Universidad Nacional Autónoma de México

Facultad de Ingeniería

Detección de covid-19 y neumonía en imágenes rayos x de pulmones utilizando redes neuronales convolucionales

T E S I S

que para obtener el título de:

Ingeniero en Computación

PRESENTA:

Mauricio Gómez Macedo

DIRECTOR DE TESIS:
Dra. Jimena Olveres Montiel

Ciudad Universitaria, CDMX., 2022

Agradezco profundamente a la Facultad de Ingeniería y a la Universidad, por la formación que me han dado.
A mi mamá Alejandra, mi hermano Edgar y mi papá Ignacio que me apoyaron incondicionalmente a lo largo de mi vida, sin ellos no lo hubiera logrado.
A Eloísa por motivarme a ser mejor persona y ayudarme en los momentos difíciles

Reconocimientos

Investigación realizada gracias a los programas

-UNAM-PAPIIT PAPIIT IV100420
-PAPIIT TA101121
-SECTEI/202/2019

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

El coronavirus que se descubrió por primera vez en Wuhan, China en 2019, ha impactado de manera negativa en México y todo el mundo, se vieron afectados distintos sectores como el de salud, económico y social. Al 7 de septiembre de 2021 se han contabilizado 220,563,227 casos de coronavirus en todo el mundo y de los cuales han fallecido 4,565,483 personas, en México se tiene registro de 3,420,880 personas infectadas y 262,868 fallecidos [4].

Al principio de la pandemia, México y muchos otros países no contaban con el equipo necesario para realizar las pruebas de detección de coronavirus, además de que eran costosas y escasas. Las pruebas más comunes para la detección de COVID-19 son PCR y antígenos, a pesar de que la prueba PCR tiene un 97.2 % de efectividad es costosa y tarda días en obtener el resultado, lo cual es crucial para la recuperación del paciente. Por su parte la prueba de antígenos tiene una efectividad de 72 % para personas con síntomas y por lo que es probable que entregue falsos positivos, por esta razón es necesario encontrar nuevas formas para la detección del coronavirus[5][6].

El coronavirus crea daño en los pulmones, este daño se puede apreciar con imágenes rayos X de tórax, el problema es que se puede confundir con daños producidos por otras enfermedades como la neumonía. Es necesario estar seguro de cuál es la enfermedad que produjo el daño en los pulmones del paciente, por esta razón es importante crear métodos que ayuden a diferenciar los tipos de daños que identifican a la enfermedad.

Existen algoritmos de inteligencia artificial que tienen un índice de error bajo y pueden solucionar problemas si estos aprenden de forma correcta. En el área de la medicina son populares las redes neuronales convolucionales para identificar daños ya que estas pueden identificar bordes y figuras para encontrar enfermedades en imágenes médicas.

Es de gran ayuda para los médicos tener una inteligencia artificial que indique si un paciente tiene coronavirus, neumonía o que esté sano, se pueden diagnosticar más pacientes de lo habitual debido a que este proceso es rápido. La problemática de tener un problema multiclase es que tienden a un error más alto que los problemas binarios, por lo que se busca probar un algoritmo que pueda detectar 3 categorías y compararlo con 2 algoritmos binarios en cascada.

En el capítulo 1 se describen los objetivos y la metodología que se utilizó para el desarrollo de este proyecto. El capítulo 2 aborda el estado del arte de los algoritmos utilizados, se profundiza en ellos y se explican sus fórmulas y cómo funcionan. En el capítulo 3 se explican las bases de datos que se utilizaron para que el modelo aprenda y las características de cada una. En el capítulo 4 se explica el proceso del entrenamiento de cada uno de los algoritmos, se mencionan los hiperparámetros que se utilizaron y sus respectivas funciones. En el capítulo 5 se muestran todos los resultados obtenidos para medir la eficiencia de los modelos creados y analizar el modelo con mejor resultado. Por último en el capítulo 6 se mencionan las conclusiones del proyecto.

1.1. Objetivo

Desarrollar un algoritmo que realice el diagnóstico de las enfermedades COVID-19 o neumonía mediante la implementación de una arquitectura de redes neuronales convolucionales en imágenes rayos X de tórax. El algoritmo también será capaz de distinguir si los pulmones se encuentran sanos.

A partir de esto, surgen los siguientes objetivos secundarios:

- Verificar si un algoritmo de inteligencia artificial puede distinguir distintas enfermedades utilizando imágenes médicas.
- Comparar una red neuronal convolucional multiclase con 2 binarias en cascada.
- Justificar si las imágenes rayos X de tórax tienen la calidad suficiente para que se pueda detectar enfermedades con inteligencia artificial.

1.2. Justificación

La inteligencia artificial se debe utilizar para el beneficio de la humanidad y aplicarlo al área de la salud puede traer grandes beneficios al desarrollo de nuevas herramientas que pueden encontrar otras formas de curar a la gente. El coronavirus está afectando de manera grave al mundo por lo que se deben encontrar todas las formas posibles para defendernos ante este virus, es importante crear algoritmos que sean capaces de identificar oportunamente si una persona está enferma de COVID-19, esto puede ser vital en la recuperación de algún paciente.

Este trabajo forma parte del proyecto PAPIIT IV100420 iniciado en el Centro Médico Nacional La Raza donde los médicos radiólogos hicieron la validación clínica de este.

1.3. Contribución

Se creó un sistema basado en redes neuronales convolucionales capaz de reconocer si una persona se encuentra enferma de COVID-19 o neumonía. Se entrenaron 2 modelos distintos con la finalidad de compararlos y encontrar cuál es más eficiente, el primer modelo consiste en entrenar 2 redes neuronales convolucionales binarias para hacer las detección de las 3 clases, la primer red neuronal va a identificar si la imagen está enferma o sana, en la clase enfermo se juntaron las imágenes de COVID-19 y neumonía, si la detección indica que es enfermo, se utiliza la segunda red neuronal para identificar si la imagen pertenece a COVID-19 o neumonía. El segundo modelo consiste en solo entrenar una red neuronal convolucional con 3 salidas para detectar todas las clases por sí sola.

Las imágenes rayos X de tórax con las que se entrenaron los modelos se dividieron de tal forma que el conjunto de entrenamiento esté balanceado, es decir, en el primer modelo la clase enfermo y la clase sano tienen aproximadamente el mismo número de imágenes y dentro de la clase enfermo, las imágenes son las mismas para las clases COVID-19 y neumonía. En el segundo modelo las 3 clases están balanceadas para que tengan aproximadamente el mismo número de imágenes. El conjunto de validación también está balanceado debido a que se toma el 20 % del conjunto de entrenamiento y el conjunto de prueba no están balanceadas las clases debido a que no influye dicho conjunto en el aprendizaje de los algoritmos.

Capítulo 2

Estado del Arte

2.1. Perceptrón

El perceptrón es la unidad básica de una red neuronal multicapa, es un algoritmo de inteligencia artificial de aprendizaje supervisado y se utiliza para problemas de tipo binario [7].

El algoritmo cuenta con η entradas y η pesos, cada entrada es multiplicada por un único peso, se hace una suma del resultado de las entradas multiplicadas por los pesos y el resultado se ajusta con un sesgo, la ecuación se muestra en (2.1), donde z es el resultado de la ecuación, las entradas se representan con la variable x, los pesos con la variable w y el sesgo con la variable b, el subíndice i indica a que entrada y peso se está refiriendo. El resultado de la operación (2.1) se evalúa en una función de activación que, dependiendo de la función de activación, sólo puede dar resultados dentro de un rango de valores específicos. La representación gráfica del perceptrón se encuentra en 2.1.

El resultado de la función de activación es la salida del perceptrón, este valor tendrá que ser evaluado con los ejemplos preestablecidos y con una función de pérdida va a medir el error que tiene el resultado, con el algoritmo de propagación hacia atrás y la medición del error se actualizarán los pesos para poder reducir la pérdida. Cuando se concluye el algoritmo de propagación hacia atrás, se repite el proceso desde el inicio hasta lograr tener un error bajo o aceptable.

$$z = \sum_{i=1}^{n} w_i \cdot x_i + b \tag{2.1}$$

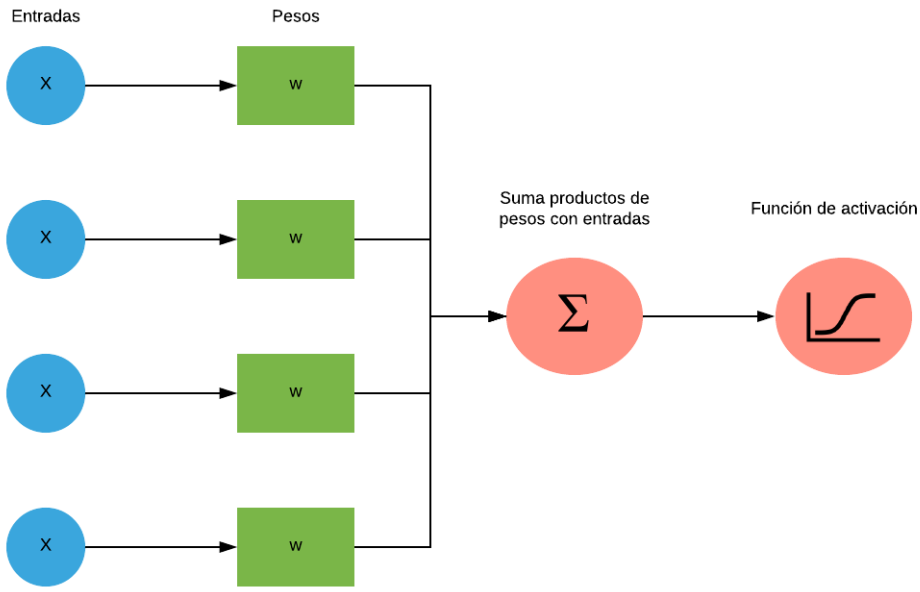

Figura 2.1: Imagen representativa del perceptrón.

2.2. Función de activación

La función de activación en el perceptrón es necesaria porque en caso de no tenerla, solo se trataría de un producto punto y una adición, por lo que sería una función lineal y se necesita una función no lineal, que sea derivable, para el correcto funcionamiento del algoritmo de propagación hacia atrás[8].

2.2.1. Función Sigmoide

La función sigmoide es una función real diferenciable acotada entre 0 y 1, que se define para todos los valores de entrada reales y que tiene una derivada positiva en todas partes [9]. La fórmula está en la ecuación (2.2), la variable z representa el resultado de (2.1). La gráfica de la función sigmoide se encuentra en la figura 2.2.

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{2.2}$$

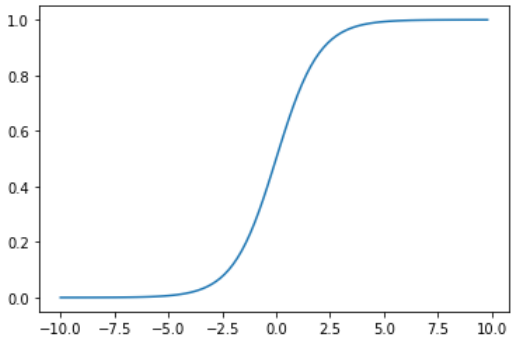

Figura 2.2: Gráfica función sigmoide.

2.2.2. Función Unidad Lineal Rectificada

Es una función de activación que consiste en igualar a 0 todos los valores negativos y mantener los valores positivos. En 2011 se demostró que esta función mejora el entrenamiento de redes neuronales profundas [10]. La fórmula se encuentra en la ecuación (2.3) y la gráfica en la figura 2.3.

$$\sigma(z) = \begin{cases} z & si\ z > 0 \\ 0 & si\ z \leq 0 \end{cases} \tag{2.3}$$

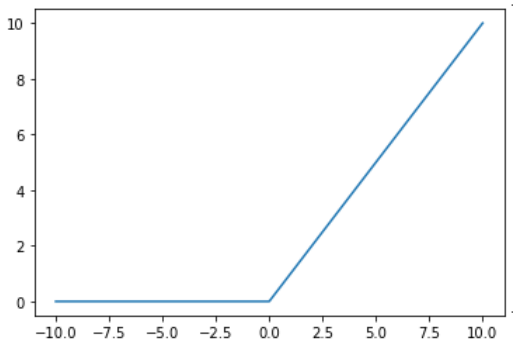

Figura 2.3: Gráfica función Unidad Lineal Rectificada.

2.3. Función de pérdida

El modelo se debe ajustar a los ejemplos preestablecidos que se proporcionen y se utiliza la función de pérdida, que va a comparar el resultado de salida del modelo con el dato real de los ejemplos, para saber el desempeño de nuestro algoritmo y medir cuán alejado se encuentra el resultado del modelo con el real, saber esto ayuda a la función

de optimización para actualizar los valores del modelo y tener un resultado más cercano al real. Existen funciones de pérdida para datos de tipo continuo y categórico, se debe elegir la función que mejor se adecue al problema.

2.3.1. Entropía cruzada binaria

La entropía cruzada binaria (BCE, por las siglas en inglés *Binary Cross Entropy*) es una función de pérdida que trabaja con probabilidades y funciona para problemas de tipo categórico y al ser binaria solo va a funcionar en problemas con 2 clases. Esta función hace que las etiquetas de clase correctas de los ejemplos de entrenamiento sean más probables y modifica las variables para que las etiquetas correctas sean más factibles de que sucedan [11].

En (2.4) se muestra la ecuación de BCE, donde $\hat{y}$ representa el resultado de salida del perceptrón, y representa los valores reales de los ejemplos de entrenamiento y N representa el número de ejemplos de entrenamiento. Esta ecuación funciona diferente cuando la etiqueta es 0 y cuando es 1. Cuando la etiqueta es 0 la parte $y \cdot log(\hat{y})$ resultará 0 y solo utiliza la parte de $(1-y) \cdot log(1-\hat{y})$ y viceversa cuando la etiqueta es 1.

Debido a que los logaritmos van a obtener valores entre 0 y 1, darán como resultado valores negativos, por esto se debe ajustar con el primer signo negativo para que el resultado de BCE sea un valor positivo.

$$L(y, \hat{y}) = -\frac{1}{N} \sum y \cdot log(\hat{y}) + (1-y) \cdot log(1-\hat{y}) \tag{2.4}$$

2.3.2. Suavizado de etiquetas

La función BCE da una pérdida baja cuando acierta el modelo con una probabilidad alta, sin embargo, por error humano hay algunas etiquetas que están incorrectas y la pérdida con estos datos será particularmente alta, lo que en consecuencia causará problemas en el aprendizaje. El suavizado de etiquetas consiste en reducir ligeramente el valor de las etiquetas, por ejemplo, una etiqueta con valor 1 convertirlo en 0.9 y una etiqueta con valor 0 convertirlo a 0.1, esto hará que cuando se esté evaluando un dato mal etiquetado, la pérdida será más baja y por el contrario cuando la etiqueta sea correcta, se siga manteniendo un costo menor. En esencia, el suavizado de etiquetas ayudará al modelo a entrenar en torno a datos mal etiquetados y, en consecuencia, mejorará la precisión, solidez y rendimiento [12].

2.4. Propagación hacia atrás

La propagación hacia atrás (BP, por las siglas en inglés *Back-Propagation*) es un algoritmo que se utiliza en redes neuronales y permite que la información del costo de la función de pérdida fluya del final de la red hasta el inicio para calcular el gradiente y con

la función de optimización actualizar los pesos. Es necesario modificar los parámetros de la capa de salida y para hacer el ajuste se deben modificar los valores de las capas anteriores, de esta forma los gradientes de una capa van a depender de la capa que le precede, por esta razón el cálculo se realiza del final hacia el inicio. BP se refiere al método para calcular el gradiente utilizando derivadas y regla de la cadena[13].

2.5. Función de optimización

Los optimizadores son algoritmos que utilizan el gradiente obtenido por la función de pérdida para actualizar los valores de los pesos y el sesgo, como resultado, la siguiente iteración generalmente debe tener una pérdida menor. Las redes neuronales generalmente se entrenan mediante el uso de optimizadores iterativos basados en gradientes que simplemente llevan la función de costo a un valor muy bajo [13].

2.5.1. Adam

Adam es una función de optimización que está dirigido a problemas de aprendizaje automático con grandes conjuntos de datos[14]. El algoritmo tiende a converger considerablemente rápido, debido a que modifica el valor de la tasa de aprendizaje, por esto, se utiliza a menudo para los algoritmos de machine learning [15]. En (2.5) se muestran las ecuaciones del optimizador Adam para los pesos y en (2.6) para el sesgo, donde V_{dw} y V_{db} son variables que modifican su valor en cada iteración y su valor inicial es 0, β_1 y β_2 son hiperparámetros que se les asigna el valor según se convenga y van a determinar lo rápido o lo lento que avanza a una pérdida menor, generalmente β_1 es igual a 0.9 y β_2 es igual a 0.999, dw y db representan los gradientes de w y b respectivamente que son obtenidos por el algoritmo de BP. Si S_{dw} o S_{db} resultan ser 0, se utiliza la variable ε que es igual a 1 x 10^{-8} para evitar que el denominador sea 0.

$$V_{dw} = \beta_1 \cdot V_{dw} + (1-\beta_1) \cdot dw \qquad V_{db} = \beta_1 \cdot V_{db} + (1-\beta_1) \cdot db$$
$$S_{dw} = \beta_2 \cdot S_{dw} + (1-\beta_2) \cdot dw^2 \qquad S_{db} = \beta_2 \cdot S_{db} + (1-\beta_2) \cdot db^2$$

$$V_{dw} = \frac{V_{dw}}{(1-\beta_1)} \qquad V_{db} = \frac{V_{db}}{(1-\beta_1)} \tag{2.5}$$
$$S_{dw} = \frac{S_{dw}}{(1-\beta_2)} \qquad S_{db} = \frac{S_{db}}{(1-\beta_2)} \tag{2.6}$$

$$w = w - \alpha \frac{V_{dw}}{\sqrt{S_{dw}} + \varepsilon} \qquad w = w - \alpha \frac{V_{db}}{\sqrt{S_{db}} + \varepsilon}$$

2.6. Redes Neuronales Multicapa

Una red neuronal multicapa (MLP, por las singlas en inglés *Multilayer perceptron*) es un algoritmo de aprendizaje supervisado, donde una gran cantidad de perceptrones se agrupan en capas y solo existen conexiones directas entre ellos, esto proporciona al algoritmo potentes ventajas como el mapeo no lineal y la tolerancia al ruido [16]. Una capa representa una serie de η perceptrones y tiene 3 tipos de capas, la capa de entrada, las capas ocultas y la capa de salida. En la capa de entrada se introducen los datos para su procesamiento y el número de perceptrones será igual al número de variables que tenga el problema. Las capas ocultas estarán entre la entrada y la salida, se tienen κ capas ocultas y este número varía dependiendo el problema, cada capa oculta puede tener el mismo número de perceptrones o distinto número, para referirse a un perceptrón de la red se utiliza el superíndice para indicar la capa y el subíndice para indicar el número de perceptrón en esa capa. Solamente se utiliza una capa para la capa de salida y este puede variar el número de perceptrones dependiendo el tipo de problema. En la figura 2.4 se muestra la representación gráfica de una MLP. Todos los perceptrones de una capa estan conectados con todos los perceptrones de la siguiente capa y así sucesivamente hasta la capa de salida.

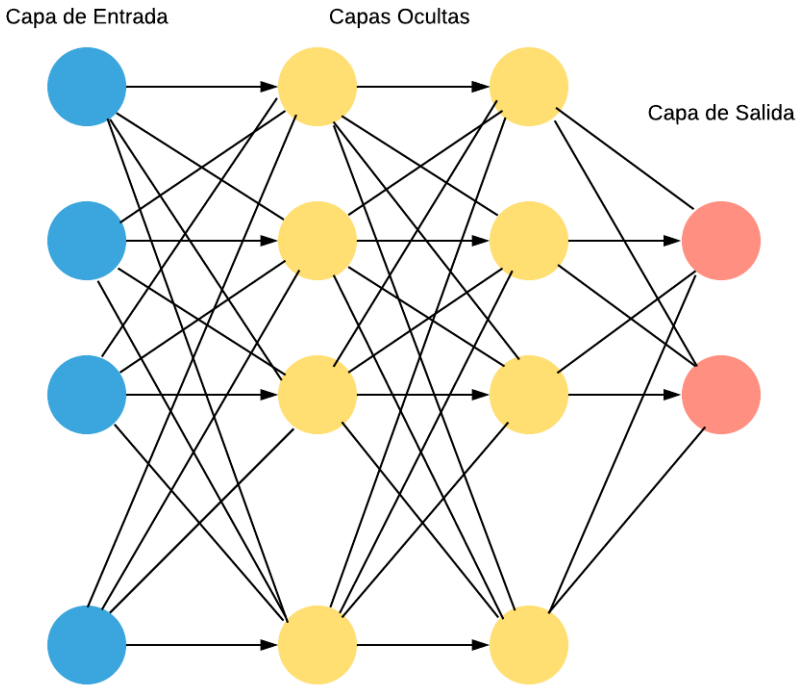

Figura 2.4: Representación gráfica de una red neuronal multicapa.

Cada conexión entre perceptrones cuenta con un único peso y cada capa oculta cuenta con un sesgo. Las capas ocultas y de salida pueden tener diferentes funciones de activación, al final se utiliza la función de pérdida que mejor se ajuste al problema. El algoritmo de BP va a funcionar de la capa final hacia el inicio de la red, cada capa va a obtener su gradiente de la capa siguiente y así sucesivamente hasta la capa de entrada.

2.7. Convolución

La convolución es una operación entre 2 funciones y se denota con un asterisco, en (2.7) se muestra la ecuación en caso discreto, donde x representa los datos de entrada w es el kernel o filtro, el cual le da más peso a los datos requeridos.

$$s(t) = (x * w)(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a) \tag{2.7}$$

En las aplicaciones de aprendizaje automático, la entrada suele ser una matriz multidimensional de datos y el kernel o filtro suele ser una matriz multidimensional de parámetros que se adaptan mediante el algoritmo de aprendizaje, estos arreglos multidimensionales se les conoce como tensores[13], la ecuación de convolución para matrices de dos dimensiones sería (2.8). El kernel debe ser volteado horizontalmente al operarse con la matriz de entrada y la convolución al ser conmutativa, la ecuación para matrices bidimensionales es (2.9).

$$s(i,j) = (I * K)(i,j) = \sum_{m}\sum_{n} I(m,n)K(i-m, j-n) \tag{2.8}$$

$$s(i,j) = (I * K)(i,j) = \sum_{m}\sum_{n} I(i+m, j+n)K(m,n) \tag{2.9}$$

2.8. Capa de submuestreo

La capa de submuestreo es importante porque ejecuta el muestreo descendente en los mapas de características procedentes de la capa anterior y produce nuevos mapas de características con una resolución condensada. Esta capa reduce drásticamente la dimensión espacial de la entrada y tiene dos propósitos principales.

El primero es reducir el número de parámetros o pesos, disminuyendo así el costo computacional. El segundo es controlar el sobreajuste de la red. Se espera que un método de agrupación ideal extraiga solo información útil y descarte detalles irrelevantes [17].

Existen distintos algoritmos para submuestrear, entre los más comunes son muestreo máximo y muestreo promedio.

- Muestreo máximo (*Max Pooling*) obtiene el valor máximo de una ventana de valores, con un alto y ancho específico
- Muestreo promedio (*Mean Pooling*) obtiene el promedio de de una ventana de valores, con un alto y ancho específico

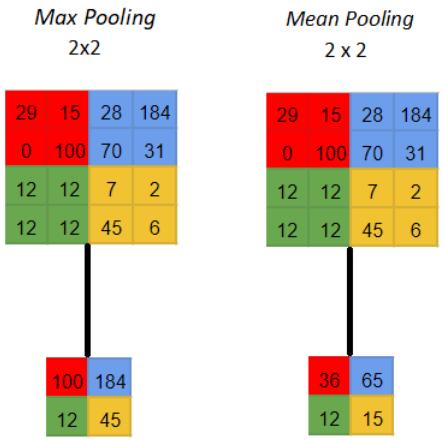

Figura 2.5: Imagen representativa de *Max Pooling* y *Mean Pooling*.

2.9. Normalización

La normalización es una aplicación en estadística que ajusta valores de variables a un rango en específico y en el entrenamiento del modelo ayuda que la media de todos los valores sea 0 y la desviación estándar 1. Generalmente las imágenes utilizan valores entre 0 y 255, con este rango de valores no se puede conseguir una media igual a 0 y una desviación estándar igual a 1, por esta razón es beneficioso aplicar una normalización a los datos [18]. La ecuación para normalizar es (2.10) donde x_i es el valor de la entrada que se va a normalizar, μ y σ^2 son la media y desviación estándar respectivamente del conjunto de datos a normalizar.

$$\hat{x}_i = \frac{x_i - \mu}{\sqrt{\sigma^2 + \varepsilon}} \tag{2.10}$$

2.10. *Dropout*

Dropout es una técnica que consiste en igualar a 0 las salidas de cierto número de perceptrones o valores de los filtros, este método es ampliamente usado en *deep learning*. El *dropout* ayuda a reducir el sobreajuste y evita que perceptrones sean dependientes de

otros. Los parámetros de los perceptrones que se eligen para el *dropout* se toman aleatoriamente y esto ha brindado grandes mejoras en tareas relacionadas para reconocimiento de objetos [19].

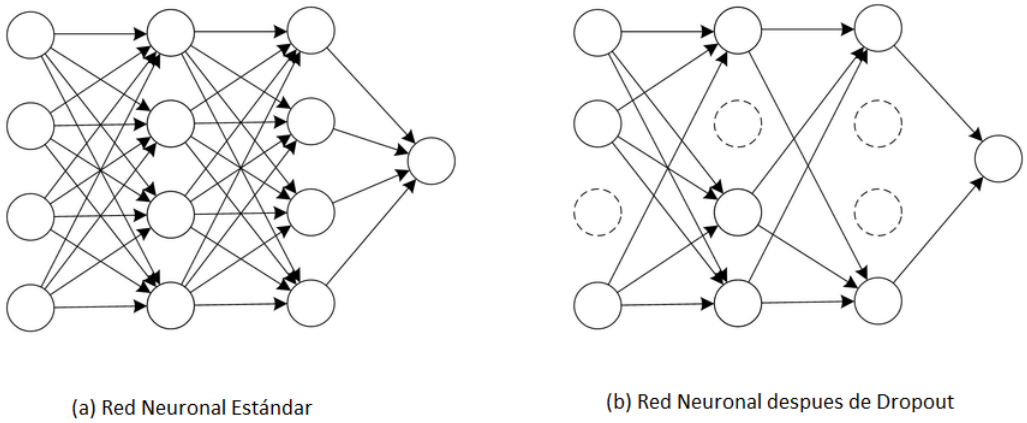

Figura 2.6: Imagen representativa de *Dropout*. Imagen tomada de [1].

2.11. *Padding* y *Stride*

El *padding* consiste en agregar un número apropiado de filas y columnas, generalmente con valores igual a 0, a cada lado de la matriz de características de entrada para que sea posible ajustar las ventanas de convolución central alrededor de cada mosaico de entrada [8]. Esto se utiliza porque al hacer la operación de convolución, se pierde información de los bordes de la imagen, al agregar filas y columnas a la imagen, se obtiene información de los bordes aunque ligeramente sesgadas debido a los valores 0 agregados.

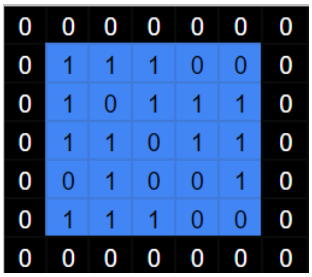

Figura 2.7: Imagen representativa de *Padding*

La distancia entre dos ventanas sucesivas es un parámetro de la convolución, llamado *stride*, que por defecto es 1 [8]. Al tener un valor alto de *stride*, se pierde información de

la imagen pero también se obtiene menor costo computacional porque no opera algunos renglones y columnas.

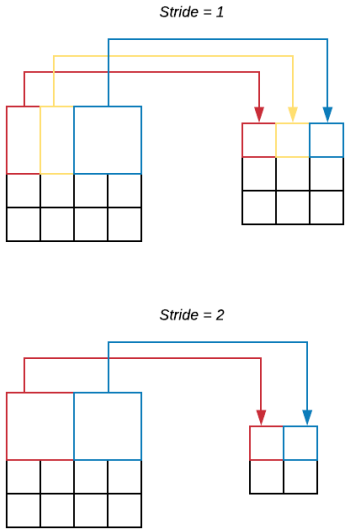

Figura 2.8: Imagen representativa de *Stride*.

2.12. Red Neuronal Convolucional

Las redes neuronales convolucionales (CNN, por las siglas en inglés *Convolutional Neural Networks*) son simplemente redes neuronales que utilizan la convolución en lugar de la multiplicación de matrices general en al menos una de sus capas. [13]. Las CNN son una herramienta prometedora para resolver el problema del reconocimiento de patrones [20]. Una capa de CNN está conformada por la etapa de convolución, la etapa de detección donde se aplica la función de activación y la etapa de submuestreo.

La etapa de convolución utiliza la imagen de entrada para hacer la operación de convolución con m filtros y después sumar el sesgo, el alto y ancho de los filtros tienen el mismo valor y se expresan con la variable f. El valor del alto y ancho de la matriz resultante de la convolución se obtiene de las ecuaciones (2.11) y (2.12), donde el superíndice indica la capa, p es el *padding*, f es el tamaño del filtro y s es el *stride*. Las dimensiones de la nueva matriz son $[n_H^\ell, n_W^\ell, m^\ell]$.

$$n_H^\ell = \frac{n_H^{\ell-1} + 2p^\ell - f^\ell}{s^\ell} + 1 \qquad (2.11) \qquad n_W^\ell = \frac{n_W^{\ell-1} + 2p^\ell - f^\ell}{s^\ell} + 1 \qquad (2.12)$$

La etapa de detección aplica la función de activación a todos los valores obtenidos en la etapa de convolución y el resultado pasa a la etapa de submuestreo, el nuevo valor de alto y ancho, n_{PH} y n_{PW} respectivamente, se obtienen de las fórmulas (2.13) y (2.14), donde n_H^ℓ Y n_w^ℓ son los valores obtenidos en la etapa de convolución, p y s son el padding y stride de la etapa de submuestreo y f es el ancho y alto de la ventana para submuestreo, las dimensiones resultantes de esta etapa son $[n_{PH}, n_{PW}, m]$, esta es la salida de la capa de convolución.

$$n_{PH} = \frac{n_H^\ell + 2p - f}{s} + 1 \qquad (2.13) \qquad n_{PW} = \frac{n_W^\ell + 2p - f}{s} + 1 \qquad (2.14)$$

La salida de la capa de convolución puede tener otra capa de convolución y el procedimiento se repite, generalmente las CNN tienen más de una capa de convolución y para los problemas de clasificación, en la última capa se aplica el método de aplanado que consiste en transformar una matriz en un vector. A la matriz resultante se le aplica un aplanado y se conecta con una MLP con el objetivo de que los últimos perceptrones puedan elegir una clase.

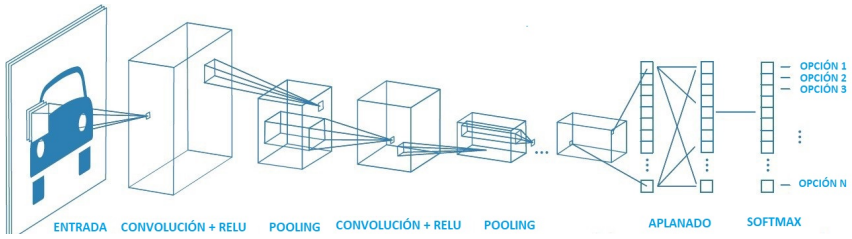

Figura 2.9: Imagen representativa de CNN. Imagen tomada de [2].

2.13. *Residual Network*

Las *residual networks* (ResNet) son un tipo de arquitectura de redes neuronales, las cuales son las más utilizadas en la actualidad [18]. Las redes neuronales profundas, mientras más capas tienen, son más difíciles de entrenar debido a que el gradiente tiende a desvanecerse, lo que significa que el gradiente llega a valores muy cercanos al 0 y la siguiente actualización prácticamente tendrá el mismo valor y no habrá un cambio sustancial.

Las ResNet pueden facilitar la formación de redes más profundas debido a su arquitectura [21], porque además de las conexiones consecutivas entre las capas, tiene conexiones no consecutivas como se muestra en la figura 2.10. Estas conexiones no consecutivas generalmente hacen los enlaces cada 2 capas, por esto, habrán capas que tengan información de la capa anterior y también de 2 capas atrás. En este caso si los valores de $\mathcal{F}(X)$ se desvanecen, la suma de X evita que los valores se hagan 0, esto

permite que los gradientes no se desvanezcan y puedan ser más profundas.

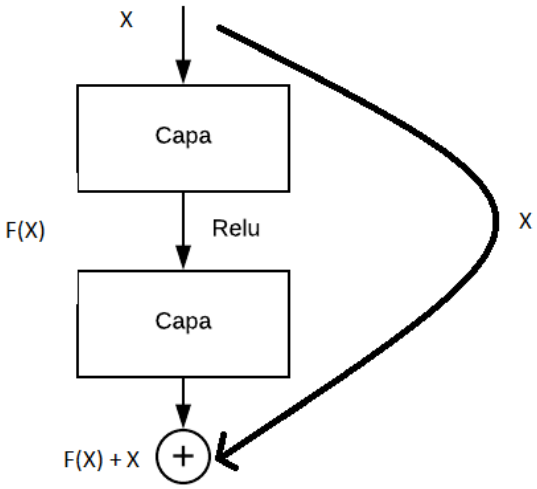

Figura 2.10: Conexión entre capas ResNet.

ResNet 152 es una red neuronal residual de 152 capas que es conocida por ser muy profunda y dar buenos resultados en reconocimiento de objetos y ganó el ILSVRC, una competencia de reconocimiento de objetos, en 2015 con 3.57 % de error en el conjunto de datos de prueba de ImageNet [21]. La ResNet 152 termina con un aplanado en su última capa convolucional y una capa de tipo MLP y el número de neuronas será igual al número de clases que se tenga en el problema.

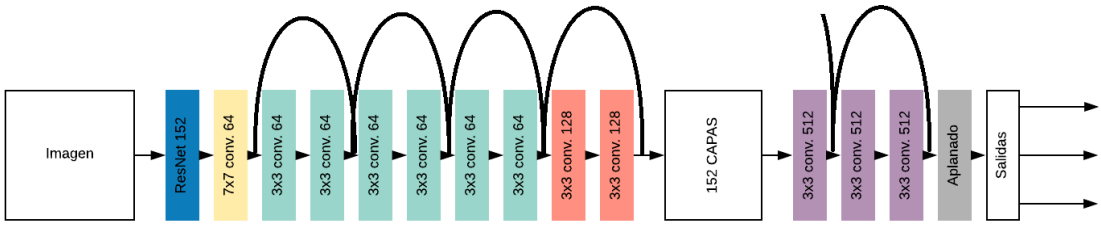

Figura 2.11: ResNet 152.

2.14. Aumento de datos

Las CNN dependen en gran medida de grandes cúmulos de datos para evitar el sobreajuste, desafortunadamente, muchos problemas de aplicaciones no tienen acceso a un alto número de imágenes, por ejemplo, análisis de imágenes médicas. Una solución para los datos limitados es el aumento de datos, que es un conjunto de técnicas que mejoran el tamaño y calidad de los datos. Estas técnicas pueden ser rotar, cortar,

saturar, acercar, alejar las imágenes entre otras [22].

Figura 2.12: Imagen representativa de Aumento de datos. Imagen tomada de [3].

2.15. *Learning rate finder*

La tasa de aprendizaje es el hiperparámetro más importante en el entrenamiento de una red neuronal por esto se debe elegir el mejor valor posible, si se elige un valor muy bajo el modelo puede no llegar a la pérdida más baja y si se elige un valor muy alto el modelo puede divergir y tener una pérdida alta.

El método de *learning rate finder* (LRF) consiste en aplicar varios valores en la tasa de aprendizaje con un modelo a un lote de datos, se empieza por un valor bajo y se va aumentando al doble el valor de la tasa de aprendizaje en cada iteración. Se mide la pérdida que se tiene con cada iteración y se detiene hasta que la pérdida deje de bajar. Con un lote de datos se puede obtener el comportamiento de todo el conjunto de datos. Para el entrenamiento se usa el valor que tuvo la menor pérdida y de esta forma se puede obtener una tasa de aprendizaje óptima para entrenar el modelo [23].

2.16. *Transfer Learning, Fine tuning* y *Freezing*

Transfer Learning (TL) es un método utilizado frecuentemente en *deep learning* que consiste en reutilizar parámetros entrenados para un problema nuevo. El estudio de *transfer learning* está motivado por el hecho de que las personas pueden aplicar de manera inteligente los conocimientos aprendidos previamente para resolver nuevos problemas de forma más rápida y con mejores soluciones[24]. En las CNN generalmente se utiliza TL con redes entrenadas con el dataset de ImageNet, que tiene 15 millones de imágenes de alta resolución y que cuenta con aproximadamente 22,000 categorías. Las

imágenes fueron recopiladas de internet y etiquetadas por personas [25].

Fine tuning es un proceso que se utiliza después de hacer TL y consiste en tener un algoritmo pre-entrando y ajustarlo a los datos que requiera el problema, en redes neuronales se ajustan las capas ocultas y la capa de salida. Esto es un ajuste de modelo en *deep learning* y es un método de aprendizaje sencillo y eficaz [26].

Freezing es una técnica utilizada en modelos de *deep learning* que consiste en no modificar los valores de los parámetros de las capas ocultas en el entrenamiento, solamente se van a actualizar los parámetros de la capa de salida. Esta técnica se utiliza en CNN después de usar *fine tuning* debido a que la capa de salida se asignan perceptrones con pesos aleatorios y sirve para ajustar esos pesos sin alterar la información de las capas ocultas [18].

2.17. Métricas

2.17.1. Matriz de confusión

La matriz de confusión es una medida popular para problemas de clasificación, puede ser aplicado para clasificación binaria o multiclase. Se crea una matriz con renglones y columnas de todas las clases que tiene el problema, las clases de los renglones indican la categoría real y las clases de las columnas indican la categoría predicha.

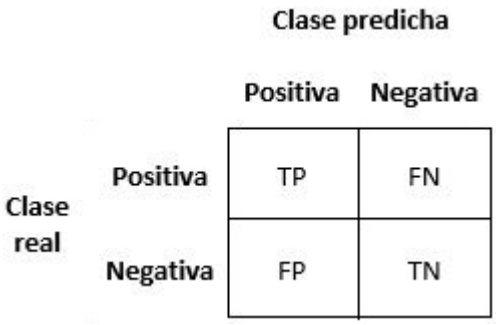

Figura 2.13: Ejemplo de Matriz de confusión.

En la matriz se indican los verdaderos positivos (TP, por las siglas en inglés *True Positive*), verdadero negativos (TN, por las siglas en inglés *True Negative*), falsos positivos (FP, por las siglas en inglés *False Positive*) y falsos negativos (FN, por las siglas en inglés *False Negative*) [27]. Los TP y TN indican las predicciones correctas que hizo el modelo, por lo que FP y FN indican las predicciones incorrectas.

2.17.2. Exactitud

La exactitud (*Accuracy*) es una métrica para evaluar un modelo de clasificación e indica cuántas predicciones detectó correctamente y se calcula con el número total de predicciones correctas dividido por el número total de predicciones.

$$Exactitud = \frac{TP + TN}{TP + FN + FP + TN} \tag{2.15}$$

2.17.3. Precisión

La precisión es una métrica que indica que tan probable es que sea correcta una predicción dada y se obtiene con las predicciones correctas de la clase dividido por las predicciones correctas de la clase más los falsos positivos.

$$Precisin = \frac{TP}{TP + FP} \tag{2.16}$$

2.17.4. Exhaustividad

La exhaustividad (*Recall*) es una métrica que indica la probabilidad de que detecte correctamente una clase y se calcula con las predicciones correctas de la clase dividido por las predicciones correctas de la clase más los falsos negativos.

$$Exhaustividad = \frac{TP}{TP + FN} \tag{2.17}$$

2.17.5. Valor F1

El valor F1 es otra medida de exactitud y se calcula utilizando la precisión y la exhaustividad, esta medida es útil cuando las clases tienen distinto número de ejemplos.

$$ValorF1 = 2 * \frac{precisión * exhaustividad}{precisión + exhaustividad} \tag{2.18}$$

2.18. FastAI

FastAI es una biblioteca de aprendizaje profundo que proporciona componentes de alto nivel y su objetivo es tener el estado del arte en algoritmos de aprendizaje profundo, proporcionando facilidad de uso y rendimiento. Esta biblioteca tiene la ventaja de que aprovecha el dinamismo del lenguaje de programación Python y la flexibilidad de la biblioteca PyTorch, utilizada para algoritmos de *deep learning*.

Capítulo 3

Base de datos

La base de datos está compuesta por imágenes rayos X de tórax de pulmones con COVID-19, neumonía y sanos, solo se cuenta con 3 opciones para clasificar y las imágenes provienen de México y varios países. Hay un amplio rango de edades de los pacientes y se tiene hombres y mujeres, por lo que se tiene diversidad en las imágenes, lo que significa que no están sesgados los datos.

La base de datos contiene dos modalidades de imágenes rayos X de tórax, las anteroposteriores (AP) que se toman del pecho hacia la espalda por lo que se verá la parte delantera del tórax y las posteroanteriores (PA) que son lo opuesto, se toman de la espalda hacia el pecho y se verá la parte trasera del tórax.

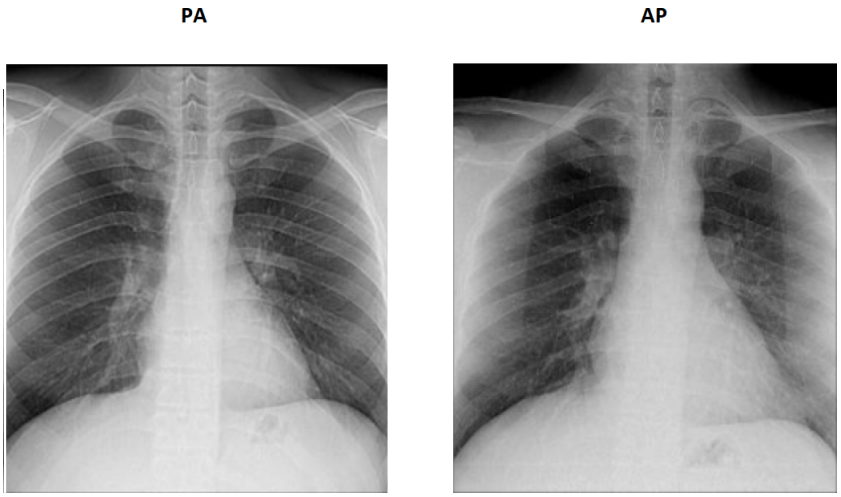

Figura 3.1: Imágenes rayos X de tórax PA y AP.

3.1. COVID-19

Las imágenes de COVID-19 recabadas se tomaron de diferentes sitios, siendo las más importantes las que provienen del Instituto Mexicano del Seguro Social (IMSS), Cohen y las recopiladas por la universidad Qatar junto con la universidad de Dhaka.A continuación se especifica más sobre sus características y su origen. La importancia de tener la disponibilidad de todas estas bases de datos es que se puede desarrollar investigación de cualquier parte del mundo.

3.1.1. Instituto Mexicano del Seguro Social

Se obtuvieron 3 lotes de imágenes rayos X de tórax de pulmones con COVID-19 por hospitales pertenecientes al IMSS, el primer lote de imágenes provino del Centro Médico Nacional La Raza (La Raza) y del Hospital General de Zona 48 de las cuales fueron 24 y 36 respectivamente. Los pacientes están entre el rango de edad de 15 a 73 años, hay personas de ambos sexos y se cuenta con 37 imágenes PA y 23 AP.

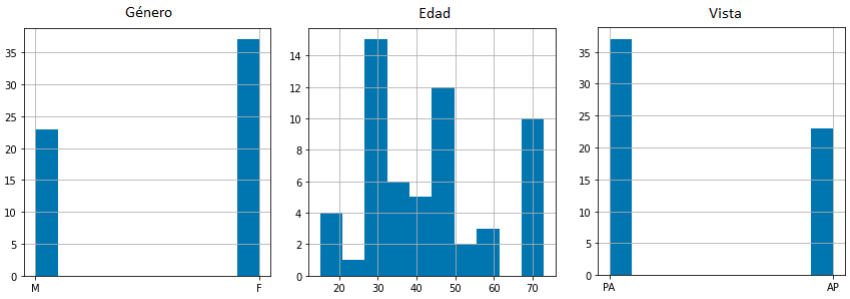

Figura 3.2: Histogramas lote 1 IMSS.

El segundo lote cuenta con 219 imágenes provenientes del hospital de infectología "Dr. Daniel Méndez Hernández" en La Raza. Hay personas de ambos sexos y se encuentran entre el rango de edad de 7 a 90 años y solamente se cuentan con imágenes AP.

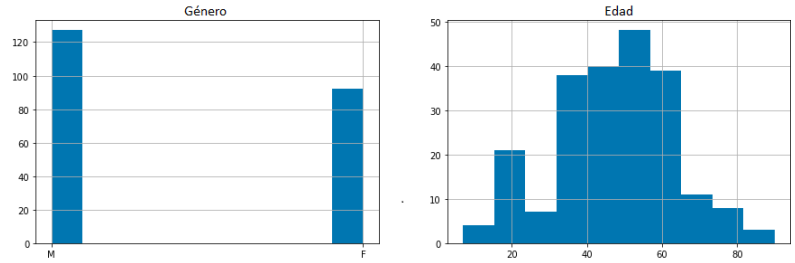

Figura 3.3: Histogramas lote 2 IMSS.

El tercer lote tiene 153 imágenes que fueron proporcionadas por el hospital de infectología de la raza, las personas se encontraron entre los 28 y los 88 años de edad, 104 pacientes fueron hombres y 49 mujeres, todas las imágenes son AP.

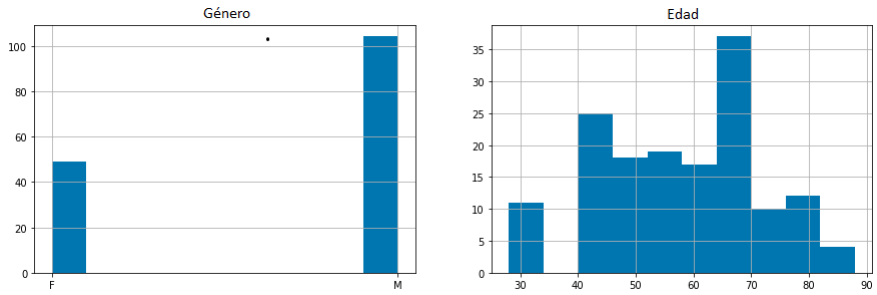

Figura 3.4: Histogramas lote 3 IMSS.

3.1.2. Cohen

Investigadores provenientes del Instituto Montreal para el aprendizaje de algoritmos, la facultad de medicina de la Universidad de Montreal y el departamento de matemáticas de la Universidad de Fontbonne crearon un repositorio en GitHub de imágenes rayos X de tórax de casos positivos de COVID-19, las imágenes las obtuvieron de bases de datos abiertas [28]. Este repositorio cuenta con 429 imágenes rayos X de tórax de COVID-19, los pacientes están en el rango de edad de 12 a 94 años y hay mujeres y hombres dentro de la base de datos.

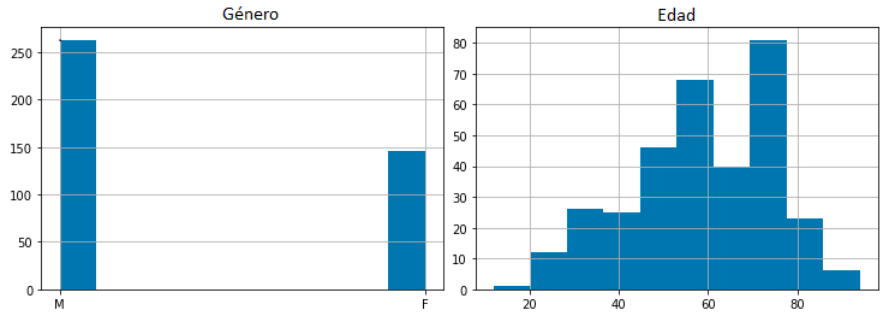

Figura 3.5: Histogramas GitHub.

3.1.3. Universidad de Qatar y Universidad de Dhaka

Un equipo de investigadores de la Universidad de Qatar, Doha, Qatar, y la Universidad de Dhaka, Bangladesh, junto con colaboradores de Pakistán y Malasia en colaboración con médicos, han creado una base de datos de imágenes de rayos X de tórax para casos positivos de COVID-19.

Esta base de datos cuenta con imágenes del Banco de Datos de Imagen Médica de la Comunidad de Valencia, la Escuela de Medicina de Hannover, la Sociedad Italiana de Radiología Médica e Intervencionista, de la Sociedad Radiológica Europea (EURORAD) e imágenes de repositorios como Kaggle y Github. Son 3,616 datos en total y hay imágenes rayos X AP y PA las cuales están en formato png y tienen una resolución de 256×256 píxeles [29] [30].

3.1.3.1. Banco de Datos de Imagen Médica de la Comunidad Valenciana

El Banco de Datos de Imagen Médica de la Comunidad Valenciana (BIMCV) es un banco de imágenes médicas que obtiene los datos de los hospitales de Valencia, tienen el objetivo de integrar imágenes médicas para el desarrollo de proyectos de tecnología relacionados con imágenes médicas.

3.1.3.2. Escuela de Medicina de Hannover

La escuela de medicina de Hannover en Alemania es una universidad con un hospital de máxima atención con una zona de captación internacional. En la universidad se imparten medicina, odontología, bioquímica, biomedicina, partería y ciencias de la salud. Las principales áreas de investigación son la investigación de trasplantes y células madre, la investigación de infecciones e inmunidad, así como la tecnología biomédica y la investigación de implantes.

3.1.3.3. Sociedad Italiana de Radiología Médica e Intervencionista

La Sociedad Italiana de Radiología Médica e Intervencionista (SIRM) es la sociedad de médicos radiólogos y radiólogos intervencionistas y, en 2021, cuenta con más de 11.000 miembros, que representan una de las principales sociedades científicas italianas y europeas, acreditada por el Ministerio de Salud.

3.1.3.4. Sociedad Radiológica Europea

EURORAD es una base de datos revisada por radiológicos, proporcionada y operada por la Sociedad Europea de Radiología. Es una herramienta educativa con el propósito de ayudar con un entorno de aprendizaje para radiólogos, residentes de radiología y estudiantes.

3.1.4. Universidad de Waterloo

El laboratorio de visión y procesamiento de imágenes de la Universidad de Waterloo junto con la empresa de inteligencia artificial DarwinAI y la universidad Jaime I crearon una base de datos de imágenes rayos X de tórax de personas con COVID-19. El conjunto

cuenta con 58 imágenes de pacientes con COVID-19 y hay 32 en formato PA y 26 en AP.

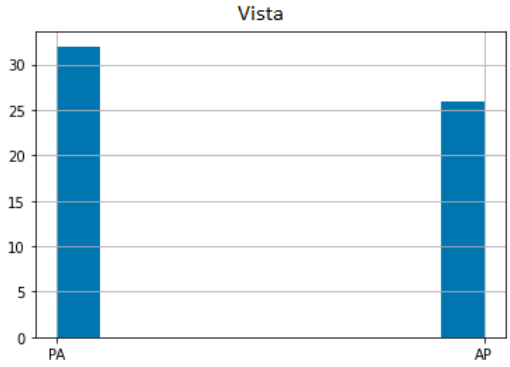

Figura 3.6: Formato imágenes universidad de Waterloo.

3.1.5. Hospital Universitario San Cecilio

Un grupo de radiólogos del hospital universitario San Cecilio proporcionó una base de datos de imágenes rayos X de tórax de personas con COVID-19 y de personas sanas, el conjunto cuenta con 852 imágenes de las cuales 456 pertenecen a la clase COVID-19 y 456 a la clase sano, las imágenes solo vienen en formato PA y están categorizadas en suaves, moderadas y severas dependiendo el daño que se tenga en los pulmones [31].

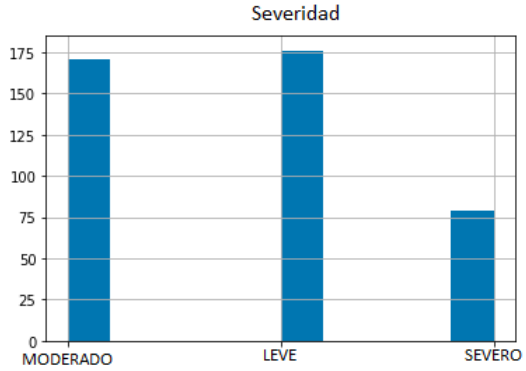

Figura 3.7: Histograma severidad de daño en pulmones por COVID-19.

3.2. Neumonía y Sano

Las imágenes obtenidas para las clases neumonía y sano se consiguieron de distintas fuentes, principalmente de institutos de Estados Unidos como la Sociedad Radiológica

de América del norte y el Instituto Nacional de Salud, estas instituciones cuentan con una gran base de datos, estas imágenes se pueden obtener de forma gratuita y esto beneficia al desarrollo de algoritmos de inteligencia artificial porque cualquier persona puede desarrollar.

3.2.1. Radiological Society of North America

La Sociedad Radiológica de América del Norte (RSNA) es una organización sin fines de lucro que cuenta con 31 subespecialidades radiológicas de 145 países. La RSNA proporcionó una base de datos de imágenes rayos X de tórax, la cual contenía 9,555 imágenes de pulmones con neumonía y 8,851 de imágenes de pulmones sanos [32].

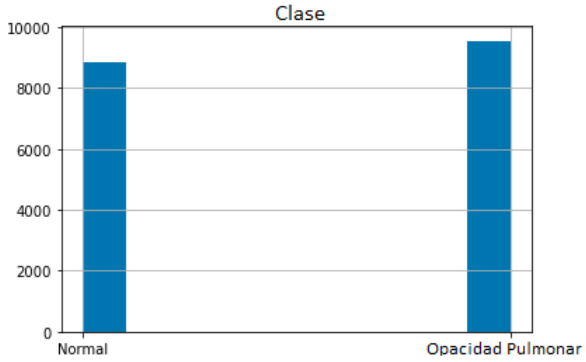

Figura 3.8: Histogramas RSNA.

3.2.2. Universidad de California San Diego

La universidad de California San Diego junto con el Centro Médico para Mujeres y Niños de Guangzhou obtuvo una base de datos con 5,856 imágenes rayos X de tórax con neumonía y sanos. Del total de imágenes, 4,273 imágenes pertenecen a la clase neumonía y 1,583 imágenes pertenecen a la clase sano, las imágenes son AP y PA [33].

3.2.3. National Institutes of Health

Los Institutos Nacionales de Salud (NIH) que son parte del departamento de salud y servicios humanos en Estados Unidos recolectó 112,120 imágenes de rayos X de tórax de 30,805 pacientes de distintos hospitales en el país. La base de datos cuenta con 13 tipos de enfermedades y daños en los pulmones, entre los cuales están atelectasia, consolidación, infiltración, neumotórax, edema, enfisema, fibrosis, efusión, neumonía, engrosamiento pleural, cardiomegalia, nódulo y hernia. Estas imágenes también contienen la clase sano, las cuales son 60,361 y de la clase neumonía son 150. La base de datos

cuenta con imágenes de personas de hasta 95 años, hay hombres y mujeres y hay de tipo AP y PA [34].

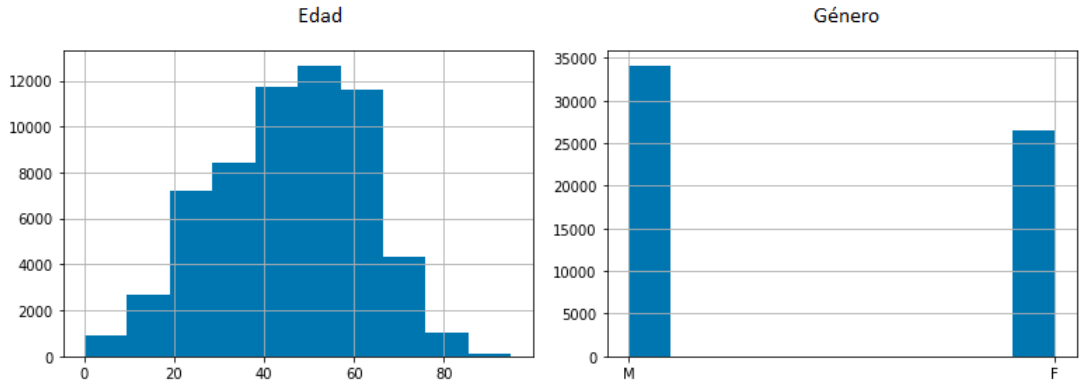

Figura 3.9: Histogramas NIH.

3.2.4. Hospital Angeles Lindavista

Se recuperaron 220 imágenes rayos X de tórax con pulmones sanos del hospital Angeles Lindavista, los pacientes tienen entre 4 y 58 años, son 119 hombres y 101 mujeres. Las imágenes son 215 PA y 5 AP.

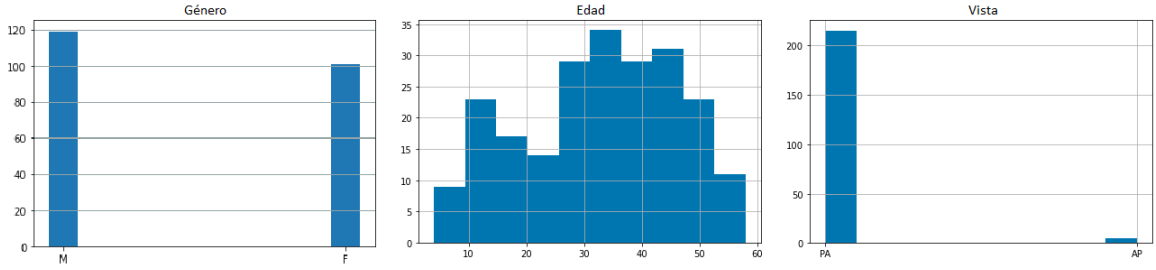

Figura 3.10: Histogramas Angeles Lindavista.

Capítulo 4

Metodología

Para llevar a cabo este proyecto fue necesario investigar diferentes modelos de arquitectura de redes neuronales convolucionales e identificar cuál es la que mejor resultados entrega para imágenes médicas. Se consiguieron bases de datos de diversas fuentes para que el sistema sea diverso en cuanto a que el aprendizaje no esté sesgado, los datos son imágenes rayos X de tórax de pulmones y las imágenes obtenidas pertenecen a 3 clases, las cuales son COVID-19, neumonía y pulmones sano, cada imagen sólo pertenece a una clase.

En este proyecto se entrenaron 2 ResNet 152 en forma de cascada para hacer la detección de enfermedades en pulmones, la primer ResNet se entrenó para que pudiera detectar pulmones sanos y enfermos, por lo que su salida es binaria, en este primer entrenamiento se seleccionaron las 4,121 imágenes rayos X de tórax con pulmones sanos y se utilizaron 4,490 imágenes rayos X de tórax de pulmones enfermos de neumonía o COVID-19.

La segunda ResNet se entrenó para que pudiera diferenciar entre imágenes rayos X de tórax con neumonía y COVID-19, se eligieron 2,340 imágenes para la clase COVID-19 y 2,150 imágenes para la clase neumonía.

Ambas redes utilizaron para COVID-19 las 432 imágenes proporcionadas por el IMSS, las 58 imágenes de la universidad de Waterloo, 200 imágenes seleccionadas al azar del Hospital Universitario San Cecilio, 250 de la base de datos de Cohen que se seleccionaron aleatoriamente y 1,400 de la recolección de imágenes de la Universidad de Qatar y la Universidad de Dhaka. Para neumonía se obtuvieron 150 imágenes de la base de datos de NIH, 900 imágenes de la Universidad de California San Diego y 1100 imágenes de RSNA, todas las imágenes se eligieron aleatoriamente de sus respectivas bases de datos. En la clase sano se utilizaron las 221 imágenes del hospital Angeles Lindavista, 900 de la Universidad de California San Diego, 1500 imágenes de RSNA y NIH cada una, se eligieron de forma aleatoria.

Se entrenó una tercera red neuronal que pudiera ser capaz de diferenciar entre imágenes rayos X de pulmones con neumonía, COVID-19 y sanos, a esta red neuronal se le colocaron 3 neuronas en la capa final por lo que su detección fue multiclase. Para el entrenamiento se utilizaron las mismas imágenes de COVID-19 y neumonía que las 2

ResNet anteriores y para las imágenes de la clase sano se seleccionaron las 221 imágenes del hospital Angeles Lindavista y aleatoriamente se seleccionaron 400 de la universidad California San Diego, 800 de RSNA y 600 de NIH.

El proyecto se programó en el lenguaje de programación Python y se utilizó la biblioteca FastAI para la implementación de las redes neuronales convolucionales, así como todos los hiperparámetros requeridos.

4.1. Redes Neuronales en Cascada

La forma de cascada utiliza 2 redes neuronales con salidas binarias en vez de utilizar una sola red neuronal con salidas múltiples. Se combinaron 2 ResNet 152, el proceso consiste en primero utilizar la ResNet que detecta pulmones sanos y enfermos, si la red lo detecta como sano ahí termina la detección y se queda en la clase sano. Si la red neuronal lo detecta como enfermo, pasa a la ResNet entrenada para detectar neumonía y COVID-19 y hará el proceso para detectar la enfermedad adecuada y será la clase detectada.

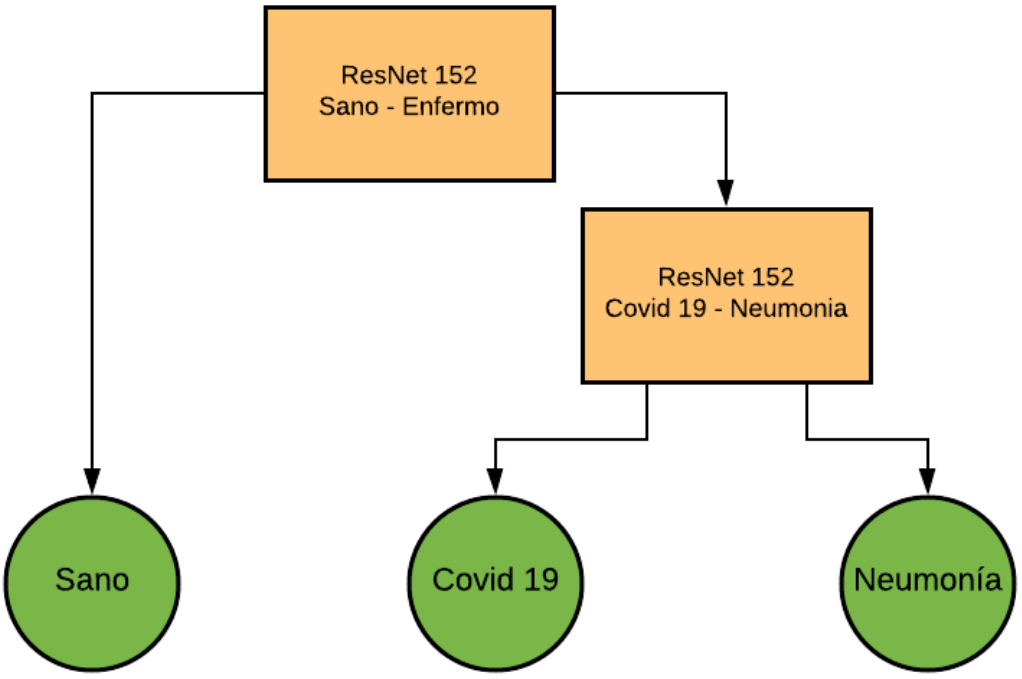

Figura 4.1: Diagrama Escalera 2 ResNet.

4.1.1. Clasificación de imagen Sano y Enfermo

Se utilizaron 8,611 imágenes rayos X de tórax para la detección de pulmones sanos y enfermos, de las cuales 4,121 pertenecen a la clase sano y 4,490 a la clase enfermo.

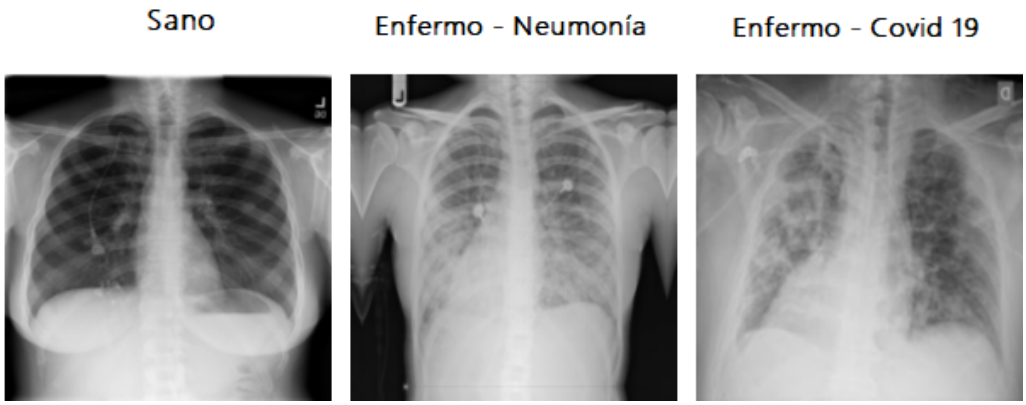

Figura 4.2: Imágenes rayos X de tórax de pulmones sanos, con neumonía y con COVID-19.

Las imágenes se redimensionaron a 224 x 224 píxeles y se utilizaron lotes de 32 imágenes para su entrenamiento, esto significa que al entrenar cada 32 imágenes se obtendrá el error de ese lote de imágenes y con esa información se actualizarán los valores de los parámetros.

Se dividió el conjunto total de imágenes en 2 grupos de datos, los cuales serían para el entrenamiento y la validación. Se utilizaron 6,944 imágenes para el entrenamiento y 1,667 imágenes para validación.

Para el entrenamiento se utilizó una red neuronal convolucional ResNet 152 y el método de *transfer learning* de una ResNet 152 pre entrenada con la base de datos de ImageNet para hacer *fine tuning* con la base de datos de imágenes rayos X de pulmones sanos y enfermos. Se utilizaron las métricas de exactitud, precisión, exhaustividad y valor F1 para medir el aprendizaje del algoritmo.

Las imágenes se normalizan para que el aprendizaje no se haga lento en las capas profundas. Se usó BCE con suavizado de etiquetas como función de pérdida.

Para identificar pulmones sanos y enfermos se le agregaron 2 neuronas al final de la ResNet 152, las cuales van a funcionar para elegir las 2 clases, estos valores se colocan de forma aleatoria. Es necesario ajustar esos valores de las 2 neuronas debido a que todas las capas de la ResNet ya están ajustados y no son valores aleatorios, para ajustar los valores se utilizó la técnica de *freezing*, el método de *learning rate finder* y se entrenó una época con el valor obtenido de LRF que fue de 0.0001.

En las figuras 4.3, 4.4 y 4.5 se muestran los *learning rate finder* obtenidos a través del entrenamiento y se explica los parámetros utilizados en cada etapa.

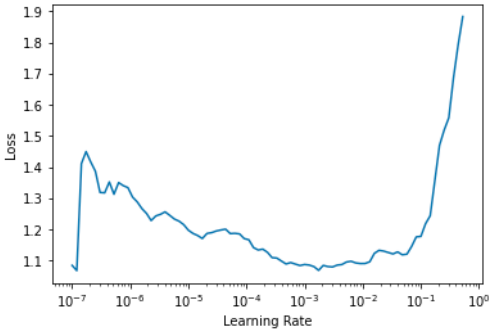

Figura 4.3: Primer *Learning Rate Finder* Sano y Enfermo.

Sin la técnica de *freezing* se volvió a utilizar el método de LRF para obtener el valor máximo de la tasa de aprendizaje para que se tenga la menor pérdida y el resultado fue $3.98x10^{-7}$ y se entrenó por 5 épocas.

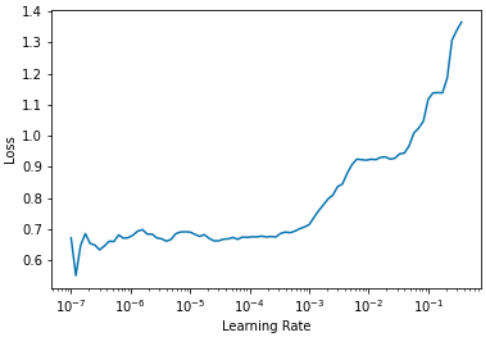

Figura 4.4: Segundo *Learning Rate Finder* Sano y Enfermo.

Por último se volvió a utilizar la técnica de *freezing* y se utilizó el método de LRF dando como resultado $6.918x10^{-7}$ y se entrenó por 1 época.

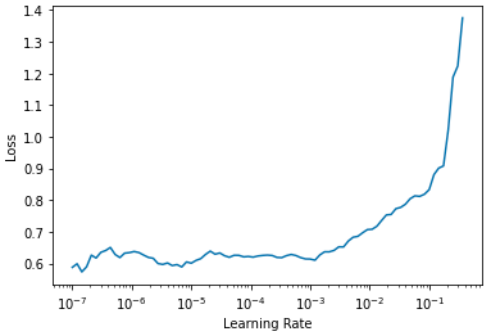

Figura 4.5: Tercer *Learning Rate Finder* Sano y Enfermo.

En la figura 4.6 se muestra de forma clara el diagrama del proceso del entrenamiento para la detección de pulmones sanos o enfermos con imágenes rayos X de tórax.

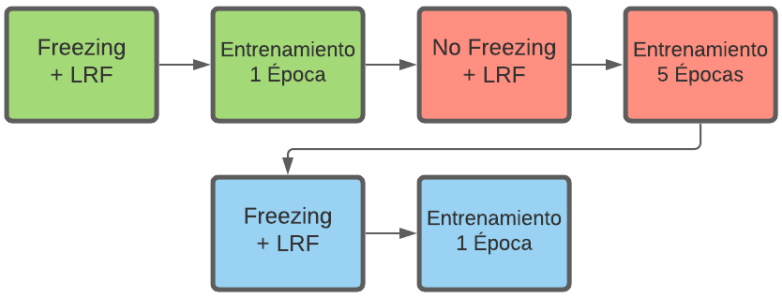

Figura 4.6: Proceso de Entrenamiento Sano y Enfermo.

4.1.2. Clasificación de imagen Neumonía y COVID-19

El entrenamiento para detectar imágenes rayos X de tórax con pulmones con neumonía o COVID-19 contó con 4,490 imágenes, de las cuales 2,150 pertenecen a la clase de neumonía y 2,340 son de la clase COVID-19.

Las imágenes se redimensionaron a 224 x 224 píxeles y se sustituyó el 20 % de las imágenes para aumento de datos, el cual consistía en aplicar un acercamiento, alejamiento, voltear horizontalmente, rotar y agregar más iluminación. Al rotar la imagen se hizo con un máximo de 10^o y podía ser hacia la derecha o hacia la izquierda. El alejamiento y acercamiento se hizo con un máximo de 10 % para ambos.

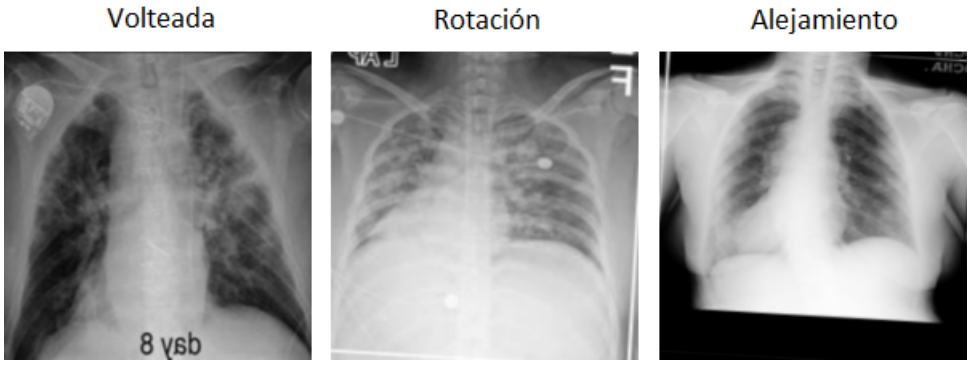

Figura 4.7: Aumento de datos de volteada, rotación y alejamiento.

Las imágenes se dividieron en 2 conjuntos de datos para hacer el entrenamiento y la validación, se seleccionaron 3,647 imágenes para el entrenamiento y 843 imágenes para la validación.

Se utilizó una red neuronal convolucional ResNet 152 con *transfer learning* de una ResNet 152 con los datos de ImageNet, con las imágenes de neumonía y COVID-19 se hizo *fine tuning* para ajustar la red neuronal. Al medir los resultados y rendimiento de la ResNet 152 se utilizaron las métricas de exactitud, precisión, exhaustividad y valor F1.

Las imágenes se dividieron en lotes de 32 imágenes y se normalizaron en el entrenamiento para evitar que se haga lento el entrenamiento y se usó BCE con suavizado de etiquetas como función de pérdida.

Se usó la técnica de *freezing* en la ResNet y se aplicó el método de *learning rate finder* para encontrar el valor adecuado de la tasa de aprendizaje, el cual fue de 0.00575. Se entrenó la red neuronal por una época para ajustar los valores de las neuronas que van a elegir entre neumonía y COVID-19.

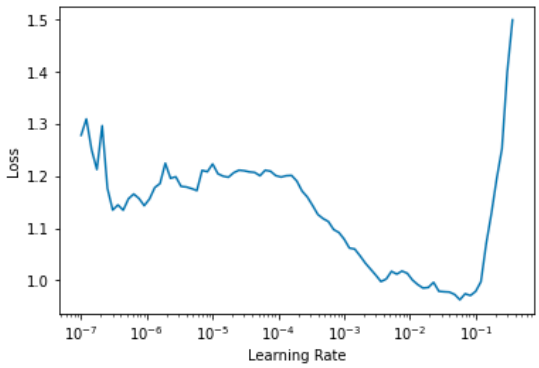

Figura 4.8: Primer *Learning Rate Finder* Neumonía y COVID-19.

Se colocó la red neuronal sin *freezing* y se obtuvo la tasa de aprendizaje con LRF el cual fue de $9.12x10^{-6}$ y se entrenó la ResNet 152 por 3 épocas.

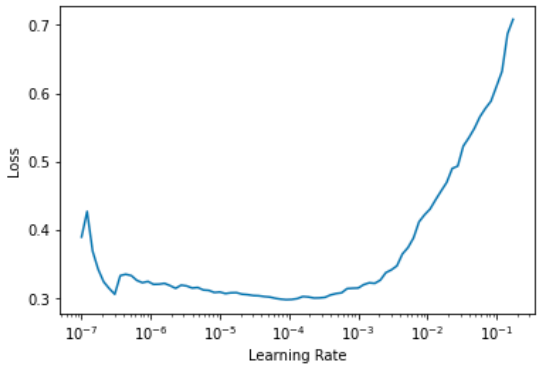

Figura 4.9: Segundo *Learning Rate Finder* Neumonía y COVID-19.

Se continuó sin la técnica de *freezing* y se usó LRF para obtener la tasa de apren-

dizaje, que fue de $6.3x10^{-8}$ y se entrenó por 2 épocas.

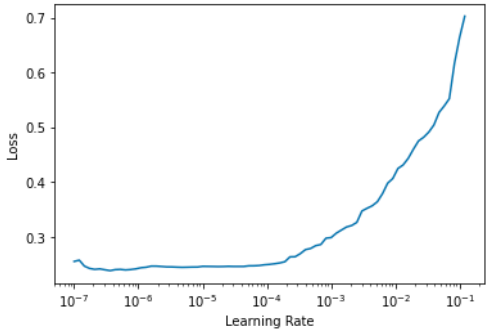

Figura 4.10: Tercer *Learning Rate Finder* Neumonía y COVID-19.

Se le volvió a aplicar el método LRF el cual fue de $7.58x10^{-8}$ y se entrenó por 1 época.

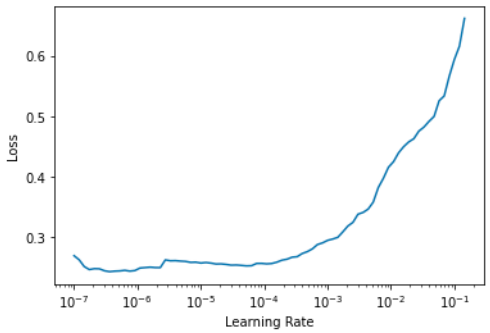

Figura 4.11: Cuarto *Learning Rate Finder* Neumonía y COVID-19.

Por último se utilizó *freezing* a la red neuronal, se le aplicó LRF que fue de $6.3x10^{-8}$ y se entrenó por una época.

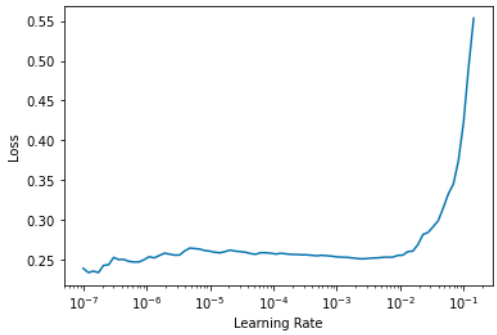

Figura 4.12: Quinto *Learning Rate Finder* Neumonía y COVID-19.

En la figura 4.13 se representa con un diagrama los pasos realizados para el entrenamiento de la ResNet 152 para la detección de pulmones con neumonía y COVID-19 en imágenes rayos X de tórax.

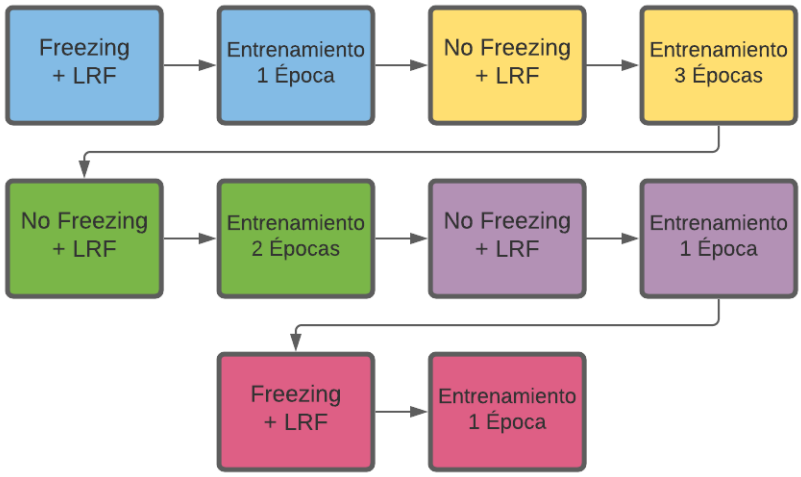

Figura 4.13: Proceso de Entrenamiento Neumonía y COVID-19.

4.2. Clasificación de imagen Sano, Neumonía y COVID-19

Se utilizó una ResNet 152 para la detección de las 3 clases y se usaron para el entrenamiento imágenes rayos X de tórax, 2,340 con COVID-19, 2,150 con neumonía y 2,021 sanas, en esta red se utilizaron menos imágenes sanas a diferencia de las 2 ResNet anteriores para que las 3 clases tengan un número balanceado de datos y no afecte el entrenamiento.

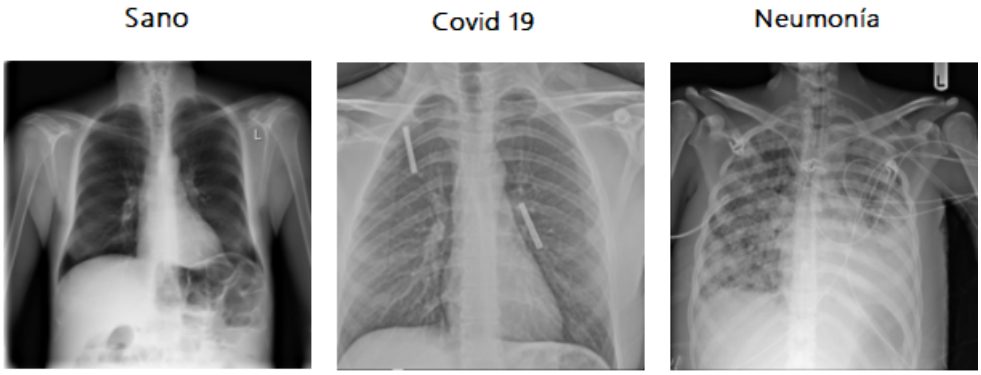

Figura 4.14: Imágenes de las clases Sano, Neumonía y COVID-19.

Las imágenes se redimensionaron a 224 x 224 píxeles y se dividieron en lotes de 32 imágenes cada uno, se seleccionaron 5,264 imágenes para el entrenamiento y 1,247 para la validación. La ResNet 152 se utilizó con *transfer learning* de una ResNet 152 preentrenada con la base de datos de ImageNet y se aplicó *fine tuning* para ajustar la red para la detección de las 3 clases. Se utilizó la métrica de exactitud para medir la eficacia de las detecciones.

En el entrenamiento primero se utilizó la técnica de *freezing* y el método de *learning rate finder* para que los valores de las 3 neuronas de la última capa puedan tener más precisión en la detección, el LRF fue de 0.0069 y se entrenó por 1 época.

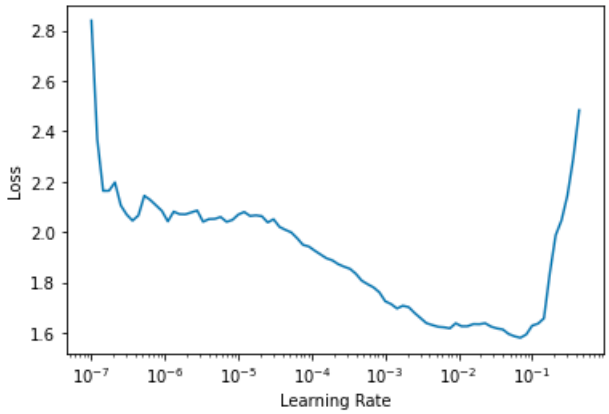

Figura 4.15: Primer *Learning Rate Finder* Sano, Neumonía, COVID-19.

El siguiente paso fue quitar la técnica de *freezing* y se usó LRF para poder ajustar los valores de las 152 capas de la ResNet, el LRF fue de $5.248x10^{-6}$ y se entrenó por 3 épocas.

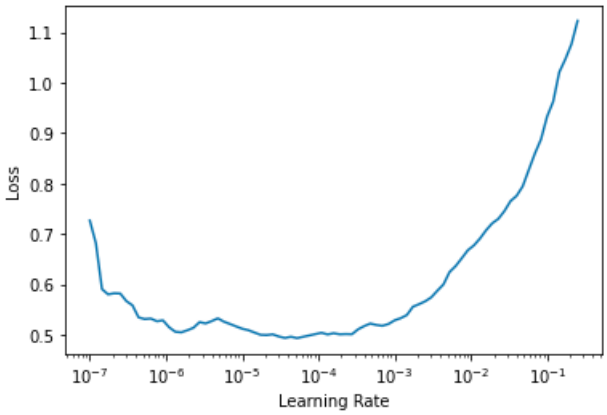

Figura 4.16: Segundo *Learning Rate Finder* Sano, Neumonía, COVID-19.

Por último se entrenó la red con 2 épocas, para la primer época se repitió el mismo

proceso que el paso anterior, se obtuvo el LRF sin *freezing* y para la segunda época se utilizó *freezing*, se tuvo una tasa de aprendizaje óptima con LRF y se entrenó por 1 época. Los LRF's fueron de $1.096x10^{-7}$ y $3.019x10^{-6}$ respectivamente.

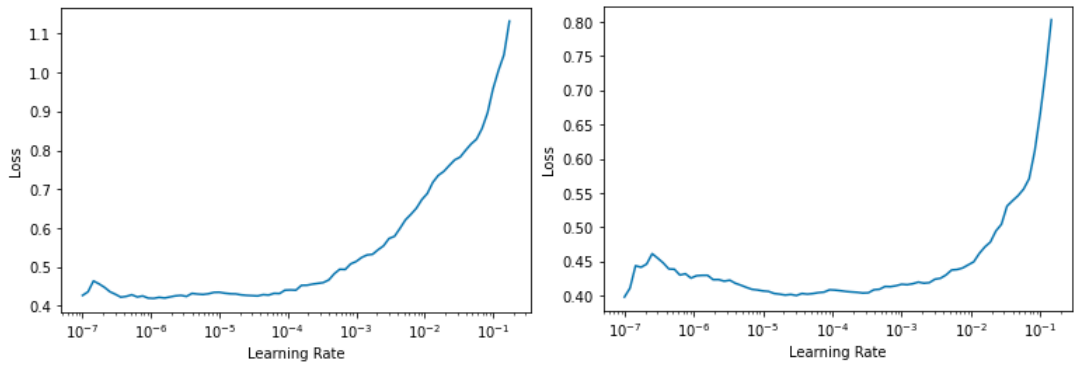

Figura 4.17: Tercer y Cuarto *Learning Rate Finder* Sano, Neumonía, COVID-19.

Para una mejor visualización del proceso del entrenamiento, en la figura 4.18 se muestran los pasos que se siguieron para crear la ResNet 152 para la detección de neumonía, COVID-19 y sano.

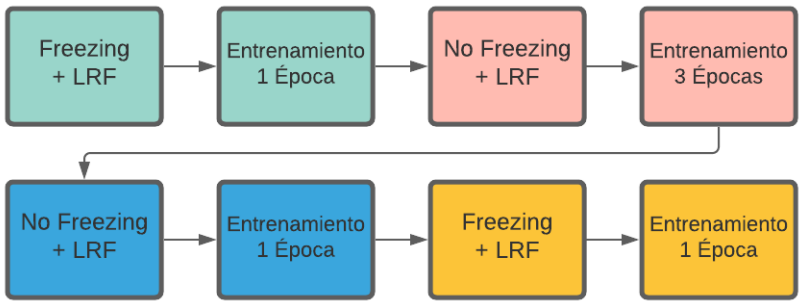

Figura 4.18: Proceso de Entrenamiento Sano, Neumonía, COVID-19.

Capítulo 5

Resultados

En este capítulo se presentarán los resultados obtenidos de las 3 redes neuronales convolucionales entrenadas, las primeras 2 que se encuentran en cascada se probaran individualmente y en conjunto, la tercera ResNet multiclase se probará para comparar los resultados obtenidos con los 2 modelos anteriores en cascada. Las imágenes para prueba son imágenes que no se utilizaron para el entrenamiento, serán imágenes nuevas para las redes neuronales.

En los resultados de cada red neuronal convolucional se muestran las métricas del conjunto de validación de cada ResNet, además se probaron con imágenes pertenecientes a las bases de datos presentadas en el capítulo 3 que no se utilizaron en el conjunto de entrenamiento ni en el de validación.

El conjunto de prueba utilizó 17,179 imágenes de las cuales 7,050 corresponden a la clase neumonía, 2,736 pertenecen a la clase COVID-19 y 7,393 a la clase sano. La clase COVID-19 tiene 226 imágenes de la base de datos del Hospital Universitario San Cecilio, 294 imágenes de Cohen y 2,216 imágenes pertenecen a la base de datos de la universidad de Qatar y universidad de Dhaka. La clase neumonía contiene 3,373 imágenes del conjunto de datos de la Universidad California San Diego y 3,677 imágenes de RSNA. La clase sano tiene 683 imágenes de la base de datos de la Universidad California San Diego, 4,210 imágenes de RSNA y 2,500 imágenes aleatorias de NIH.

Entrenamiento y Validación					
COVID-19		**Neumonía**		**Sano**	
Fuente	***No. Imágenes***	***Fuente***	***No. Imágenes***	***Fuente***	***No. Imágenes***
Hospital U San Cecilio	**200**	**NIH**	**150**	**Hospial Angeles Lindavista**	**221**
Cohen	**250**	**U C San Diego**	**900**	**U C San Diego**	**900**
Universidad Qatar/Dhaka	**1400**	**RSNA**	**1100**	**RSNA**	**1500**
IMSS	**432**			**NIH**	**1500**
Waterloo	**58**				
Total	**2340**	**Total**	**2150**	**Total**	**4121**

Tabla 5.1: Imágenes utilizadas para entrenamiento y validación del modelo en cascada

Entrenamiento y Validación					
COVID-19		Neumonía		Sano	
Fuente	*No. Imágenes*	*Fuente*	*No. Imágenes*	*Fuente*	*No. Imágenes*
Hospital U San Cecilio	200	NIH	150	Hospial Angeles Lindavista	221
Cohen	250	U C San Diego	900	U C San Diego	400
Universidad Qatar/Dhaka	1400	RSNA	1100	RSNA	800
IMSS	432			NIH	600
Waterloo	58				
Total	2340	Total	2150	Total	2021

Tabla 5.2: Imágenes utilizadas para entrenamiento y validación del modelo multiclase

5.1. Resultados ResNet 152 en Cascada

En esta sección se presentarán los resultados de las redes neuronales convolucionales con salidas binarias, primero estará la ResNet que detecta sano y enfermo, y después la ResNet que detecta neumonía y COVID-19. Se mostrarán los resultados individualmente y después en conjunto para verificar cómo funcionan finalmente ambos modelos. Por último estarán los resultados de la red neuronal convolucional multiclase capaz de seleccionar las 3 clases.

5.1.1. Resultados ResNet 152 Sano y Enfermo

La ResNet encargada de detectar sano y enfermó utilizó 1,667 imágenes para la validación, utilizó las métricas de exactitud, precisión, exhaustividad y valor F1 para medir la eficiencia de la red neuronal convolucional.

La primera época se aplicó *freezing* para ajustar los valores de los 2 perceptores de la última capa y se midió la pérdida con la función de entropía cruzada con suavizado de etiquetas para 2 salidas.

Época	Pérdida	Exactitud	Tasa Error	Exhaustividad	Precisión	Valor F1
1	**0.337583**	**0.925094**	**0.074906**	**0.919900**	**0.929204**	**0.924528**

Tabla 5.3: Primer Resultado ResNet Sano y Enfermo.

En la primera época da buenos resultados ya que la ResNet que se utilizó ya estaba previamente entrenada con el conjunto de ImageNet. La siguiente etapa de entrenamiento se quitó la técnica de *freezing* para entrenar las capas intermedias y se entrenó por 5 épocas.

Época	Pérdida	Exactitud	Tasa Error	Exhaustividad	Precisión	Valor F1
2	**0.326566**	**0.935705**	**0.064295**	**0.917397**	**0.951948**	**0.934353**
3	**0.313149**	**0.942572**	**0.057428**	**0.932416**	**0.951469**	**0.941846**
4	**0.310716**	**0.945069**	**0.054931**	**0.937422**	**0.951715**	**0.944515**
5	**0.312566**	**0.945069**	**0.054931**	**0.933667**	**0.955186**	**0.944304**
6	**0.313432**	**0.945693**	**0.054307**	**0.941176**	**0.949495**	**0.945317**

Tabla 5.4: Segundo Resultado ResNet Sano y Enfermo.

Modificar las capas intermedias mejoró a la red neuronal convolucional debido a que las 5 últimas épocas obtuvieron mejores resultados en todas las métricas. Por último se entrenó el modelo por 1 época con la técnica de *freezing*.

Época	Pérdida	Exactitud	Tasa Error	Exhaustividad	Precisión	Valor F1
7	**0.316814**	**0.945693**	**0.054307**	**0.917397**	**0.972149**	**0.943979**

Tabla 5.5: Tercer Resultado ResNet Sano y Enfermo.

Con la última época se logró el mejor resultado posible debido a que el método de *learning rate finder* solo mostraba pérdidas mayores en caso de que se entrenará por más épocas.

Se utilizó la red neuronal convolucional para clasificar las 1,667 imágenes del conjunto de validación y se utilizó una matriz de confusión.

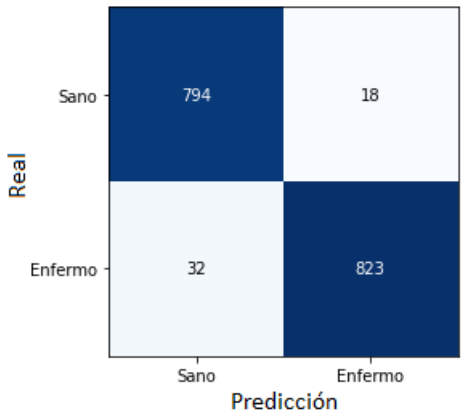

Figura 5.1: Matriz de Confusión Conjunto Validación Sano y Enfermo.

Se utilizó la ResNet 152 con todas las imágenes del conjunto de prueba, que contiene datos de las distintas bases de datos. Todos los resultados obtenidos se muestran en la

siguiente tabla que contiene información del número de imágenes, la clase, el número de clasificaciones correctas e incorrectas y la exactitud.

Se presentan los resultados de cada base de datos en la tabla 5.6, se distingue con color azul la clase COVID-19, con naranja la clase neumonía y con verde la clase sano. Las clases COVID-19 y neumonía pertenecen a la clase enfermo en esta red neuronal convolucional.

Dataset	Clase	No. Imágenes	Correctas	Incorrectas	Accuracy
San Cecilio	**COVID-19**	**226**	**213**	**13**	**94.247 %**
Cohen	**COVID-19**	**294**	**270**	**24**	**91.836 %**
U. Qatar y U. Dhaka	**COVID-19**	**2216**	**2193**	**23**	**98.962 %**
U. California San Diego	**Neumonía**	**3373**	**3258**	**115**	**96.590 %**
RSNA	**Neumonía**	**3677**	**3037**	**640**	**82.594 %**
U. California San Diego	**Sano**	**683**	**615**	**68**	**90.043 %**
RSNA	**Sano**	**4210**	**3978**	**232**	**94.489 %**
NIH	**Sano**	**2500**	**2145**	**355**	**85.800 %**

Tabla 5.6: Resultados Prueba ResNet 152 Sano y Enfermo.

Se sumó todos los ejemplos de la clase COVID-19 y se obtuvieron las clasificaciones correctas e incorrectas, también se obtuvo la exactitud del total de ejemplos, el cual fue un resultado bueno.

Total COVID-19	Correctas COVID-19	Incorrectas COVID-19	Exactitud COVID-19
2,736	**2,676**	**60**	**97.807 %**

Tabla 5.7: Resultados Prueba ResNet 152 Sano y Enfermo - COVID-19.

Se obtuvo el total de imágenes de la clase neumonía y se colocó el número de ejemplos correctos e incorrectos que clasificó la ResNet 152 y se obtuvo la exactitud.

Total Neumonía	Correctas Neumonía	Incorrectas Neumonía	Exactitud Neumonía
7,050	**6,295**	**755**	**89.290 %**

Tabla 5.8: Resultados Prueba ResNet 152 Sano y Enfermo - Neumonía.

Se obtuvo el total de las clases COVID-19 y neumonía debido a que ambas clases pertenecen a la clase enfermo, se tuvo el número total de imágenes, el número de predicciones correctas, incorrectas y la exactitud.

Total Enfermo	Correctas Enfermo	Incorrectas Enfermo	Exactitud Enfermo
9,786	8,971	815	91.671 %

Tabla 5.9: Resultados Prueba ResNet 152 Sano y Enfermo - Enfermo.

En la clase sano se sumaron los ejemplos correctos e incorrectos, el total de las imágenes y la exactitud que tuvieron.

Total Sano	Correctas Sano	Incorrectas Sano	Exactitud Sano
7,393	6,738	655	91.140 %

Tabla 5.10: Resultados Prueba ResNet 152 Sano y Enfermo - Sano.

Se sumó el total de la clase enfermo y la clase sano, así como sus predicciones correctas e incorrectas y también la exactitud de todos los ejemplos pertenecientes a ambas clases.

Total Sano y Enfermo	Correctas Sano y Enfermo	Incorrectas Sano y Enfermo	Exactitud Sano y Enfermo
17,179	15,709	1,470	91.144 %

Tabla 5.11: Resultados Prueba ResNet 152 Sano y Enfermo - Total.

Todas las imágenes del conjunto de prueba se usaron para probar la ResNet para clasificar entre las clases sano y enfermo, los resultados se colocaron en una matriz de confusión para evaluar la efectividad de cada clase.

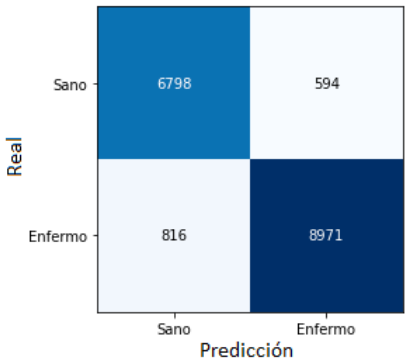

Figura 5.2: Matriz de Confusión Conjunto Prueba Sano y Enfermo.

Se usaron las métricas de precisión, exhaustividad y valor F1 de los resultados de la matriz de confusión para medir cada clase, también se obtuvo la exactitud 5.1 de ambas clases .

Clase	Precisión	Exhaustividad	Valor F1
Sano	**91.964 %**	**89.283 %**	**90.603 %**
Enfermo	**91.662 %**	**93.79 %**	**92.713 %**

Tabla 5.12: Matriz de Confusión Sano y Enfermo.

$$Exactitud = \frac{6,798 + 8,971}{17,179} = 91.792\,\% \tag{5.1}$$

5.1.2. Resultados ResNet 152 Neumonía y COVID-19

Se utilizaron las métricas de exactitud, precisión, exhaustividad y valor F1 para medir los resultados obtenidos de la ResNet que detecta neumonía y COVID-19, en el entrenamiento se midieron los resultados con el conjunto de validación, en este caso fueron 843 imágenes para validar.

En la primer época que se utilizó la técnica de *freezing* se midió la pérdida con la función de entropía cruzada con suavizado de etiquetas.

Época	Pérdida	Exactitud	Tasa Error	Exhaustividad	Precisión	Valor F1
1	**0.286240**	**0.972717**	**0.027284**	**0.970917**	**0.977477**	**0.974186**

Tabla 5.13: Primer Resultado ResNet Neumonía y COVID-19.

Al tener una ResNet 152 previamente entrenada la red detectaba bordes y figuras, por lo que entrenar solo las últimas 2 neuronas arrojó resultados buenos. En las siguientes 3 épocas se entrenó sin la técnica de *freezing*.

Época	Pérdida	Exactitud	Tasa Error	Exhaustividad	Precisión	Valor F1
2	**0.257553**	**0.978648**	**0.021352**	**0.979866**	**0.979866**	**0.979866**
3	**0.245778**	**0.989324**	**0.010676**	**0.991051**	**0.988839**	**0.989944**
4	**0.242670**	**0.986951**	**0.013049**	**0.986577**	**0.988789**	**0.987682**

Tabla 5.14: Segundo Resultado ResNet Neumonía y COVID-19.

Los resultados van teniendo mejoría, cada vez todas las métricas tienen mejores valores, lo que indica que la red neuronal convolucional se está entrenando adecuadamente. En la siguiente etapa se volvió a entrenar sin *freezing* y por 2 épocas.

Época	Pérdida	Exactitud	Tasa Error	Exhaustividad	Precisión	Valor F1
5	**0.241982**	**0.986951**	**0.013049**	**0.988814**	**0.986607**	**0.987709**
6	**0.241063**	**0.986951**	**0.013049**	**0.986577**	**0.988789**	**0.987682**

Tabla 5.15: Tercer Resultado ResNet Neumonía y COVID-19.

Los resultados son buenos y se aprecia que están convergiendo las métricas, por lo que se intentará afinar con pequeñas mejoras en los resultados. Se entrenó sin la técnica de *freezing* por una época.

Época	Pérdida	Exactitud	Tasa Error	Exhaustividad	Precisión	Valor F1
7	**0.244617**	**0.985765**	**0.014235**	**0.979866**	**0.993197**	**0.986486**

Tabla 5.16: Cuarto Resultado ResNet Neumonía y COVID-19.

En esta última época empezó a disminuir la eficiencia de la ResNet, sin embargo el LRF todavía indicaba que podía mejorar, por esta razón se entrenó una última época y al igual que la anterior, fue sin *freezing*.

Época	Pérdida	Exactitud	Tasa Error	Exhaustividad	Precisión	Valor F1
8	**0.242822**	**0.989324**	**0.010676**	**0.991051**	**0.988839**	**0.989944**

Tabla 5.17: Quinto Resultado ResNet Neumonía y COVID-19.

Este fue el mejor resultado de la red neuronal convolucional, se dejó de entrenar en este punto debido a que el método de LRF solo indicaba que la pérdida iba a ser mayor si se seguía entrenando. Se utilizaron las 786 imágenes del conjunto de validación y se colocaron en una matriz de confusión dependiendo como clasifico la ResNet 152.

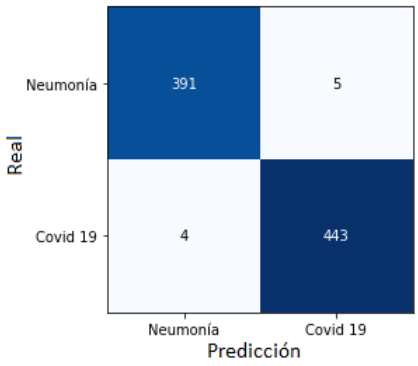

Figura 5.3: Matriz de Confusión Conjunto Validación Neumonía y COVID-19.

El conjunto de validación tuvo muy buenos resultados, es importante probar el modelo con más imágenes para comprobar que este siga funcionando correctamente con otros ejemplos.

Se utilizaron de conjunto de prueba todas las imágenes de las bases de datos que pertenecen a la clase COVID-19 y neumonía que no se utilizaron en el entrenamiento ni en la validación y probar la eficiencia de la red neuronal convolucional que distingue entre ambos daños.

Dataset	Clase	No. Imágenes	Correctas	Incorrectas	Exactitud
San Cecilio	**COVID-19**	**226**	**223**	**3**	**98.672 %**
Cohen	**COVID-19**	**294**	**277**	**17**	**94.217 %**
U. Qatar y U. Dhaka	**COVID-19**	**2216**	**2205**	**11**	**99.503 %**
U. California San Diego	**Neumonía**	**3373**	**3364**	**9**	**99.733 %**
RSNA	**Neumonía**	**3677**	**3563**	**114**	**96.899 %**

Tabla 5.18: Resultados Prueba ResNet 152 COVID-19 y Neumonía.

Se juntaron todos los ejemplos de la clase COVID-19 para un resultado general de esta clase de todas las bases de datos, se tuvo el número de predicciones correctas, incorrectas y la exactitud.

Total COVID-19	Correctas COVID-19	Incorrectas COVID-19	Exactitud COVID-19
2,736	**2,705**	**31**	**98.866 %**

Tabla 5.19: Resultados Prueba ResNet 152 COVID-19 y Neumonía - COVID-19.

Se realizó el mismo proceso con las imágenes de neumonía para tener la exactitud general de esta clase y se representa en la siguiente tabla, donde también se tiene el total de imágenes, predicciones correctas e incorrectas.

Total Neumonía	Correctas Neumonía	Incorrectas Neumonía	Exactitud Neumonía
7,050	**6,927**	**31**	**98.255 %**

Tabla 5.20: Resultados Prueba ResNet 152 COVID-19 y Neumonía - Neumonía.

Se sumó el total de ambas clases para tener un resultado general de la red neuronal convolucional, en la siguiente tabla se muestra el total de imágenes, las imágenes que se detectaron correctamente e incorrectamente, y la exactitud.

Total Covid y Neumonía	Correctas Covid y Neumonía	Incorrectas Covid y Neumonía	Exactitud Covid y Neumonía
9,786	9,632	154	98.426 %

Tabla 5.21: Resultados Prueba ResNet 152 COVID-19 y Neumonía - Total.

Se utilizaron solamente las imágenes pertenecientes a las clases de neumonía y COVID-19 como conjunto de prueba para analizar la ResNet y se colocaron los resultados en una matriz de confusión.

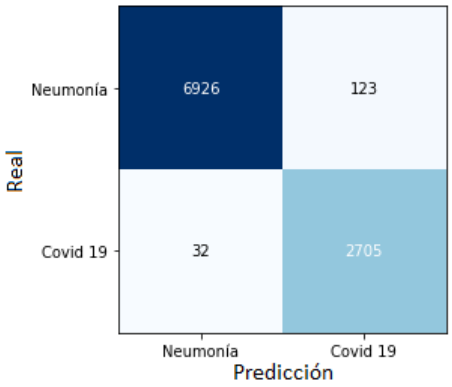

Figura 5.4: Matriz de Confusión Conjunto Prueba Neumonía y COVID-19.

Una vez obtenida la matriz de confusión, a partir de sus valores, se calcularon las métricas de precisión, exhaustividad y valor F1 para las 2 clases. Además se obtuvo la exactitud 5.2 de todas las predicciones.

Clase	Precisión	Exhaustividad	Valor F1
Neumonía	**98.255 %**	**99.54 %**	**98.893 %**
COVID-19	**98.831 %**	**95.651 %**	**97.215 %**

Tabla 5.22: Matriz de Confusión Prueba Neumonía y COVID-19.

$$Exactitud = \frac{6,926 + 2,705}{9,786} = 98.416\,\% \tag{5.2}$$

5.1.3. Resultados ResNet 152 en Cascada

Las 17,179 imágenes del conjunto de prueba se utilizaron en las redes neuronales convolucionales 4.1.1 (Sano-Enfermo) y 4.1.2 (Neumonía-COVID-19) en forma de cascada como se muestra en la figura 4.1 para medir su resultado en la clasificación de

3 clases. Los resultados se muestran en la matriz de confusión 5.5 para identificar los resultados de cada clase y verificar la precisión de cada una.

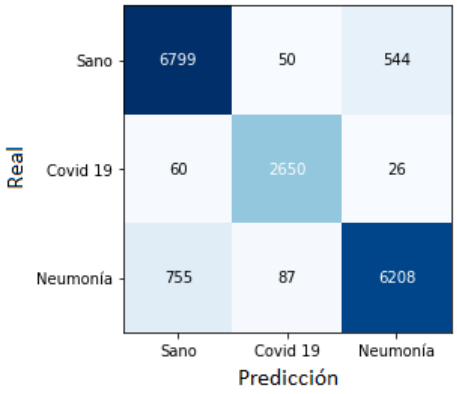

Figura 5.5: Matriz de Confusión Conjunto Prueba Cascada.

Se midieron las métricas de precisión, exhaustividad y valor F1 de cada clase para un mejor análisis, también se obtuvo la exactitud de todos las predicciones 5.3.

Clase	Precisión	Exhaustividad	Valor F1
Sano	**91.965 %**	**89.296 %**	**90.61 %**
COVID-19	**96.857 %**	**95.084 %**	**95.962 %**
Neumonía	**88.057 %**	**91.59 %**	**89.788 %**

Tabla 5.23: Métricas Conjunto Prueba Cascada.

$$Exactitud = \frac{6,799 + 2,650 + 6,208}{17,179} = 91.14\,\% \tag{5.3}$$

5.2. Resultados ResNet 152 Sano, Neumonía y COVID-19

La ResNet fue validada con 1,247 imágenes y se utilizó la métrica de exactitud y error para medir los resultados. Esta ResNet a diferencia de las anteriores es multiclase, al tener que detectar más clases es posible que tenga un error más alto. La primera época fue para ajustar los valores de la última época, por lo que se entrenó con la técnica de *freezing*.

Época	Pérdida	Exactitud	Tasa Error
1	**0.455976**	**0.911257**	**0.088743**

Tabla 5.24: Primer Resultado ResNet Sano, Neumonía y COVID-19.

La red neuronal convolucional con el entrenamiento de la última capa tiene un buen resultado debido a que se utilizó una ResNet que estaba entrenada previamente. El siguiente entrenamiento fue sin la técnica de *freezing* para ajustar los valores de las capas intermedias y se entrenó por 3 épocas.

Época	Pérdida	Exactitud	Tasa Error
2	**0.427947**	**0.933443**	**0.066557**
3	**0.417625**	**0.940016**	**0.059984**
4	**0.416760**	**0.939195**	**0.060805**

Tabla 5.25: Segundo Resultado ResNet Sano, Neumonía y COVID-19.

En cada época va disminuyendo la pérdida y mejora la exactitud, esto demuestra que el modelo está aprendiendo adecuadamente. En la siguiente etapa se entrenó sin *freezing* y por una época.

Época	Pérdida	Exactitud	Tasa Error
5	**0.461797**	**0.933440**	**0.066560**

Tabla 5.26: Tercer Resultado ResNet Sano, Neumonía y COVID-19.

En el entrenamiento empezó a tener menor rendimiento debido a que la pérdida aumentó y la exactitud disminuyó, por esta razón solo se va a entrenar con la técnica de *freezing* para no modificar las capas intermedias debido a que puede tener peor resultado.

Época	Pérdida	Exactitud	Tasa Error
6	**0.415466**	**0.943303**	**0.056697**

Tabla 5.27: Cuarto Resultado ResNet Sano, Neumonía y COVID-19.

La última época de entrenamiento fue la que mejor resultado tuvo, debido a que la pérdida fue la menor que se tuvo e igualmente la exactitud fue la mejor. Se paró de

entrenar porque el método de LRF solo indicaba que iba a aumentar la pérdida con cualquier valor de la tasa de aprendizaje.

Se clasificaron las 1,247 imágenes y se utilizó una matriz de confusión para un mejor entendimiento y poder analizar las clases que tienen más error.

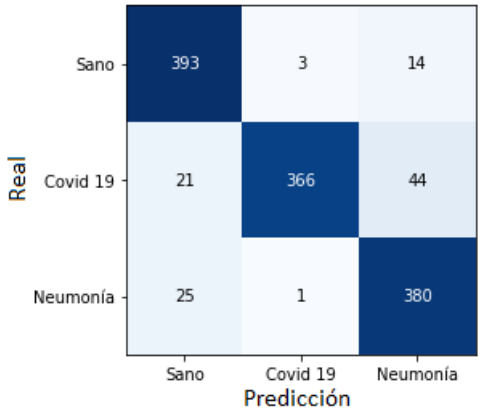

Figura 5.6: Matriz de Confusión Conjunto Validación Sano, COVID-19 y Neumonía.

Se utilizó el conjunto de prueba que contiene 17,179 imágenes para probar el modelo para las 3 clases y se colocaron los resultados en la matriz de confusión 5.7 para visualizar los resultados por cada clase.

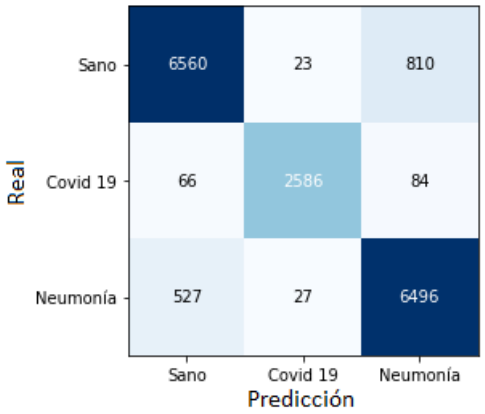

Figura 5.7: Matriz de Confusión Conjunto Prueba Sano, Neumonía y COVID-19.

Se utilizaron las métricas de precisión, exhaustividad y valor F1 para medir la efectividad de cada clase y también se obtuvo la exactitud 5.4 de todo el conjunto de prueba con esta ResNet.

Clase	Precisión	Exhaustividad	Valor F1
Sano	**88.733 %**	**91.71 %**	**90.196 %**
COVID-19	**94.518 %**	**98.103 %**	**96.277 %**
Neumonía	**92.142 %**	**87.903 %**	**89.972 %**

Tabla 5.28: Métricas Conjunto Prueba 3 Clases.

$$Exactitud = \frac{6,560 + 2,586 + 7,390}{17,179} = 91.053\,\% \tag{5.4}$$

Capítulo 6

Conclusiones

Con el desarrollo de este proyecto se logró crear un método basado en IA, el cual utiliza imágenes rayos X de tórax para identificar si una persona tiene COVID-19, neumonía o simplemente se encuentra sana.

Se utilizaron dos modelos diferentes, un modelo en cascada que utiliza 2 redes neuronales convolucionales ResNet 152 con salidas binarias y el otro modelo solo era una ResNet 152 multiclase con 3 las salidas, ambos tuvieron prácticamente el mismo porcentaje de exactitud, el modelo de cascada solo estuvo más alto por 0.087 %, aunque en las otras métricas si hubieron más diferencias. En los resultados de la red neuronal convolucional multiclase se tuvo mejor precisión con la clase de neumonía por 4.085 % y en exhaustividad se obtuvo mejor resultado en las clases sano y COVID-19 por 2.414 % y 3.019 % respectivamente. En cuanto al modelo en cascada se tuvo mejor precisión en las clases sano y COVID-19 por 3.232 % y 2.339 % respectivamente y se tuvo mejor exhaustividad en la clase neumonía con 3.687 % más a diferencia del modelo de 3 clases. En la métrica valor F1 se tuvieron valores similares en ambos modelos, tuvieron diferencias no mayor a 0.5 % y neumonía tuvo el resultado más bajo junto con sano, la clase COVID-19 tuvo un resultado mejor de aproximadamente un 5 % más que las otras clases en ambos modelos.

Aunque inicialmente se esperaba que el modelo encargado de detectar sano y enfermo iba a ser mejor debido a que las imágenes enfermas presentan manchas en los pulmones y las imágenes de pulmones sanos se encuentran limpias, pero el resultado fue diferente. Mientras que el algoritmo para detectar neumonía y COVID-19 tuvo una exactitud de 98.416 %, el algoritmo para detectar sano y enfermo tuvo 91.792 %.

Las imágenes médicas rayos X de tórax son viables para hacer la detección de enfermedades como neumonía y COVID-19 utilizando redes neuronales convolucionales, lo cual es una buena noticia debido a que este tipo de imágenes son más accesibles a la población en general en comparación con otro tipo de imágenes como las tomografías computarizadas (CT), que probablemente si se entrena un modelo de inteligencia artificial puedan entregar mejores resultados pero este tipo de tecnología es más costoso y no se encuentra en muchas partes de países en vías de desarrollo.

Como trabajo futuro se buscará realizar el mismo proceso de entrenamiento con la

diferencia de utilizar las imágenes en formato dicom que utiliza valores más exactos y una resolución más alta, por esta razón las imágenes son más pesadas y se requiere un procesamiento computacional con más potencia. Además de la clasificación se buscará hacer segmentación de las imágenes en donde indique específicamente en donde se encuentran los daños, para realizar este proceso se requiere de médicos especialistas que seleccionen las regiones de donde se encuentran los daños. Esto implica otro tipo de algoritmo para segmentación en el que la entrada es una imagen y la salida será una matriz del mismo tamaño de la imagen de entrada.

Bibliografía

[1] Amine ben khalifa and Hichem Frigui. Multiple instance fuzzy inference neural networks. 10 2016. VII, 13

[2] A comprehensive guide to convolutional neural networks. https://towardsdatascience.com/. Accessed: 2021-11-17. VII, 15

[3] Data augmentation in deep learning. https://medium.com/. Accessed: 2021-11-17. VII, 17

[4] World Health Organization. Who coronavirus (covid-19) dashboard. https://covid19.who.int/. Accessed: 2021-09-6. 1

[5] Fachi M. M. Vilhena R. O. Cobre A. F. Tonin F. S. Pontarolo R. Böger, B. Systematic review with meta-analysis of the accuracy of diagnostic tests for covid-19, 2021. URL https://doi.org/10.1016/j.ajic.2020.07.011. 1

[6] Deeks JJ Berhane S Taylor M Adriano A Davenport C Dittrich S Emperador D Takwoingi Y Cunningham J Beese S Domen J Dretzke J Ferrante di Ruffano L Harris IM Price MJ Taylor-Phillips S Hooft L Leeflang MMG McInnes MDF Spijker R Dinnes, J and A Van den Bruel. Rapid, point-of-care antigen and molecular-based tests for diagnosis of sars-cov-2 infection, 2021. ISSN 1465-1858. URL https://doi.org//10.1002/14651858.CD013705.pub2. 1

[7] Stephen I. Gallant. Perceptron-based learning algorithms. *IEEE Transactions on neural networks*, 1:179–191, 1990. 5

[8] François Chollet. *Deep Learning with Python*. Manning, 2018. 6, 13

[9] Claudio Moraga Jun Han. The influence of the sigmoid function parameters on the speed of backpropagation learning. *International Workshop on Artificial Neural Networks*, 1995. 6

[10] Abien Fred M. Agarap. Deep learning using rectified linear units (relu). 2018. 7

[11] Daniel Jurafsky James H. Martin. *Speech and Language Processing*. Prentice Hall, 2020. 8

[12] Aakash Srivastava. Label smoothing: Making model robust to incorrect labels, Jun 2019. URL https://towardsdatascience.com/label-smoothing-making-model-robust-to-incorrect-labels-2fae037ffbd0. 8

[13] Aaron Courville Ian Goodfellow, Yoshua Bengio. *Deep Learning*. 2017. 9, 11, 14

[14] Diederik P. Kingma Jimmy Lei Ba. Adam: A method for stochastic optimization. 2015. 9

[15] Kunin et al. Katanforoosh. Parameter optimization in neural networks, 2019. URL https://www.deeplearning.ai/ai-notes/optimization/. 9

[16] Yaima Filiberto Rafael Bello Lenniet Coello, Yumilka Fernandez. Improving the multilayer perceptron learning by using a method to calculate the initial weights with the similarity quality measure based on fuzzy sets and particle swarms. *Computación y Sistemas*, 2015. 10

[17] Hossein Gholamalinezhad and Hossein Khosravi. Pooling methods in deep neural networks, a review. 2020. 11

[18] Jeremy Howard Sylvain Gugger. *Deep Learning for Coders with Fastai and PyTorch: AI Applications Without a PhD*. O'Reilly Media, 2020. 12, 15, 18

[19] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors, 2012. 13

[20] M.V. Valueva, N.N. Nagornov, P.A. Lyakhov, G.V. Valuev, and N.I. Chervyakov. Application of the residue number system to reduce hardware costs of the convolutional neural network implementation. *Mathematics and Computers in Simulation*, 177:232–243, 2020. ISSN 0378-4754. doi: https://doi.org/10.1016/j.matcom.2020.04.031. URL https://www.sciencedirect.com/science/article/pii/S0378475420301580. 14

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. URL http://arxiv.org/abs/1512.03385. 15, 16

[22] Connor Shorten and Taghi M. Khoshgoftaar. A survey on image data augmentation for deep learning. *J Big Data*, pages 6 – 60, 2019. doi: https://doi.org/10.1186/s40537-019-0197-0. URL https://journalofbigdata.springeropen.com/articles/10.1186/s40537-019-0197-0#citeas. 17

[23] Leslie N. Smith. No more pesky learning rate guessing games. *CoRR*, abs/1506.01186, 2015. URL http://arxiv.org/abs/1506.01186. 17

[24] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010. doi: 10.1109/TKDE.2009.191. 17

[25] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. 2012. URL http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf. 18

[26] Peng Peng and Jiugen Wang. How to fine-tune deep neural networks in few-shot learning? *CoRR*, abs/2012.00204, 2020. URL https://arxiv.org/abs/2012.00204. 18

[27] Ajay Kulkarni, Deri Chong, and Feras A. Batarseh. 5 - foundations of data imbalance and solutions for a data democracy. In Feras A. Batarseh and Ruixin Yang, editors, *Data Democracy*, pages 83–106. Academic Press, 2020. ISBN 978-0-12-818366-3. doi: https://doi.org/10.1016/B978-0-12-818366-3.00005-8. URL https://www.sciencedirect.com/science/article/pii/B9780128183663000058. 18

[28] Joseph Paul Cohen, Paul Morrison, and Lan Dao. Covid-19 image data collection. *arXiv 2003.11597*, 2020. URL https://github.com/ieee8023/covid-chestxray-dataset. 23

[29] Muhammad E. H. Chowdhury, Tawsifur Rahman, Amith Khandakar, Rashid Mazhar, Muhammad Abdul Kadir, Zaid Bin Mahbub, Khandakar Reajul Islam, Muhammad Salman Khan, Atif Iqbal, Nasser Al Emadi, Mamun Bin Ibne Reaz, and Mohammad Tariqul Islam. Can ai help in screening viral and covid-19 pneumonia? *IEEE Access*, 8:132665–132676, 2020. doi: 10.1109/ACCESS.2020.3010287. 24

[30] Tawsifur Rahman, Amith Khandakar, Yazan Qiblawey, Anas Tahir, Serkan Kiranyaz, Saad Bin Abul Kashem, Mohammad Tariqul Islam, Somaya Al Maadeed, Susu M. Zughaier, Muhammad Salman Khan, and Muhammad E.H. Chowdhury. Exploring the effect of image enhancement techniques on covid-19 detection using chest x-ray images. *Computers in Biology and Medicine*, 132:104319, 2021. ISSN 0010-4825. doi: https://doi.org/10.1016/j.compbiomed.2021.104319. URL https://www.sciencedirect.com/science/article/pii/S001048252100113X. 24

[31] S. Tabik, A. Gómez-Ríos, J. L. Martín-Rodríguez, I. Sevillano-García, M. Rey-Area, D. Charte, E. Guirado, J. L. Suárez, J. Luengo, M. A. Valero-González, P. García-Villanova, E. Olmedo-Sánchez, and F. Herrera. Covidgr dataset and covid-sdnet methodology for predicting covid-19 based on chest x-ray images, 2020. 25

[32] Kang K. Rui P. Rsna pneumonia detection challenge, Oct 2018. URL https://www.kaggle.com/c/rsna-pneumonia-detection-challenge/overview. 26

[33] Michael Goldbaum Daniel Kermany, Kang Zhang. Large dataset of labeled optical coherence tomography (oct) and chest x-ray images, 2018. URL https://data.mendeley.com/datasets/rscbjbr9sj/3. 26

[34] Xiaosong Wang, Yifan Peng, Le Lu, Zhiyong Lu, Mohammadhadi Bagheri, and Ronald Summers. Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases, 2017. 27