiantes pueden desarrollar una mejor comprensión de los patrones visuales asociados a distintas enfermedades, reforzando su capacidad diagnóstica de forma asistida.

Finalmente, la implementación de este tipo de soluciones contribuye a sentar las bases para un ecosistema de salud más automatizado, accesible y resiliente. La combinación de experiencia médica e inteligencia artificial permite avanzar hacia una medicina más precisa, preventiva y personalizada, donde el tiempo de respuesta y la calidad del diagnóstico se ven beneficiados por el uso estratégico de tecnologías emergentes.

Así como también, a nosotros como estudiantes de Ciencias Computacionales nos sirve de gran ayuda para implementar los aprendizajes vistos a lo largo del curso de Inteligencia Artificial

Resultados Esperados

El objetivo principal de este proyecto es desarrollar y entrenar un modelo de clasificación multi-etiqueta utilizando DenseNet121 como backbone principal para la extracción de características, con el fin de detectar múltiples patologías torácicas a partir de radiografías del conjunto de datos NIH Chest X-Ray. Para mejorar la capacidad de representación del modelo, se ha incorporado una Feature Pyramid Network (FPN), que permite integrar información en diferentes escalas, y un Classification Wrapper personalizado, que adapta las salidas del modelo a la naturaleza multi-etiqueta del problema.

Este enfoque busca lograr un desempeño competitivo en métricas relevantes para clasificación médica, tales como la precisión, recall, F1-score y Area Under the ROC Curve (AUC) para cada etiqueta. Se espera que el modelo pueda generalizar adecuadamente sobre imágenes no vistas, manteniendo un equilibrio entre sensibilidad y especificidad, crucial en aplicaciones clínicas donde los falsos negativos pueden tener consecuencias importantes.

Además del rendimiento técnico, uno de los objetivos clave es garantizar que el modelo resultante sea fácilmente reutilizable y extensible por otros investigadores, profesionales de la salud o estudiantes interesados en mejorar sus capacidades. Por esta razón, el proyecto ha sido estructurado con un enfoque modular y reproducible, facilitando el ajuste de hiperparámetros, la integración de nuevos conjuntos de datos o la incorporación de mejoras arquitectónicas.

Se espera que, como resultado de este trabajo, se obtenga un modelo funcional que no solo sirva como prueba de concepto para la clasificación automatizada de enfermedades torácicas, sino también como una base sólida para investigaciones futuras. Al proporcionar una implementación abierta y documentada, el modelo podrá ser reutilizado y refinado en distintos contextos clínicos o educativos, adaptándose a diferentes poblaciones o necesidades específicas.

Metodología

A continuación, se presenta toda la metodología seguida a lo largo del proyecto, se especifica el proceso de desarrollo, investigación, así como también la justificación de cada una de las decisiones tomadas en el desarrollo del modelo.

Dataset: NIH-Chest-X-Ray

El dataset se obtuvo directamente del repositorio de Hugging Face, a continuación, se describe información relevante del Dataset

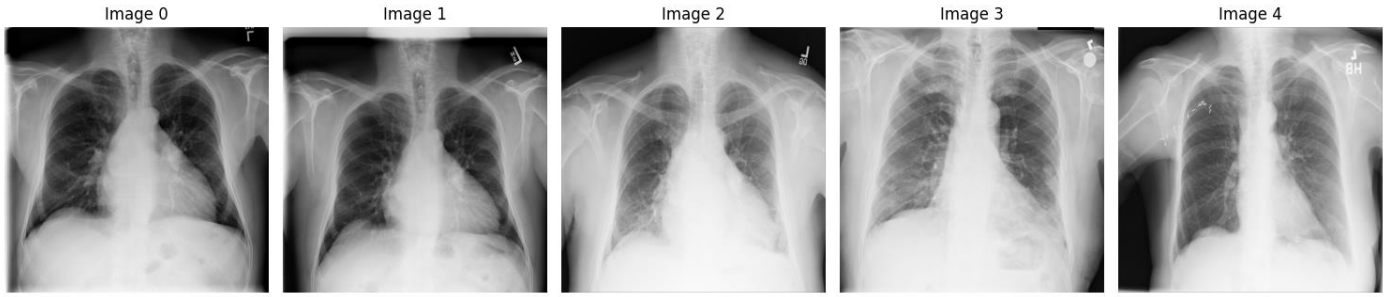


Figure 1: Imágenes de Muestra contenidas en NIH-Chest-X-ray-dataset

Descripción del dataset

El dataset ChestX-ray8 contiene 112,120 imágenes de radiografías de tórax en vista frontal, correspondientes a 30,805 pacientes únicos. Cada imagen está etiquetada con hasta doce posibles patologías torácicas, extraídas automáticamente de los informes radiológicos mediante técnicas de procesamiento de lenguaje natural. Las imágenes pueden tener múltiples etiquetas por muestra, lo que permite representar la presencia simultánea de varias enfermedades en una sola radiografía.

Contenido

- 112,120 imágenes de rayos X de tórax en vista frontal en formato PNG con resolución de 1024×1024 (dentro de la carpeta images).
- Metadatos de todas las imágenes (Data_Entry_2017.csv): índice de imagen, etiquetas de hallazgos, número de seguimiento, ID del paciente, edad del paciente, sexo del paciente, posición de la vista, tamaño original de la imagen y espaciado original de los píxeles.
- Cajas delimitadoras (bounding boxes) para aproximadamente 1000 imágenes (BBox_List_2017.csv): índice de imagen, etiqueta del hallazgo, caja delimitadora [x, y, ancho, alto]. [x y] son las coordenadas de la esquina superior izquierda de cada caja. [ancho alto] representan el ancho y la altura de cada caja.
- Dos archivos de partición de datos (train_val_list.txt y test_list.txt) están disponibles. Las imágenes del conjunto de datos ChestX-ray se dividen en estos dos conjuntos a nivel de paciente. Todos los estudios de un mismo paciente aparecerán solo en el conjunto de entrenamiento/validación o en el conjunto de prueba, pero no en ambos.

Etiquetado

Cada uno de los registros del Dataset puede ser clasificado en las siguientes enfermedades:

- Atelectasis - Atelectasia
- Consolidation - Consolidación
- Infiltration - Infiltración
- Pneumothorax - Neumotórax
- Edema - Edema

- Emphysema - Enfisema
- Fibrosis - Fibrosis
- Effusion - Derrame
- Pneumonia - Neumonía
- Pleural Thickening - Engrosamiento pleural
- Cardiomegaly - Cardiomegalia
- Nodule - Nódulo
- Mass - Masa
- Hernia - Hernia
- No Findings - Sano

Distribución de Etiquetas

A continuación, se muestra un diagrama circular que muestra la proporción del etiquetado de las imágenes que cuentan con alguna de las enfermedades, ya que como se mencionó anteriormente, una misma radiografía, puede contar con diferentes enfermedades etiquetadas.

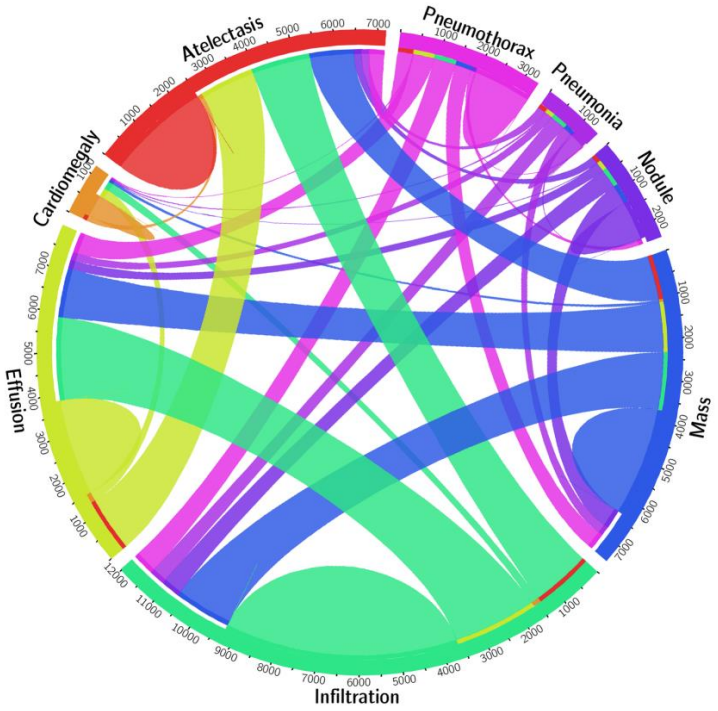


Figure 2:Distribución multi-etiqueta de las Enfermedades en NIH-Chest-X-ray-dataset

Así mismo, se presenta una tabla de frecuencias para cada una de las enfermedades presentes:

Etiqueta	Cantidad	Frecuencia	Etiqueta	Cantidad	Frecuencia
Sin hallazgos	60,361	0.426468	**Engrosamiento pleural**	3,385	0.023916
Infiltración	19,894	0.140557	**Cardiomegalia**	2,776	0.0196132
Derrame	13,317	0.0940885	**Enfisema**	2,516	0.0177763
Atelectasia	11,559	0.0816677	**Edema**	2,303	0.0162714
Nódulo	6,331	0.0447304	**Fibrosis**	1,686	0.0119121
Masa	5,782	0.0408515	**Neumonía**	1,431	0.0101104
Neumotórax	5,302	0.0374602	**Hernia**	227	0.00160382

Consolidación	4,667	0.0329737	-	-	-

Tabla 1: Tabla de Frecuencia de las etiquetas presentes en NIH-Chest-X-ray-dataset

Y esta información se muestra de forma gráfica en la siguiente figura:

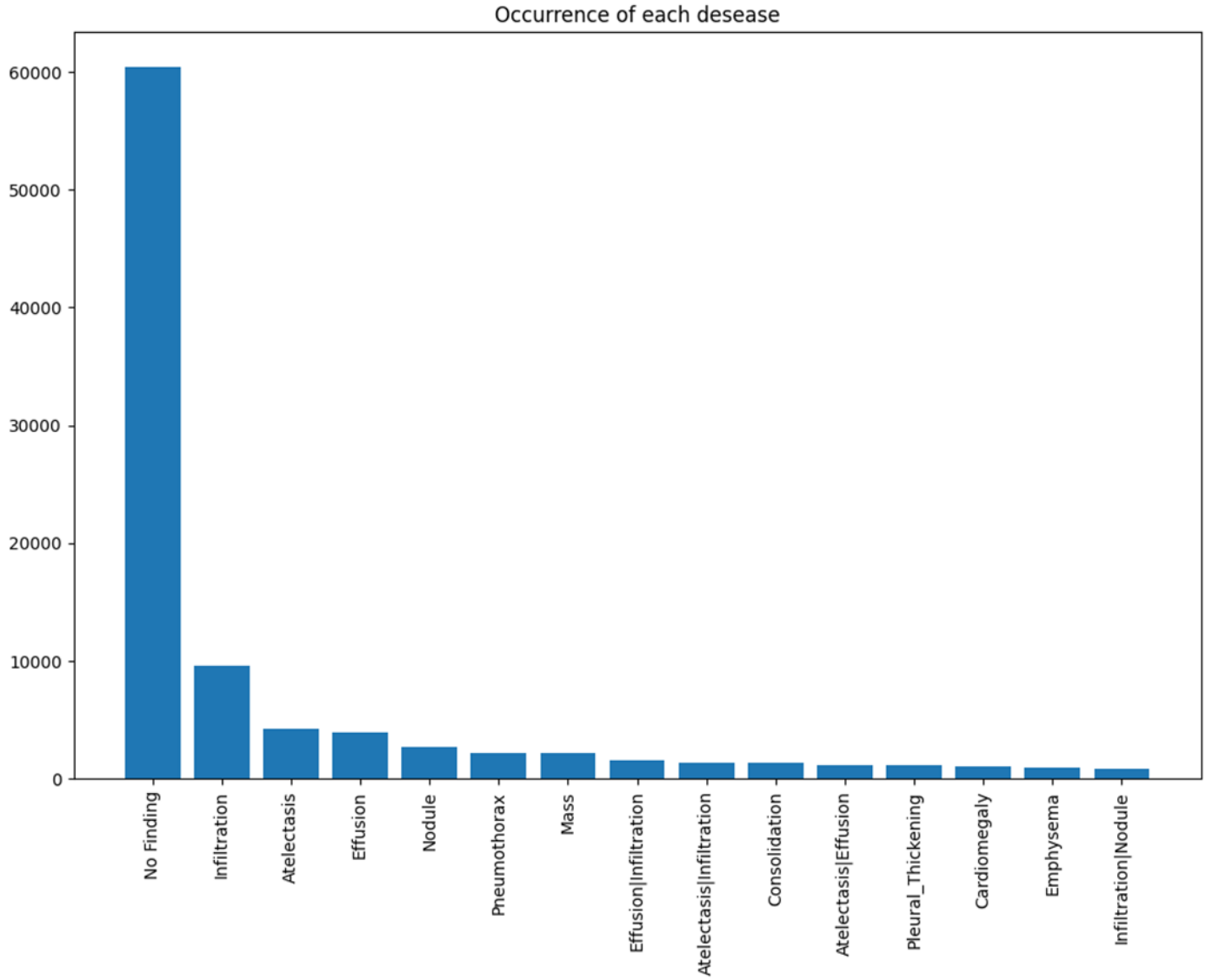


Figure 3: Distribución de las Enfermedades en NIH-Chest-X-ray-dataset

información de los Pacientes

Se generaron representaciones gráficas que resumen información demográfica contenida en el conjunto de datos, específicamente la distribución de edad de los pacientes y la proporción entre géneros. Estas visualizaciones permiten comprender mejor la composición del dataset y evaluar posibles sesgos en los datos.

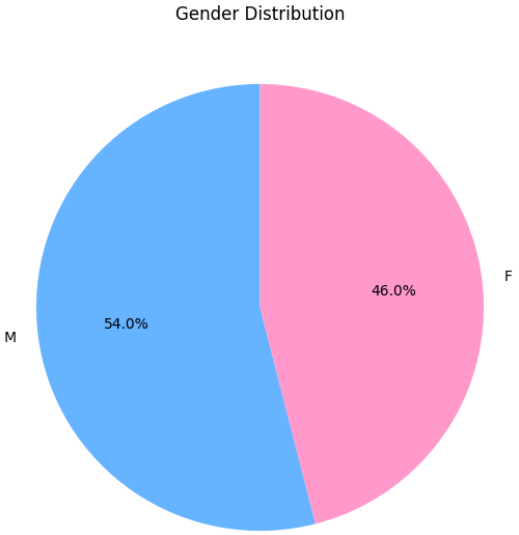


Figure 4: Proporción de Genero en NIH-Chest-X-ray-dataset

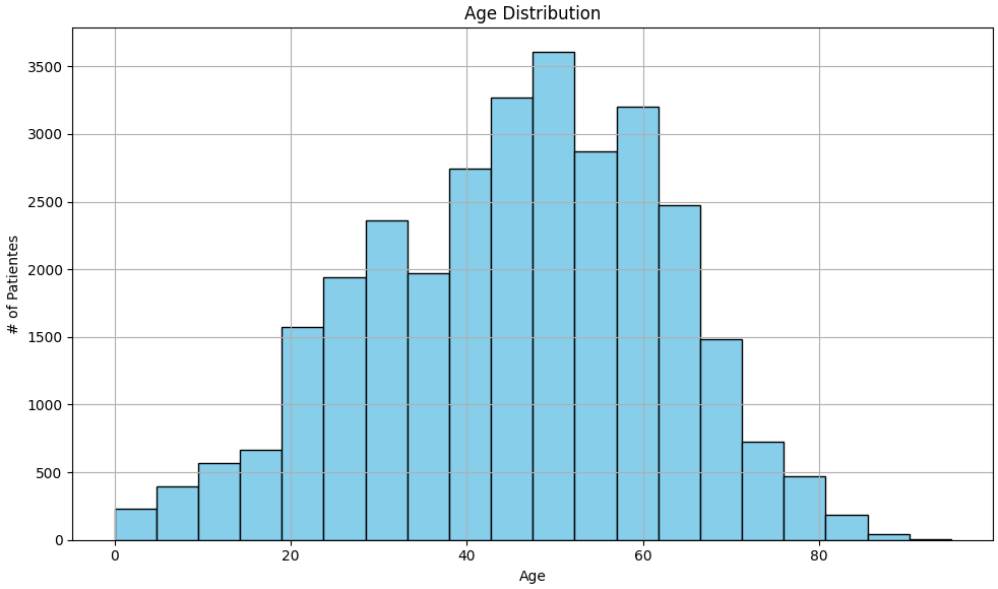


Figure 5: Distribución de Edades en NIH-Chest-X-ray-dataset

Esta información es clave para nuestro modelo, ya que como se verá en la sección de Arquitectura de la red Neuronal, se recibe como datos de entrada información del paciente

Entorno de Desarrollo

A continuación, se describe el entorno de desarrollo usado a lo largo del proyecto, se especifica información relevante relacionada con la base sobre la que fue desarrollado el proyecto.

Entorno de Ejecución

El entorno de ejecución seleccionado fue Google Collab, ya que ofrece recursos de gran potencia para el entrenamiento de modelos de Machine Learning, tanto en su versión gratuita como en la versión Pro. Se contrató la suscripción de **Google Collab Pro**, para poder acceder a hardware más potente y tener sesiones de ejecución más largas.

El notebook final lo podemos encontrar en el siguiente link, así como también, en el apartado de Anexos

Especificaciones Técnicas

El hardware que proporciona Google Collab y el usado cuenta con las siguientes características:

- **CPU**: Intel(R) Xeon(R) CPU @ 2.20GHz (6 núcleos / 12 Hilos)
- **RAM**: 53 GB
- **GPU**: NVIDIA L40 (22.5 GB)
- **Almacenamiento**: 235.7 GB

Framework de Machine Learning

Como se mencionó en secciones anteriores, se opto por usar el framework de Pytorch para el desarrollo, creación y entrenamiento del modelo de Machine Learning, así como también, se enlistan las librerías que se usaron a lo largo del proceso de desarrollo.

Librerías

Las librerías que se utilizaron durante el desarrollo del modelo son las siguientes:

Procesamiento de datos

- Os
- Re
- Ast
- Time
- Sleep
- Timedelta
- Path
- Zipfile
- Csv
- Glob
- Shutil
- Json
- Numpy
- Pandas
- Sklearn.utils.resample
- Sklearn.model_selection.train_test_split
- Sklearn.metrics.

Imágenes y visualización

- Cv2
- PIL.image
- Torchvision.transforms.v2
- Torchvision.transforms.functional
- Matplotlib.pyplot

Red neuronal / Machine Learning

- Torch
- Torch.nn
- Torch.nn.functional
- Torch.utils.data.Dataset
- Torch.utils.data.Dataloader
- Torch.utils.data.Subset
- Torch.utils.data.random_split
- Huggingface_hub.hf_hub_download

Preprocesamiento de los Datos

El proceso de preprocesamiento de los datos se explica en las siguientes secciones, todo este preprocesamiento es de gran importancia, ya que nos permite filtrar, mejorar y aumentar la calidad de los datos que recibirá nuestro modelo, entre mejor sea la calidad de los datos, el modelo podrá aprender patrones de forma más sencilla.

Hot One Encoding de las enfermedades

Para cada categoría diagnóstica, se crean 14 nuevas columnas binarias que indica con un valor de 1 la presencia de la enfermedad N en la imagen correspondiente, o un 0 en caso contrario. De este modo, cada imagen queda representada por un vector de indicadores que resumen sus características clínicas relevantes.

Out[17]:

spacing[x	...	Atelectasis	Pneumothorax	Fibrosis	Edema	Cardiomegaly	Consolidation	No Finding	Infiltration	Pleural_Thickening	Hernia
0.168	...	0	0	0	0	0	0	1	0	0	0
0.143	...	0	0	0	0	0	0	0	1	0	0
0.168	...	0	0	0	0	0	0	0	0	0	0

Figure 6: Hot One Encoding de Finding Labels

Undersampling

No Findings

Como se evidenció en la sección de Distribución de Etiquetas, el conjunto de datos presenta un desbalance significativo, con una mayor proporción de pacientes sin hallazgos patológicos en comparación con aquellos que presentan alguna enfermedad. Este desequilibrio puede inducir al modelo a favorecer la predicción de casos sanos, comprometiendo su capacidad para detectar condiciones clínicas relevantes. Por ello, es necesario aplicar técnicas de balanceo o reestructuración del conjunto de datos con el fin de mitigar este sesgo y mejorar el desempeño del modelo en la clasificación de las clases minoritarias. Para realizar esto solo nos vamos a quedar solo con el 10% de las personas sanas, así que ahora la distribución se ve de la siguiente manera.

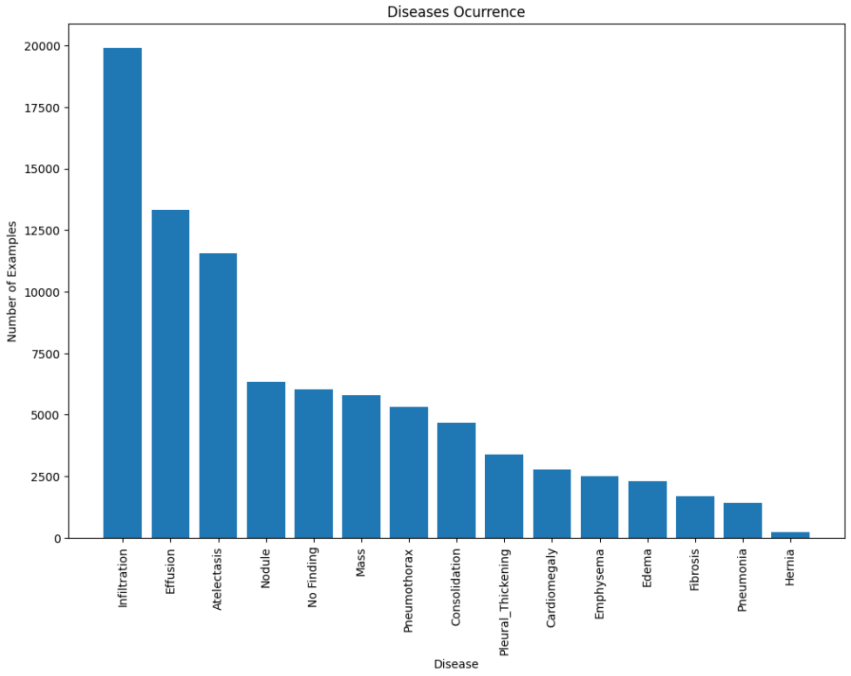


Figure 7: Distribución después del Undersampling de No Findings

Infiltration

Se realiza el mismo proceso para la enfermedad de Infiltración, ya que cuenta con una gran cantidad de observaciones, en este caso, se seleccionó solamente el 15% de los casos únicos de dicha enfermedad, es decir, no se eliminaron registros en donde se encontraba infiltración junto con alguna otra enfermedad

Effusion

Mismo proceso que el anterior, solamente que, en este caso, se seleccionaron solamente el 50% de los casos únicos de Effusion

Atelectasis

Ya, por último, se selecciono el 50% de los casos únicos de Atelectasis

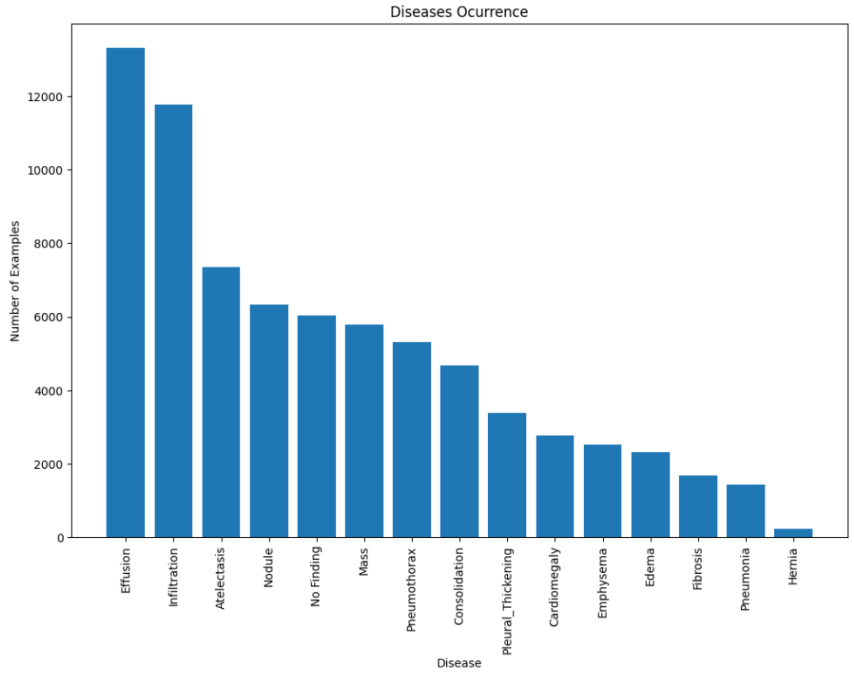


Figure 8: Distribución Final de las Etiquetas en el Dataset

Creación de la columna disease_vec

Posteriormente, de obtener una muestra balanceada del conjunto de datos, se incorpora una nueva columna denominada "disease_vec". Esta columna contiene, para cada imagen, un vector binario que representa la presencia (1) o ausencia (0) de cada una de las posibles condiciones médicas.

La construcción de dicho vector se realiza a partir de las columnas correspondientes a las etiquetas previamente generadas, las cuales contienen valores binarios. Estas se agrupan en un arreglo que permite representar, de forma compacta y estructurada, el conjunto de enfermedades asociadas a cada imagen. Este formato es especialmente útil para el entrenamiento de modelos de clasificación multi-etiqueta, ya que permite capturar múltiples condiciones simultáneamente en una misma instancia.

Esta columna representará la salida del modelo, o también conocida como Y, ya que se trata de los valores a predecir para cada una de las imágenes que reciba el modelo

Información del Paciente

El modelo recibirá los datos del paciente, pero es necesario realizar las siguientes conversiones:

Edad

Se procede a normalizar la variable correspondiente a la edad de los pacientes, escalándola al rango [0, 1] para garantizar una contribución uniforme durante el entrenamiento del modelo.

Tipo de Radiografía

La posición de la vista radiográfica (View Position) se codifica de la siguiente forma:

- 0: proyecciones posteroanteriores (PA).
- 1: proyecciones anteroposteriores (AP).

Sexo

El sexo se representa de la siguiente manera:

- 0: sexo masculino (Male)
- 1: sexo femenino (Female)

Toda esta información demográfica y técnica se transforma posteriormente en otro vector denominado **"patient_info"** el cual, se dará como dato de entrada al modelo.

División del Dataset

En esta etapa, el conjunto de datos balanceado se divide en tres subconjuntos:

- **Conjunto de entrenamiento (train_df):** 72% del conjunto de datos completo
- **Conjunto de validación (validation_df):** 8% del conjunto de datos completo (equivalente al 10% del conjunto de entrenamiento)
- **Conjunto de prueba (test_df):** 20% del conjunto de datos completo

Para garantizar que la distribución de las etiquetas se mantenga de manera consistente en cada partición, se utiliza la opción **stratify**, la cual agrupa las muestras en función de las primeras letras de las etiquetas diagnósticas. Esto asegura que las proporciones de clases no se vean alteradas significativamente en el subconjunto de prueba, lo que es crucial para obtener una evaluación justa del desempeño del modelo.

Pre-procesado para train_df

Se aplicó una segunda etapa de limpieza exclusivamente sobre el subconjunto de entrenamiento (train_df), con el objetivo de evitar cualquier tipo de fuga de datos (data leakage) hacia los conjuntos de validación o prueba.

En primer lugar, se duplicaron las imágenes correspondientes a las enfermedades con menos de 3000 ocurrencias mediante una transformación por simetría vertical. Esta técnica permite aumentar la representación de dichas clases sin introducir nuevas instancias externas al conjunto.

Seguidamente, para aquellas enfermedades cuya frecuencia es inferior a 1500 muestras, se replicaron las instancias necesarias hasta alcanzar dicho umbral, con el fin de reducir el desbalance entre clases.

Por último, dado que las categorías "No Finding" y "Atelectasis" siguen estando sobrerrepresentadas, se optó por submuestrearlas, conservando únicamente el 25% de los casos correspondientes a "No Finding" y el 50% de los asociados a "Atelectasis".

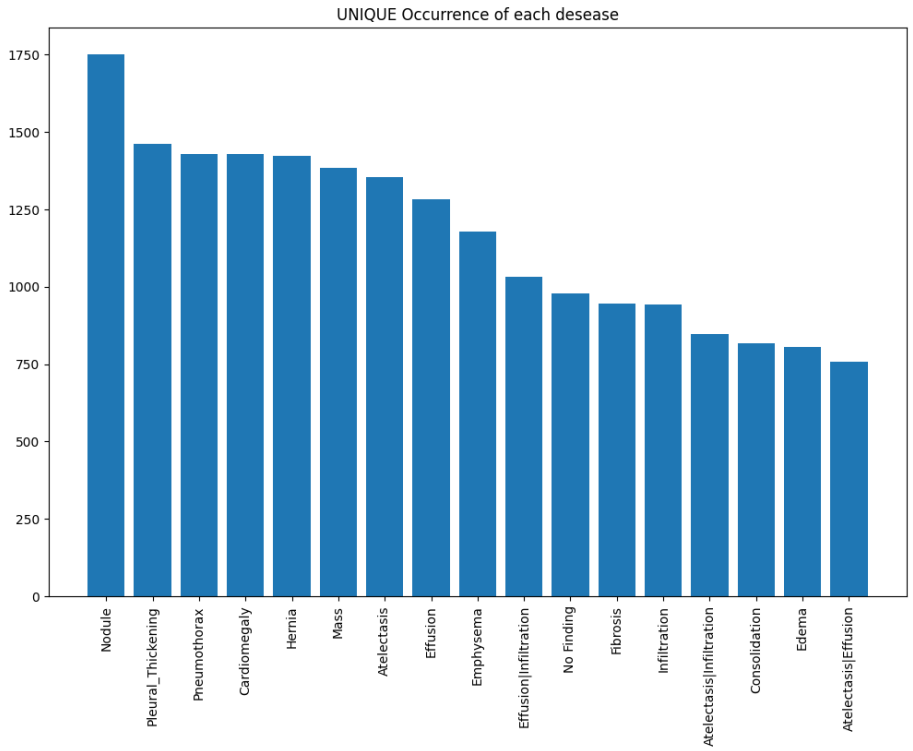


Figure 9: Distribución Final del conjunto de Datos de Entrenamiento

Data Augmentation

En esta fase, se crean dos objetos **DataLoader** y se definen dos conjuntos de transformaciones con el fin de mejorar la generalización del modelo durante el entrenamiento, así como mantener la coherencia en las etapas de validación y prueba.

Transformaciones para entrenamiento: train_transform

El conjunto de transformaciones aplicado durante el entrenamiento está diseñado para ampliar artificialmente el conjunto de datos, introduciendo variabilidad y robustez en los datos de entrada. Las transformaciones aplicadas incluyen:

- Conversión de las imágenes a tensores con valores de píxeles normalizados entre 0 y 1.
- Recorte aleatorio de aproximadamente el 90% del tamaño original de la imagen, lo que introduce variación espacial.
- Redimensionamiento a un tamaño fijo definido por la variable `IMAGE_SIZE` (por ejemplo, 640×640 píxeles), garantizando uniformidad en la entrada del modelo.
- Aplicación de transformaciones afines aleatorias, incluyendo rotaciones suaves (±7 grados) y traslaciones (hasta un 6%) para simular diferentes ángulos de adquisición.
- Aplicación de desenfoque gaussiano con kernel de tamaño 3 y sigma en el rango [0.2, 1.5], con una probabilidad del 40%, para simular ruido en las imágenes.
- Normalización utilizando las estadísticas del conjunto **ImageNet** (media = [0.485, 0.456, 0.406], desviación estándar = [0.229, 0.224, 0.225]) para estandarizar los datos de entrada.

Transformaciones para validación y prueba: val_test_transform

Durante la validación y la evaluación final, el enfoque se centra en la consistencia de las entradas más que en la variabilidad. Por ello, las transformaciones aplicadas son más conservadoras:

- Conversión de las imágenes a tensores con valores de píxeles escalados entre 0 y 1.

- Recorte del 90% del tamaño original para eliminar artefactos de borde, manteniendo la coherencia espacial.
- Redimensionamiento a `IMAGE_SIZE` para que coincida con las dimensiones esperadas por el modelo.
- Normalización utilizando las mismas estadísticas de **ImageNet**, asegurando que las condiciones de inferencia sean coherentes con las del entrenamiento.

Estas transformaciones permiten que el modelo sea entrenado con datos enriquecidos, pero evaluado bajo condiciones controladas y representativas de un entorno real.

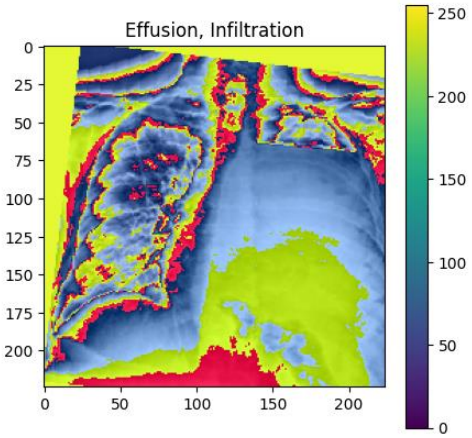


Figure 10: Imagen después de aplicar transformaciones (train)

Arquitectura de la Red Neuronal

A continuación, se describe la arquitectura la red neuronal desarrollada, como se puede apreciar en la siguiente sección, está compuesta por 4 grandes bloques (BackBone-FPN, Proyección de Imagen, Proyección de datos Tabulares y combinación de Imagen con Datos tabulares)

Datos de Entrada

Imagen médica (x_image)

Tipo: Tensor

Forma: [B, 3, 224, 224]

Descripción: Imagen de entrada con 3 canales (RGB), con dimensiones de 224x224 pixeles

Datos tabulares (x_tabular)

Tipo: Tensor

Forma: [B, 3]

Descripción: Vector numérico con información del paciente estructurado por muestra.

Datos de Salida

Predicciones (logits)

Tipo: Tensor

Forma [B, 14]

Descripción: Valores logit (sin activar) para cada una de las 14 clases. Estos valores representan la evidencia sin normalizar que el modelo asigna a cada clase.

BackBone – DenseNet

Las Dense Convolutional Networks o DenseNet, representan una arquitectura de redes neuronales profundas diseñada para mejorar la eficiencia del aprendizaje y la reutilización de características a lo largo de la red. A diferencia de arquitecturas tradicionales donde cada capa recibe la salida únicamente de la capa anterior, en DenseNet cada capa está conectada directamente con todas las capas siguientes de forma concatenada. Es decir, la entrada de cada capa consiste en las salidas de todas las capas anteriores.

Este enfoque de conectividad densa presenta múltiples ventajas:

- Mejor flujo de gradientes durante el entrenamiento, lo que mitiga el problema del desvanecimiento del gradiente en redes profundas.
- Reutilización de características, ya que las capas pueden acceder directamente a los mapas de activación previos, reduciendo la redundancia.
- Eficiencia en parámetros, al requerir menos filtros y capas para lograr un rendimiento competitivo.

DenseNet ha demostrado un alto desempeño en tareas de clasificación de imágenes y ha sido aplicada exitosamente en contextos como el análisis médico, incluyendo radiografías de tórax y otras modalidades de imágenes biomédicas.

El backbone de denseNet será el encargado de extraer las características de las imágenes, debido a todas las ventajas que se enlistaron anteriormente.

Se extrajeron las siguientes capas intermedias de denseNet:

- features.denseblock2 → c3 (512 channels)
- features.denseblock3 → c4 (512 channels)

- features.denseblock4 → c5 (1024 channels)

Estas capas serán usadas y entrenadas por medio del FPN

FPN - Feature Pyramid Network

Se decidió implementar en la arquitectura del modelo una Feature Pyramid Network, ya que se ha presentado evidencia positiva al hacer uso de este tipo de red en la tarea de extracción de características.

En forma de contexto, una FPN, proporciona una ruta de arriba hacia abajo para construir capas de mayor resolución a partir de una capa rica en semántica.

Nuestra FPN tiene la siguiente Arquitectura:

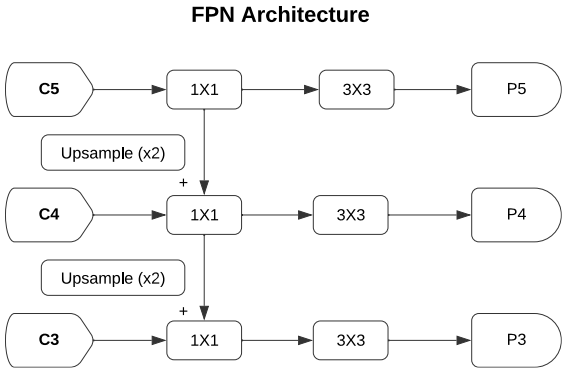


Figure 11: Arquitectura del FPN

Como se puede apreciar, la FPN recibe como inputs las capas C5, C4 y C3 de DenseNet y posteriormente aplica una conversión de tamaño 1x1 para cada uno de los canales por medio de la función **Conv2d,** esta conversión se hace para unificar la cantidad de canales de cada una de las capas y poder sumarlas entre sí.

Posteriormente para los canales laterales se aplica un upsample (x2) por medio de la función **F.interpolate** para poder igualar la resolución espacial entre capas, esta función reduce el tamaño espacial a la mitad, ya que la FPN se construye de forma top-down, es decir, desde las capas más profundas (más semánticas, pero de baja resolución, como C5) hacia las más superficiales (más resolución, pero menos semántica, como C3).

Al final, se aplica una conversión de 3x3 a cada una de las salidas de la FPN para eliminar artefactectos creados por el upsampling y homogeneizar la salida.

Classification Wrapper

Esta parte de la red neuronal combina los 2 componentes anteriores y agrega 3 componentes nuevos

El proceso que hace esta red neuronal es el siguiente:

BackBone CNN

- Usa DenseNet121 preentrenado en ImageNet.
- Extrae 3 niveles jerárquicos (c3, c4, c5) del backbone.
- Estos 3 niveles se alimentan a la FPN para generar un "pyramid" de características.

FPN

- La FPN recibe c3, c4, c5 y los convierte a 3 mapas de características de 256 canales cada uno.
- Integra características de bajo, medio y alto nivel con top-down + lateral connections.

Salida del FPN

- Cada salida de la FPN (P3, P4, P5) se reduce a 1x1 usando AdaptiveAvgPool2d.
- Luego se concatena en un solo vector de tamaño 256 × 3 = 768 por muestra.

Proyección de la Imagen

- Convierte las características de 768 dimensiones a 256 con una capa densa + ReLU.

información Tabular

- Convierte los datos tabulares (por ejemplo, edad, género, historial médico) a un vector de tamaño 8.
- Agrega la funcion de activación ReLU para intentar aprender relaciones complejas entre los datos del paciente

Combinación de información tabular con Imagen

- Combina la imagen y los datos tabulares.
- Pasa por capas densas, aplica Dropout para regularización.
- La salida final tiene num_classes=14 (cada clase representa una enfermedad)

A continuación, se muestra la arquitectura completa de la red neuronal entrenada:

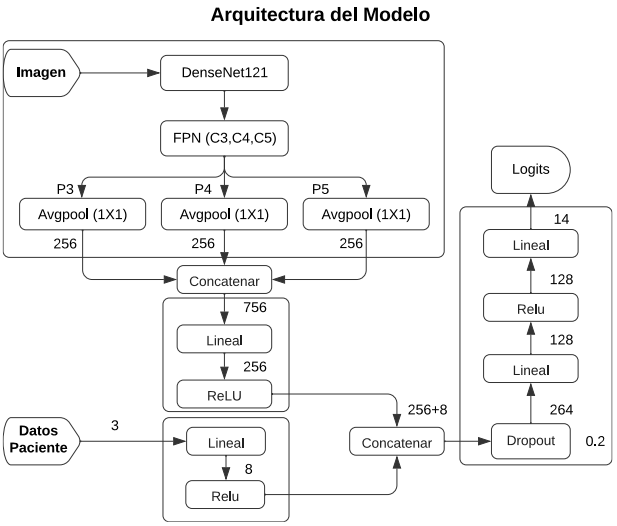


Figure 12: Arquitectura del Modelo

Como se puede apreciar en la figura anterior, el modelo está compuesto de 4 grandes bloques, se puede ver el código que se encarga de crear esta red neuronal en el Notebook de Google Collab

Proceso de Entrenamiento

Como se mencionó en la sección de entorno de desarrollo, los recursos que teníamos disponibles nos fueron de gran ayuda para poder realizar el entrenamiento del modelo múltiples ocasiones, con distintos parámetros y configuraciones (Todas estas versiones se pueden ver en el siguiente link), el modelo final fue entrenado con los siguientes parámetros:

Parámetros de Entrenamiento

BatchSize

Se estableció un tamaño de batch de **160**, con este valor, se hacía uso de toda la VRAM con la que cuenta la GPU Nvidia L4, si se quiere entrenar el modelo en un entorno con especificaciones menores es recomendable bajar el tamaño del batch, para la tarjeta T4 hicimos pruebas con un tamaño de batch de **100**

Resolución de Imagen

La resolución de las imágenes fue de 224x224 pixeles, ya que se trata de un punto intermedio entre calidad y rendimiento, además, que DenseNet fue entrenado con esta misma resolución.

También se hicieron pruebas con 580,512,480 pixeles, pero el proceso de entrenamiento se ve afectado drásticamente, además que el extractor de características se ve afectado negativamente.

Criterio de Optimización

Se uso Binary Cross Entropy with Logits Loss (nn.BCEWithLogitsLoss), tratándose de una function que combina:

- Sigmoid activation.
- Binary Cross Entropy Loss.

Esta función es usada para problemas de clasificación multi-Etiqueta

A esta función se le agregaron en forma de argumento, los pesos personalizados para cada una de las clases del Dataset, para poder contrarrestar el desbalanceo que existe. El calculo de los pesos se describe a continuación.

Pesos personalizados de perdida

De acuerdo con el número de observaciones que tiene el dataSet de entrenamiento de cada una de las enfermedades, se calcula un peso personalizado, se asigna un peso mayor a las clases que tienen menos ejemplos positivos (menos frecuentes), y un peso menor a las clases más comunes. Estos pesos obligan al modelo a aprender mas de las clases menos fruentes, los pesos obtenidos son los siguientes:

Enfermedad	**Peso**
Hernia	3.0943
Nodule	2.1768
Fibrosis	3.0209
Edema	2.8127
Cardiomegaly	2.5823
Effusion	1.6668
Pneumothorax	2.3087
Pleural_Thickening	2.4494
Consolidation	2.4346
Emphysema	2.6907
Pneumonia	3.0943
Mass	2.2459

Atelectasis	1.8179
Infiltration	1.6402

Table 1: Pesos personalizados para la función de perdida

Cabe resaltar que el modelo fue sometido a 3 ciclos de entrenamiento, donde las primeras 2 veces se usaron los pesos contenidos en la tabla anterior, pero para el tercer ciclo, se establecieron los pesos de forma manual para las siguientes enfermedades, debido a que el modelo no lograba identificar correctamente dichas enfermedades

Enfermedad	**Peso**
Pneumonia	4.5
Fibrosis	3.8
Consolidation	4.3

Table 2: Pesos manuales para las clases debiles

Learning Rate

Se estableció un Learning Rate de **0.001** en conjunto con un scheduler que se especifica a continuación

Scheduler

Supervisa la métrica de pérdida de validación (val_loss) durante el entrenamiento. Si dicha métrica no mejora durante 2 épocas, se reduce la tasa de aprendizaje a la mitad.

Epochs

Se hicieron 3 ciclos de entrenamiento de 5 épocas cada uno de ellos.

Ciclo de Entrenamiento

El ciclo de entrenamiento se registró el valor de perdida de entrenamiento, así como el tiempo por cada uno de los pasos que se hacían a lo largo de la época.

En promedio una época constaba de 180 pasos con un tiempo de 25 minutos por Época de Entrenamiento. El proceso completo de entrenamiento tuvo una duración de 280 Minutos (4 con 40 Minutos)

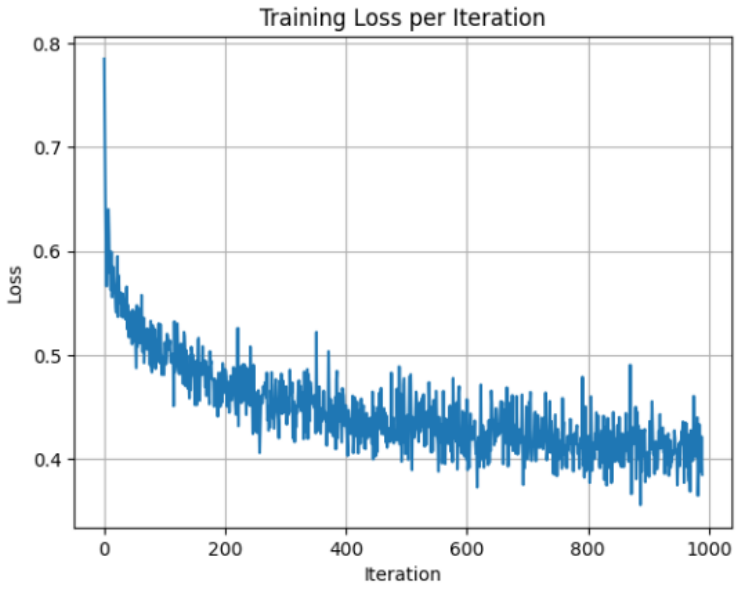


Figure 13: Perdida de Entrenamiento por Interacción (Época 10-15)

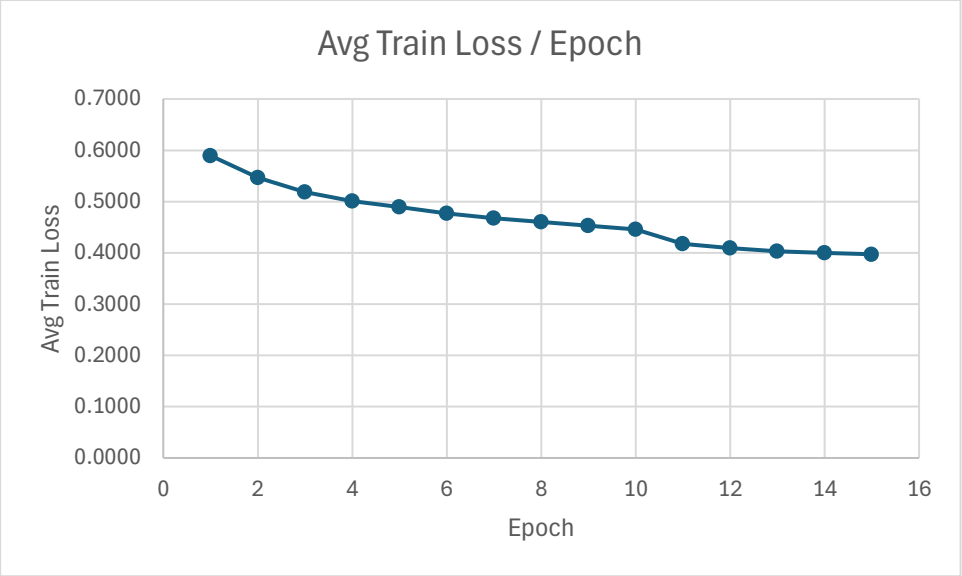


Figure 14: Perdida de entrenamiento por época

Ciclo de Validación

El ciclo de validación es de gran ayuda para saber que tan bien el modelo está aprendiendo, a continuación, se enlistan los puntos principales contenidos en este ciclo.

Limite Dinámico por Clase

Nuestro modelo da como salida logits (presencia de cada clase en forma de datos crudos), antes de hacer predicciones con estos datos, es necesario aplicar la función de torch.sigmoid, que da como salida las probabilidades de que cada una de las clases esté presente, pero debido al desbalanceo que se tiene en el dataset, es recomendable tener un umbral dinámico por cada una de las clases (en lugar de que todas tengan una probabilidad fija) y poder decidir si dicha enfermedad se encuentra presente o no de acuerdo a su umbral personalizado. Hicimos pruebas calculando estos umbrales, contamos con 3 funciones que calculan los umbrales:

1. find_optimal_thresholds_by_roc

Esta función busca el umbral óptimo para cada clase utilizando la curva ROC, con el objetivo de maximizar la estadística de Youden (J), que se define como la sensibilidad menos (1 - especificidad). En otras palabras, busca el punto en la curva ROC que proporcione el mejor equilibrio entre la tasa de verdaderos positivos y la tasa de falsos positivos. Para cada clase, se calcula la curva ROC usando sklearn.metrics.roc_curve, se obtiene el valor de J para cada posible umbral, y se elige aquel que lo maximiza. En los casos en que una clase esté ausente o completamente presente (es decir, solo ceros o unos en las etiquetas), devuelve un umbral predeterminado de 0.5. Esta función es útil cuando se quiere ajustar los umbrales basándose en las características del modelo en la curva ROC, sin priorizar directamente precisión o recall.

2. find_optimal_thresholds_with_precision_constraint

Esta función encuentra el mejor umbral por clase con base en la máxima F1-score, siempre y cuando se cumpla una precisión mínima establecida por el usuario. Para ello, recorre un conjunto de umbrales posibles (por defecto, desde 0.1 hasta 0.85 en incrementos de 0.05) y, para cada clase, descarta aquellos que no alcanzan la precisión mínima requerida (por defecto, 0.30). De los umbrales válidos, selecciona aquel que produce el F1-score más alto. Esta función es especialmente útil en contextos donde es prioritario mantener un control estricto sobre los falsos positivos, como en diagnósticos médicos, donde una predicción incorrecta puede tener consecuencias significativas.

3. find_optimal_thresholds_with_dynamic_adjustment

Esta variante también busca maximizar el F1-score bajo una restricción de precisión mínima, pero añade un ajuste dinámico en caso de que el rendimiento (medido por F1-score) sea insuficiente.

Si después de aplicar el criterio anterior el mejor F1-score sigue siendo inferior a un umbral definido (por defecto, 0.30), la función reduce el umbral aún más (según un factor de ajuste, por defecto 0.05) para permitir mayor recall, lo que puede ser crucial en casos donde se necesita detectar más positivos, aun si ello implica aumentar los falsos positivos.

La función que nos dio mejores resultados fue la numero 3

Estos umbrales dinámicos son almacenados con estructura de diccionario en un archivo json.

Perdida de Validación

El valor de la perdida de validación en el entrenamiento se muestra en la gráfica siguiente:

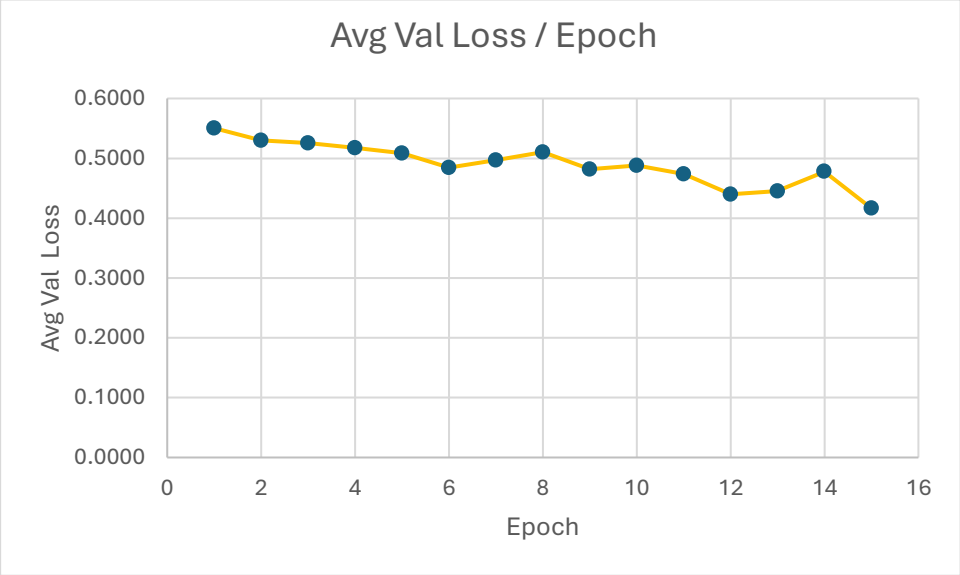


Figure 15: Perdida de validación por época

Como se puede apreciar en la gráfica, el modelo logro reducir su valor de perdida de validación, pero cuenta con mesetas a lo largo de varias épocas, en veces empeoraba entre épocas.

Precisión (Promedio)

El valor de la precisión obtenida en el entrenamiento se muestra en la gráfica siguiente:

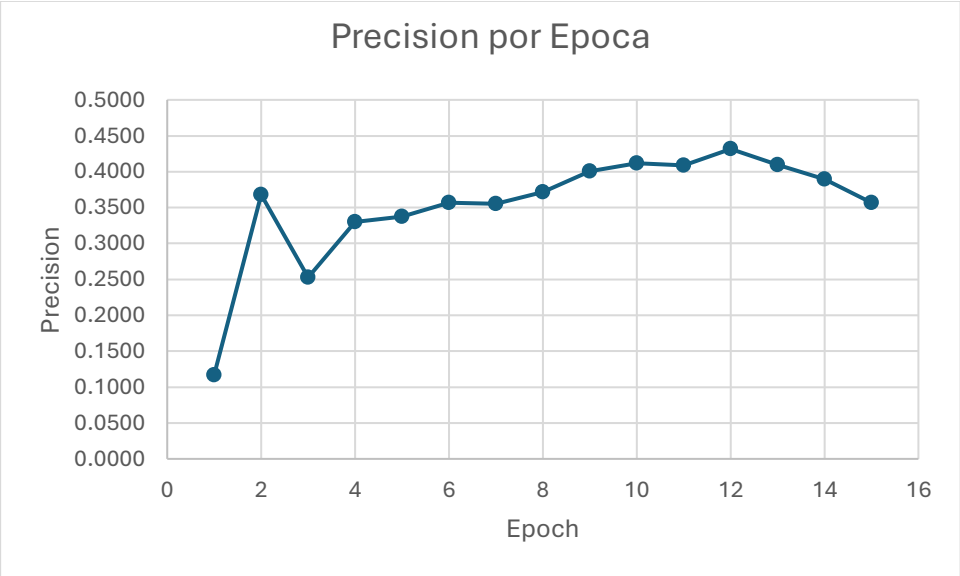


Figure 16: Precisión por época

Como se puede apreciar, después de la época 12, el modelo empezó a disminuir su precisión

Recall (Promedio)

El valor del recall obtenido en el entrenamiento se muestra en la gráfica siguiente:

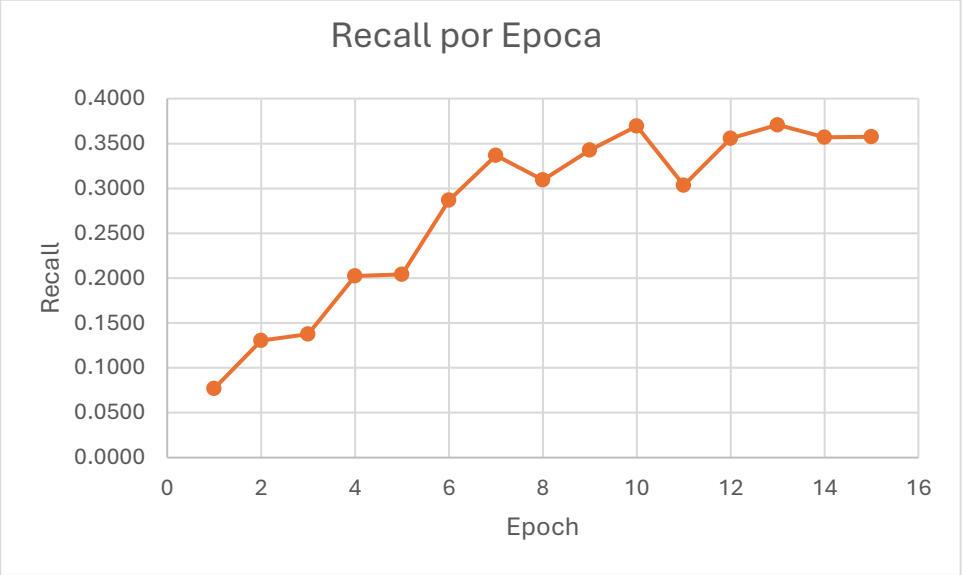


Figure 17: Recall por época

A diferencia de la precisión, el recall se mantuvo constante después de la época 12

Macro F1 (Promedio)

El valor del Macro F1 obtenido en el entrenamiento se muestra en la gráfica siguiente:

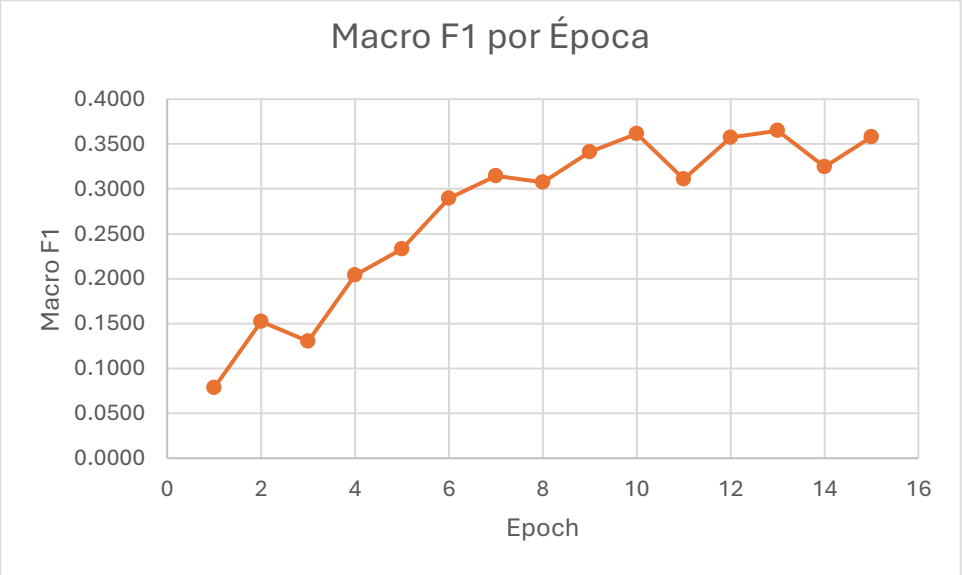


Figure 18: Macro F1 por época

El F1 Score (Macro) no obtuvo mejoría contundente después de la época 10

AUC (Promedio)

El valor AUC obtenido en el entrenamiento se muestra en la gráfica siguiente:

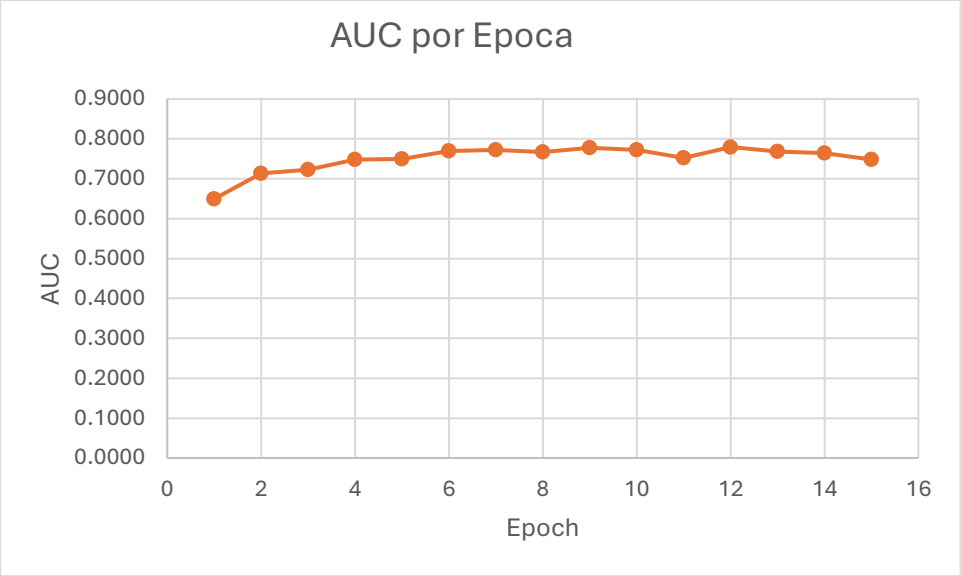


Figure 19: AUC por época

El valor AUC se mantuvo constante a lo largo del proceso de entrenamiento, el mejor valor se obtuvo en la época 12

Todos estos datos se pueden ver en la siguiente tabla

E	Avg Loss	Val Loss	Precisión (mean)	Recall (mean)	F1 Score (mean)	Macro F1	AUC (mean)
1	0.5895	0.5504	0.1171	0.0770	0.0785	0.0785	0.6495
2	0.5466	0.5296	0.3679	0.1307	0.1520	0.1520	0.7138
3	0.5181	0.5257	0.2527	0.1375	0.1302	0.1302	0.7231
4	0.5009	0.5175	0.3303	0.2021	0.2041	0.2041	0.7484
5	0.4894	0.5083	0.3376	0.2039	0.2330	0.2330	0.7493
6	0.4772	0.4844	0.3569	0.2865	0.2897	0.2897	0.7698
7	0.4672	0.4972	0.3555	0.3365	0.3147	0.3147	0.7717
8	0.4606	0.5107	0.3717	0.3091	0.3074	0.3074	0.7669
9	0.4533	0.4820	0.4006	0.3428	0.3416	0.3416	0.7779
10	0.4452	0.4883	0.4121	0.3696	0.3617	0.3617	0.7723
11	0.4174	0.4739	0.4089	0.3032	0.3110	0.3110	0.7526
12	0.4095	0.4399	0.4316	0.3556	0.3571	0.3571	0.7794
13	0.4030	0.4457	0.4095	0.3708	0.3649	0.3649	0.7687
14	0.4003	0.4775	0.3896	0.3568	0.3249	0.3249	0.7639
15	0.3972	0.4485	0.3570	0.3513	0.3577	0.3577	0.7485

Figure 20: Tabla con métricas de Entrenamiento y Validación

Criterio de Guardado

Después de cada ciclo de validación se guarda el mejor modelo hasta el momento, en proceso de entrenamiento probamos con las siguientes métricas para considerar un modelo como “El mejor”

AUC

Evalúa la capacidad del modelo para distinguir entre clases positivas y negativas para cada etiqueta.

Por qué importa: En multietiqueta, cada clase se trata como un problema binario, y AUC proporciona una medida robusta de la discriminación del modelo por clase, independientemente del umbral de decisión.

Ventaja: No se ve afectada por el desbalance entre clases, lo cual es común en tareas multietiqueta (algunas enfermedades, por ejemplo, son mucho más raras que otras).

Útil para evaluar qué tan bien separa el modelo los positivos de los negativos, sin depender de un umbral específico.

Macro F1 (Seleccionado)

Qué mide: Es el promedio del F1-score de cada clase, dándole igual peso a cada etiqueta, sin importar cuán frecuente o rara sea.

Por qué importa: En multietiqueta, las etiquetas poco frecuentes tienden a ser ignoradas si se usa una métrica global como la accuracy o el micro F1. El Macro F1 fuerza al modelo a desempeñarse bien en todas las etiquetas por igual.

Ventaja: Favorece modelos que son equilibrados en su rendimiento entre clases comunes y raras.

Ideal cuando todas las clases son igualmente importantes.

Este fue el criterio que nosotros consideramos **mas importante** para la selección del "Mejor modelo", siendo el modelo de le época 13 el seleccionado. Y sobre el cual se muestran los resultados

Precisión

Qué mide: De los casos que el modelo predijo como positivos, cuántos eran realmente positivos.

Por qué importa: En aplicaciones sensibles (como diagnósticos médicos), minimizar los falsos positivos es clave. Un modelo con alta precisión es conservador, solo predice positivo cuando está relativamente seguro.

Desventaja: Puede venir a costa de un bajo recall (el modelo puede omitir positivos verdaderos).

Clave cuando es más costoso cometer falsos positivos que falsos negativos.

Archivos Resultado

Cuando el proceso de entrenamiento y análisis de resultados termina, se guardan 5 archivos:

validation_metrics_[Nombre del Modelo].csv

Contiene las métricas del modelo obtenidas con el dataset de prueba (test_df)

configuración.txt

Contiene la configuración y logs el proceso de entrenamiento

optimizer_classification.pt

Contiene el estado actual del optimizador

model_classification.pt

Contiene los pesos de la arquitectura del modelo

thresholds.json

Contiene los umbrales personalizados para cada una de las enfermedades

Los archivos del modelo final se encuentran en el siguiente archivo: FINAL DenseNetTuned.zip

Resultados

Umbrales de Aceptación

En primer lugar, se analizarán los umbrales calculados en el proceso de validación y almacenados en el archivo thresholds.json:

Enfermedad	Peso
Hernia	0.20
Nodule	0.50
Fibrosis	0.65
Edema	0.75
Cardiomegaly	0.40
Effusion	0.50
Pneumothorax	0.25
Pleural_Thickening	0.45
Consolidation	0.50
Emphysema	0.40
Pneumonia	0.45
Mass	0.25
Atelectasis	0.35
Infiltration	0.45

Table 3 Umbrales Personalizados por Enfermedad

Recordando que se uso la función numero 3 para encontrar los mejores umbrales, que en forma de recapitulación

1. Explora umbrales entre 0.10 y 0.85 (en pasos de 0.05).
2. Para cada umbral t, calcula:
   - La precisión
   - El F1-score
1. Si el umbral cumple con la condición precision ≥ 0.40, y el F1-score es el mejor encontrado hasta ese momento, lo guarda como candidato.
2. Luego de revisar todos los umbrales:
   - Si el mejor F1-score encontrado aún es menor que 0.30, se aplica un ajuste descendente al umbral para mejorar el recall, bajándolo 0.05 unidades.

Es decir, podemos anticipar lo siguiente (al menos para el dataset de Validación):

Los valores de umbral por enfermedad obtenidos reflejan el desempeño individual del modelo para cada clase, con base en su capacidad para equilibrar precisión y F1-score. Por ejemplo, enfermedades como Edema (umbral de 0.75) y Fibrosis (0.65) presentan umbrales elevados porque el modelo logra hacer predicciones confiables para estas clases, alcanzando una buena precisión sin necesidad de ajustar el umbral hacia abajo. En contraste, clases como Pneumothorax y Mass, con umbrales de 0.25, requieren decisiones más permisivas. Esto sugiere que el modelo tiene dificultad para detectarlas con confianza, por lo que se reduce el umbral para incrementar el recall y evitar que los positivos pasen desapercibidos.

El caso más extremo es Hernia, con un umbral de 0.20, indicando que el modelo probablemente presenta muchos falsos negativos para esta enfermedad si se usaran umbrales más altos. Por ello, se aplica el ajuste adaptativo para permitir que cualquier probabilidad moderadamente alta ya cuente como predicción positiva.

Métricas de Validación

Al momento de utilizar el set de evaluación en el modelo final, se obtuvieron las siguientes métricas:

Exact Match Ratio

Esta métrica mide el porcentaje de veces que el modelo predijo exactamente todas las enfermedades presentes en una radiografía torácica. Justamente es la métrica más estricta utilizada para evaluar la precisión de los modelos de predicción multiclase. Se obtuvo un porcentaje de aciertos de alrededor del **14.21%**, lo cual es un porcentaje bajo, considerando que la mayoría de las imágenes cuenta con solo una enfermedad a predecir. Esto puede indicar, que, si bien se predice el valor correcto, aparte se agregan otras enfermedades predichas erróneas.

Hamming Loss

El Hamming Loss mide el porcentaje de falsos positivos y falsos negativos arrojados por el modelo para cada clase, del total de predicciones. Esto implica que entre más cercano este el valor de cero, más preciso ha sido el modelo en predecir el valor de cada clase. Se obtuvo como resultado un valor de 0.1326, o básicamente un **13.26%** de desaciertos en todas las clases. Esto implica que, en la mayoría de las veces, alrededor del 87%, sí se predijo correctamente la presencia o ausencia de una enfermedad en una imagen. Esto puede indicar un buen rendimiento general, ya que implica que el modelo no estuvo simplemente poniendo todos los posibles valores a predecir, para acertar el 100% de las veces.

Recall

Hay dos tipos de Recall, el micro, obtenido al considerar el recall, ratio de verdaderos positivos del total de positivos, de todas las clases; y el macro, obtenido al promediar el recall individual de todas las clases. Entre más cercano sea al uno, implica que hubo más verdaderos positivos y falsos negativos, o sea que se tuvo más precisión. Se obtuvo un recall micro de **0.43** y uno macro de **0.38**, indicando que básicamente cerca del 40% de las veces que la enfermedad estaba presenta, se predijo un valor positivo. Es un valor deficiente, ya que por lo menos el modelo debería de ser capaz de identificar cuando una enfermedad se encuentra en una imagen, independientemente de si se identificaron otras erróneas.

Label Ranking Average

Esta métrica mide el porcentaje de veces que se asignó el mayor porcentaje de predicción a todas las clases que se presentan en una imagen. En otras palabras, si hay 3 clases en la imagen, un acierto implicaría que las clases presenten, independientemente del orden, los mayores porcentajes de acierto de entre las demás clases. En este caso, se obtuvo un valor de **64%** de clases rankeadas correctamente. Es un valor bueno, ya que se acerca al 70%, sin embargo, esto no significa que las clases rankeadas hayan tenido el porcentaje necesario para ser consideradas como presentes en la imagen, por lo que no es un indicador que contundentemente evalúe el rendimiento del modelo.

Brier Score

El Brier score sirve para conocer el nivel de precisión del modelo, al añadir peso a precisiones incorrectas, especialmente a predicciones incorrectas ocasionadas por una sobre confianza del modelo en la predicción (se penaliza más un valor como 0.99 si es que el verdadero valor era 0, que ha predecir 0.5 a un valor que era 0). En la siguiente tabla se muestran las predicciones obtenidas para todas las clases:

Clase	Brier Score
Hernia	0.0046
Nodule	0.1189

Fibrosis	0.0721
Edema	0.0447
Cardiomegaly	0.0397
Effusion	0.1487
Pneumothorax	0.0782
Pleural_Thickening	0.1237
Consolidation	0.0922
Emphysema	0.0372
Pneumonia	0.0431
Mass	0.1011
Atelectasis	0.1587
Infiltration	0.1836

Table 4: Brier Score

Al ser la mayoría de los valores obtenidos menores al 0.10, esto implica que por lo general la diferencia entre el porcentaje predicho y el valor real no es muy grande. Esto indica que el modelo por lo general tuvo una buena precisión al momento de predecir las enfermedades.

F1 Score Global

El F1 score evalúa el porcentaje de verdaderos positivos, con respecto a los verdaderos positivos, falsos negativos y falsos positivos; hace un promedio entre todas las clases si es macro, y con todas las clases si es micro. En este caso es parecida al recall, solo que se agrega al porcentaje todos los falsos positivos. Para el macro es obtuvo en micro un valor de **0.41** y en macro **0.32.** Justamente son valores similares a los obtenidos al recall, pero ahora con el F1 Global, se observa que por lo general el 32% de las veces se identificó en la imagen correctamente una enfermedad, el resto de las veces simplemente se equivocó.

F1 Score (Class)

Ahora a continuación se muestra el F1 Score pero por clase en la siguiente tabla:

Clase	F1 Score
Hernia	0.4237
Nodule	0.2261
Fibrosis	0.1538
Edema	0.1548
Cardiomegaly	0.5475
Effusion	0.6020
Pneumothorax	0.4736
Pleural_Thickening	0.2565
Consolidation	0.0259
Emphysema	0.4416
Pneumonia	0.0330
Mass	0.3814
Atelectasis	0.4695
Infiltration	0.4147

Table 5: F1 Score

En general los resultados rondan el 40% y 30% de aciertos. Sin embargo, en la tabla se observa que hay algunos valores donde la tasa de acierto es muy baja, específicamente al identificar enfermedades como Pneumonia, Consolidation, Edema y Fribrosis. De esta forma se puede observar claramente los casos en donde se necesita realizar más entrenamiento.

Accuracy

Esta métrica promedio la precisión al momento de predecir el valor de cada clase en todas las imágenes, o sea checa la proporción de verdaderos positivos y negativos. Se obtuvo un resultado general del 86.74%, lo cual, si bien es un buen indicador de precisión, no necesariamente significa que el modelo está prediciendo correctamente. Como también se toma en cuenta los verdaderos negativos, la precisión aumenta considerablemente a las métricas obtenidas en el recall y en el F1-score, ya que es común que una enfermedad falte en la mayoría de las imágenes. El indicador de precisión básicamente nos confirma que por lo menos casi siempre no predice una enfermedad cuando está no está presente.

Matriz de confusión

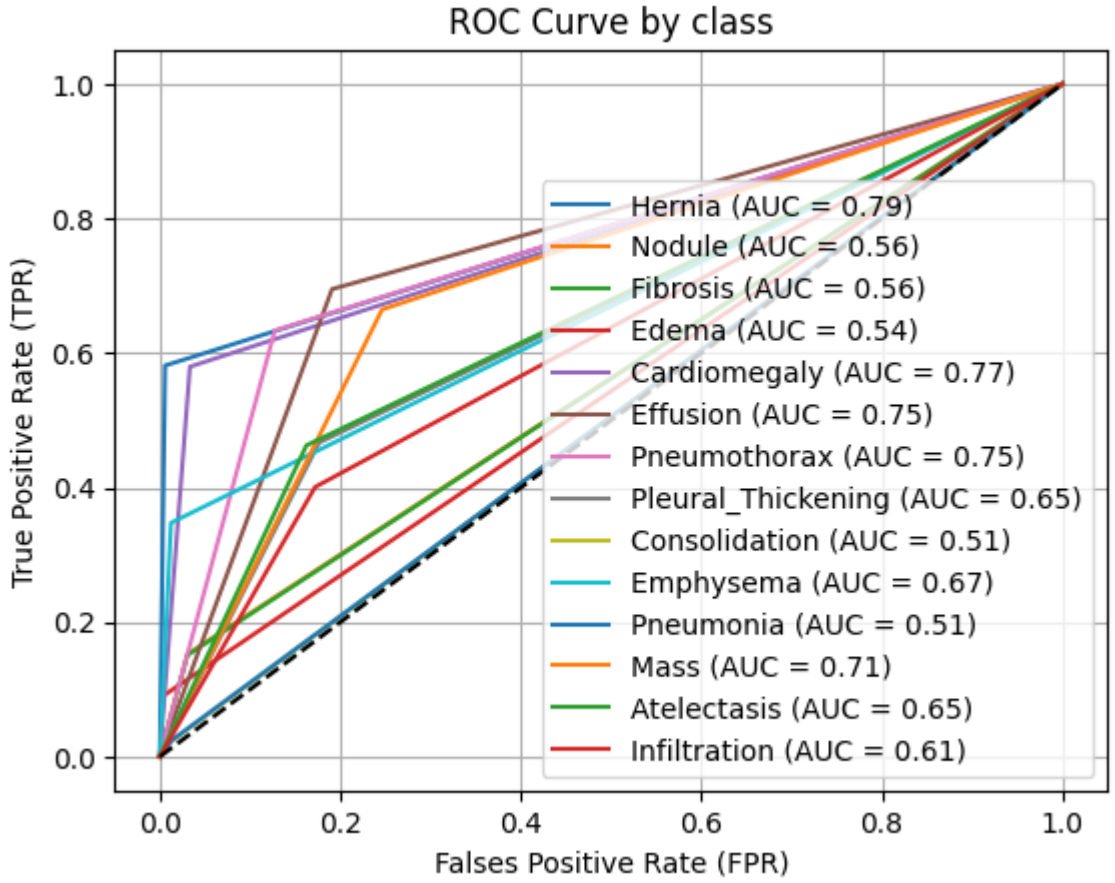
Las matrices de confusión muestran el número de verdaderos positivos y negativos, y falsos positivos y negativos predichos en torno a una clase. A continuación, se muestran las matrices de confusión de todas las clases:

Clase	Matrices de confusión
Hernia	$\begin{pmatrix} 8494 & 50 \\ 18 & 25 \end{pmatrix}$
Nodule	$\begin{pmatrix} 7186 & 230 \\ 982 & 177 \end{pmatrix}$
Fibrosis	$\begin{pmatrix} 8023 & 252 \\ 265 & 47 \end{pmatrix}$
Edema	$\begin{pmatrix} 8146 & 32 \\ 372 & 37 \end{pmatrix}$
Cardiomegaly	$\begin{pmatrix} 7815 & 270 \\ 211 & 291 \end{pmatrix}$
Effusion	$\begin{pmatrix} 5301 & 1250 \\ 621 & 1415 \end{pmatrix}$
Pneumothorax	$\begin{pmatrix} 6676 & 975 \\ 343 & 593 \end{pmatrix}$
Pleural_Thickening	$\begin{pmatrix} 6541 & 1401 \\ 344 & 301 \end{pmatrix}$
Consolidation	$\begin{pmatrix} 7749 & 25 \\ 802 & 11 \end{pmatrix}$
Emphysema	$\begin{pmatrix} 8040 & 101 \\ 291 & 155 \end{pmatrix}$
Pneumonia	$\begin{pmatrix} 8229 & 110 \\ 242 & 6 \end{pmatrix}$
Mass	$\begin{pmatrix} 5701 & 1863 \\ 343 & 680 \end{pmatrix}$
Atelectasis	$\begin{pmatrix} 5451 & 1057 \\ 1117 & 962 \end{pmatrix}$
Infiltration	$\begin{pmatrix} 5372 & 1117 \\ 1257 & 841 \end{pmatrix}$

Table 6: Matriz de confusión

ROC Curve

La curva ROC (Receiver Operating Characteristic) grafica los valores entre la ratio de falsos positivos (en el eje x) y en ratio de verdaderos positivos (en el eje y), obtenidos al mover entre el 0 y el 1 el valor mínimo de la probabilidad predicha para considerar la predicción como positiva (o sea que sí se considera presente la clase). En una buena gráfica de ROC, la punta de la gráfica tiene que tender a la parte izquierda superior. A continuación, se muestra la gráfica obtenida para el modelo entrenado:



La gráfica muestra una leve tendencia por los datos a la izquierda, demostrando que por lo menos el modelo no está haciendo, en la mayoría de los casos, predicciones al azar. Aun así, se observa que el modelo todavía tiene un margen de mejora.

Ejemplos

A continuación, se muestran algunas predicciones realizadas por el modelo.

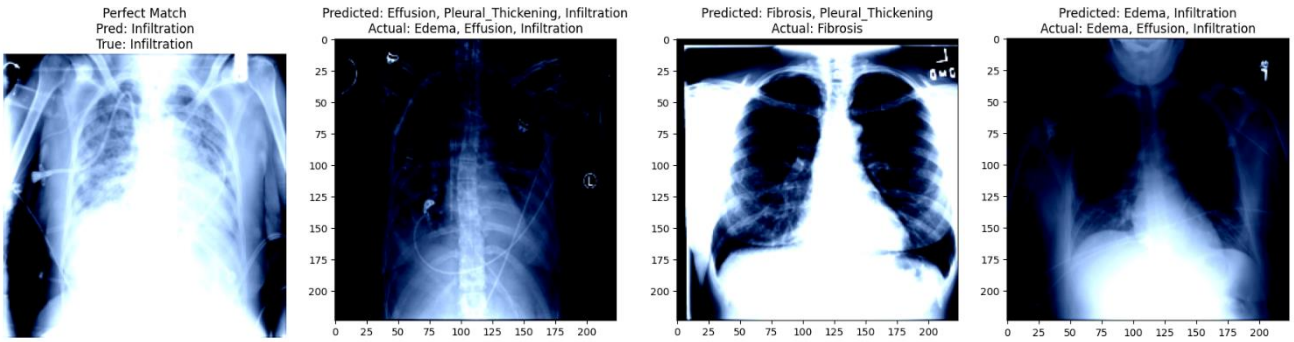


Figure 21: Ejemplos de Predicciones

Como se puede apreciar, el modelo acierta en algunas enfermedades, pero cuenta con algunos falsos positivos, así como también algunos falsos negativos, nuestro modelo cuenta con grandes áreas de oportunidad y mejora.

Conclusiones

A partir del análisis de los resultados obtenidos, se pueden extraer las siguientes observaciones relevantes sobre el comportamiento del modelo de clasificación multietiqueta en el conjunto de prueba:

Desempeño adecuado en clases frecuentes

El modelo presentó un rendimiento sólido en clases con mayor número de muestras, tales como Effusion (F1-score = 0.60), Atelectasis (F1-score = 0.47) e Infiltration (F1-score = 0.41). Este comportamiento indica que el modelo logra aprender representaciones efectivas cuando dispone de una cantidad suficiente de ejemplos durante el entrenamiento.

Resultados aceptables en clases complejas y poco frecuentes

A pesar de su baja frecuencia en el conjunto de datos, algunas clases como Hernia (F1-score = 0.42) y Pneumothorax (F1-score = 0.47) obtuvieron métricas razonablemente satisfactorias. Esto sugiere que el modelo es capaz de detectar patrones útiles incluso en condiciones clínicas menos representadas.

Bajo rendimiento en ciertas clases específicas

Las clases Pneumonia (F1-score = 0.03) y Consolidation (F1-score = 0.03) reflejan un rendimiento deficiente. Estas condiciones podrían presentar características visuales menos distintivas o solapadas con otras patologías, lo cual dificulta su detección automática. Además, la baja cantidad de ejemplos podría estar limitando la capacidad de generalización del modelo para estas etiquetas.

Impacto del desbalance de clases

Se observó un impacto negativo significativo del desbalance en el conjunto de datos, particularmente en clases como Fibrosis (F1-score = 0.15) y Pleural_Thickening (F1-score = 0.26), que tienen menor representación. Esto evidencia la necesidad de aplicar técnicas de mitigación como reponderación de clases, oversampling o generación de datos sintéticos (e.g., SMOTE).

Desajuste entre precisión y sensibilidad

El modelo mostró una tendencia a favorecer la sensibilidad (recall) en algunas clases (Mass, Hernia) y la precisión en otras (Edema, Emphysema), lo cual podría ser ajustado según los requerimientos clínicos. Por ejemplo, en contextos médicos, es habitual preferir modelos con alta sensibilidad para minimizar los falsos negativos.

Importancia del ajuste de umbrales específicos por clase

Los resultados obtenidos refuerzan la importancia de utilizar umbrales adaptativos por clase para convertir las probabilidades en predicciones binarias, tal como se realizó en este trabajo. Esta estrategia es particularmente útil en contextos multietiqueta y de alta desproporcionalidad de clases, permitiendo mejorar el F1-score y otras métricas relevantes para cada etiqueta.

Anexos

- Notebook Google Colllab
- Repositorio Github
- Repositorio Hugging Face DataSet Original
- Repositorio Hugging Face DataSet Pre-procesado
- Carpeta del proceso de Desarrollo

Bibliografía

Hasanah U, A. C. (27 de Diciembre de 2023). *CheXNet and feature pyramid network: a fusion deep learning architecture for multilabel chest X-Ray clinical diagnoses classification*. Obtenido de Pubmed: https://pubmed.ncbi.nlm.nih.gov/38150139/#:~:text=contribute%20to%20this%20state,other%20approaches%20and%20has%20become

Hui, J. (26 de Marzo de 2018). *Comprensión de las redes piramidales de características para la detección de objetos (FPN)*. Obtenido de Medium: https://jonathan-hui.medium.com/understanding-feature-pyramid-networks-for-object-detection-fpn-45b227b9106c

Ito Aramendia, A. (1 de Marzo de 2024). *DenseNet: una guía completa*. Obtenido de Medium: https://medium.com/@alejandro.itoaramendia/densenet-a-complete-guide-84fedef21dcc

Liz, H., Huertas-Tato, J., Sánchez-Montañés, M., Del Ser, J., & Camacho, D. (s.f.). *Deep learning for understanding multilabel imbalanced Chest X-ray datasets*. Obtenido de ar5iv.labs.arxiv: https://ar5iv.labs.arxiv.org/html/2207.14408#:~:text=sufficient,to%20focus%20on%20minority%20classes