Clasificación de diagnósticos Covid-19 basado en radiografías de tórax

T. Delgado, F. Quisel

ARTICLE INFO

Keywords:
Radiografía
Covid-19
Machine Learning

ABSTRACT

El artículo explica en detalle la propuesta de solución a uno de los mayores problemas existentes en la actualidad, como lo es el **Covid-19 enfermedad del nuevo coronavirus conocido como SARS-CoV-2**, esta solución está basada en un DataSet extraído de la plataforma Kaggle el cual formará parte de un input al un modelo basado en las redes neuronales de convolución, especiales para el procesamiento de imágenes, conocido como Computer Vision. Por lo anterior es que el enfoque y problemática que aborda el proyecto es sobre un diagnostico de Covid-19, donde los pacientes presentan dificultades de respiración leves o graves, las que afectan al sistema humano respiratorio. Para la propuesta en cuestión, los diagnósticos se realizan a través de las radiografías convertidas en un formato de imagen, que contiene información visual de lo que captan los rayos X en los espacios clínicos, poniendo sobre la mesa una forma de solucionar y optimizar los procesos médicos actuales relacionados a la pandemia.

1. Introducción

El mundo está experimentando una pandemia única en la vida, que causa incontables sufrimientos y muertes humanas, el desmoronamiento de las relaciones sociales y priva a las personas de sus rutinas de vida y a los países de prosperidad económica, un ejemplo; la situación de los Chilenos. La pandemia de coronavirus ha afectado los sistemas de salud, de gran manera y ha revelado desigualdades inconcebibles y ha trastornado a las instituciones internacionales. El impacto de la pandemia de coronavirus está haciendo que todas las empresas, o pymes revisen los objetivos de negocios durante el último siglo. Junto con las tecnologías emergentes y la incertidumbre política y económica, el Covid-19 cambiará sistemas económicos y geopolíticos enteros [1].

Por otro lado, la naturaleza del comercio está experimentando un cambio profundo en un mundo cibernético cada vez más digital e interconectado. La resiliencia y la conectividad serán las nuevas consignas a medida que las organizaciones busquen adaptarse a este futuro impredecible.

Para entender el motivo de esta pandemia es importante entender de qué se trata esta enfermedad, y es que el Covid-19 es la enfermedad causada por el nuevo coronavirus conocido como SARS-CoV-2. La OMS (Organización Mundial de la Salud) tuvo noticia por primera vez de la existencia de este nuevo virus el 31 de diciembre de 2019, al ser informada de un grupo de casos de ɐneumonía víricaɐ que se habían declarado en Wuhan (República Popular China).

Los síntomas más habituales de la enfermedad del Covid-19 son las siguientes:

- Fiebre y tos
- Cansancio
- Dolor de garganta
- Dolor de cabeza
- Dolores musculares o articulares
- Pérdida del gusto o el olfato

Entre los síntomas de un cuadro grave de la COVID-19 se incluyen:

- Disnea (dificultad respiratoria)
- Pérdida de apetito
- Confusión
- Dolor u opresión persistente en el pecho
- Temperatura alta (por encima de los 38° C)

Las personas de cualquier edad que tengan fiebre o tos y además respiren con dificultad, sientan dolor u opresión en el pecho o tengan dificultades para hablar o moverse deben solicitar atención médica inmediatamente. De ser posible, llame con antelación a su dispensador de atención de salud, al teléfono de asistencia o al centro de salud para que puedan indicarle el dispensario adecuado [2].

Entre las personas que desarrollan síntomas, la mayoría (alrededor del 80%) se recuperan de la enfermedad sin necesidad de recibir tratamiento hospitalario. Alrededor del 15% desarrollan una enfermedad grave y requieren oxígeno y el 5% llegan a un estado crítico y precisan cuidados intensivos.

Entre las complicaciones que pueden llevar a la muerte se encuentran la insuficiencia respiratoria, el síndrome de dificultad respiratoria aguda, la septicemia y el choque septicémico, la tromboembolia y/o la insuficiencia multiorgánica, incluidas las lesiones cardíacas, hepáticas y renales.

La radiografía de tórax utiliza una dosis muy pequeña de radiación ionizante para producir imágenes del interior del tórax. Se utiliza generalmente para evaluar los pulmones y sus afecciones pulmonares, como neumonía, enfisema, cáncer y hasta ahora y muy probablemente el Covid-19, debido a que la radiografía de tórax es amplia en su disponibilidad, de bajo coste, rápida y sencilla, es particularmente útil en el diagnóstico y tratamiento de emergencia.

La tomografía computarizada torácica tiene una mayor sensibilidad que la radiografía de tórax y permite valorar tanto la afectación pulmonar como posibles complicaciones, además de proporcionar diagnósticos alternativos [3].

2. Desarrollo

2.1. Problema a solucionar

El problema a resolver es la detección del covid-19, mediante radiografía de tórax donde se plantea clasificar si el diagnóstico del paciente es positivo o no.

Como se menciona anteriormente, los problemas que ha causado la enfermedad Covid-19 ha causado problemas graves. Por ello, la detección inmediata de esta enfermedad será un plus importante en la gestión del tiempo y así otorgar un tratamiento inmediato.

2.2. Posible Solución

El problema planteado es importante solucionar ya que el covid-19 (SARS-CoV-2) ha producido inmensas consecuencias a nivel mundial sea en las personas como también en la economía de los países, como se puede observar en la figura 1 (datos reales, muertes por Covid-19) en un periodo de estudio del 11 de junio 2021.

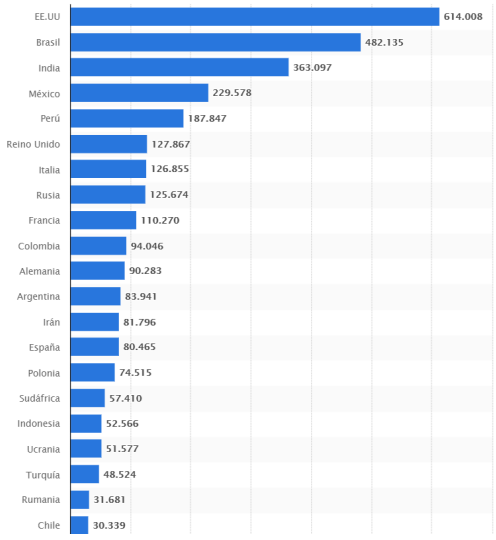

Figure 1: Número de personas fallecidas a causa del coronavirus en el mundo

El dataset será entrada a un modelo de Machine Learning basado en las redes neuronales de convolución, especiales para el procesamiento de imágenes, conocido como Computer Vision, En donde se clasifican estas imágenes para saber si el paciente es covid-19 positivo o negativo.

2.3. Data set

El dataset a utilizar tiene su origen en Kaggle, el cual contiene una recopilación de imágenes de radiografía de tórax de pacientes COVID-19 positivos y negativos correspondientes a una **competencia** para obtener el modelo en resultados. Este conjunto de datos comprende aproximadamente 16.351 imágenes de radio x.[4]

Type	COVID-19 Negative	COVID-19 Positive	Total
train	13793	2158	15951
test	200	200	400

Figure 2: Distribución del dataset

En la figura 2 se muestra la distribución del dataset a utilizar, compuesto por dos carpetas train y test, que contienen imágenes mixtas entre las 2 clases, y etiquetadas mediante los archivos train_COVIDx3.txt y test_COVIDx3.txt, los que contienen además información de las imágenes con el formato mostrado en la figura 3 , estas imágenes de rayos X serán las que se utilizaran para entrenar y probar el modelo.

```
[patient id] [filename] [class] [data source]
```

Figure 3: Head de los archivo txt

Por último se encuentra una carpeta con archivos de imágenes sin etiquetar destinadas al proceso de clasificación esperada por la competencia.

Este conjunto de datos de COVIDx se construye a partir de distintos conjuntos de datos de radiografía de tórax de código abierto como por ejemplo covid-chestxray-dataset también RSNA International COVID-19 Open Radiology Database (RICORD), en donde dataset se creó mediante una colaboración entre la RSNA y la Sociedad de Radiología Torácica (STR).

Luego de esto, se realiza un pre-procesamiento de las imágenes mixtas en las carpetas train y test, para obtener una forma de entrada al modelo ordenada y etiquetada, de manera similar a la vista en el proyecto estudiado anteriormente como se muestra en la figura 4.

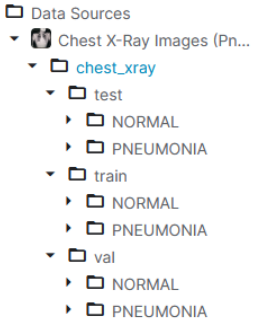

Figure 4: Ejemplo directorio de entrada

2.4. EDA (Análisis exploratorio de datos)

Ya teniendo el dataset a trabajar como primer experimento elegido, se comenzó a analizar este, visualizando sus datos de los archivos train, test y val.

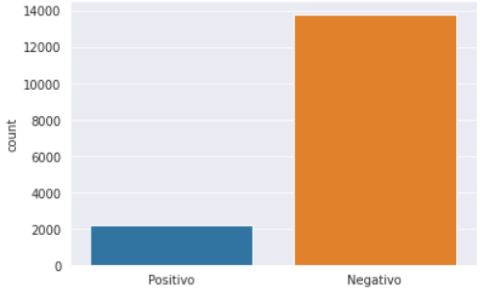

Figure 5: Cantidad de imágenes para entrenamiento

Para el test y validación se utilizan estos conjuntos ya balanceados, ya que test cuenta con 185 imágenes de cada clases y val cuenta con 15 para Positivo y Negativo. En cambio el archivo train esta de desequilibrada, ya que train tiene 2158 imágenes de la clase Positiva y solo 13793 de la clase Negativa.

El segundo experimento consisitió únicamente a diferencia del primero en usar un aumento artificial de las imágenes que se utilizaron como entrada anteriormente.

Como tercer experimento se agregaron 500 imágenes correspondientes a un dataset de similares características al trabajado, resultando entonces en ejemplos extras de clase Positiva al conjunto de entrenamiento, Figura 6 .

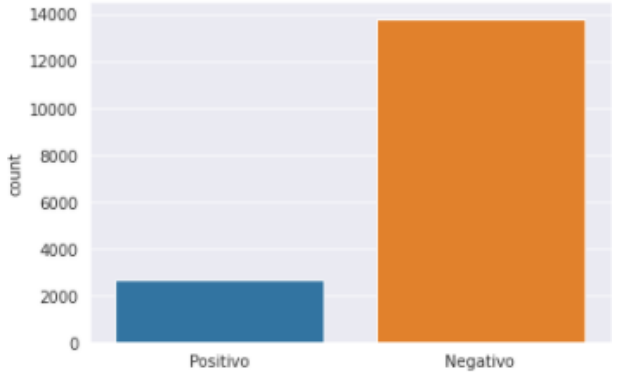

Figure 6: Entrada dataset con registros extras

Por último se realizó un cuarto experimento en donde la entrada de datos corresponde a la misma aplicada en el tercer experimento, pero esta vez haciendo una duplicación de todas las imágenes existentes en el conjunto de entrenamiento únicamente de clase Positiva, para lograr en cierto grado, un balance, resultando como entrada lo siguiente, Figura 7.

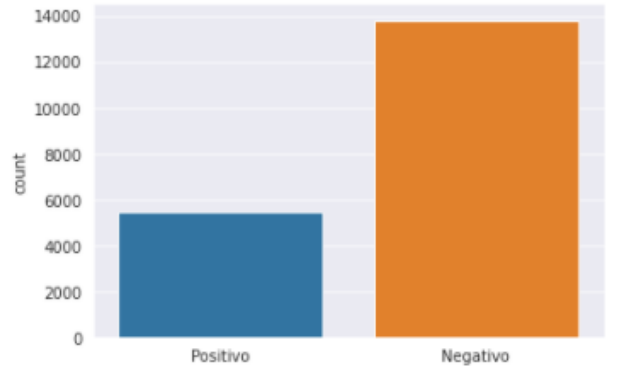

Figure 7: Entrada con registros extras duplicados

A la vez también se hizo una vista previa de las imágenes de ambas clases, como se muestra en la imagen siguiente en donde se visualiza una radiografía de clase positiva con una de clase negativa. En la imagen figure 8 se muestra la radiografía del tórax de una persona con covid-19 o sea que es de la clase Positiva, por otra parte en la imagen figure 9 se muestra una radiografía con resultados negativo.

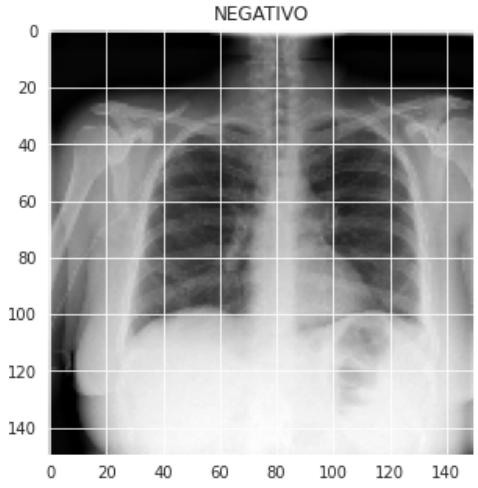

Figure 8: Radiografía Negativo en Covid-19

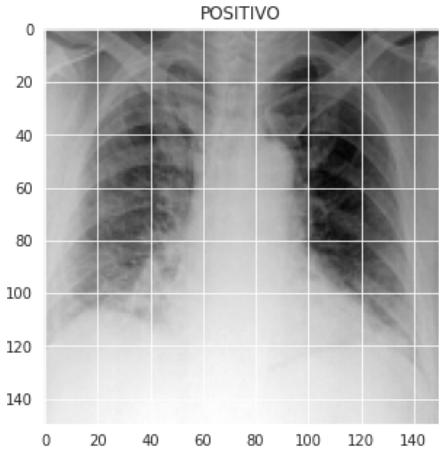

Figure 9: Radiografía Positivo en Covid-19

2.5. Biblioteca Keras

Primeramente se debe comprender que una librería es parecida a las convencionales librerías donde se va a buscar cualquier libro, no obstante, en vez de libros son módulos, los cuales se aplicarán en los procesos de programación.
Keras es una librería de código abierto con licencia MIT escrita en Python, se basa principalmente en el trabajo de Francois Chollet, el cual es un desarrollador de google. El objetivo de keras es la aceleración de la creación de redes neuronales. Por esto no funciona como un framework independiente, sino como una interfaz de uso intuitivo (API) que permite acceder a varios frameworks de aprendizaje automático y desarrollarlos.
En resumen Keras es una librería basada en Python para desarrollar modelos de aprendizaje profundo como también compatible con otras librerías Python (así como TensorFlow o Theano).

2.6. Modelo mediante CNN

Durante los años, se han desarrollado algunas arquitecturas de CNN, lo cual dio sitio a adelantos impresionantes en el campo del aprendizaje profundo[**?**]. Una buena medida de este avance son las tasas de error en competencias como

el desafío ILSVRC ImageNet.
En este problema el tipo de aprendizaje aplicado fue el llamado supervisado, ya que cada grupo de imágenes está identificado con la entrada correspondiente a su clase, es decir si es positivo o negativo en covid-19.

Por otra parte, uno de los problemas que mayormente existe en la mayoría de los modelos es el sobreajuste originado en la etapa de entrenamiento, esto es cuando el modelo se ajusta a los datos de entrenamiento concluyendo en una mala generalización en los datos de test, es decir que el algoritmo no puede funcionar con precisión en la entrada de nuevos datos, por lo que para evitar este problema de sobreajuste como también balancear la cantidad de ejemplos, es necesario expandir artificialmente el conjunto de datos.La idea principal del aumento consiste en alterar los datos de entrenamiento con pequeñas transformaciones para reproducir las variaciones, de manera que cambien su representación en la matriz, pero sin cambiar su etiqueta el cual lo identifica (Figura 10). Algunos de los aumentos más populares aplicados son escalas de grises, giros horizontales, giros verticales, recortes aleatorios, fluctuaciones de color y traslaciones. Para el aumento, se aplicó solo un par de las transformaciones mencionadas en los datos de entrenamiento, para duplicar o triplicar la cantidad de ejemplos y crear un modelo robusto

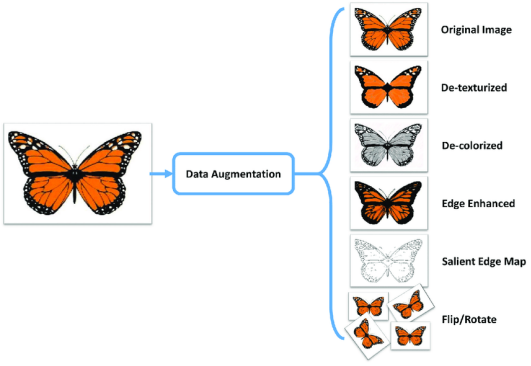

Figure 10: Data Augmentation

Para los modelos en los que se aplicó el aumento de datos, se declaró lo siguiente:

- Rotación aleatoriamente algunas imágenes de entrenamiento 30 grados
- Zoom aleatorio en un 20% algunas imágenes de entrenamiento
- Cambiar las imágenes de forma aleatoria horizontalmente en un 10% del ancho
- Cambiar las imágenes verticalmente al azar en un 10% de la altura
- Giro en las imágenes horizontalmente al azar.

La construcción del modelo y fase de entrenamiento se basa en la arquitectura de CNN (Convolutional Neural Network) ya que es un tipo especializado de red neuronal diseñada para trabajar con datos de imágenes bidimensionales como lo es en este caso, y así mismo se pueden usar con datos unidimensionales y tridimensionales [**?**].

- Convolutional layer
- Pooling layer
- Fully-connected layer

La Convolutional Layer es el bloque de construcción central de una CNN, y es donde ocurre la mayor parte de los cálculos. Requiere algunos componentes, que son datos de entrada, un filtro y un mapa de características, además existe un detector de características conocido como Kernel o filtro el cual se moverá a través de los campos receptivos de la imagen, verificando si la característica está presente. Este proceso se conoce como convolución.

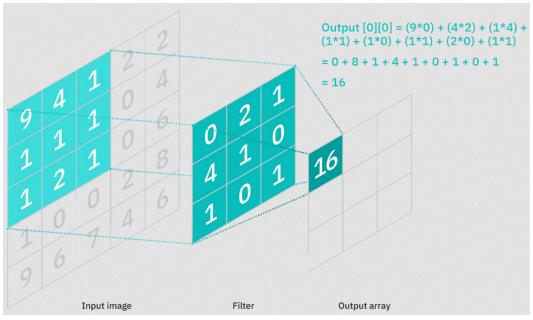

Figure 11: Convolución

Para este proyecto se observa la implementación de este por medio de 5 convoluciones bidimensionales [**?**], donde se define lo siguiente:

- *Filter*, define el número de filtros de salida en la convolución (la dimensionalidad) aplica 32 para la primera capa de convolución, mientras que en las siguientes implementa 64, 65, 128 y 256.
- $Kernel_size$ es el filtro convolucional que realiza una validación cruzada y especifica la altura y el ancho de la ventana de convolución 2D que se aplica en una matriz (2,2) para las 5 capas convolucionales.
- *Stride* es la distancia, o número de píxeles, en la que el núcleo se moverá sobre la matriz de entrada, y que para las 5 capas se aplicará en 1, es decir pixel por pixel.
- *Padding*, usado para cuando los filtros no se ajustan a la imagen de entrada, en este caso se aplica de tipo $\zeta same\eta$, es decir, que para el stride 1 la salida será la misma que la entrada y se aplica padding a la imagen de entrada para que la imagen de entrada quede completamente cubierta por el filtro y el stride especificado.
- *Activation*, alimenta a la neurona actual y su salida que va a la siguiente capa, donde deciden si la neurona debe activarse o no, para las 5 capas de convolución se

aplica una transformación de unidad lineal rectificada (ReLU) al mapa de características, lo que introduce la no linealidad (Figura12).

Figure 12: Funciones de Activación

Después de la primera capa de convolución se aplica el *BatchNormalization* el cual transforma las entradas para que estén estandarizadas, lo que significa que tendrán una media de cero y una desviación estándar de uno. Para las siguientes capas el BatchNormalization se aplica luego de cada Dropout - este actúa para prevenir el sobreajuste en el entrenamiento ocasionando que algunas neuronas no propaguen su aprendizaje en esta etapa-, exceptuando el último.

Luego de cada ejecución del BatchNormalization el modelo convolucional ejecuta MaxPool2D que va resumiendo todos los valores de la entrada por píxel en una nueva matriz con los máximos de los filtros establecidos en una matriz de salida (figura 13), de manera que para el proyecto el tamaño de estos filtros (size) se define en una matriz de (2,2) y con un stride de 2.

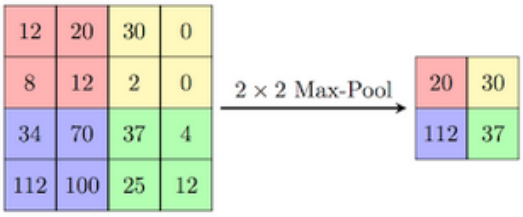

Figure 13: Max Pool

Después se realiza Flattening para transformar la matriz en un vector la cual sera la entrada de la red neuronal.
Por último se compone la capa Fully Connected y para construir la red neuronal se utiliza Dense, quien define la capa oculta de la red con 128 neuronas, para después definir la capa de salida, la cuál es la predicción que se forma en conjunto a la función de activación, la que depende de la cantidad de clases a predecir.Por lo que en el proyecto se usa Sigmoide (Existencia de 2 clases),ya que puede definir si algo pertenece o no a la clase en cuestión.

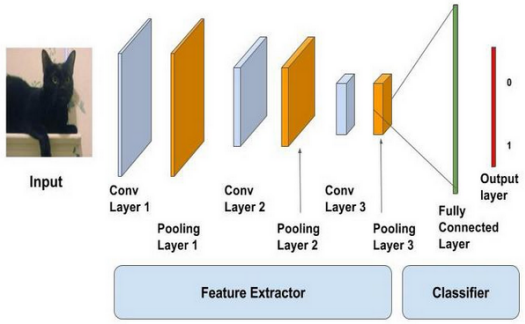

Figure 14: Modelo CNN

2.7. Optimizador y Entrenamiento en 4 etapas

Con un conjunto de dato se realiza el entrenamiento para así entrenar el algoritmo de Machine Learning, Por lo que el conjunto correspondiente a este entrando a nuestro modelo influye en la salida, de esta forma existe una correlación en la salida procesada con la salida de muestra original del dataset.El resultado de esta correlación se utiliza para modificar el modelo. Una vez que las capas se agregan al modelo, se configura una función de puntuación, una función de pérdida y un algoritmo de optimización.
Por lo tanto se define una función de pérdida para medir el rendimiento del modelo en imágenes con etiquetas conocidas,lo que se traduce en una tasa de error entre las etiquetas observadas y las predichas.
El proyecto utiliza **binarycrossentropy** para las clasificaciones binarias (2 clases) con un valor de salida entre 0 y 1, Por lo que en este punto la función más importante es el optimizador. Esta función mejorará iterativamente los parámetros (filtra los valores del kernel, los pesos y el sesgo de las neuronas) para minimizar la pérdida en el proyecto se ocupa **RMSprop** (con valores predeterminados) optimizador muy efectivo.

"Accuracy" es la función métrica definida en el parámetro esta se utiliza para evaluar el desempeño de nuestro modelo, ya que esta función es similar a la función de pérdida, excepto que los resultados de la evaluación métrica no se utilizan al entrenar el modelo (solo para evaluación).

Se utilizó un método de annealing de la tasa de aprendizaje (Learning Ratio) para hacer que el optimizador converja más rápido y más cerca del mínimo global de la función de pérdida que representa el paso por el cual el optimizador recorre el "loss landscape", es decir, que cuanto mayor sea el LR más rápida es la convergencia.Sin embargo, el muestreo es muy pobre con un LR alto y el optimizador probablemente podría caer en un mínimo local, por ello, es mejor tener una tasa de aprendizaje decreciente durante el entrenamiento para alcanzar de manera eficiente el mínimo global de la función de pérdida. Para mantener la ventaja del tiempo de cálculo rápido con un LR alto, se disminuye el LR dinámicamente cada X epochs dependiendo de si es necesario (cuando no se mejora la precisión).

Para un modelo,es imposible saber cómo avanzar, pero hay una cosa que puede saber: si está progresando o perdiendo progreso. Eventualmente, si sigue dando pasos que la lleven hacia abajo, llegará a la base o mínimo. Con la función ReduceLROnPlateau de Keras.callbacks, se definió reducir el LR a la mitad si la precisión no mejora después de 2 epochs, definido en la entrada del parámetro *patience*.
Por último se procede a entrenar los 4 modelos especificando una cantidad de 10 epochs.

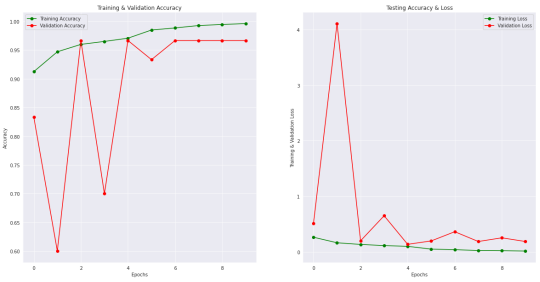

Figure 15: Precisión y Pérdida Experimento 1

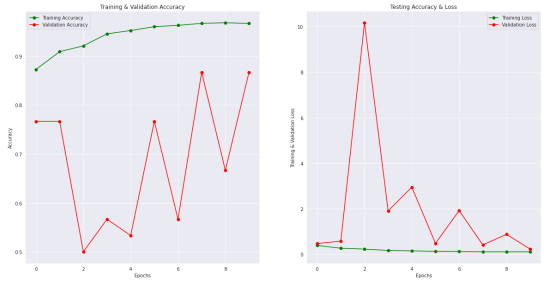

Figure 16: Precisión y Pérdida Experimento 2

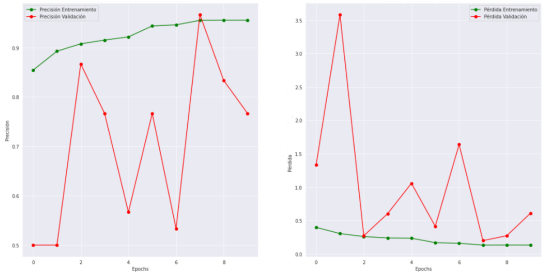

Figure 17: Precisión y Pérdida Experimento 3

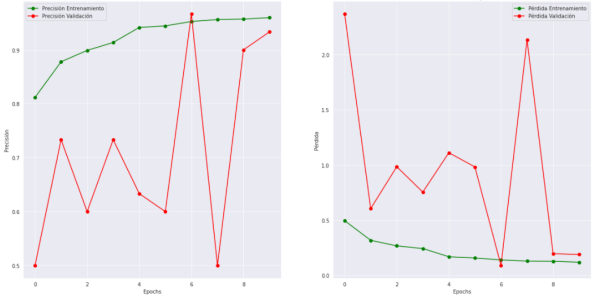

Figure 18: Precisión y Pérdida Experimento 4

Expresado lo anterior se puede visualizar que de forma general la mayor precisión de los modelos alcanza su máximo entre los 6 a 7 epochs en convergencia con la pérdida, la que es menor en ese rango.

Por último se exporta el modelo a utilizar en producción como archivo .h5.

2.8. Predicciones y evaluación del Modelo en 4 Etapas

En base al funcionamiento de las redes neuronales artificiales, en la etapa de predicción resuelven un problema complejo en el reconocimiento de patrones basado en las imágenes, analizando relaciones no lineales en datos. En la última etapa de evaluación es imprescindible medir las variaciones a las que incurre el modelo mediante múltiples formas de medir la predicción. El proyecto en cuestión utiliza un reporte de clasificación que mide la precisión en la generalización, donde Accuracy es la medida de rendimiento que corresponde a una relación entre un resultado predicho correctamente y el total de ejemplos, por lo que una alta precisión, concluye en un modelo mejor. Así mismo, Recall (sensibilidad) es la relación entre los resultados positivos predichos correctamente y todos los ejemplos de la clase, donde en conjunto a la precisión y el promedio ponderado de estos se obtiene el f1-score.

	precision	recall	f1-score	support
Positivo (Class 0)	0.49	0.73	0.59	185
Negativo (Class 1)	0.47	0.24	0.32	185
accuracy			0.48	370
macro avg	0.48	0.48	0.45	370
weighted avg	0.48	0.48	0.45	370

Figure 19: Reporte de clasificación experimento 1

	precision	recall	f1-score	support
Positivo (Class 0)	0.71	1.00	0.83	185
Negativo (Class 1)	1.00	0.60	0.75	185
accuracy			0.80	370
macro avg	0.86	0.80	0.79	370
weighted avg	0.86	0.80	0.79	370

Figure 20: Reporte de clasificación experimento 2

	precision	recall	f1-score	support
POSITIVO (Class 0)	0.91	0.99	0.95	185
NEGATIVO (Class 1)	0.99	0.90	0.94	185
accuracy			0.94	370
macro avg	0.95	0.94	0.94	370
weighted avg	0.95	0.94	0.94	370

Figure 21: Reporte de clasificación experimento 3

POSITIVO (Class 0)	0.80	0.99	0.88	185
NEGATIVO (Class 1)	0.99	0.75	0.85	185
accuracy			0.87	370
macro avg	0.89	0.87	0.87	370
weighted avg	0.89	0.87	0.87	370

Figure 22: Reporte de clasificación experimento 4

En las imágenes anteriores se determina la precisión del modelo en donde obtiene un porcentaje de efectividad 0.94 como máximo para el testing, donde su detalle se aprecia en una matriz de confusión el cual es un resumen de los resultados de la predicción sobre un problema de clasificación, es decir corresponde al número de predicciones correctas e incorrectas resumidas con valores de recuento y se desglosa por clase. Por definición; *la matriz de confusión muestra las formas en que su modelo de clasificación se confunde cuando hace predicciones.*

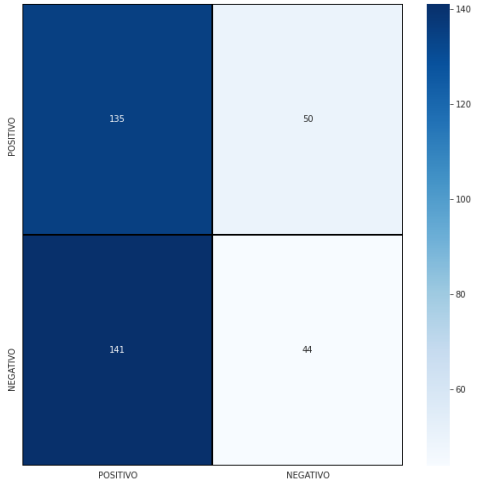

Figure 23: Matriz de confusión experimento 1

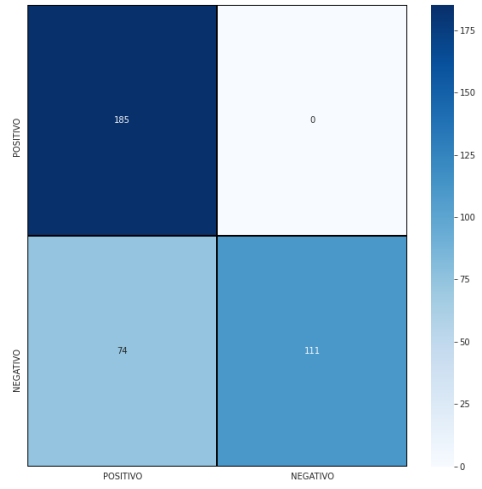

Figure 24: Matriz de confusión experimento 2

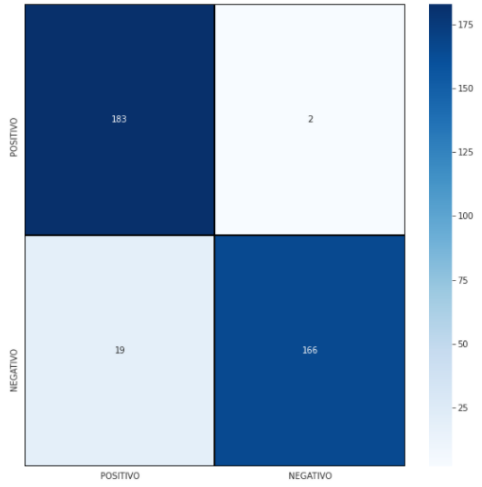

Figure 25: Matriz de confusión experimento 3

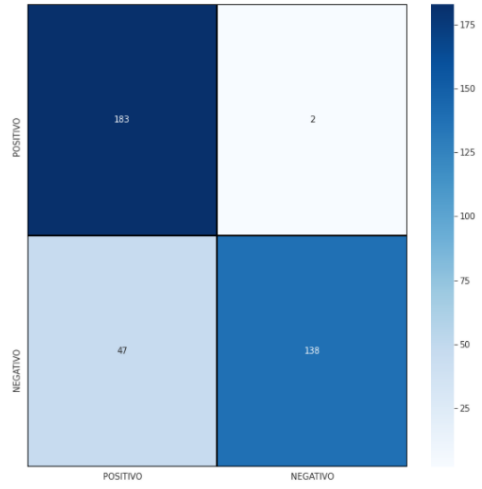

Figure 26: Matriz de confusión experimento 4

La figuras anteriores se ve la matriz de confusión del proyecto donde se aprecian valores mas o menos buenos para la conclusión del modelo en el sentido de la precisión, ya que para el diagnóstico de covid-19 (Positivo) el modelo resulta en visible en el experimento 3.

3. Conclusión

Con la creciente demanda de atención médica, en particular respecto de un médico en tiempos de pandemia, el poder identificar a un paciente con un diagnostico positivo en Covid-19 tendría un impacto positivo en el sistema de salud, solo basta obtener una precisión en los modelos que permitan generar una alta confiabilidad para la comunidad médica y optimizar recursos humanos, recordando además que la radiografía de tórax tiene una gran disponibilidad, con bajo coste, rápida y muy útil en el diagnóstico y tratamiento de emergencia.

Un modelo con características ideales tienen aplicaciones clínicas y que podrían estar en el mercado, para agilizar el proceso médico, teniendo en cuenta que los sistemas de salud han colapsado en gran medida y han quedado en segundo plano enfermedades con un menor perfil, ya que la disponibilidad de tiempo se ha reducido.

Si se logra un buen resultado de modelo CNN, los diagnósticos correspondientes a Covid-19 podrían reconocerse con mayor precisión. Por lo tanto, en la práctica clínica, se podría utilizar un sistema de identificación automática para la monitorización de radiografías.

La propuesta de modelo mediante en el diagnóstico de Covid-19 en CNN podría utilizarse para hacer este tipo de reconocimiento de manera eficiente a partir de las imágenes provenientes de las radiografías, y arrojar un diagnóstico de inmediato.

References

[1] Network, J., 2020. The great coronavirus pandemic of 2020. https://jamanetwork.com/journals/jama/fullarticle/2772746.

[2] de la Salud, O.M., 2020. Coronavirus. https://jamanetwork.com/journals/jama/fullarticle/2772746.

[3] techUK, 2020. The importance of technology in the era of covid-19. https://www.techuk.org/resource/the-importance-of-technology-in-the-era-of-covid-19.html.

[4] Wang, L., Lin, Z.Q., Wong, A., 2020. Covid-net: a tailored deep convolutional neural network design for detection of covid-19 cases from chest x-ray images. Scientific Reports 10, 19549. URL: https://doi.org/10.1038/s41598-020-76550-z, doi:10.1038/s41598-020-76550-z.