Facultad de Energía, las Industrias y los Recursos Naturales no Renovables

CARRERA DE INGENIERÍA EN SISTEMAS

Diagnóstico de Covid-19 Mediante el Análisis de Radiografías Pulmonares Empleando un Modelo Basado en Redes Neuronales Convolucionales (CNN)

Línea de investigación: Inteligencia Artificial

TESIS DE GRADO PREVIA A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN SISTEMAS.

Autor:

- Jonathan Ricardo Tillaguango Jiménez

Director:

- Ing. Oscar Miguel Cumbicus Pineda, Mg.Sc.

LOJA - ECUADOR
2022

CERTIFICACIÓN

Ing. Oscar Miguel Cumbicus Pineda
DOCENTE DE LA CARRERA DE INGENIERÍA EN SISTEMAS

CERTIFICA:

Que el egresado **Jonathan Ricardo Tillaguango Jiménez** autor del presente trabajo de titulación, cuyo tema versa sobre “DIAGNÓSTICO DE COVID 19 MEDIANTE EL ANÁLISIS DE RADIOGRAFÍAS PULMONARES EMPLEANDO UN MODELO BASADO EN REDES NEURONALES CONVOLUCIONALES (CNN)”, ha sido dirigido, orientado, discutido bajo mi asesoramiento y ha sido culminado al 100%, reúne a satisfacción los requisitos exigidos en una investigación de este nivel por lo cual autorizo su presentación y sustentación.

Loja, 17 de septiembre del 2021

Firmado electrónicamente por:
OSCAR MIGUEL CUMBICUS PINEDA

Ing. Oscar Miguel Cumbicus Pineda, Mg. Sc.
DIRECTOR DEL TRABAJO DE TITULACIÓN

AUTORÍA

Yo **Jonathan Ricardo Tillaguango Jiménez,** declaro ser autor del presente trabajo de titulación y eximo expresamente a la Universidad Nacional de Loja y a sus representantes jurídicos de posibles reclamos o acciones legales por el contenido del mismo.

Adicionalmente acepto y autorizo a la Universidad Nacional de Loja, la publicación de mi trabajo de titulación en el Repositorio Institucional - Biblioteca Virtual.

Firmado electrónicamente por:
JONATHAN RICARDO TILLAGUANGO JIMÉNEZ

C.I.: 1106090507
Fecha: Loja, 17 de septiembre del 2021
jonathan.tillaguango@unl.edu.ec
0000-0001-5143-0201

CARTA DE AUTORIZACIÓN POR PARTE DEL AUTOR, PARA LA CONSULTA, REPRODUCCIÓN PARCIAL O TOTAL, Y PUBLICACIÓN ELECTRÓNICA DEL TEXTO COMPLETO

Yo, JONATHAN RICARDO TILLAGUANGO JIMÉNEZ, declaro ser autor del trabajo de titulación que versa: "DIAGNÓSTICO DE COVID-19 MEDIANTE EL ANÁLISIS DE RADIOGRAFÍAS PULMONARES EMPLEANDO UN MODELO BASADO EN REDES NEURONALES CONVOLUCIONALES (CNN)", como requisito para optar al grado de: INGENIERO EN SISTEMAS; autorizo al Sistema Bibliotecario de la Universidad Nacional de Loja para que, con fines académicos, muestre al mundo la producción intelectual de la Universidad, a través de la visibilidad de su contenido de la siguiente manera en el Repositorio Digital Institucional: Los usuarios pueden consultar el contenido de este trabajo en el (RDI), en las redes de información del país y del exterior, con los cuales tenga convenio la Universidad.

La Universidad Nacional de Loja, no se responsabiliza por el plagio o copia de la tesis que realice un tercero.

Para constancia de esta autorización, en la ciudad de Loja, a los 11 días del mes de enero del 2022.

Firma: 

Autor: Jonathan Ricardo Tillaguango Jiménez

Cédula: 1106090507

Dirección: Loja (Bernardo Valdivieso y Leopoldo Palacios, Nro. 160-12)

Correo Electrónico: jrtillaguango@gmail.com

Celular: 0981832948

DATOS COMPLEMENTARIOS

Director de tesis: Ing. Oscar Miguel Cumbicus Pineda, Mg. Sc.

Tribunal de grado: Ing. Pablo Fernando Ordoñez Ordoñez Mg.Sc
Ing. Luis Antonio Chamba Eras, Ph.D.
Ing. Roberth Gustavo Figueroa Diaz

DEDICATORIA

El presente trabajo de titulación, se lo dedico con mucho cariño y afecto a mis padres, Lidio Tillaguango y Gladys Jiménez, quienes me han brindado su apoyo incondicional, el cual ha sido fundamental para lograr mis objetivos desde el primer día en que decidí salir y forjar mi carrera profesional. De igual forma, quiero dedicarlo a mis hermanos Daniela y Alex, los cuales ven en su hermano mayor un ejemplo de superación y constancia. A Evelyn Alexandra, quien ha sido mi sustento emocional y espiritual, esto va para ti, que has hecho de mí una mejor persona, siempre ofreciéndome su amor y apoyo incondicional. A mis amigos, que, gracias a su apoyo moral y consejos, me ayudaron a superar obstáculos desde el primer día en el que empezamos esta maravillosa etapa, gracias por ofrecerme su amistad sincera y ayuda cuando más lo necesite.

Por último, quiero dedicar este trabajo de titulación a todas aquellas personas que han sufrido en carne propia, los efectos de esta pandemia, ustedes son el motivo por el cual decidí llevar a cabo este trabajo y espero haber aportado con un granito de arena en estos tiempos tan difíciles.

Jonathan Tillaguango

AGRADECIMIENTO

Quiero agradecer de forma especial al Ing. Oscar Miguel Cumbicus Pineda, Mg. Sc, quien fue el tutor y guía del presente trabajo de titulación. Sin sus virtudes, su paciencia y constancia exigida, no habría sido posible lograr el cumplimiento de este trabajo. Sus recomendaciones, consejos y guía, me fueron siempre útiles cuando mis ideas me sobrepasaban y no encontraba la forma de canalizarlas. Usted formó parte importante de esta historia, con sus aportes profesionales, los cuales lo caracterizan como un excelente tutor y docente en el alma mater de nuestra prestigiosa Universidad. Muchas gracias por sus palabras de aliento, por el grado de exigencia y perseverancia, por estar allí cuando mis horas de trabajo se hacían confusas.

De igual forma, quiero agradecer a cada uno de los docentes de la carrera de Ingeniería en Sistemas, quienes me impartieron sus conocimientos, consejos y experiencias, a ustedes, les debo todo lo que se. Donde quiera que vaya, los llevaré conmigo en mí transitar profesional. Gracias por su paciencia, por compartir sus conocimientos de manera profesional e invaluable, por su dedicación, perseverancia y tolerancia.

A mis padres, quienes siempre me han apoyado para cumplir mis sueños y aspirar a cosas mejores cada día, gracias por los valores que me han inculcado desde mi niñez, ustedes siempre han sido mi ejemplo a seguir. Ahora que estoy a punto de cumplir una etapa muy importante en mi vida, quiero expresar mis más sinceros agradecimientos a ustedes, que siempre creyeron en mí, y lo demostraron dándome lo más preciado para el ser humano, la educación y el conocimiento, que son la oportunidad de trascender como persona, y así poder aportar para hacer de la sociedad un lugar mejor para nosotros y las futuras generaciones.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1. TÍTULO

Diagnóstico de Covid-19 mediante el análisis de radiografías pulmonares empleando un modelo basado en redes neuronales convolucionales (CNN).

2. RESUMEN

Desde la declaración de la emergencia sanitaria provocada por el Covid-19 en marzo del 2020, hasta la fecha, existen aproximadamente 219 millones de contagiados, de los cuales 4,5 millones han muerto. En Ecuador, se estima que existen 508 mil casos confirmados y aproximadamente 32 mil muertes a causa de esta enfermedad. Pese a disponer de métodos verificados para diagnosticar Covid-19, las pruebas PCR o RT-PCR, tienden a generar falsos positivos y negativos entre el 30% y el 40%. Por tanto, ayudar a los métodos tradicionales a realizar un diagnóstico clínico preciso, usando como datos de entrada radiografías pulmonares, supone un cambio radical en la detección de Covid-19, puesto que, es una alternativa mucho más cómoda para el paciente y lo que es más importante, aumenta el nivel de precisión reduciendo a la vez, las tasas de falsos positivos y falsos negativos. Por tal razón, en el presente trabajo de titulación, se plantea la creación de un modelo basado en la arquitectura de Redes Neuronales Convolucionales (CNN), capaz de analizar radiografías pulmonares para el diagnóstico de Covid-19.

Para obtener la materia prima del modelo, se realizó un protocolo de búsqueda basado en los trabajos relacionados, obteniendo como resultado, un conjunto de datos completo, variado y con radiografías pulmonares aptas para ser usadas por el modelo en la fase de entrenamiento y test. En cuanto a la fase de construcción del modelo diagnos19, se optó por el uso de VGG-16 como modelo base, al cual se le realizó un ajuste fino de sus hiperparámetros usando la técnica de fine tuning, que tras validar y evaluar los resultados mediante la aplicación de técnicas como zero shot learning y human validation, obteniendo como resultado un nivel de precisión de 99.167%, con una sensibilidad igual a 99.167%, con estos resultados se logro superar el 90% de precisión establecido en la pregunta de investigación, además, nuestros resultados son comparables con los valores obtenidos en los modelos de los trabajos relacionados, cuyo valor más elevado en cuanto a precisión fue de 98.27%, con un nivel de sensibilidad igual a 98.93%.

Palabras clave: Covid-19, CNN, VGG16, radiografías pulmonares, Rayos X.

ABSTRACT

Since the declaration of the health emergency caused by Covid-19 in March 2020, approximately 219 million people have been infected to date, of which 4.5 million have died. In Ecuador, it is estimated that there are 508,000 confirmed cases and approximately 32,000 deaths due to this disease. Despite the availability of verified methods to diagnose Covid-19, PCR or RT-PCR tests tend to generate false positives and negatives between 30% and 40%. Therefore, helping traditional methods to make an accurate clinical diagnosis, using lung radiographs as input data, represents a radical change in the detection of Covid-19, since it is a much more comfortable alternative for the patient and, more importantly, increases the level of accuracy while reducing false positive and false negative rates. For this reason, in this degree work, we propose the creation of a model based on the Convolutional Neural Network (CNN) architecture, capable of analyzing pulmonary radiographs for the diagnosis of Covid-19.

In order to obtain the raw material for the model, a search protocol based on related works was carried out, obtaining as a result a complete and varied data set with lung radiographs suitable for use by the model in the training and test phase. Regarding the construction phase of the diagnos19 model, VGG-16 was chosen as the base model, to which a fine tuning of its hyperparameters was performed using the fine tuning technique, which, after validating and evaluating the results by applying techniques such as zero shot learning and human validation, resulted in an accuracy level of 99. 167%, with a sensitivity level equal to 99.167%, with these results it was possible to exceed the 90% accuracy established in the research question, in addition, our results are comparable with the values obtained in the models of the related works, whose highest value in terms of accuracy was 98.27%, with a sensitivity level equal to 98.93%.

Key words: Covid-19, CNN, VGG16, lung radiographs, X-rays.

3. Introducción

En diciembre del año 2019, la Organización Mundial de la Salud (OMS) recibió reportes de pacientes que presentaban cuadros de neumonía aguda, causados por un patógeno de desconocido, proveniente de la ciudad de Wuhan, en la República Popular China [1]. A principios de enero, las autoridades del país asiático identificaron la causa de dicha enfermedad, misma que era provocada por un nuevo tipo de coronavirus, conocido como síndrome respiratorio agudo grave (SARS, por sus siglas en inglés, Severe Acute Respiratory Syndrome) [2]. Este tipo de enfermedad perteneciente a las variaciones de coronavirus (CoV), debido a la apariencia externa de la membrana que la recubre, son muy comunes en especies animales como ganado, camellos, murciélagos y gatos, aunque en ocasiones se puede transmitir entre humanos y propagarse entre su población, tal como ocurrió en el año 2012, tras la aparición del MERS-CoV, en el Reino de Arabia Saudí, dejando como resultado, alrededor de 858 fallecidos [3].

Tanto el MERS-CoV como el SARS-CoV-2 (variante de CoV que provoca Covid-19), según los estudios realizados a las muestras obtenidas, indican que están relacionados a los coronavirus que se originan en los murciélagos [3], [4], [5], [6]. A demás, el epicentro del nuevo Covid-19, tuvo lugar en un mercado de mariscos y animales vivos, lo cual es un indicativo de una propagación de característica zoonótica. No obstante, esta última (SARS-CoV-2), está teniendo efectos mucho más devastadores en la especie humana, que su predecesor el MERS-CoV, debido a su considerable capacidad de transmisión entre humanos, razón por lo cual, la OMS clasificó al brote como una pandemia en marzo del 2020, disparando las alarmas a nivel mundial y la implementaron medidas políticas y sanitarias como respuesta por parte de los países con el mayor número de casos registrados [4], [5].

Desde entonces, la pandemia causada por el COVID-19, se ha expandido a todos los continentes, dejando tras de sí, cerca de 270 millones de casos conformados, de los cuales 5,3 millones han muerto, según cifras oficiales manejadas por Google [7]. Según esta misma fuente, en Ecuador, se estima que existen más de 530 mil casos confirmados y aproximadamente 33 mil muertes. Pese a que se ha desarrollado 13 variedades de vacunas distintas [8], el virus SARS-COV-2, sigue suponiendo una potencial amenaza a la salud, sobre todo en países en los que a pesar de todas las medidas implementadas para su contención y mitigación, existe una escasez de equipo médicos para el diagnóstico oportuno del virus.

Este proceso, al igual que en los inicios de la emergencia sanitaria, se viene

desarrollando mediante la prueba de Reacción en Cadena de la Polimerasa o PCR (Por sus siglas en inglés, Polymerase Chain Reaction) y la RT-PCR o Real-Time PCR, mismas que han dado buenos resultados. No obstante, presentan varios puntos débiles, como los largos periodos de tiempo para la obtención de resultados, con una variación de entre 6 horas hasta 2 a 3 días respectivamente [9], [10]. Además, las pruebas RT-PCR, requieren mucho más tiempo para generar los resultados comparado a la rapidez con la que el virus se propaga, inclusive, estos resultados no son del todo precisos, pues existe una alta tasa de probabilidad de generar tanto, falsos positivos como negativos [10], [11], [12], sobre todo en las etapas iniciales de la enfermedad, las cuales oscilan entre los 5 primeros días luego de que el virus haya ingresado al organismo del portador, convirtiéndolo en una fuente de contagio activa, no obstante, los síntomas ”comunes” como tos seca, fiebre, dificultad para respirar, dolor de garganta, diarrea y cansancio, se presentan cuando la enfermedad se encuentra en una etapa avanzada, por lo general, en un promedio de 8 días luego de contagiarse. Hasta este punto, la probabilidad de que el portador del virus contagie a otros individuos fluctúa entre el 21% [13], [14]. Por tal razón, que este tipo de pruebas no son eficientes para el diagnóstico oportuno en las primeras etapas de la enfermedad, donde el paciente es un potencial foco de contagio para la población en la que reside, suponiendo además de un grave peligro para quienes padecen de enfermedades crónicas como la presión arterial, diabetes, problemas al corazón, problemas pulmonares o algún tipo de cáncer, puesto que tienen un alto riesgo de desarrollar síntomas mucho más graves que aquellas personas que no las padecen, llegando inclusive a causar la muerte [3]. Por lo tanto, uno de los principales problemas de muchos países (entre ellos Ecuador), es la falta de infraestructura para atender a pacientes afectados por la pandemia, y de mecanismos que permitan una identificación oportuna evitando así, que los grupos más vulnerables se vean expuestos ante esta grave situación. Obviamente, esta problemática desencadena en casos de infección cruzada, que se producen cuando existen aglomeraciones de personas las cuales acuden a estos centros de salud sin las debidas precauciones sanitarias, aumentando potencialmente las posibilidades de contraer o transmitir el virus, tanto a los demás pacientes, como al personal médico que trabajan en las primeras líneas de control [10]. Es por ello, que casos como estos suponen nuevos retos para todos los campos los cuales puedan aportar con métodos y herramientas que ayuden a contrarrestar esta emergencia sanitaria, y en nuestro caso, como el área tecnológica, disponemos de las herramientas para hacerlo.

Por ejemplo, una opción viable, es la aplicación de técnicas de inteligencia artificial (IA),

debido a los recientes avances en los campos de ingeniería y medicina, los cuales han permitido implementar una serie de métodos capaces de diagnosticar con un alto grado de precisión, si una persona es o no portadora del virus Covid-19. Estos modelos van desde el análisis de tomografías computarizadas (TC) e imágenes de rayos X [5], [15]–[16], análisis de sangre [17], [18], análisis de voz [16], [19], [20], hasta el diagnóstico a partir de síntomas que el paciente ingresa en el sistema en cuestión [9].

No obstante, de todos los métodos de diagnóstico antes mencionados, existe un alto nivel de aceptación y precisión en los modelos basados en TC y rayos X. De estos dos últimos, el más asequible en cuanto a disponibilidad y factibilidad, es el método que emplea radiografías para el diagnóstico de Covid-19 [5], [15]–[16].

Por lo tanto, en el presente proyecto de TT, se ha implementado un modelo basado en redes neuronales convolucionales (CNN) capaz de diagnosticar de forma precisa el Covid-19, en etapas tempranas, a partir del análisis de radiografías pulmonares, justamente donde las pruebas PCR convencionales presentan la mayor tasa de incertidumbre. Este tipo de modelo de diagnóstico se ha venido desarrollando en los últimos años, teniendo su principal acogida tras la aparición de la nueva pandemia, además, numerosos estudios basados en aprendizaje profundo (DL), fueron capaces de detectar la enfermedad utilizando este enfoque, obteniendo resultados prometedores con tasas de precisión y sensibilidad superiores al 90% [15], [21]–[26]. Tras su implementación y desarrollo se espera un ahorro médico, logístico y de recursos humanos reflejados en un diagnóstico rápido y preciso disponible para los lugares en los que las pruebas de RT-PCR no son suficientes.

4. Revisión de literatura

A continuación, se presentan las bases teóricas que sustentan el presente Trabajo de Titulación (TT), dicha información ha sido obtenida a través de un proceso de revisión bibliográfica. Esta sección inicia con los conceptos más relevantes entorno a la temática, y posteriormente, se presenta una tabla con todos los estudios que han sido seleccionados como resultado del proceso de revisión [27].

4.1. Conceptos preliminares

4.1.1. Visión por computadora

La visión por computadora o "*computer vision*", es la extracción automatizada de información de las imágenes, las cuales van desde modelos 3D, posición de la cámara, reconocimiento de objetos, agrupación y búsqueda de contenido. También puede incluir

la deformación de las imágenes, eliminación de ruidos y realidad aumentada. Muchas aplicaciones de este tipo de aprendizaje tratan de imitar la visión humana, sumados a enfoques complementarios como la geometría y la estadística, puede ser fundamentales para resolver problemas que involucren esta rama de la inteligencia artificial, pues la visión por computadora contiene una mezcla de programación, modelado y matemática que lo convierte en un campo de estudio con muchas alternativas para la resolución de problemas puntuales [28].

4.1.2. Redes Neuronales

Las Neural Networks o Redes Neuronales, son un tipo de inteligencia artificial (IA), creado con el objetivo de reproducir las funciones elementales que el cerebro humano emplea para la adquisición de conocimiento. Este tipo de arquitectura, está basada en una estructura de capas, las cuales, se constituyen por un gran número de unidades llamadas neuronas, siendo esta, su estructura más básica. Las neuronas por su parte, se encuentran interconectadas con neuronas pertenecientes a otras capas mediante un enlace [29].

Algo similar ocurre con las neuronas que conforman nuestro cerebro (**Fig. 1**.**a**), debido a que estas, también se interrelacionan unas con otras, con el fin de obtener el conocimiento, para ello hacen uso de sus tres elementos fundamentales, que son:

- **Dendritas:** se encargan de captar los impulsos nerviosos emitidos por otras neuronas.
- **Soma:** se encarga de procesar los impulsos eléctricos captados por las dendritas.
- **Axón:** envía el impulso nervioso hacia las neuronas contiguas.

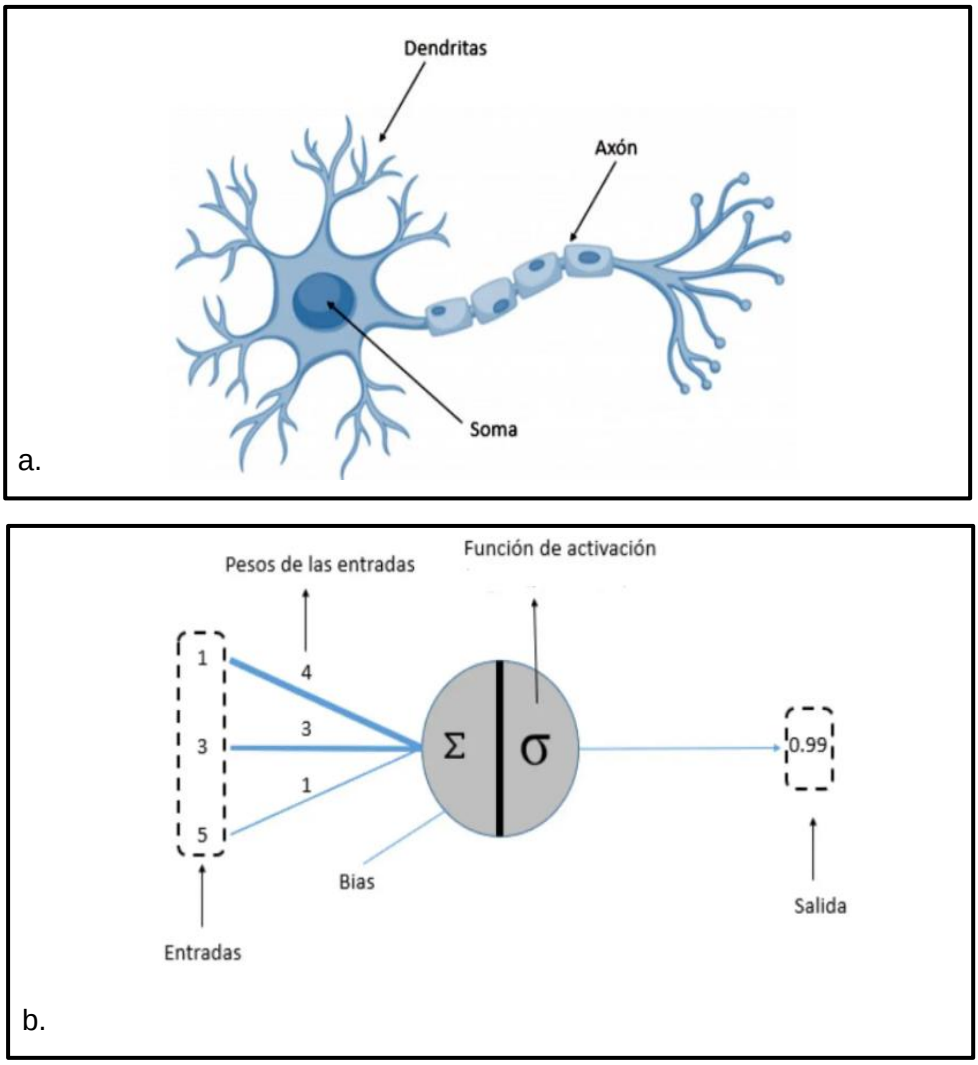


Fig. 1 *a) Estructura de una neurona. b) Estructura de una neurona artificial.*

Como resultado de esta interacción, el impulso nervioso atraviesa un conjunto de neuronas desencadenando una respuesta, la cual puede ser corporal o cognitiva [30].

En el caso de las neuronas artificiales (**Fig. 1*.b***), los enlaces producidos entre las neuronas, son el resultado de la suma de las entradas multiplicada por sus respectivos pesos (impulso nervioso), el cual se trasmitirá dentro de la red neuronal, conformando así, una estructura compleja en forma de red, cuyos enlaces son los pesos de las conexiones. Un esquema generalizado se muestra en la **Fig. 2**.

Por tanto, en una situación particular, los pesos de las conexiones tienen que modificarse para producir la salida deseada. Estructuralmente, tomando como referencia la gráfica b de la **Fig. 1***,* dentro de la base misma de las redes neuronales se pueden distinguir tres tipos de capas, las cuales se pueden relacionar con las partes que conforman las neuronas cerebrales:

- **Capas de entrada:** Comprende el conjunto de neuronas mediante las cuales la información accede a la red. Por lo general, en las capas de entrada no se realiza ningún tipo de procesamiento que requiera enlaces neuronales, más bien la entrada de la información se lo realiza con procesos que son ajenos a los de la red neuronal.
- **Capas ocultas:** Son las capas que reciben la información proveniente de las capas de entrada. Esta información es sumada mediante una función de transferencia, en la que es posible aplicar un peso diferente a la información que llega de cada neurona. A esta unión se le aplica una función de activación, es por ello que es una de las partes más importantes de la red. Es obligatorio que esta función sea no lineal, ya que de lo contrario al concatenar capas con funciones lineales se podría resumir finalmente en una sola capa lineal, y de esta manera no se podrían obtener buenos resultados en el proceso [29].
- **Capa de salida:** Por último, esta capa es la encargada de recopilar la información y proveer una salida dependiendo del tipo de red y la clasificación que se desee obtener.

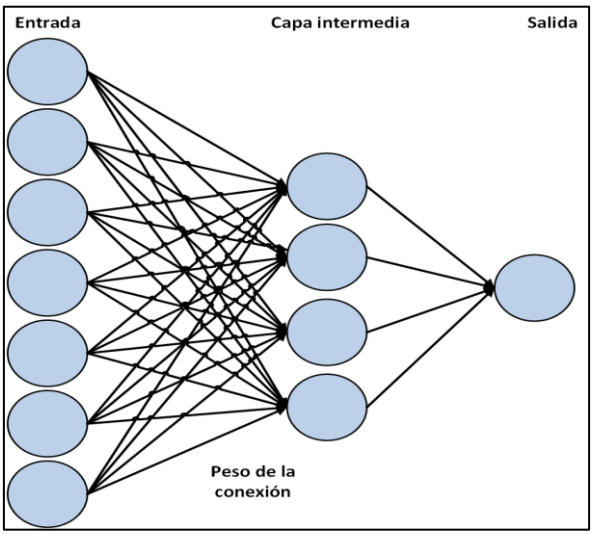


Fig. 2 *Esquema general de una Red Neuronal.*

La implementación de este tipo de arquitecturas es factible en dominios de procesos donde el conocimiento es explícito, es decir, donde se requiere que el modelo sea capaz de capturar las relaciones entre los datos. En estos casos, las redes neuronales tienen la capacidad de agregar y predecir un estado, deducir medidas, reconocimiento de modelos y patrones, clasificar elementos, entre otros. Estas aplicaciones se han potenciado en los últimos años debido al avance del procesamiento computacional, lo

que ha permitido la simulación de redes mucho más complejas y el descubrimiento de potentes algoritmos de aprendizaje [30], [31].

4.1.3. Convolucional Neural Networks (CNN)

Las Redes Neuronales Convolucionales o CNN por sus siglas en inglés, son un tipo de Red Neuronal Artificial de aprendizaje supervisado, que procesa sus capas imitando al córtex visual del ojo humano para identificar distintas características que le permiten a un computador, clasificar y detectar patrones en un conjunto de datos de entrada, que generalmente son imágenes. Para ello, la CNN contiene varias capas ocultas especializadas y ordenadas de forma jerárquica, tal como se muestra en la **Fig. 3**, esto quiere decir, que las primeras capas pueden detectar líneas, curvas y, a medida que se avanza a las capas más profundas, estas se van especializando hasta llegar a reconocer formas mucho más complejas, como el rostro humano, figuras en 3D, e incluso, aplicaciones médicas como reconocimiento de tumores, lesiones de órganos internos, etc [27].

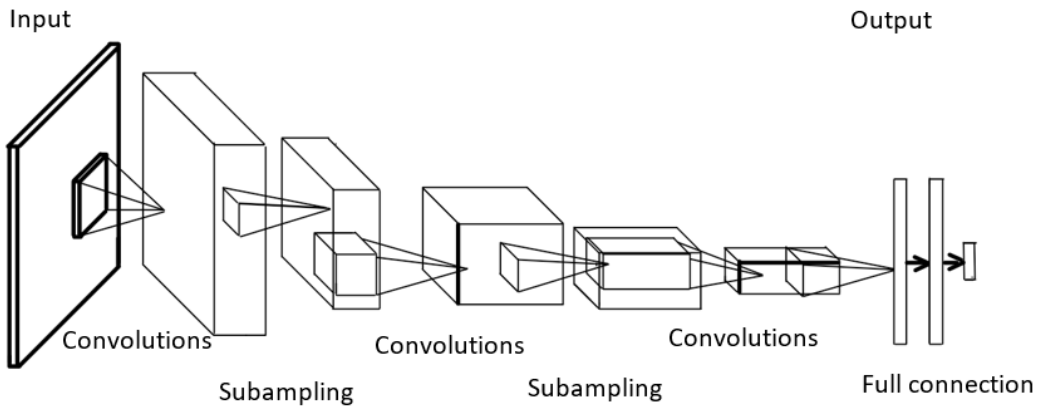


Fig. 3 *Estructura de una CNN, empleada para el reconocimiento de una entrada x. Se puede asumir que x puede ser una imagen la cual será el objeto de entrada de la red.*
Fuente: [32]

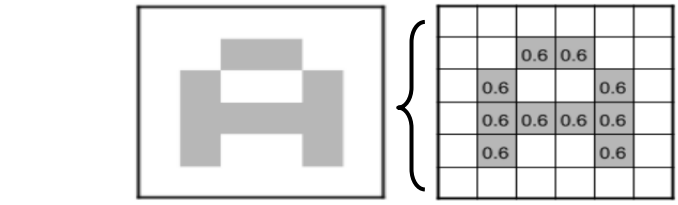
Considerando la figura anterior, el objetivo de la CNN será aprender, bajo cierto nivel de supervisión, una diversidad de patrones contenidos de forma implícita dentro de las imágenes. Estos patrones serán obtenidos a medida que las imágenes de entrada atraviesen las diferentes capas especializadas de convolución, por lo tanto, las primeras capas se encargarán de detectar curvas, líneas, contrastes y colores, para luego especializarse a medida que llega a las capas más profundas.

Asumiendo que la entrada de la CNN es una imagen, se requiere una cantidad considerable de ellas para que el modelo sea capaz de aprender a reconocer por sí sola, una diversidad de objetos y patrones dentro de las imágenes.

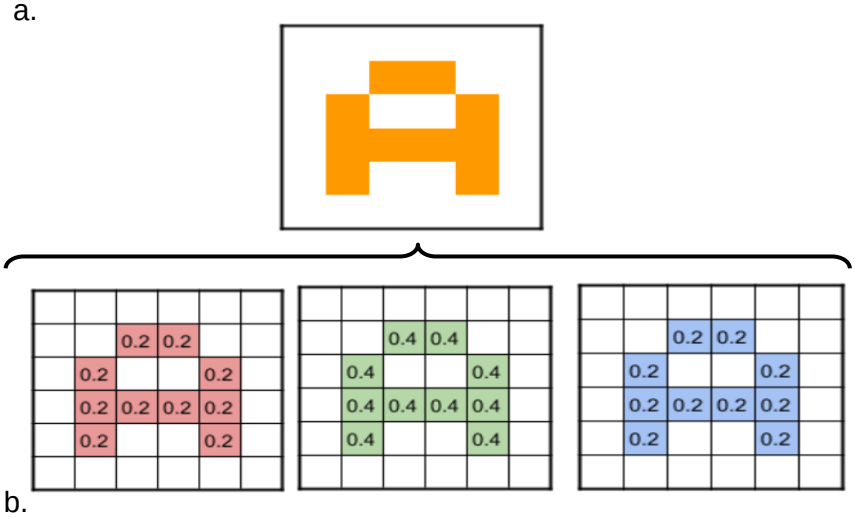
4.1.3.1. Pixeles y neuronas

El pixel es la unidad básica que constituye una imagen digital, cada pixel puede tomar un valor de entre 0 y 255, conformado la totalidad de la imagen, por lo que se puede decir, que una imagen es una matriz de pixeles, y es precisamente este concepto de donde parte la entrada de las redes neuronales, ya que la CNN toma como entrada la imagen como si fuese una matriz, tal como se muestra en la **Fig. 4**.

Para comprender la relación existente entre pixeles y neuronas, se plantea un ejemplo, el cual toma como referencia la representación de una imagen en escala de grises, tal como se muestra en la **Fig. 4 a,** cuyas dimensiones son de 28 x 28 (pixeles de ancho y alto) eso equivale a 784 neuronas necesarias para la primera convolución [33]. En el caso de la **Fig. 4 b,** es un poco más complejo, puesto que se trata de una imagen compuesta por más de un color, específicamente los del formato RGB (Red, Green, Blue) por lo que se requieren 3 canales para la representación de cada uno de los colores primarios que conforman la imagen original (véase la **Fig. 4 b),** por ende, las mismas dimensiones del caso anterior 28x28 se debe multiplicar por el número de capas adicionales, es decir, 28 x 28 x 3 = 2352 neuronas de entrada necesarias para la primera convolución.



a.



b.

Fig. 4 *a) Representación de una imagen en escala de grises y su respectiva matriz de pixeles b) Representación de una imagen RGB y sus respectivas matrices que constituyen la gama de colores.*

4.1.3.2. *Convoluciones*

El proceso de convolución, consiste en agrupar pixeles cercanos de la imagen de entrada, que en realidad es una matriz de pixeles. Una forma más específica de describir el proceso de convolución se encuentra en [27], quien define a este proceso como una operación matemática, en la que una función se "aplica" a otra función, cuyo resultado es una "mezcla" de las dos funciones. Esta mezcla de funciones, hace referencia a las operaciones que se realizan entre la matriz de entrada, cada vez que atraviesa una capa de la CNN, es decir, si la matriz de entrada tiene por dimensiones 28 x 28, esta operará escalarmente contra una pequeña matriz llamada kernel [33]. Suponiendo que la dimensión en pixeles del kernel es de 3 x 3, este recorrerá todas las neuronas de entrada y generará una nueva matriz de salida que se obtiene mediante una función de activación, $f(x) = \max(0, x)$ conocida como ReLu (Rectifier Linear Unit). El resultado de la función de activación, es un mapa de detección de características, la cual es enviada como input a las capas neuronales adyacentes de la CNN (véase la **Fig. 5**).

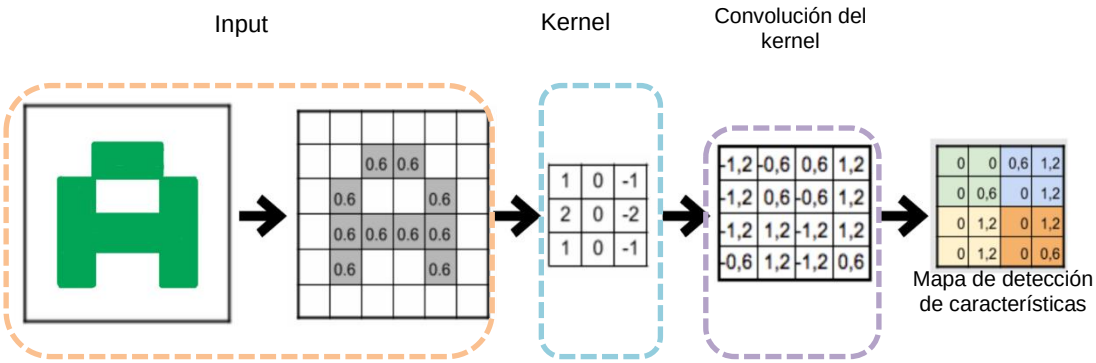


Fig. 5 *Esquema general de la entrada (input) y su kernel con el cual se realizarán las operaciones en la matriz. El resultado de la convolución es una matriz de detección de características.*

4.1.3.3. *Subsampling*

Es el proceso en el cual se reduce la cantidad de neuronas previo a realizar la siguiente operación de convolución en las capas neuronales adyacentes. Este proceso evita que el número de neuronas se dispare, optimizando así, el uso de recursos y el aumento de la capacidad de procesamiento. Por ejemplo, tomando como referencia la operación de convolución de la **Fig. 5**, donde las dimensiones de entrada son de 28x28 pixeles, cuyo resultado en la primera capa de entrada es de 784 neuronas, y luego de la primera convolución se obtiene una capa oculta de 25 088 neuronas, que en realidad equivalen a 32 mapas de características de 28 x 28, si el kernel es de 2x2. Obviamente sin la operación de reducción de neuronas o Subsampling, la cantidad de

neuronas necesarias para calcular los siguientes mapas de características tendería a crecer desmesuradamente. Es por ello que el Subsampling, tiene mucha importancia porque es capaz de reducir el tamaño de la próxima capa neuronal tomando las características más importantes detectadas por cada uno de los filtros (operación del input y su respectivo kernel) [33]. Este proceso se puede apreciar gráficamente en la **Fig. 6**.

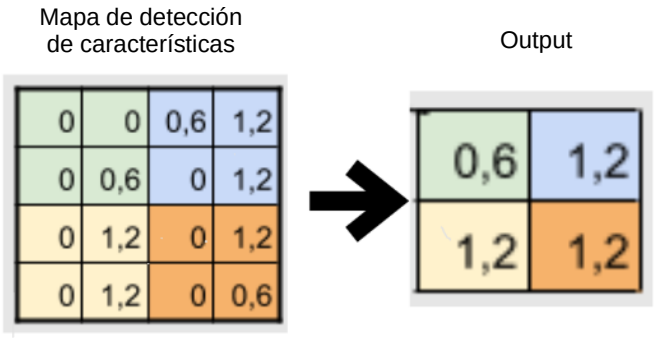


Fig. 6 *Aplicación de la función Subsampling de 2x2 en donde se reduce el mapa de detección de características a la mitad, siendo la salida, y posterior entrada de las demás capas neuronales*

4.1.4. Coronavirus

Los coronavirus son una amplia familia de virus, que se subdividen en cuatro géneros: *alphacoronavirus, betacoronavirus, gammacoronavirus* y *deltacoronavirus*. Estos tipos de coronavirus, son los causantes de provocar enfermedades de tipo respiratorio, comúnmente, en animales domésticos y silvestres, por lo que suele ser de interés veterinario más que de la rama de la medicina humana. Aunque existen casos particulares que son de importancia médica, los cuales pertenecen a al grupo de los betacoronavirus. Desde el punto de vista ecoepidemiológico, se pueden clasificar en dos grupos: coronavirus adquiridos en la comunidad (o coronavirus humanos, HCoV) y coronavirus zoonóticos[1] [34], es decir, tienen la capacidad de transmitirse de animales a personas generando a su paso grandes brotes epidemiológicos de enfermedad respiratoria grave.

En cuanto a su estructura, los coronavirus tienen una forma esférica irregular, con una

[1] La zoonosis es una enfermedad infecciosa que ha pasado de un animal a los humanos. Según la OMS [101], los patógenos zoonóticos pueden ser bacterias, virus, parásitos o agentes no convencionales y propagarse a los humanos por contacto directo o a través de los alimentos, el agua o el medio ambiente, siendo una de las principales causas de origen las alteraciones en la producción y el comercio de productos de origen animal destinados a la alimentación y otros usos.

envoltura lipídica compuesta por tres proteínas ancladas en ella, denominadas E (envoltura), M (membrana) y S (del inglés, spike, o espícula), que otorga al virión (partícula infecciosa) la apariencia de una corona (véase la **Fig. 7 A**)

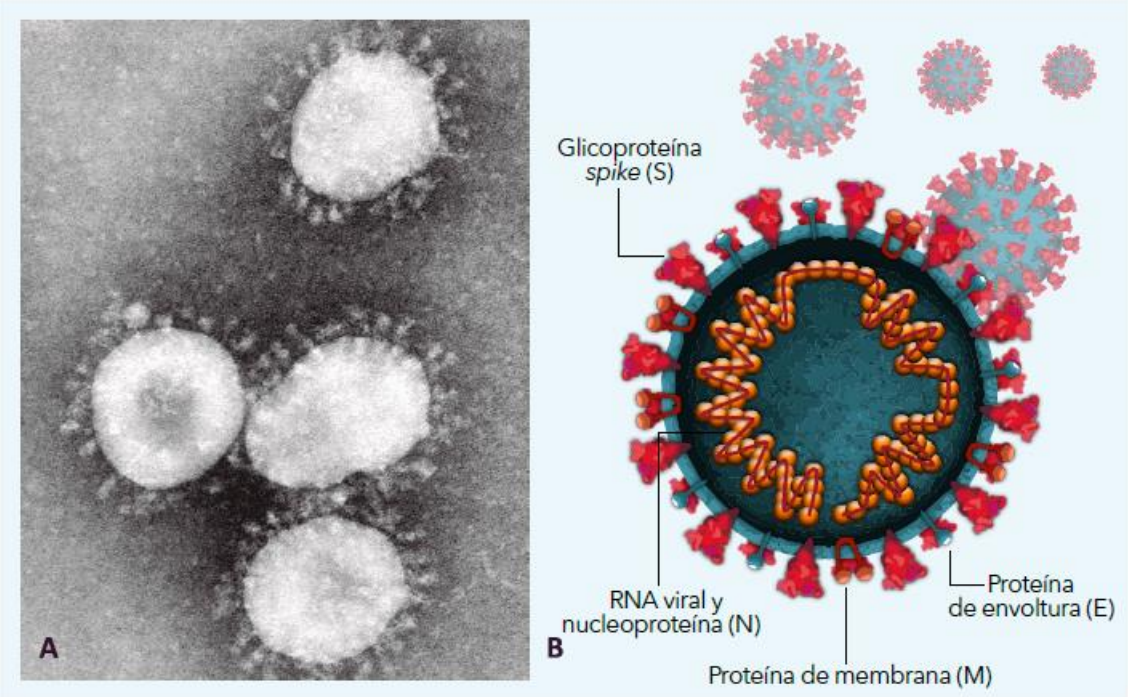


Fig. 7 A) *Microfotografía del virus SARS-CoV-2****. B)*** *Esquema de la estructura del SARS-CoV-2, que muestra los diferentes componentes estructurales.*
Fuente*:* [35].

De los 7 tipos de coronavirus capaces de provocar infecciones respiratorias en humanos, dos son las variantes que provocan cuadros mucho más graves, llegando a causar la muerte en los casos más severos, estas variantes son: el síndrome respiratorio agudo grave (SARS-CoV-2), causante de la enfermedad por coronavirus originada a finales de 2019 (COVID-19) y el coronavirus causante del síndrome respiratorio de Oriente Medio (MERS-CoV) [35], [36].

4.1.4.1. MERS-CoV

El síndrome respiratorio de oriente medio (MERS, por sus siglas en inglés, se reportó por primera vez en Arabia Saudita en 2012 y luego se propagó a otros países) es una enfermedad respiratoria grave que involucra principalmente al tracto respiratorio superior. Causa fiebre, tos y dificultad para respirar. Aproximadamente el 30% de las personas que han contraído esta enfermedad han muerto y hay casos en los que las personas solo presentaron síntomas leves [37], [38].

4.1.4.2. SARS-CoV-2

Denominado como virus del síndrome respiratorio agudo severo tipo-2 (SARS-CoV-2) o Covid-19, es un nuevo tipo de coronavirus de origen zoonótico, detectado por primera vez en diciembre de 2019 en la ciudad de Wuhan, provincia de Hubei, en China.

Este tipo de virus se propaga principalmente, por medio de gotas respiratorias entre personas que hayan tenido contacto directo con un portador del virus. Estas partículas varían en cuanto a tamaño, es decir, van desde las más grandes llamadas 'gotículas respiratorias', hasta las más pequeñas, llamadas 'aerosoles' [35].

En cuanto al periodo de incubación y replicación viral, por lo general, varía de entre 2 a 7 días, extendiendo su sintomatología a 2 semanas aproximadamente, provocando en las personas infectadas, síntomas de leves afecciones respiratorias y los pacientes que desarrollan un cuadro grave de la enfermedad requieren oxígeno y aquellos que llegan a un estado crítico precisan cuidados intensivos.

Existen estudios como [35], [39], [40] en los cuales se especifica que existe una tendencia a que los niveles de criticidad de la enfermedad es mayor en cierto intervalo de edades, siendo los más afectados las personas de entre 30 y 80 años. En el caso de personas menores a los 19 años, pese a ser expuestos a una carga viral relativamente alta, el número de contagios no supera el 1% del total a nivel mundial. En cuanto al total de contagiados, se estima que entre el 7% y el 10%, progresan a enfermedad severa, y que la tasa de letalidad fluctúa entre el 1% y 3%, aunque estas tasas varían dependiendo de las comorbilidades en los pacientes y del país en que reside, sin embargo, estos porcentajes son estimados en base al número de muertes y al número de casos confirmados.

4.1.5. Inteligencia Artificial

La inteligencia artificial (IA) o AI por sus siglas en inglés (Artificial Intelligent), es la ciencia e ingeniería que permite diseñar y programar ordenadores de forma que realicen tareas que requieren inteligencia, es decir que pueden razonar, aprender, reunir conocimiento, comunicarse, manipular y percibir los objetos. En otras palabras, la IA es la ciencia e ingeniería que permitirá replicar la inteligencia humana mediante máquinas [27], [41].

4.1.6. Machine Learning

El aprendizaje automático o Machine Learning (ML) por sus siglas en inglés, es una disciplina científica vinculada a la IA, que busca el desarrollo de sistemas con la capacidad de aprender por sí mismos usando el aprendizaje del contexto sobre un problema particular, por medio de la extracción de patrones complejos en un conjunto extenso de datos [42] , es decir, descubre una estructura compleja en grandes conjuntos de datos mediante el uso del algoritmo de retropropagación para indicar cómo una máquina debe cambiar sus parámetros internos que se utilizan para calcular la representación en cada capa a partir de la capa anterior, de tal manera que el tiempo y la intervención humana se reduzcan al mínimo. El ML ha sido aplicado con éxito en campos que van desde el reconocimiento de patrones, visión por computadora, ingeniería de naves espaciales, finanzas, entretenimiento, biología computacional, hasta aplicaciones biomédicas y médicas.

4.1.7. Polimerase Chain Reaction (PCR)

La PCR o 'Reacción en Cadena de la Polimerasa', es una prueba de diagnóstico que permite detectar un fragmento del material genético de un patógeno. En la pandemia de coronavirus, como en tantas otras crisis de salud pública relacionadas con enfermedades infecciosas, se está utilizando para determinar si una persona está infectada o no. Mediante esta prueba se localiza y amplifica un fragmento de material genético que, en el caso del coronavirus, es una molécula de ARN. El análisis de este tipo de muestras obtenidas de los pacientes se lleva a cabo en un laboratorio de microbiología. Si la prueba detecta ARN del virus, el resultado es positivo y se confirma que esa persona está infectada por el SARS-CoV-2. Si la técnica de PCR no detecta el material genético del virus, la persona no estaría infectada [9], [17], [19].

4.1.8. Real Time Polymerase Chain Reaction (RT-PCR)

Es un método nuclear capaz de detectar la presencia de material genético específico de los patógenos, como los virus (SARS-COV-2). Inicialmente el método utilizaba marcadores de isótopos radiactivos para detectar cierto tipo de material genético, pero, tras la realización de mejoras, el marcado isotópico se ha sustituido por marcadores especiales, que suelen ser colorantes fluorescentes. A diferencia de la PCR convencional, que solo arroja los resultados al final, esta técnica permite a los científicos observar los resultados de manera casi inmediata mientras el proceso sigue en curso. Aunque actualmente la RT-PCR en tiempo real es el método que más se utiliza para

detectar los coronavirus, muchos países siguen necesitando ayuda para poner en marcha esta técnica y utilizarla [9], [14].

4.1.9. Radiografías

Según [45], [46] las radiografías son un proceso médico rápido e indoloro, que se aplica al paciente con el fin de generar imágenes de la estructura interna del cuerpo. Este proceso consiste en exponer al individuo al barrido de haces de rayos X, los cuales atraviesan los órganos internos y son absorbidos en diferentes cantidades según la densidad del tejido, es decir, los materiales densos, como huesos y metales, aparecen de color blanco, mientras que tejidos menos densos como órganos y músculos, tienden a ser representados mediante colores más obscuros.

Existen diferentes tipos de radiografías que se aplican dependiendo de la zona de interés, no obstante, para el presente TT se requieren de radiografías pulmonares debido a que el virus que provoca Covid-19 (SARS-CoV-2) afecta principalmente a los pulmones, razón por la que las radiografías pulmonares son de nuestro absoluto interés. En el caso particular de las radiografías pulmonares o rayos X de tórax, se aplican al paciente de tal forma que se produzcan imágenes de los pulmones, el corazón, los vasos sanguíneos, las vías respiratorias y los huesos del tórax, incluyendo la columna

Fig. 8 *Radiografía Pulmonar en la que se aprecian los diferentes órganos internos, los cuales se represente en una escala de grises dependiendo de la densidad de su composición.*

vertebral (véase la **Fig. 8**). De estos órganos, los pulmones, al tratarse de órganos con una densidad de tejidos relativamente baja, aparecerán de color negro, evidenciado así, la presencia de líquidos dentro de los pulmones o alrededor de ellos, la presencia de aire rodeando a los pulmones y daños al tejido pulmonar provocados por diferentes afecciones respiratorias, entre ellos la Covid-19, que provoca en la mayoría de los casos la opacidad de vidrio esmerilado.

4.1.10. Opacidad en vidrio esmerilado (GGO)

La opacidad de vidrio esmerilado, conocido también como Ground-Glass Opacity (GGO, por sus siglas en inglés), es un área nebulosa provocada por el aumento de la atenuación del pulmón que, a su vez, se produce cuando el material genético de un virus (SARS-CoV-2) se ha multiplicado, infectando en el proceso a las células de los órganos afectados (pulmones), produciendo proteínas que complementen su estructura viral y le permiten propagarse. Durante este proceso, el virus destruye la célula infectada, la abandona e infecta nuevas células produciendo lesiones, las cuales son denominadas como “lesiones radiológicas” debido a que las consolidaciones y nódulos deteriorados de los pulmones son observables en radiografías pulmonares o tomografías computarizadas. Véase la figura **Fig. 9**.

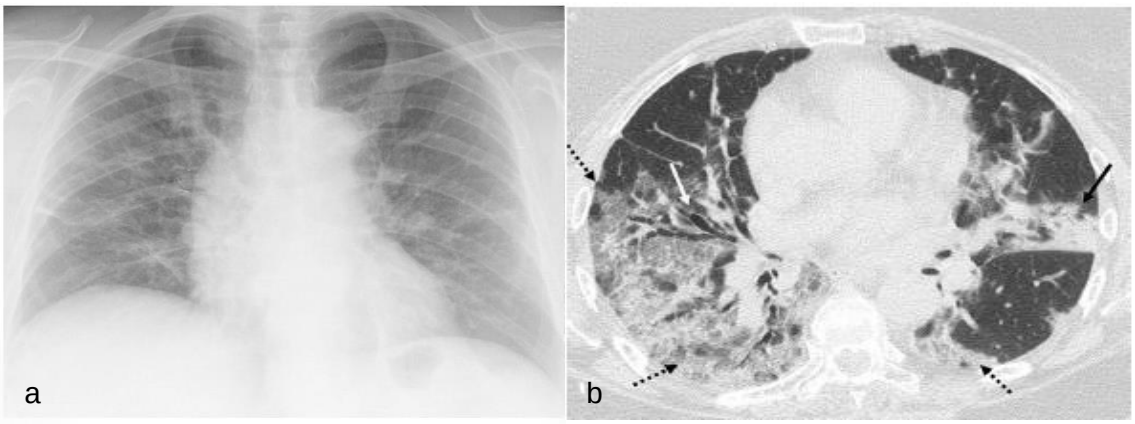


Fig. 9 *a) Radiografía pulmonar de paciente COVID-19. b) Tomografía computacional de tórax, realizada a paciente Covid-19*

Las alteraciones mostradas en la imagen anterior, suelen ser evidentes a partir del quinto día de contagio, aunque sea posible observar GGO durante todo el proceso de desarrollo de los síntomas, es mucho más evidente durante el periodo comprendido desde el día 5 al 11, según un estudio publicado en Elsevier [47] y respaldado por los trabajos relacionados TR01, TR02, TR03, TR06, TR11, TR12, TR13, TR14, TR19, TR21, TR22, TR26, TR25, TR27, TR33 y TR41. Estos estudios sugieren que el GGO, es una característica consistente entre los pacientes sospechosos para diagnóstico

clínico Covid-19+, así como para los pacientes pertenecientes a la Unidad de cuidados intensivos (UCI), como a los diagnosticados con la enfermedad pero que no requieren ingresar a la UCI.

4.1.11. Matriz de confusión

Es una herramienta utilizada en el campo de la IA, sobre todo en el ML, la cual permite visualizar el desempeño de un algoritmo, conocido generalmente como modelo. Cada columna de la matriz representa el número de predicciones de cada clase, mientras que cada fila representa las instancias en la clase real (véase la **Fig. 10**) [48]. En pocas palabras la matriz de confusión permite observar que tipos de aciertos y errores tiene nuestro modelo al momento de procesar una entrada.

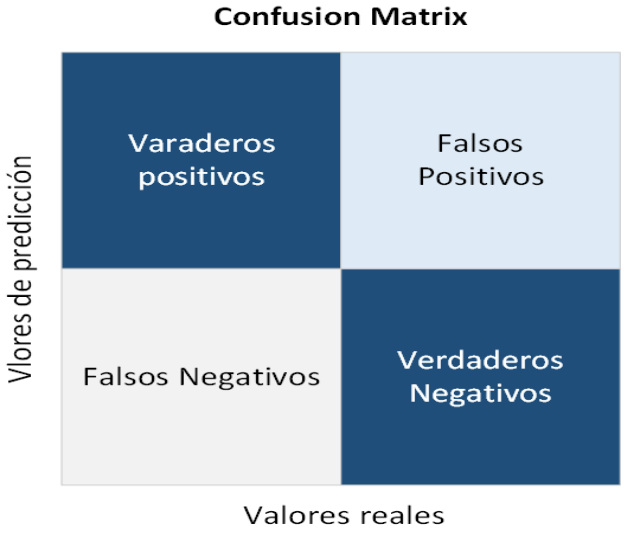


Fig. 10 *Estructura general de una Matriz de confusión.*

En nuestro caso, un modelo diseñado para diagnosticar (clasificar) Covid-19 a partir de radiografías pulmonares, el modelo deberá clasificar la entrada entre dos clases, positivos o negativos para Covid-19, por lo que existen 4 opciones dentro de la matriz de confusión:

1. *Paciente que tiene covid-19* y el modelo lo clasificó como positivo para covid19. Esto equivale a un verdadero positivo o VP.
2. *Paciente que no tiene covid-19* y el modelo lo clasifico como negativo para covid19. Este equivale a un verdadero negativo o VN.
3. *Paciente que tiene covid-19* y el modelo lo clasificó como negativo para covid19. Éste equivale a un falso negativo o FN.
4. *Paciente que no tiene covid-19* y el modelo lo clasificó como positivo para covid19. Esto equivale a un falso positivo o FP.

4.1.12. Accuracy (Exactitud)

Es la proporción del número total de predicciones que fueron correctas. Es decir, su valor hace referencia a lo cerca que está el resultado de una medición del valor verdadero. En términos estadísticos, la exactitud se relaciona con el sesgo de estimación, representando la proporción de los resultados verdaderos positivos y verdaderos negativos) dividido entre el número total de casos examinados (verdaderos positivos, falsos positivos, verdaderos negativos y falsos negativos) [48], por lo que se define de la siguiente manera:

Ecuación 1

$$Accuracy = \frac{VP + VN}{VP + FP + VN + FN}$$

Donde:

$VP = Verdaderos\ positivos$

$FP = Falsos\ positivos$

$VN = Vedaderos\ negativos$

$FN = Falsos\ negativos$

4.1.13. Precisión (Precision)

La precisión se refiere a la dispersión del conjunto de valores obtenidos a partir de un conjunto de mediciones realizadas a un fenómeno específico con una magnitud definida. Se representa por la proporción de verdaderos positivos dividido entre todos los resultados positivos (tanto verdaderos positivos, como falsos positivos) [48]. Cuanto menor es la dispersión mayor la precisión.

Está definido por:

Ecuación 2

$$Precision = \frac{VP}{VP + FP}$$

Donde:

$VP = Verdaderos\ positivos$

$FP = Falsos\ positivos$

4.1.14. Sensibilidad (Recall o sensitivity)

Conocida como tasa de verdaderos positivos, calcula la proporción de los casos positivos que el modelo ha clasificado correctamente [48]. Se define como:

Ecuación 3

$$Sensibilidad = \frac{VP}{VP + FN}$$

Donde:

$VP = Verdaderos\ positivos$

$FN = Falsos\ negativos$

4.1.15. Especificidad

Conocida como tasa de verdaderos negativos, calcula la tasa de los casos negativos que el modelo ha clasificado correctamente [18]. Se define como:

Ecuación 4

$$Especificidad = \frac{VN}{VN + FP}$$

Donde:

$VN = Verdaderos\ negativos$

$FP = Falsos\ positivos$

4.2. Bibliotecas o Librerías

4.2.1. OpenCV

OpenCV es una biblioteca de enlaces diseñada para resolver problemas de visión por computadora, disponible en varios lenguajes de programación, entre ellos Python. Dentro del entorno de desarrollo con Python, OpenCV simplifica la codificación de ideas orientadas a la visión por computadora en menos líneas de código sin perder la legibilidad, permitiendo así la creación de código escalable y sostenible.

Dentro de las librerías que OpenCV utiliza, destaca Numpy, que es una biblioteca altamente optimizada para operaciones numéricas con una sintaxis de estilo MATLAB. Todas las estructuras de matrices de OpenCV se convierten desde y hacia matrices Numpy, facilitando la integración con otras bibliotecas que usan Numpy, como SciPy y Matplotlib [49]. El uso de cada una de las funciones descritas a continuación, se pueden verificar en el apartado 1 del Anexo 4, disponible en el CD número 2.

En cuanto al procesamiento de imágenes, esta librería utiliza las siguientes funciones:

- **Imread:** Esta función permite cargar una o más imágenes, para lo cual, solicita como primer argumento la ruta de la(s) imagen(es) y como segundo argumento, aunque opcional, el formato de la imagen para importar una sola imagen usando la función `imread()`. En cambio, si se desea trabajar con más de una imagen,

es recomendable utilizar un array, en el cual se almacenarán todas las direcciones de las imágenes que se necesitan leer. Este array, será recorrido posición por posición utilizando la función `imread()`.

Independientemente de la cantidad de imágenes que se deseen utilizar como entrada, el valor de retorno será una matriz como la que se muestra en la **Fig. 11**. En caso de que la ruta asignada no contenga imágenes o no sea posible resolver la ruta, el valor de retorno será una matriz vacía, en el mejor de los casos, el entorno de desarrollo mostrará un mensaje de error indicando parte del problema.

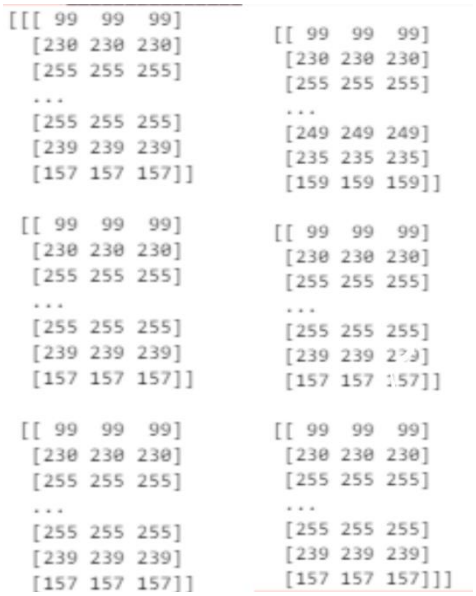


Fig. 11 *Valor de salida para la función imread(), con varias imágenes como entrada.*

Para referencia, se ha utilizado el ejemplo en la cual se lee varias imágenes, debido a que el modelo a desarrollar requiere no una, sino miles de imágenes, lo que se evidencia en la **Fig. 12**, donde se muestra el progreso y estado de la carga de las imágenes:

```
1 import cv2
2 import os
3 from tqdm import tqdm
4
5 # Directorio de las imágenes
6 imagePaths = []
7  for dirname, _, filenames in os.walk(:\Users\jr-98\Documents\
DataCovid19imgX\RAHMAN\COVID-19_Radiography_Dataset10'):
8    for filename in filenames:
9        if (filename[-3:] == 'png'):
```

```
            imagePaths.append(os.path.join(dirname, filename))
        else:
            print('Error con la lectura de archivos')

# Delimita la dimensión de las imágenes
imgSize = 224

# Transformación y segmentación de imágenes
X = []
Y = []
hmap = {'NORMAL': 'Normal', 'COVID': 'Covid-19'}
for imagePath in tqdm(imagePaths):
    label = imagePath.split(os.path.sep)[-2]
    image = cv2.imread(imagePath)
    image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    image = cv2.resize(image, (imgSize, imgSize))

    X.append(image)
    Y.append(hmap[label])

# Comprobación por lotes de las imágenes cargadas
print('Covid-19:', Y.count('Covid-19'))
print('Normal:', Y.count('Normal'))
```

```
100
%|██████████████████████████████████████████████████████████████████████████████|
6000/6000 [00:08<00:00, 683.77it/s]
Covid-19: 3000
Normal: 3000
```

Fig. 12 *Estado de carga de las imágenes que serán usadas en la fase de entrenamiento y test del modelo. El total de imágenes utilizadas es de 6000 distribuidas entre 3000 por cada clase.*

- **cvtColor:** Esta función se encarga de convertir una imagen de espacio de color a otro color, debido a los diferentes formatos existentes de imágenes y las inconsistencias que estos presentarían si no se hiciese un tratamiento previo. Como ejemplo [50] especifica que, en caso de una transformación de un espacio de color RGB, el orden de los canales debe especificarse explícitamente (RGB o BGR), considerando que el RGB es el formato de color por defecto con el que OpenCv trabaja, aunque en realidad la configuración de los canales es BGR, ya

que el orden de los bits se encuentra invertido, es decir si disponemos de una imagen de 24 bits, los primeros 8 bits no corresponden al rojo sino al azul, y los últimos 8 bits, no corresponden al azul sino al rojo (el orden de bits correspondientes al color verde permanecen invariables en ambos casos).

Para utilizar la conversión de BGR a RGB, se requiere dos parámetros indispensables, el primero, un array de entrada que se obtiene con la función imread(), detallado anteriormente, y el segundo, un array de salida, que está compuesto por el valor de retorno de la función de conversión, las cuales se encuentra disponible en la documentación oficial de OpenCv [50], para este caso en particular se empleará la corrección del orden de los canales BGR a RGB, tal como se muestra en la línea 23 de la sección de código anterior.

- **resize:** Esta función permite transformar imágenes de cierta escala, a una escala específica. Esta función es especialmente útil cuando se procesa una cantidad considerable de imágenes, las cuales pueden variar tanto en escala como en resolución, por lo que se requiere especificar de forma explícita su valor, de tal forma que el conjunto de imágenes se acoplen a dicha escala, es decir, el tamaño de la imagen se puede especificar manualmente o se puede especificar el factor de escala [51]. Para ello, se utilizan diferentes métodos de interpolación, siendo los siguientes métodos los más utilizados, `cv.INTER_AREA` para encoger y `cv.INTER_LINEAR` para hacer zoom. Por defecto, el método de interpolación `cv.INTER_LINEAR` se usa para todos los propósitos de cambio de tamaño.

4.2.2. Tensorflow

Es un framework de código abierto, desarrollado inicialmente por Google Brain Team, para el desarrollo de algoritmos de Machine Learning (ML, aprendizaje automático) y cálculos de datos descentralizados, ofreciendo una oportunidad para la investigación abierta y experimental de algoritmos que usados para descifrar y detectar patrones y las correlaciones entre ellos, lo cual la convierte en una de las plataformas open source de extremo a extremo más utilizadas e importantes para el desarrollo de algoritmos que empleen ML [52]. Este framework, cuenta con un conjunto integral y flexible de herramientas, bibliotecas y recursos que permite a los investigadores y desarrolladores compilar, implementar e innovar en el aprendizaje automático [52], [53]. En cuanto a disponibilidad, TensorFlow (TF) se encuentra disponible para ambientes locales. Su uso y despliegue se encuentra en el README[2] del proyecto.

[2] Repositorio general

En cuanto a los ambientes de trabajos locales, además de instalar TF, se requiere de otras herramientas y librerías adicionales, de tal forma que sea posible obtener el máximo rendimiento de los equipos en la fase de entrenamiento de los modelos (esta librerías y drivers adicionales se describen más adelante). Así mismo, los modelos se pueden construir en lenguajes de programación como java, C, C++ y Python, debido a la disposición de las librerías que permiten la aceleración del procesamiento mediante la GPU (CPU suele ser el método de procesamiento por defecto, pero la GPU es mucho más veloz).

Una vez establecidos las principales características de TF, es necesario detallar su funcionamiento, la página principal de Google Colab [54], explica mediante un ejemplo su funcionamiento general, allí se especifica que TF es un grafo, o al menos que sus operaciones se rigen a la estructura de grafos, con la particularidad que TF separa la definición de las operaciones de su ejecución, es decir, primero define la estructura del grafo como tal, para luego ejecutarse mediante un modelo de programación paralela llamado DataFlow, así pues, la forma que los nodos representan unidades de computación y las aristas representan los datos consumidos o producidos, conocidos también como tensores[3]. Como ejemplo, en la **Fig. 13**, se representa la suma de dos tensores de 0 dimensiones, con la operación `tf.add`, misma que recibe como entrada 0 o más tensores, los cuales, mediante un conjunto de operaciones (en este caso la adición) constituye un nuevo tensor.

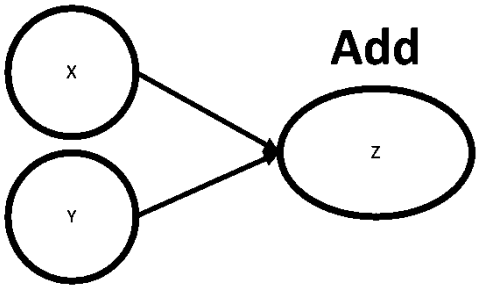


Fig. 13 *Representación gráfica del proceso de adición de dos tensores tf.add, que suma dos vectores de 0 dimensiones.*

El número de tensores y su estructura varía según la complejidad del algoritmo y la cantidad de operaciones realizadas a los datos entrantes, partiendo del ejemplo anterior, la suma de dos tensores converge en la creación de un nuevo tensor, que en teoría contiene el resultado de la suma de los dos tensores de entrada, pero si la cantidad de

[3] Un tensor es un matriz de n dimensiones, conocido también como rank donde un tensor de 0 dimensiones es un escalar y un tensor de 1 dimensión es un vector.[54]

operadores aumenta, por ejemplo, en un grafo que represente las dependencias entre las operaciones, como se muestra en la **Fig. 14**, en la cual se ha construido un grafo que resuelve la siguiente operación $(a * b)\hat{}(c + d)$, donde se involucran tres operaciones en concreto, mismas que poseen una jerarquía que el grafo debe respetar, de tal forma que se obtenga el resultado equivalente a esa operación.

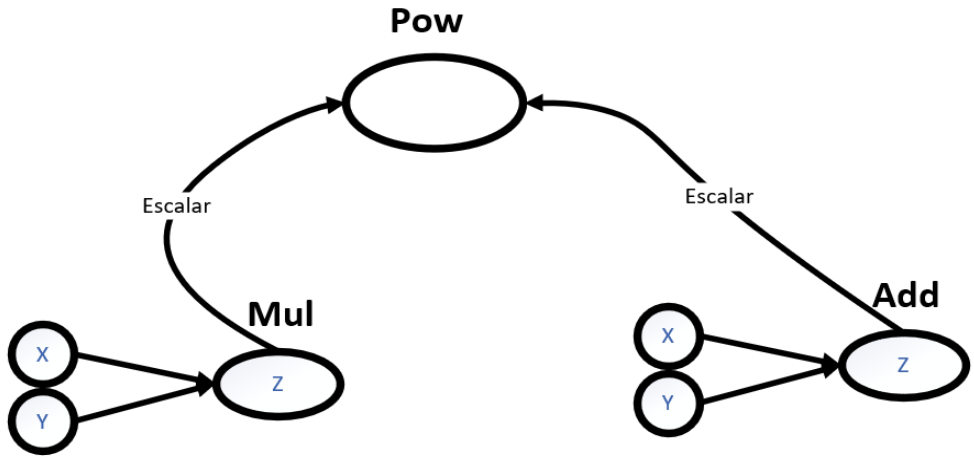


Fig. 14 *Representación gráfica de las dependencias de las operaciones con un grafo que se construye para resolver la operación (a*b)^(c+d).*

Al inicio, se menciona que TF necesita de librerías adicionales para operar correctamente, siendo las más importantes, tanto para ambientes locales como ambientes orientados a la nube, NumPy [54], la cual es una librería especializada en el cálculo numérico y el análisis de datos, especialmente para un gran volumen de ellos, como lo es trabajar con matrices y arreglos los cuáles serán las entradas de los modelos desarrollados con TF. Estas matrices se calculan a partir de cientos y miles de imágenes, como es el caso de las aplicaciones más destacadas de TF, como el mejoramiento de fotografías en smartphones, procesamiento de imágenes y asistencia al diagnóstico médico [53], [54], (parte principal del presente TT).

Como se manifestó anteriormente, una de las aplicaciones de TF es la creación de algoritmos para la ayudar al diagnóstico de enfermedades mediante el análisis de imágenes médicas, como pueden ser las radiografías o las tomografías computarizadas (TC), además [53] menciona que uno de los sectores donde el aprendizaje automático o mejor conocido como ML ha tenido aplicaciones prometedoras, es la medicina, puesto que reduce significativamente los tiempos de análisis de las imágenes médicas y lo más importante, brinda una segunda opinión médica con la cual el profesional de la salud puede contrastar la suya.

4.2.3. Keras

Es una API[4] de alto nivel incluida dentro de la librería de TensorFlow (TF), que permite desarrollar y entrenar modelos de aprendizaje profundo de forma rápida y flexible, otorgando ciertas ventajas como información clara sobre la causa de los errores, la facilidad de creación y modificación de bloques configurables y dinámicos entre sí mediante capas, funciones y modelos [55]. Dentro de esta librería existen muchas funciones de las cuales se resaltan las siguientes:

- **keras.preprocessing:** se utiliza para el procesamiento de los datos ubicados en el disco (local o servicio de almacenamiento en la nube) a un objeto que pueda ser utilizado por el modelo para su entrenamiento. En cuanto a las imágenes, se recomienda la utilizar keras.preprocessing.image, que recibe como argumentos, el directorio donde se almacenan las imágenes, las etiquetas, la escala de colores, el tamaño de los lotes, la dimensión de las imágenes, etc. Como salida, produce tensores con imágenes codificadas en función de las etiquetas que se le asignen para cada uno de los lotes.
- **keras.models**: se utiliza para crear un modelo para lo cual se debe especificar sus entradas y salidas en las capas de las gráficas.
- **keras.applications**: permite crear instancias de un modelo en específico el cual se almacenará en ~/.keras/models/. Tras haber creado dicha instancia, los modelos se construirán tomando en cuenta formato de las imágenes predefinidos con anterioridad [56]. Algunos de los modelos de ML más importantes que se encuentran disponibles en esta librería son:
    - Xception
    - VGG16
    - VGG19
    - ResNet50-101-152
    - ResNet50V2-101V2-152V2
    - Inception3
    - InceptionResNetV2
    - MobileNet

[4] *Application Programming Interface* (API) o Interfaz de Programación de Aplicaciones, es un conjunto de herramientas y protocolos construidos para desarrollar he integrar aplicaciones y se puedan ejecutar en dispositivos diferentes que funcionen sobre el mismo sistema operativo sin la necesidad de saber cómo estan implementados [102].

- **keras.layers**: permite trazar una flecha desde las “entradas” (inputs) a la capa que se haya creado.
- **keras.optimizer**: Es utilizado como uno de los argumentos más importantes para compilar un modelo de Keras. Específicamente, su función consiste en crear una instancia de un optimizador (*Adam*, es el optimizador más popular dentro de keras) antes de compilar el modelo. En caso de no existir parámetros establecidos, se utilizará los parámetros predeterminados del optimizador [56].

4.2.4. Sklearn

Es una librería open source empleada en modelos de regresión lineal, clustering, reducción dimensional, modelos de selección y procesamiento de datos. De las aplicaciones mencionadas anteriormente, es de nuestro interés aquellas orientadas al procesamiento de datos, ya que, al trabajar con imágenes como datos de entrada, se asume que estas se transformarán en matrices y vectores que serán usadas por el modelo, por lo que se requiere un procesamiento previo de los datos, lo que se consigue con la librería `sklearn.preprocessing`, debido a las funciones y transformadores que posee. Es decir, los algoritmos de aprendizaje automático se benefician de la estandarización del conjunto de datos de tal forma que, si algunos valores atípicos están presentes en el conjunto, los transformadores o escaladores acoplarán estas anomalías. Cabe recalcar, que el comportamiento de los diferentes escaladores, transformadores y normalizadores en un conjunto de datos que contiene valores atípicos marginales se resaltan en comparar el efecto de diferentes escaladores en datos con valores atípicos [57], siendo la estandarización de conjuntos de datos es un requisito común para muchos estimadores de aprendizaje automático, y justamente, es en este aspecto que sklearn simplifica el trabajo.

4.3. Arquitecturas de redes convolucionales

4.3.1. VGG-16

Es un modelo de red neuronal convolucional con 16 capas de convolución de 3x3, el cual alcanza niveles de precisión superiores al 90% en el mejor de los casos, por lo que es considerado como uno de los modelos con mejor arquitectura en cuanto al procesamiento de imágenes se refiere. Este modelo fue propuesto por K. Simonyan y A. Zisserman, en el artículo "Redes convolucionales muy profundas para el reconocimiento de imágenes a gran escala" [58] y también se menciona en los trabajos relacionados TR11, TR16, TR28 y TR29, los cuales utilizan este modelo para el análisis

de radiografías y tomografías computarizadas, con un nivel de precisión y rendimiento óptimos, considerando su bajo nivel de complejidad computacional. En la **Fig. 15** se resumen sus características más relevantes:

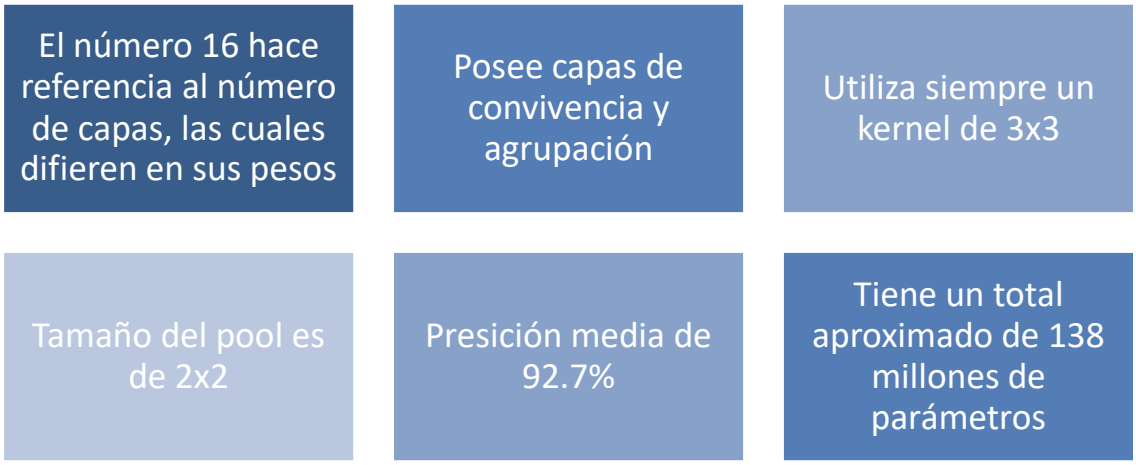


Fig. 15 *Principales características de VGG16 como modelo de Red Neuronal Convolucional (CNN)*

En cuanto a su arquitectura, VGG16 posee 16 capas convolucionales, con un kernel de 3x3 y el tamaño del pool es de 2x2, para cada una de las capas que conforman el modelo (conceptos en la sección Convolucional Neural Networks). Por ejemplo, en la **Fig. 16**, se aprecia el esquema de la arquitectura del modelo VGG16, que recibe como entrada una radiografía (imagen con formato .png) con una dimensión de 224x224, a la cual se aplica un filtro de 3x3, razón por la cual se especifican los siguientes valores 224x224x3, seguida de dos capas convolucionales, cada una con un tamaño de 224x224x64, como resultado de la primera convolución (VGG16), se obtiene una capa de agrupación que

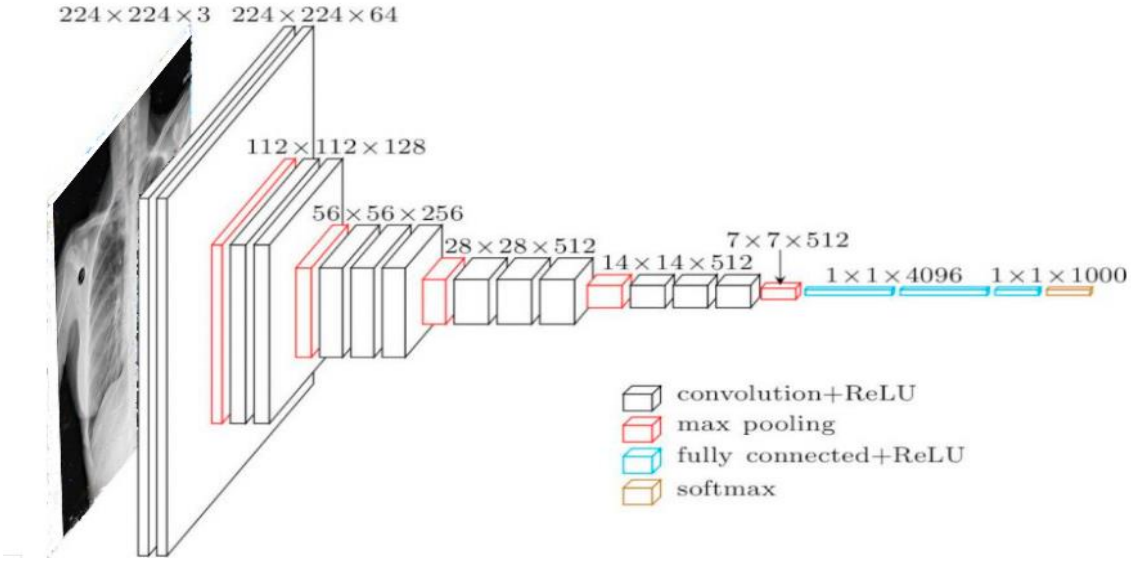


Fig. 16 *Esquema de la arquitectura del modelo VGG16*

reduce las dimensiones de la imagen a 112x112x64, que se encuentra resaltada de color rojo, esta será la entrada para las siguientes capas convolucionales del modelo.

El proceso descrito se repite con dos capas de convolución de 112x112x128, lo que da como resultado una nueva capa de agrupación que reduce las dimensiones de la entrada a 56x56x128 (segundo bloque de color rojo de la **Fig. 16**). A continuación, la última capa de agrupación es tomada como entrada en las siguientes tres capas convolucionales, cada una con un tamaño de 56x56x256. El resultado, nuevamente es una capa de agrupamiento que reduce las dimensiones de la imagen a 56x56x128 (tercer bloque de color rojo de la **Fig. 16**). Las tres siguientes capas de convolución cuyas dimensiones son 28x28x512, reducen nuevamente la imagen que recibieron como entrada de las capas de convolución anterior, en una nueva capa de agrupación de dimensiones 14x14x512 (cuarto bloque de color rojo de la **Fig. 16**).

Por último, las tres capas de conv512 con dimensiones 14x14x512, arrojará una capa de agrupación con 7x7x512 que, a su vez, es la entrada para las capas densas que se encuentran conectadas con cada uno de los 4090 nodos, dando como resultado una capa con 1000 nodos de 1x1. En la **Fig. 17**, se muestra de forma gráfica una síntesis del proceso mencionado anteriormente.

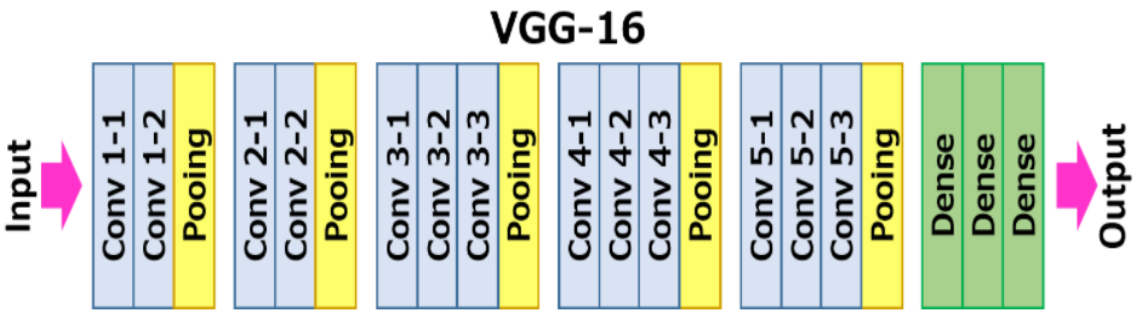


Fig. 17 *Estructura de VGG16*
Fuente: [59]

En cuanto a su implementación, existen varios métodos disponibles actualmente, en la sección de librerías Bibliotecas o Librerías, específicamente con la librería Keras, se especifica su implementación y algunos de sus hiperparámetros[5] que se deben configurar, ya que de ello dependerá el rendimiento del modelo. Es necesario recalcar que este no es el único método, pero si es uno de los más utilizados en los artículos relacionados TR12, TR18, TR38 y TR39.

[5] Los hiperparámetros son parámetros ajustables que permiten controlar el proceso de entrenamiento de un modelo, como el número de capas ocultas, el tamaño de los lotes, la tasa de aprendizaje etc.[103]

4.3.2. ResNet50

ResNet50 es un tipo de CNN compuesta por 50 capas de profundidad, cuyo nombre proviene de ResNet, abreviatura de Residual Network o red residual y el número de capas que posee. Este modelo ha sido implementado y testeado en los trabajos relacionados TR02, TR03, TR18, TR22, TR23 y TR40, arrojando muy buenos resultados en cuanto a identificación de patrones que ayuden al diagnóstico de Covid-19.

Este modelo, utiliza una técnica conocida como “Aprendizaje residual”, que según [60], es un tipo de aprendizaje que intenta obtener algunas características empleando la parte residual de las capas, entendiéndose como residuo, la resta de las características aprendidas de la entrada de esa capa. Estas entradas, a su vez, son las salidas obtenidas de las neuronas en capas posteriores, mismas que conforman la pirámide “cerebral” de la red, y es aquí donde las redes neuronales residuales o ResNet, permiten crear conexiones de salto, que no son más que atajos entre las diferentes capas que conforman la red. De forma más técnica y detallada, [60] considera $H(x)$ como la representación matemática del mapeo de la red, de tal forma que sea posible encajar n capas apiladas, siendo x la denotación de las entradas a la primera de las capas apiladas, lo cual permitió plantear la hipótesis de que es posible aproximar las funciones residuales de las capas, es decir $H(x) - x$, suponiendo que la entrada y salida son de la misma dimensión (se ha empleado la parte residual de las capas). Entonces, partiendo de la hipótesis anterior, y tomando como referencia la **Fig. 18**, para representación de una ResNet, se puede decir que, en lugar de que la entrada se aproxime a $H(x)$, como normalmente lo hacen las CNN lineales y jerárquicas, las ResNet permiten que las capas se aproximen a una función residual $F(x) = H(x) - x$, donde x, puede ser cualquiera de los valores de x_1, x_2, x_n, por lo que son los posibles saltos o atajos para el proceso de aprendizaje. Este proceso ha demostrado ser más fácil que entrenar redes neuronales convolucionales profundas, resolviendo el problema de la degradación de la precisión, el cual sugiere que las soluciones pueden presentar dificultades para aproximar las asignaciones de identidad por múltiples capas no lineales. Con la formulación del aprendizaje residual, los mapeos de identidad son óptimos, los solucionadores pueden simplemente impulsar los pesos de las múltiples capas no lineales hacia cero para abordar las asignaciones de identidad.

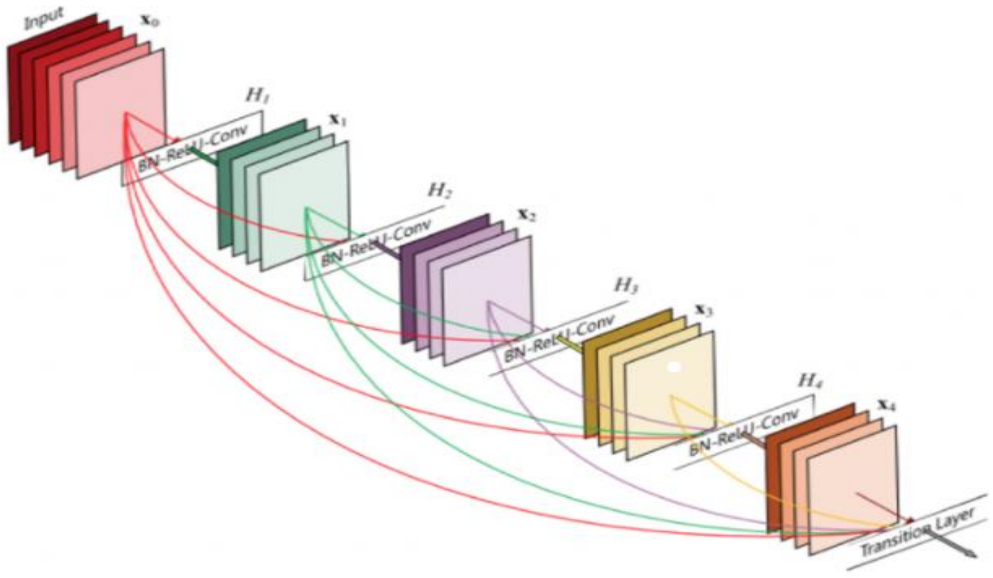


Fig. 18 *Representación gráfica de la estructura de un Red Neuronal Residual o mejor conocida como ResNet. Las capas de color más saturado representan las entradas de las capas contiguas y las capas de colores menos saturados, representan las salidas de dichas capas. Las líneas que interconectan las capas son atajos a capas posteriores a su origen lo que permite a la red "saltar" sobre ciertas capas para llegar a capas muy por debajo de su jerarquía.*

Para una mejor descripción de la arquitectura de la ResNet50, es necesario tomar como referencia la **Fig. 19**, pues se necesita realizar una comparación con otro tipo de modelo de red para comprender su funcionamiento e infraestructura. Como punto de comparación, la arquitectura que se encuentra en la parte izquierda de la **Fig. 19**, es una red basada en la infraestructura básica de las redes neuronales convolucionales VGG, donde las capas de convolución poseen una dimensión de 3x3 (como se explicó en el apartado 4.3.1), cuyo funcionamiento se puede sintetizar en dos aspectos fundamentales, el primero, para la misma salida, el tamaño del mapa de características y las capas tienen el mismo número de filtros. Como segundo aspecto, el tamaño del mapa de características se reduce a la mitad, a la vez que el número de filtros se duplica para preservar la flexibilidad de las capas. En cuanto a las ResNet (**Fig. 19**, derecha) se observa claramente su característica distintiva, la red de conexiones de acceso directo, las cuales son utilizados cuando las dimensiones del objeto de entrada y de salida son las mismas (conexión de atajo con línea continua **Fig. 19**, derecha) y cuando aumentan las dimensiones (conexión de atajo con líneas entrecortadas **Fig. 19**, derecha). Para cada uno de estos casos, se considera que el acceso directo realiza un mapeo de identidad para aumentar la dimensión de la entrada (no se introducen parámetros extra) ademas se realiza una proyección de la ecuación $F(x) = H(x) - x$, con el fin de hacer coincidir las dimensiones mediante un proceso de convolución. En el caso del ejemplo

de la **Fig. 19** nótese que, cuando los atajos van a través de mapas de características de 2x2 y un salto de 2 capas.

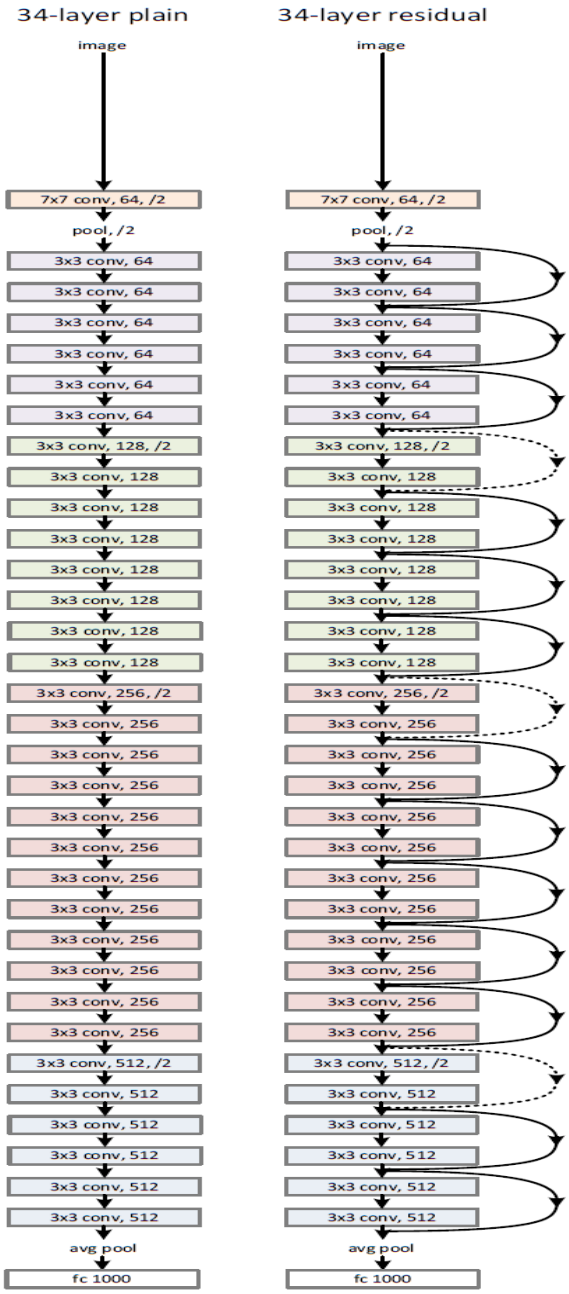


Fig. 19 *Estructura de arquitecturas de redes neuronales convolucionales. En la* ***parte izquierda****, se presenta el modelo de red simple con 34 capas, con un total de 3 600 millones de parámetros. En la* ***parte derecha****, se presenta una ResNet con 34 capas, con 3 600 millones de parámetros, cuya particularidad son sus líneas entrecortadas que representan los "atajos" lo cual aumenta las dimensiones de la red.*
Fuente*:* [60]

Por último, la implementación de ResNet50 se puede realizar (al igual que el modelo anterior) con el API de keras, junto con TensorFlow (se describe en las secciones posteriores) tanto en ambientes locales y en aquellos servicios orientados a la nube como Google Colab. Otra opción para implementar este modelo, es usando Matlab, que también dispone de paquetes que ayudan a la implementación de este modelo [61].
En este caso se utilizará el API de Keras, el cual permitirá utilizar el modelo mediante las siguientes líneas de comando:

```
tf.keras.applications.ResNet50(
    include_top=True,
    weights="imagenet",
    input_tensor=None,
    input_shape=None,
    pooling=None,
    classes=1000,
    **kwargs
)
```

Cada implementación de Keras requiere diferentes entradas dependiendo del modelo, en este caso, los parámetros solicitados son los siguientes [62]:

- **include_top**: indica si se debe incluir la capa completamente conectada en la parte superior de la red.
- **weights**: permite utilizar tanto `none` (inicialización aleatoria), como 'imagenet' (entrenamiento previo en ImageNet) o la ruta al archivo de pesos que se cargará.
- **input_tensor**: permite utilizar tensores de Keras opcionales (es decir, salida de `layers.Input()`) para usar como entrada de imagen para el modelo.
- **input_shape**: es una tupla opcional, que solo se especificará si la función `include_topes` tiene un valor de `False` (de lo contrario, la forma de entrada debe ser (224, 224, 3) con '`channels_last`' como formato de datos) o (3, 224, 224) con '`channels_first`' como formato de datos. Debe tener exactamente 3 canales de entrada, y el ancho y la altura no deben ser menores que 32.
- **pooling**: el modo de pooling es opcional para la extracción de características cuando `include_topes = false.` Otros valores que se pueden utilizar son:
    - **`none`**, significa que la salida del modelo será la salida del tensor 4D del último bloque convolucional.
    - **`avg`**, significa que la agrupación de promedios globales se aplicará a la salida del último bloque convolucional, por lo tanto, la salida del modelo

será un tensor 2D.

- **max,** significa que se aplicará la agrupación máxima global.

- **classes**: es el número opcional de clases para clasificar las imágenes, solo se especificará si `include_top` es verdadero y si `weightsse` no especifica ningún argumento.

4.3.3. InceptionV3

Es un modelo implementado para el reconocimiento de imágenes utilizado en los trabajos relacionados TR2, TR11, TR14, TR36 y TR38, con un nivel de precisión superior al 80%, como promedio entre los diferentes TR en los que se ha implementado. Este modelo de red neuronal convolucional está conformado por 48 capas de profundidad, distribuidas en bloques de construcción simétricos y asimétricos que incluyen convoluciones, agrupación promedio, agrupación máxima, concats, abandonos y capas completamente conectadas [63].

Su arquitectura se basa en 5 etapas las cuales se describen a continuación:

- **Convoluciones factorizadas:** Reduce el número de parámetros que intervienen en la red con el fin de reducir los requerimientos computacionales, aumentando así la eficiencia de la red.
- **Convoluciones más pequeñas:** reduce las dimensiones de las capas convolucionales, lo que desemboca en un proceso de entrenamiento mucho más rápido. En la **Fig. 20**, se propone un ejemplo, suponiendo que se trata de unas

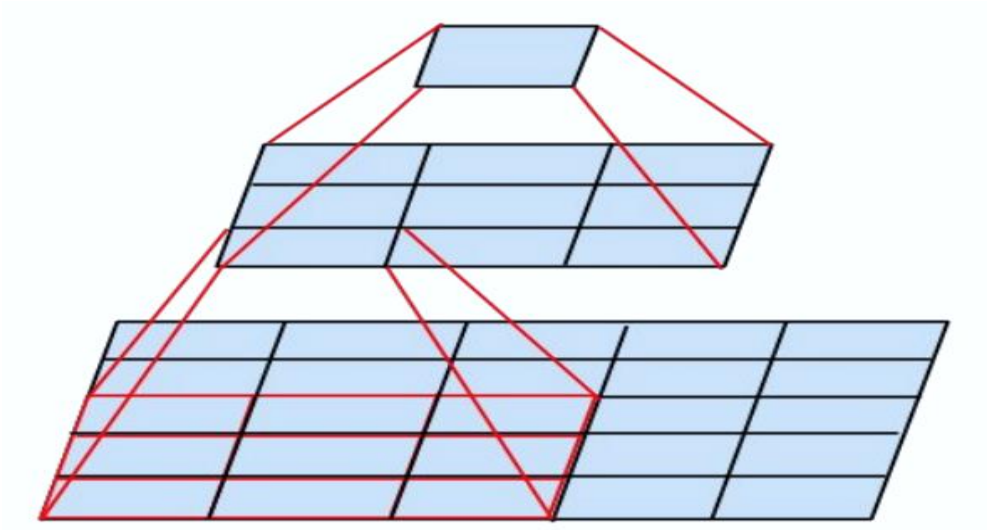


Fig. 20 *Proceso de reducción de dimensión de una capa convolucional. En la parte central se encuentra una capa convolucional de 3x3 y en la parte inferior una capa complementaria conectada. Estas dos últimas capas, al tener la misma dimensión pueden compartir sus pesos entre sí, de tal forma que los cálculos se reducen.*

capas de 5x5, con 25 parámetros en total, a la cual se aplica 2 filtros de 3x3, mismos que remplazaran los filtros convolucionales de 5x5, donde el número de parámetros también se reduce a 18 en lugar de los 25 anteriores.

- **Convoluciones asimétricas:** tomando como referencia el proceso de reducción de capas de 5x5 con la aplicación de filtros de 3x3 del ejemplo de la **Fig. 17**, es posible reducir el costo de cada transacción alterando la simetría de las cargas con un filtro de convolución aún más "pequeño" de 1x3, seguida por una convolución de 3x1, por lo que la capa de convolución de 3x3 se reducirá a una capa de convolución de 2x2, tal como se muestra la **Fig. 21**.

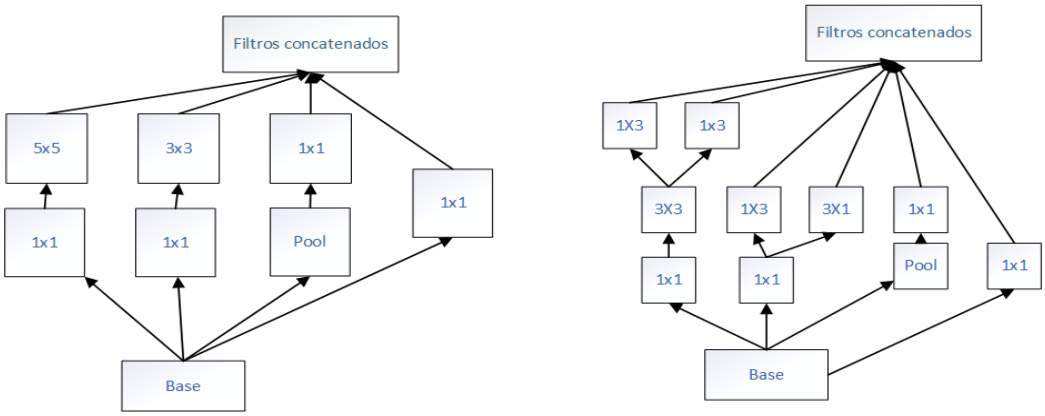


Fig. 21 *Diagramas que representan la convolución simétrica (diagrama de la parte izquierda) y la convolución asimétrica (diagrama de la parte derecha).*

- **Clasificador auxiliar:** se trata de una pequeña red neuronal convolucional que actúa como un regulador de pérdida al ser insertada entre las capas predefinidas durante la fase de entrenamiento, de tal forma que la pérdida incurrida es sumada a la pérdida de la red principal.
- **Reducción del tamaño de la cuadrícula:** consiste en aplicar operación de agrupación en las cuadrículas de las capas de entrada con el fin de reducir los cuellos de botella que se pueden generar por los procesos computacionales, en la **Fig. 22** se muestra el proceso de reducción de las cuadriculas, partiendo de una entrada de 35x35x320.

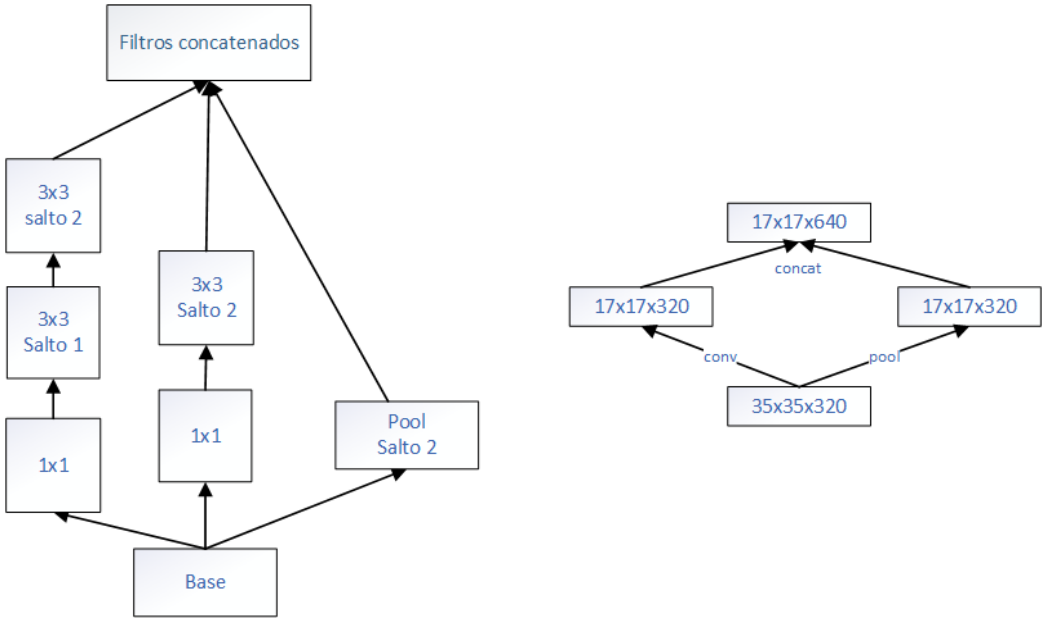


Fig. 22 *Diagrama del proceso de reducción del tamaño de las cuadriculas. En la parte izquierda, se presenta un esquema general del proceso de reducción de las cuadriculas y en la parte derecha, se muestra el esquema de las operaciones de agrupación.*

Por último, la agrupación de todos los conceptos mencionados conforma la totalidad del esquema general de la arquitectura del modelo InceptionV3, el cual se aprecia en la **Fig. 23**

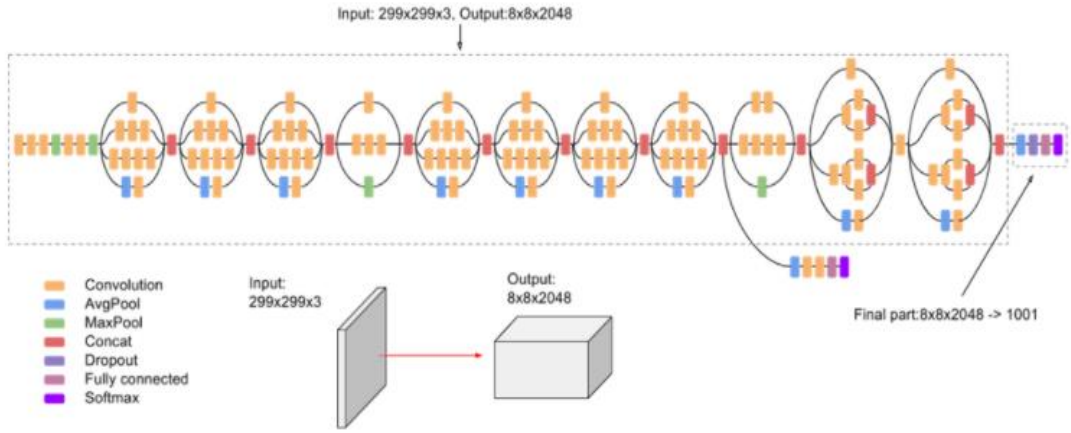


Fig. 23 *Diagrama de alto nivel de la estructura del modelo InceptionV3*

En cuanto a su implementación, este modelo está disponible en Matlab, para lo cual es necesario descargar e instalar el modelo de caja de herramientas de aprendizaje profundo para el paquete de soporte de red Inception-v3 [64], también se encuentra disponible (al igual que los casos anteriores) en el API de Keras, y para ambientes de desarrollo en la nube y locales.

4.4. Trabajos relacionados

De acuerdo a la revisión bibliográfica realizada para la constitución del anteproyecto, véase el Anexo 1 (el desarrollo completo del anteproyecto se encuentra disponible en el CD número 1), donde se detalla el proceso llevado a cabo para el desarrollo de la RSL. Aunque, el presente TT no tiene por objetivo realizar una RSL como tal, esta revisión ha permitido analizar el problema y sus posibles soluciones, siendo los trabajos o estudios relacionados (TR), el sustento a la solución del problema. Los trabajos relacionados resultado de la RSL, se detallan en la **TABLA I**.

TABLA I *Trabajos relacionados*
Código*: Código distintivo de cada trabajo relacionado*

Código	Título	Referencia
TR01	A deep learning framework to detect Covid-19 disease via chest X-ray and CT scan images.	[15]
TR02	A Deep Learning Prognosis Model Help Alert for COVID-19 Patients at High-Riskof Death: A Multi-Center Study	[40]
TR03	A multi-task pipeline with specialized streams for classification and segmentationof infection manifestations in COVID-19 scans	[65]
TR04	A Novel AI-enabled Framework to Diagnose Coronavirus COVID-19 using Smartphone Embedded Sensors: Design Study	[6]
TR05	A Novel Medical Diagnosis model for COVID-19 infection detection based on Deep Features and Bayesian Optimization	[66]
TR06	A novel perceptual two-layer image fusion using deep learning for imbalanced COVID-19 dataset	[23]
TR07	A Radiomics Signature to Quantitatively Analyze COVID-19-Infected Pulmonary Lesions	[67]
TR08	A Survey on how computer vision can response to urgent need to contribute in COVID-19 pandemics	[68]

TR09	Accelerating detection of lung pathologies with explainable ultrasound image analysis	[10]
TR10	An Attention Mechanism with Multiple Knowledge Sources for COVID-19 Detection from CT Images	[69]
TR11	Approaches based on artificial intelligence and the internet of intelligent things to prevent the spread of COVID-19: Scoping review	[70]
TR12	Artificial Intelligence and COVID-19: Deep Learning Approaches for Diagnosis and Treatment	[71]
TR13	Artificial Intelligence applied to chest X-Ray images for the automatic detection of COVID-19. A thoughtful evaluation approach	[19]
TR14	Artificial intelligence technology for diagnosing COVID-19 cases: A review of substantial issues	[72]
TR15	Benchmarking Methodology for Selection of Optimal COVID-19 Diagnostic Model Based on Entropy and OPSIS Methods	[73]
TR16	CCBlock: An Effective Use of Deep Learning for Automatic Diagnosis of COVID-19 Using X-Ray Images	[5]
TR17	Chest X-ray image analysis and classification for COVID-19 pneumonia detection using deep CNN	[24]
TR18	Computer aid screening of COVID-19 using X-ray and CT scan images: An inner comparison	[25]
TR19	COVID CT-Net: Predicting Covid-19 from chest CT images using attentional convolutional network	[74]
TR20	COVID-19 detection based on deep learning and artificial bee colony	[3]
TR21	Covid-19 Detection by Optimizing Deep Residual Features with Improved Clustering-Based Golden Ratio Optimizer	[18]
TR22	COVID-19 detection on chest X-Ray and CT Scan images using multi-image augmented deep learning model	[14]

TR23	COVID-19 Disease Diagnosis using Smart Deep Learning Techniques	[75]
TR24	Covid19 Identification from Chest X-ray Images Using Machine Learning Classifiers with GLCM Features	[16]
TR25	CT Quantification and Machine-learning Models for Assessment of Disease Severity and Prognosis of COVID-19 Patients	[76]
TR26	Deep learning analysis provides accurate COVID-19 diagnosis on chest computed tomography	[77]
TR27	Deep Learning in Detection and Diagnosis of Covid-19 using Radiology Modalities: A Systematic Review	[78]
TR28	Detection of coronavirus disease (COVID-19) from X-ray images using deep convolutional neural networks	[26]
TR29	Diagnosis of Coronavirus Disease 2019 (COVID-19) with Structured Latent Multi-View Representation Learning	[79]
TR30	Expert system for early diagnosis of covid-19	[9]
TR31	Heg.IA: an intelligent system to support diagnosis of Covid-19 based on blood tests	[80]
TR32	Inf-Net: Automatic COVID-19 Lung Infection Segmentation from CT Images	[80]
TR33	Intelligent Internet of Things and Advanced Machine Learning Techniques for COVID-19	[12]
TR34	Longitudinal proteomic profiling of high-risk patients with COVID-19 reveals markers of severity and predictors of fatal disease	[81]
TR35	Machine learning-based prediction of COVID-19 diagnosis based on symptoms	[11]

TR36	Model expert system for diagnosis of COVID-19 using naïve bayes classifier	[82]
TR37	Multi-task contrastive learning for automatic CT and X-ray diagnosis of COVID-19	[83]
TR38	Severity Detection for the Coronavirus Disease 2019 (COVID-19) Patients Using a Machine Learning Model Based on the Blood and Urine Tests	[17]
TR39	The value of artificial intelligence and imaging diagnosis in the fight against COVID-19	[84]
TR40	Transfer learning to detect COVID-19 automatically from X-ray images, using convolutional neural networks	[22]
TR41	Using machine learning of clinical data to diagnose COVID-19: A systematic review and meta-analysis	[85]

Tal como se mencionó anteriormente, cada uno de los TR de la tabla anterior, fue analizado para identificar algún aporte significativo, el cual permitiese obtener información relacionada con los modelos de IA utilizados para el diagnóstico de Covid-19. Así mismo, ha permitido extraer información adicional como los requerimientos del modelo, su nivel de precisión, su factibilidad de implementación y el banco de datos utilizado. Finalmente, la síntesis de datos permitió dar contestación a las preguntas de investigación planteadas en la **TABLA XXII**, de la RSL anexada.

a) **¿Cuáles son los enfoques existentes basados en Inteligencia Artificial (IA) o Machine Learning (ML), para la detección de Covid-19 (SARS-CoV-2), gripe o neumonía pulmonar?**

Según los artículos TR02, TR05, TR13, TR17, TR20, TR21, TR22, TR24, TR25, TR26, TR27, TR28, TR29, TR33, TR37 y TR40, proponen la creación de herramientas basadas en diferentes enfoques para el diagnóstico rápido y eficiente de Covid-19. Estos enfoques se basan en modelos que van desde el análisis de tomografías computarizadas (TC) e imágenes de rayos X TR01, TR05, TR07, TR10, TR12, TR13, TR20 TR24 y TR41, el análisis de sangre, propuestos en TR21 y TR38, el análisis de voz TR13, TR24 y TR10, incluso TR30, recomienda la implementación de un modelo para el diagnóstico de Covid-19, a partir de los síntomas que el paciente ingresa en el sistema. Otros artículos como el TR03 y TR26, proponen la construcción de un

novedoso sistema que unifica la IA y el internet de las cosas (IoT), con el fin de proveer de herramientas de diagnóstico asequibles, tanto para profesionales de la salud y para quienes no lo son.

Por lo tanto, ¿Cuál es el enfoque de aplicación para contrarrestar el Covid-19? Más de la mitad de los artículos y trabajos relacionados, apuntan a que los métodos para el diagnóstico basados en el análisis de tomografías computarizadas (TC) y radiografías pulmonares (rayos X), son los que mayor precisión arrojaron en la fase de evaluación. De estos dos últimos, el más asequible en cuanto a disponibilidad y factibilidad, es el método que emplea rayos X para el diagnóstico de Covid-19, según los artículos TR01, TR07, TR10, TR13, TR16 y TR24.

b) ¿Cuáles son los métodos, modelos o herramientas más precisos empleados hasta la fecha, para el diagnóstico de Covid-19?

En cuanto a las herramientas, métodos y modelos de diagnóstico más utilizados, en los trabajos relacionados RT02, TR07, TR11, TR16, TR18, TR23, TR24, TR30, TR31, TR32, TR35 y TR41, se han implementado modelos basados en Redes Neuronales Convolucionales (CNN) para el análisis de imágenes médicas, mismos que han obtenido los mejores resultados en su fase de evaluación. Este tipo de modelos, junto con aquellos que se basan en el diagnóstico a partir de la sintomatología del paciente, como en el caso de los artículos TR08 y TR34, presentan una mayor viabilidad en cuanto a su aplicación, ratificado en las revisiones y comparaciones de los modelos existentes TR08 y TR11, en donde se establece un análisis comparativo entre los diferentes modelos empleados para el diagnóstico de Covid-19, llegando a la conclusión, de que la implementación de modelos basados en redes neuronales convolucionales CNN poseen una mayor cantidad de aceptación y un mayor nivel de precisión, convirtiéndolos así, en los más factibles a implementar, en cuanto a modelos de IA se refiere.

Por lo tanto, ¿Cuál es el modelo, método o herramienta más precisa utilizada para identificar el virus del Covid-19? En vista de la gran variedad de modelos existentes, únicamente se resalta aquellos basados en la arquitectura de capas (CNN) para la interpretación de patrones existentes en las CT y las imágenes de rayos X (imágenes médicas). De esta forma, imita el entrenamiento médico, pero con una curva de aprendizaje mucho mayor, dado que este tipo de modelos son capaces de identificar los patrones existentes en un conjunto enorme de datos que cualquier humano, porque el enfoque de CNN tiene resultados más precisos y en periodos de tiempo mucho más cortos. En cuanto a la precisión y sensibilidad de los modelos implementados y descritos en los artículos TR01, TR02, TR07, TR11, TR14, TR16, TR17, TR25, TR26, TR28,

TR29, TR30, TR31, TR32, TR36 y TR41, existe un umbral aceptable que supera fácilmente el 90% de precisión, siendo TR02, TR11, TR16, TR28, TR28, TR31, TR32 y TR4, los que mayor nivel de precisión presentan, aproximadamente un 98%.

c) ¿Cuáles son los recursos necesarios empleados por los métodos de IA para la detección de Covid-19?

Tomando como referencia aquellos artículos en los cuales el nivel de precisión supera el 90%, se da contestación a la última pregunta de investigación, ¿Cuáles son los requisitos para su implementación?

Empezando por el artículo TR02, el cual propone la creación de un sistema potenciado por el aprendizaje automático, para el diagnóstico de COVID-19 aplicando un modelo basado en CNN llamado DeepLab-v3+. Este modelo requiere el tratamiento previo y el aumento de los datos, de tal manera, que los pequeños detalles, las texturas y el contraste de las imágenes permitan una mayor precisión del diagnóstico. Así mismo TR11, propone una red neuronal convolucional profunda para clasificar un conjunto de imágenes médicas y diagnosticar enfermedades como la neumonía y la COVID-19. Para ello utiliza una máquina de vectores de soporte lineal, VGG-16 e InceptionV3 para modelos de redes neuronales convolucionales que son utilizados para obtener los resultados, además, se utilizan algoritmos de aprendizaje como SVM, algoritmo Naive Bayes y el algoritmo de bosque aleatorio.

En cuanto al TR16, se propone la utilización de modelos prediseñados de CNN (VGG-16 o VGG-19) para la creación de un modelo propio, constituido por 15 capas (una menos que la VGG-16 que contiene 16 capas). En términos de rendimiento VGG-16, es una red con baja complejidad computacional debido a las pequeñas dimensiones de sus filtros, exactamente 9 pixeles por cada una de las 16 capas de aprendizaje que la componen.

Al igual que en los artículos anteriores, TR28 propone la implementación de un modelo basado en redes neuronales convolucionales CNN para solventar el problema actual que presentan las pruebas RT-PCR al diagnosticar COVID-19. Este modelo está enfocado en el diagnóstico a partir del análisis de radiografías de tórax, además, utiliza SVM para clasificar cada una de las clases de radiografías existentes. En lo que respecta al modelo de aprendizaje profundo utilizado en este trabajo, el VGG-16 que también se menciona en el artículo TR29, consta de diferentes capas de agrupación máxima, de activación y convolucionales completamente conectadas, por lo tanto, contiene un total de 21 capas, pero solo 16 de peso. Estas capas adicionales están conformadas a su vez por 5 capas de agrupación máxima, 3 de densidad y 13 de

convolución.

En cuanto al lenguaje de programación empleado para la construcción de los modelos, se ha utilizado Python, y en lo que a hardware corresponde, se recomienda utilizar un CPU con una capacidad de procesamiento igual o superior al AMD Ryzen 53600x o por el lado de Intel, considerar las opciones a partir del Core i5-9400f en adelante, en cuanto a una GPU, se recomienda el uso de una NVIDIA GTX 1080 o superior, además como requisito mínimo, usar 16 GB de RAM para aprovechar al máximo la capacidad de procesamiento de las CPU y la GPU.

5. Materiales y Métodos

5.1. Tipo de investigación

Durante la ejecución del presente TT, se empleó la investigación cualitativa, siendo principalmente experimental, ya que, en el transcurso del mismo, se trabajó un modelo base que fue elegido tras el análisis previo utilizando esta metodología, además se utilizó características de este método como la observación directa, el estudio de casos y la entrevista abierta para la recolección de información adicional concerniente a los métodos que los especialistas llevan a cabo para el diagnóstico de esta enfermedad.

Como resultado, se logró determinar que las redes neuronales convolucionales (CNN) son el tipo de arquitectura neuronal artificial que mejores resultados presentaron frente a las necesidades y requerimientos establecidos, mismos que se enfocaron en el desarrollo de un modelo capaz de diagnosticar Covid-19 mediante el análisis de radiografías pulmonares, donde la revisión de literatura relacionada a este campo, aportó el material bibliográfico que permitió determinar y sustentar el tipo de arquitectura CNN que mejores resultados haya obtenido al momento de analizar imágenes (radiografías). El resultado de la investigación determinó que, la arquitectura VGG16 usada en el reconocimiento de imágenes era la opción más viable a usar para alcanzar el objetivo de este TT. Asimismo, esta metodología ayudó en la búsqueda de los datos, los cuales sirvieron para entrenar y evaluar la precisión del modelo y extraer información del proceso que un especialista realiza para el diagnóstico de la enfermedad. Los artículos resultantes de esta revisión se muestran en la **TABLA I**.

5.2. Métodos de investigación

5.2.1. Método científico

Según Castán [86], el método científico es un procedimiento mediante el cual se emplea la observación sistemática, medición, experimentación, formulación, interpretación y modificación de las hipótesis, lo que permite al investigador dar respuestas a las interrogantes planteadas, considerando las variables del entorno y la naturaleza de la investigación, concretamente, es un método que relaciona la ciencia con el conocimiento científico.

Este método fue empleado para el cumplimiento de cada uno de los objetivos, siguiendo los pasos que se muestran en la **Fig. 24**.

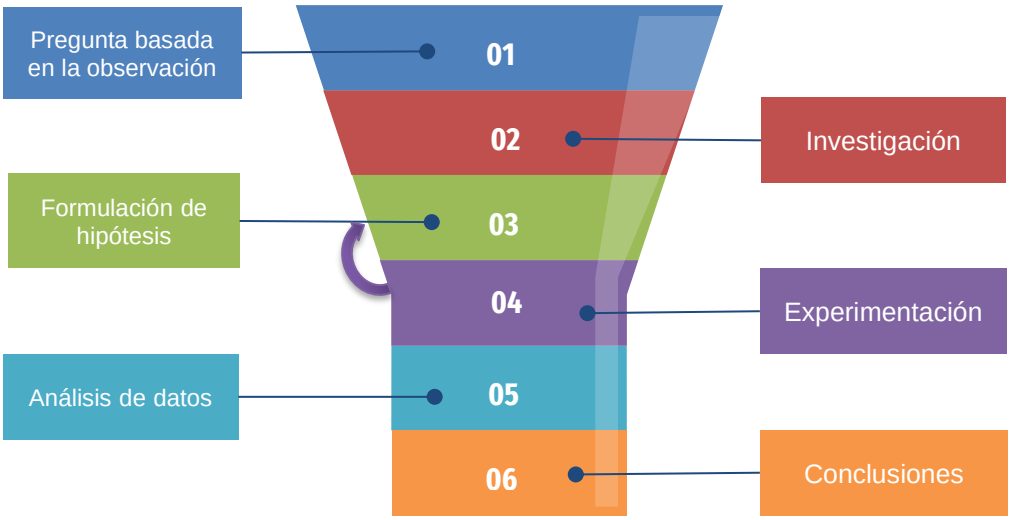


Fig. 24 *Esquema general del método científico.*

Estos pasos fueron acoplados acorde a los objetivos y a sus tareas específicas, lo cual se sintetiza en la **Fig. 25**, donde se aprecia los pasos de la metodología aplicada, en relación a cada uno de los objetivos y sus actividades. En lo que respecta al primer objetivo, la formulación de la pregunta basada en la observación, fue necesaria para determinar la fuente de los datos a utilizar durante el desarrollo del proyecto, para ello fue necesario plantear una RSL que no solo permitiera determinar la fuente de información principal, si no también establecer la problemática y la sustentación del presente TT, en base al análisis de trabajos relacionados que hayan tratado con un problema similar, de igual forma, la investigación y análisis llevados a cabo durante el desarrollo de la RSL fue de gran ayuda para comparar los resultados obtenidos por los TR y los modelos aplicados, junto con los datos que fueron utilizados para su desarrollo, lo que permitió la formulación de la pregunta de investigación, que a su vez constituye la hipótesis de nuestro proyecto, cuya conclusión fue la selección del modelo junto con

el dataset que mejor se adaptó a los recursos disponibles y a la fiabilidad de los resultados obtenidos por sus antecesores.

Fig. 25 *Aplicación del método científico por cada uno de los objetivos del presente TT.*

El mismo procedimiento fue aplicado en el objetivo número dos, que consiste en crear un modelo para diagnosticar Covid-19 utilizando la arquitectura de CNN. Para determinar el modelo a utilizar se aplicado los mismos pasos que en el objetivo anterior, y cuya información base fue el resultado de la RSL planteada inicialmente. En cuanto a la experimentación y análisis de datos, fueron empleados en la construcción del modelo, tanto en la fase de desarrollo, como en la preparación del entorno y aceleración por hardware necesario para maximizar la eficiencia del modelo y la reducción de los tiempos de entrenamiento, respecto a la conclusión, se obtuvo un modelo entrenado acorde a la configuración de los hiperparámetros que mejores resultados arrojaron en la fase de experimentación, asegurando que el modelo resultante, junto con sus respectivos pesos, fuese el más optimo posible considerando las condiciones y recursos empleados.

Por último, en el desarrollo del objetivo número tres, se utilizó la investigación y formulación de hipótesis para diseñar y crear un sistema que facilitara el uso del modelo en la fase de pruebas con ZSL y con el especialista. Este sistema a su vez, forma parte del aporte de investigación de nuestro TT, que se resume en una interfaz de usuario que involucró la construcción y consumo de recursos (API) para establecer la comunicación entre el modelo a la interfaz. El análisis de los datos y las conclusiones, se utilizaron para contrastar los resultados obtenidos tanto en las pruebas de ZSL y los diagnósticos validados con el experto en radiografías, llegando a la conclusión de que

nuestro modelo supera al especialista en aspectos que se recalcan en la sección de resultados.

5.3. Técnicas e instrumentos para la recolección de información

5.3.1. Recopilación de información

Para la recopilación de información relacionada con el presente TT, se realizó una revisión sistemática de literatura, RSL (ver Anexo 1) basada en la metodología de Bárbara Kitchenham [89], con la que se obtuvieron 102 estudios relevantes, de los cuales, 41 fueron seleccionados (ver **TABLA I**) y analizados detalladamente en la sección 4.4. La evaluación y extracción de información en cada uno de los TR, además de contribuir a la sustentación científica al presente TT, permitió determinar las técnicas y métodos utilizados para la detección de Covid-19 (SARS-CoV-2), de otras enfermedades con sintomatología similar, tales como, la gripe y la neumonía pulmonar, mediante la aplicación de técnicas de aprendizaje automático o Machine Learning, y de esta manera, determinar la viabilidad de diseño y creación de un sistema capaz de diagnosticar, con un nivel de eficiencia aceptable, si el paciente (usuario) es portador de Covid-19.

5.3.2. Entrevista

La entrevista, según [90], se define como “una técnica que permite obtener información mediante una conversación, propuesta con un fin determinado, distinto al simple hecho de conversar”. La aplicación de esta técnica fue de gran utilidad durante el desarrollo de la investigación cualitativa (expuesta en el Anexo 3), ya que con ella fue posible extraer información adicional que el especialista utilizó en la fase de evaluación del modelo para identificar las radiografías de los pacientes infectados por covid-19 de las que no lo estaban. Al mismo tiempo, se aprovechó las características de la entrevista propuesta en TR32, puesto que el autor sugiere el uso de la entrevista informativa o libre como método para la obtención de la información relacionada con el Covid-19, es por ello el uso de preguntas abiertas para corroborar información referente a las características de los pulmones infectados por covid-19, factibilidad del diagnóstico basado en radiografías como método alternativo, y a la vez, obtuvimos un retroalimentación del modelo desde el punto de vista del profesional, en cuanto a la eventual aplicación del modelo en el ámbito profesional, sobre todo, en los centros de salud y hospitales donde carezcan de insumos médicos para el diagnóstico y control de

la enfermedad.

5.3.3. Fine tuning

Según [91], Fine tuning es una técnica empleada en Dee Learning para el ajuste de los pesos en una topología de red previamente entrenada a una velocidad de aprendizaje relativamente lenta que permita obtener el mejor rendimiento del modelo al procesar los datos de entrenamiento. Esta técnica será empleada para ajustar los parámetros necesarios en el modelo VGG16, de tal forma que este sea capaz de reconocer si la imagen médica ingresada corresponde o no, a un paciente con COVID-19 o descartar dicha afección. El ajuste de sus hiperparámetros dependerá de los recursos disponibles y de los niveles de precisión alcanzados.

5.3.4. Zero Shot Learning (ZSL)

Dentro del aprendizaje automático, la técnica del Zero-Shot Learning (aprendizaje de tiro cero), ZSL por sus siglas en inglés, hace referencia al proceso mediante el cual, una máquina aprende a reconocer patrones de imágenes diferentes a las utilizadas en la fase de entrenamiento [92]. Esta particularidad ha sido utilizada en el apartado de Resultados del Objetivo 3, referentes a la primera actividad denominada *Pruebas Zero Shot Learning fase 2.* con el fin de medir la eficiencia del modelo al momento de clasificar las imágenes médicas con las que no ha sido entrenado, ya que en su implementación y posible uso profesional, el usuario ingresará imágenes totalmente diferentes a las utilizadas en la fase de entrenamiento, por lo que el modelo debe estar en la capacidad de clasificar con un nivel de precisión, sensibilidad y especificidad superiores a las presentadas por las pruebas tradicionales actualmente empleadas, garantizando así, un diagnóstico confiable y eficiente.

5.4. Técnicas, instrumentos y herramientas para el desarrollo del modelo

5.4.1. Lenguaje de programación

5.4.1.1. Python

Es el lenguaje de programación más utilizado para el desarrollo de modelos de aprendizaje automático (ML) e inteligencia artificial, debido a su gran flexibilidad, disponibilidad de módulos y librerías de acceso libre (desarrolladas por la comunidad y expertos) orientadas a la realización de proyectos de ML. Asimismo, destaca por la simplicidad en su sintaxis, lo que facilita el desarrollo de modelos óptimos y escalables,

en comparación con otros lenguajes, como R, Java o Matlab [93]. Además, la elección de Python como lenguaje de programación está justificado por los trabajos relacionados, TR01, TR02, TR03, TR16, TR22, TR24, TR38 y TR40. Esto sumado a que las librerías explicadas en la sección 4.2, están disponibles para este lenguaje. El modelo como tal utiliza la versión v 8.8.8, mismo se detalla a profundidad en el README del modelo.

5.4.2. Herramientas de software

5.4.2.1. Anaconda

Es una suite con más de 720 paquetes de código abierto (entre ellos tensorflow, keras, openCV, sklearn, etc.) que funciona como un gestor de entorno, debido a que posee una colección de herramientas y aplicaciones facilitando su despliegue desde su propia interfaz gráfica llamada *Anaconda Navigator*. De igual forma, funciona como un gestor de paquetes y librerías, de manera que se optó por utilizar este gestor para la instalación de las librerías junto con el gestor propio de Python llamada `pip`. Otra característica de esta herramienta, es que facilita la creación de diferentes ambientes virtuales o *enviromets*[6] (envs) que es donde se realizará la instalación de paquetes de forma aislada al resto de proyectos almacenados localmente, esta particularidad simplificó la tarea de realizar pruebas con las diferentes configuraciones y experimentar con los hiperparámetros del modelo preentrenado VGG16, optando así por la que mejores resultados arrojó [94]. A continuación, se describe las principales características de las herramientas utilizadas de la suite Anaconda, las cuales constan en le README del modelo, junto con las muestras de ejemplificación:

5.4.2.1.1. Conda

Es un sistema de gestión de entornos y paquetes diseñado para trabajar específicamente con Python, aunque actualmente funciona con varios lenguajes de programación. En este TT, se ha instalado `conda`, para la instalación, ejecución y actualización de paquetes y dependencias necesarios para la creación del modelo, de tal forma que no afecte al resto de librerías y proyectos alojados localmente. De igual forma, este gestor facilitó la creación y administración de entornos virtuales con diferentes versiones de las librerías, mismas que tras haber aplicado el método científico

[6] Los entornos virtuales o enviroments (env) en el caso particular de Python, se emplean para administran entornos de proyectos, puesto que cada uno de ellos tiene su propia forma de descargar, almacenar y resolver paquetes [104]. Estas condiciones pueden variar dependiendo del proyecto como versiones de librerías y frameworks, hasta la versión requerida de Python ejecutar todo el proyecto [104].

acorde a las configuraciones de los hiperparámetros sugeridas en los TR, se pudo determinar la configuración que mejor se adaptó a los requerimientos establecidos.

5.4.2.1.2. JupyterLab.

JupyterLab es una versión mejorada de Jupyter Notebook que se encuentra disponible dentro de la suite de anaconda, esta herramienta de código abierto se utiliza para el desarrollo, entrenamiento, creación y visualización los datos y graficas de modelos de IA. Estas características fueron de gran utilidad, puesto que permitió ejecutar el modelo por partes, de tal forma que la corrección de errores y optimización de código utilizando la técnica de *Fine tuning* se realizó de una forma óptima [95]. Además, JupyterLab, al ser una herramienta integrada en la suite de anaconda, hace uso de los recursos instalados en el entorno creado específicamente para el funcionamiento del modelo.

6. Resultados

En el siguiente apartado se presenta los resultados obtenidos para cada uno de los objetivos específicos del presente TT, los cuales permitieron el cumplimiento del objetivo principal. Cada objetivo se subdivide en tareas, las cuales se detallan a continuación:

6.1. Objetivo 1: Recopilar información de datasets que contengan imágenes de rayos x, que permitan entrenar y evaluar el modelo para el diagnóstico de Covid-19.

Para la recopilación de los datos que permitieron entrenar y evaluar el modelo, fue necesario delimitar las características del conjunto de datos (datasets) de las cuales se han obtenido las radiografías pulmonares. Se resolvió, además de contar con los datasets utilizados en los trabajos relacionados expuestos en la **TABLA I**, la creación de condiciones de inclusión y exclusión enfocadas a la selección de estos datos.

6.1.1. Tarea 1 Buscar en datasets accesibles imágenes de rayos X

En cuanto a la búsqueda de información, se utilizó como punto de partida los 41 artículos seleccionados, mismos que fueron el resultado de la RSL anexada en la sección 0. Del total de TR se ha incluido únicamente aquellos que hagan mención al uso de datasets que contengan imágenes de rayos X, dichos artículos se detallan en la **TABLA II**.

TABLA II *Fuente de datos proveniente de los artículos seleccionados en la RSL.* ***Fuente:*** *Sitio o repositorio del cual provienen la información.* ***Tipo de acceso****: si el dataset es de acceso libre o privado,* ***disponibilidad****: si los datos alojados en el dataset son libres o privados aún se encuentran accesibles para su utilización. L****ink:*** *dirección del recurso web.*

Código	Fuente	Tipo de acceso	Disponibilidad	URL
TR03	IEEE8023	Libre	Accessible	https://github.com/shimaaelbana/Classification-and-Segmentation-of-infectionmanifestations-in-COVID-19-scans
TR05	IEEE8023	Libre	Accesible	https://github.com/ieee8023/covid-chestxray-dataset
TR13	IEEE8023	Privado	Accesible	https://www.kaggle.com/c/rsna-pneumonia-detection-challenge/
	RSNA			

	COVIDNet			https://github.com/iliasprc/COVIDNet
TR15	IEEE8023	Libre	Accesible	https://github.com/ieee8023/covid-chestxray-dataset
TR18	RSNA	Privado	Accesible	https://www.kaggle.com/c/rsna-pneumonia-detection-challenge/data
TR22	IEEE8023	Libre	Accesible	https://github.com/ieee8023/covid-chestxray-dataset
TR24	IEEE8023	Libre	Accesible	https://github.com/ieee8023/covid-chestxray-dataset
TR31	EINSTEIN D4U	Libre	Acessible	http://www.kaggle.com/einsteindata4u/covid19
TR32	IEEE8023	Libre	Accesible	https://github.com/ieee8023/covid-chestxray-dataset
TR35	MEDRXIV	Privado	Inaccesible	https://www.medrxiv.org/node/72563.external-links.html
TR37	IEEE8023	Libre	Accessible	https://github.com/ieee8023/covid-chestxray-dataset
	LARXEL	Libre	Inaccesible	https://www.kaggle.com/andrewmvd/convid19-x-rays
		Libre	Accessible	
TR40	TAWSIFUR RAHMAN	Libre	Accessible	https://www.kaggle.com/tawsifurrahman/covid19-radiography-database

Una vez obtenidos los datasets disponibles de todos los trabajos relacionados, se procedió a su análisis y selección, para lo cual se ha establecido los siguientes criterios de inclusión y exclusión:

- **Criterios de inclusión**
  - El conjunto de datos debe ser de acceso libre (open Source).
  - Debe contener imágenes de rayos X.
  - El número de radiografías debe ser superior a 1000.
  - El año en el cual se ha generado la información no debe ser menor al

2019.

- Los datos deben estar clasificados previamente.

- **Criterios de exclusión**
  - El conjunto de datos con acceso restringido.
  - Que contenga cualquier tipo de información que coincida con imágenes de rayos x.
  - Cualquier dataset que se encuentre por debajo del umbral de tiempo establecido.

6.1.2. Tarea 2: Seleccionar aquellos datos que cumplan con los criterios de aceptación.

Tras haber establecido los criterios de inclusión y exclusión, se procedió a seleccionar los datasets que cumpliesen con los criterios establecidos, en la **TABLA III**, se muestra los resultados de los DS a los cuales se logró acceder. De estos resultados, únicamente TAWSIFUR RAHMAN cumple con todos los criterios de aceptación establecidos, por lo que este DS fue utilizado tanto en la fase de entrenamiento y evaluación del modelo. No obstante, para la fase del ZSL, se utilizó imágenes de IEEE8023, ya que es el dataset que TR03, TR05, TR13, TR15, TR22, TR24 y TR32 utilizan para la fase de entrenamiento y test.

TABLA III *Selección de los DS preseleccionados en base a los criterios de aceptación establecidos.*
***Imágenes****: Número de imágenes alojadas en el DS.*
TC/RX*: Tomografías computarizadas o rayos X,* ***Clasificación****: Si los datos (imágenes) se encuentran clasificados.*
Año: *Año de publicación de la información.*

Fuente	# Imagenes	TC/RX	Clasificación	Año
IEEE8023	930	RX	N	2020
RSNA	3000	RX	S	2019
TAWSIFUR RAHMAN	21164	RX	S	2020
LARXEL	79	RX	N	2020

En lo que respecta al dataset de RSNA, se trata de un concurso impulsado por la Sociedad Radiológica de América del Norte, el cual liberó los datos a finales del 2019, con el fin de que los participantes inscritos lograran crear un algoritmo para detectar una señal visual de neumonía en imágenes médicas, específicamente, se necesitaba ubicar automáticamente las opacidades pulmonares en las radiografías de tórax [96]. Pese a cumplir con la mayoría de las condiciones iniciales, este dataset ha sido descartado por

el hecho de que las radiografías liberadas no reflejan los efectos que el COVID-19 provoca en los pacientes.

En cuanto al dataset, TAWSIFUR RAHMAN, se trata de un conjunto de datos público, creada el 28 de marzo del 2020, con la finalidad de construir un dataset con imágenes de rayos X obtenidas de pacientes positivos para COVID-19, opacidad pulmonar y neumonía. Muchas de estas radiografías proceden principalmente de fuentes como la Sociedad Italiana de Radiología Médica e Intervencionista (SIRM) y otras 43 fuentes no especificadas por el autor [97]. Este dataset dispone de 21164 radiografías clasificadas en 4 categorías diferentes, mismas que se detallan a continuación:

- **COVID**: Contiene 3615 radiografías de pacientes diagnosticados con Covid-19 (**Fig. 26 a**).
- **Laung_Opacity**: Contiene 6012 radiografías de pacientes diagnosticados con infección de neumonía bacteriana (**Fig. 26 b**).
- **Normal**: Contiene 10192 radiografías de pacientes sin ningún tipo de anomalías pulmonar (**Fig. 26 c**).
- **Viral_Pneumonía**: Contiene 1345 radiografías de pacientes diagnosticados con Neumonía viral (**Fig. 26 d**).

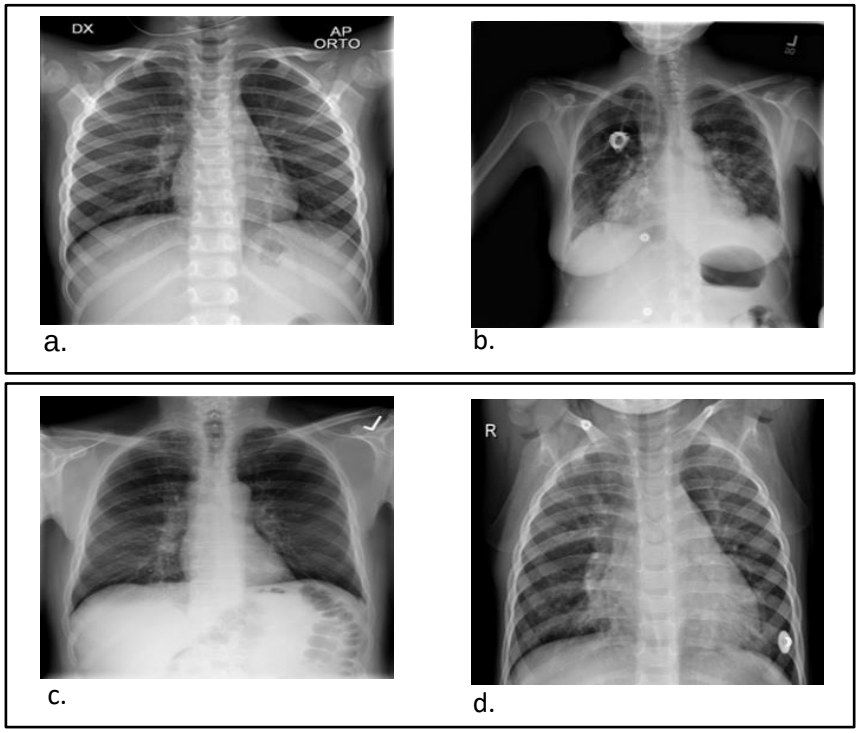


***Fig. 26 a**. Radiografía de paciente diagnosticado con Covid-19. **b.** Radiografía de paciente diagnosticado con infección de neumonía bacteriana. **c.** Radiografía de paciente sin ningún tipo de anomalía pulmonar. **d**. Radiografía de paciente diagnosticado con Neumonía viral.*

6.1.3. Tarea 3: Realizar un tratamiento previo de los datos.

Con el conjunto de datos seleccionado y al ser de acceso libre, no hubo inconveniente alguno al momento de obtener las radiografías y la información relacionada a ellas. Respecto a los datos, TAWSIFUR RAHMAN, posee cada uno de los 4 apartados descritos en la sección anterior separados en carpetas, además existen metadatos en formato .xlsx para cada uno de las secciones antes mencionadas, tal y como se evidencia en la **Fig. 27**, donde constan las segmentaciones de las radiografías en función de la afección (COVID, Lung_Opacity y Viral Pneumonia) o ausencias (Normal) de las anomalías pulmonares, con sus respectivos metadatos.

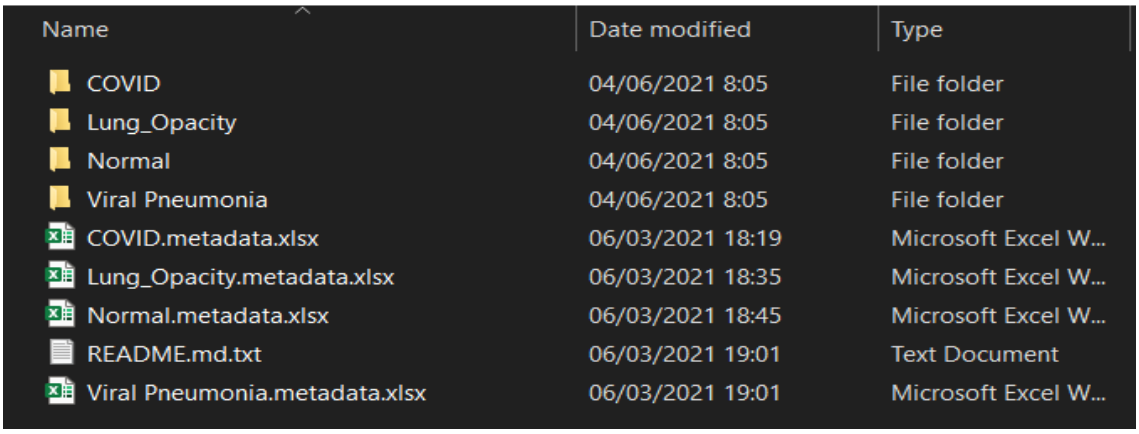


Fig. 27 *Contenido del dataset TAWSIFUR RAHMAN almacenados localmente. Las carpetas contienen las radiografías para cada uno de los casos a los que hace referencia y los archivos .xlsx son los metadatos correspondientes al conjunto de datos*

En cuanto a los datos empleados en la fase de entrenamiento y fase de evaluación, se han recopilado un total de 13807 radiografías, distribuidas entre COVID y Normal, con 3615 y 10192 respectivamente. Una vez que se definió el grupo de imágenes a utilizar, fue necesario subdividirlas en dos grupos, entrenamiento (training) y evaluación (test), para lo cual se ha tomado como referencia los siguientes trabajos relacionados TR18, TR22, TR27, TR34, TR36 y TR37, mismos que utilizaron diferentes cargas de imágenes con una proporción de 50:50, 70:30 y 80:20, siendo la proporción de 80:20 la más empleada.

6.2. Objetivo 2: Crear un modelo para diagnosticar Covid-19 utilizando la arquitectura de CNN

6.2.1. Tarea 1: Buscar un modelo base de CNN para iniciar la construcción a partir de un modelo prediseñado

Considerando los trabajos relacionados que hacen referencia a los modelos y

arquitecturas implementadas para el diagnóstico de COVID-19, mismos que utilizan como base al análisis de radiografías pulmonares, se ha considerado tres modelos, los cuales serán analizados para determinar la factibilidad de la posible implementación de uno de ellos en el presente TT. El análisis previo necesario para la preselección, se basó en la precisión y factibilidad de la implementación de los modelos en los TR seleccionados, para ello se estableció un umbral de aceptación superior al 70%, siendo TR01, TR02, TR07, TR11, TR14, TR16, TR17, TR25, TR26, TR28, TR29, TR30, TR31, TR36, TR40 y TR41, y los que superaron el umbral de aceptación con un porcentaje promedio del 90% de precisión, 87.7% de sensibilidad y 92.9% de especificidad (ver **TABLA IV**), de los cuales, TR01, TR14, TR16, TR26, TR28 y TR32, presentaron un mayor nivel de precisión, superior al 95%, a excepción de IceptionV3, que presentó un valor del 90% de precisión en comparación al resto de modelos que sobrepasan el 95% (Véase la **TABLA V**), este análisis se encuentra en el Anexo 5, disponible en el CD número 2.

TABLA IV *Niveles de precisión y parámetros de evaluación de los modelos implementados en los TR.*

TR: *Código de trabajo relacionado,* ***Modelo:*** *Nombre del modelo principal identificado en los respectivos TR,* ***Precisión:*** *Valor de precisión del conjunto de valores obtenidos a partir de mediciones repetidas de una magnitud,* ***Sensibilidad****: Porcentaje de objetos identificados correctamente,* ***Especificidad****: Se trata de los casos negativos que el algoritmo ha clasificado correctamente.*

TR	Modelo	Precisión	Sensibilidad	Especificidad
TR01	VGG-16	0.975	0.923	0.988
TR02	COVID19-Net	0.875	0.917	0.864
TR07	LASSO	0.839	0.868	0.733
TR11	ResNet	0.900	-	-
TR14	VGG-16	0.982	0.920	0.980
TR16	VGG-16	0.985	0.984	0.985
TR17	VGG-19	0.900	-	-
TR25	U-Net	0.833	0.812	0.854
TR26	ResNet-50	0.956	0.844	0.933
TR28	VGG-16	0.950	0.60	-
TR29	V-Net	0.855	0.866	0.932
TR40	Inception-V3	0.90	0.877	0.680
TR41	XGBoost	0.850	0.870	0.925

TABLA V *Modelos con un índice de precisión superiores al 90%.*

TR	Modelo	Precisión	Sensibilidad	Especificidad
TR01	VGG-16	0.975	0.923	0.988
TR14		0.982	0.920	0.980
TR16		0.985	0.984	0.985
TR26	ResNet-50	0.956	0.844	0.933
TR40	Inception-V3	0.90	0.877	0.680

Como resultado del análisis anterior, los modelos con un mayor nivel de precisión fueron VGG-16, ResNet50 e InceptionV3, razón por la cual fueron preseleccionados como posibles modelos de CNN para el diagnóstico de Covid-19 a partir del análisis de radiografías pulmonares. Por tanto, fue necesario realizar un análisis detallado de las características de cada uno de los modelos y determinar el que mejor se adaptase a los fines establecidos. Los parámetros establecidos para seleccionar el modelo que mejor se adapte al presente TT (véase la **TABLA VI**) fueron los requerimientos y la factibilidad de implementación. En el caso de los requerimientos, se consideraron aquellos utilizados en los trabajos relacionados, lo cual ha servido como punto de comparación con los recursos disponibles para el desarrollo del presente TT. Estos recursos fueron aquellos disponibles en Google Colab (en su versión libre) que ofrece 12 GB de RAM y 50 GB de almacenamiento, y las especificaciones para el desarrollo local, los cuales son un procesador Intel (R) Core (TM) i7-10750H a 2.60 Hz, 32 GB de RAM y una GPU Nvidia GeForce 2070 Super con 8 GB de NVRAM, y como sistema operativo Windows 10 de 64-bits. En vista de las características antes mencionadas, aquellos recursos disponibles para el desarrollo local, fueron los que mejores características presentan, frente a la versión libre de Google Colab (computación en la nube), razón por la cual, se optó por implementar el modelo utilizando procesamiento local.

Tras el análisis de las características de los modelos, presentado en la **TABLA VI**, el modelo VGG-16, cumplió con las especificaciones de los recursos disponibles, aunque no sean exactamente los mismos componentes, los disponibles cumplen, he incluso superan los utilizados en los TR a los que hace referencia el modelo. Con respecto a los modelos restantes, ResNet50 es parcialmente factible ya que el tipo de tarjeta gráfica utilizada en este trabajo difiere de los componentes disponibles, además no existen especificaciones de la capacidad de Memoria RAM necesaria que permita entrenar y evaluar el modelo. Por último, InceptionV3, no especifica el tipo de procesador y la cantidad de memoria RAM necesaria para su implementación.

TABLA VI *Comparación de parámetros de los modelos de CNN utilizados para el análisis de imágenes médicas utilizadas en los TR.*
SO: *Sistema Operativo,* ***LP:*** *Lenguaje de Programación,* ***LB:*** *Librerías,* ***Img:*** *Imágenes,* ***BC:*** *Balance de Carga*

Nombre	Precisión %	Requerimientos		Factibilidad
VGG16*	98	CPU	Intel (R) Core (TM) i7-5700 HQ CPU (2.70GHz)	SI
		GPU	NVIDIA GTX 970M 8 GB de GDDR	SI
		RAM	16 GB	SI
		SO	Windows 10 (64-bit)	SI
		LP	Python	SI
		LB	Keras	SI
			Tensorflow	SI
			Open CV	SI
		Img	1000	SI
		BC	20:80	SI
ResNet50	95	CPU	Intel(R) Xeon(R) CPU X5460 (3.16 GHz)	NO
		GPU	Nvidia Tesla GPU con 16 GB de VRAM	NO
		RAM	No especificado	-
		SO	Arquitectura de 64-bits	SI
		LP	Python	SI
		LB	fastai2	SI
		Img	6 868	SI
		BC	50:50	SI
InceptionV3	90	CPU	No especificado	-
		GPU	Tesla K80 GPU	NO
		RAM	No especificado	-
		SO	Windows 10 (64-bit)	SI
		LP	Python	SI
		LB	PyCM	SI
		Img	1600	SI
		BC	70:30	SI

6.2.2. Tarea 2: Construir el modelo para el diagnóstico de Covid-19.

En lo que respecta a la construcción del modelo para el diagnóstico de Covid-19 mediante el análisis de radiografías, se ha tomado como referencia la arquitectura VGG16, la cual se ha detallado en la sección 4.3.1 del presente TT, cuyo proceso de selección se muestra en la sección anterior. Esta arquitectura se caracteriza especialmente por el número de capas de convolución y sus hiperparámetros, los cuales posibilitan el refinamiento mediante técnicas como Fine tuning, de tal forma que, al obtener los resultados de las predicciones, estos sean lo más cercanos a 1 o al 100% de precisión.

Por consiguiente, para el uso de los modelos, se han empleado las librerías descritas en la sección 4.2.3, en las cuales se menciona la posibilidad de usar arquitecturas preentrenadas, donde se deben especificar los hiperparámetros necesarios (los hiperparámetros dependen del modelo a usar). Los modelos disponibles en keras, entre ellos VGG16, pertenecen a este tipo de modelos, por lo que son perfectos para utilizarlos en un proyecto de ML aplicado al reconocimiento de patrones dentro de un conjunto de imágenes y así lograr la concepción de un modelo capaz de identificar los mismos patrones en otras radiografías de pacientes con Covid-19 y normales. Estas diferencias se basan en la presencia o ausencia de los efectos provocados por el Covid-19 en los pulmones de los pacientes que lo padecen, entre aquellas radiografías obtenidas de pacientes que no porten el virus o incluso, que porten otro tipo de virus que provoquen patrones diferentes en los pulmones. Por esta razón, se ha considerado los parámetros de aceptación para el dataset ha emplear, ya que dichas radiografías permitieron construir la base de conocimiento para nuestro modelo, con la cual se han realizado las predicciones/diagnóstico de las entradas. Una vez definido el dataset a utilizar, y tanto la arquitectura como el modelo base hayan sido seleccionados, se procedió con la fase de programación del modelo, al cual de ahora en adelante se hará referencia con el nombre de Diagnos19[7]. Para la elaboración de este modelo, se tomó como referencia los trabajos relacionados TR14, TR16 y TR34, mismos que sirvieron como sustentación para la elección de la arquitectura VGG16. Además, esta investigación está sustentada en otros trabajos relacionados como TR04, TR05, TR06, TR21, TR25 y TR28, con el objetivo de definir el esquema general del modelo a desarrollar, el cual se muestra en la **Fig. 28**.

[7] Diagnos19, es el nombre del modelo resultante del presente TT. La palabra “Diagnos19 ” ha sido elaborado a partir de un juego de palabras provenientes de **Diagnós**tico y Covid-**19.**

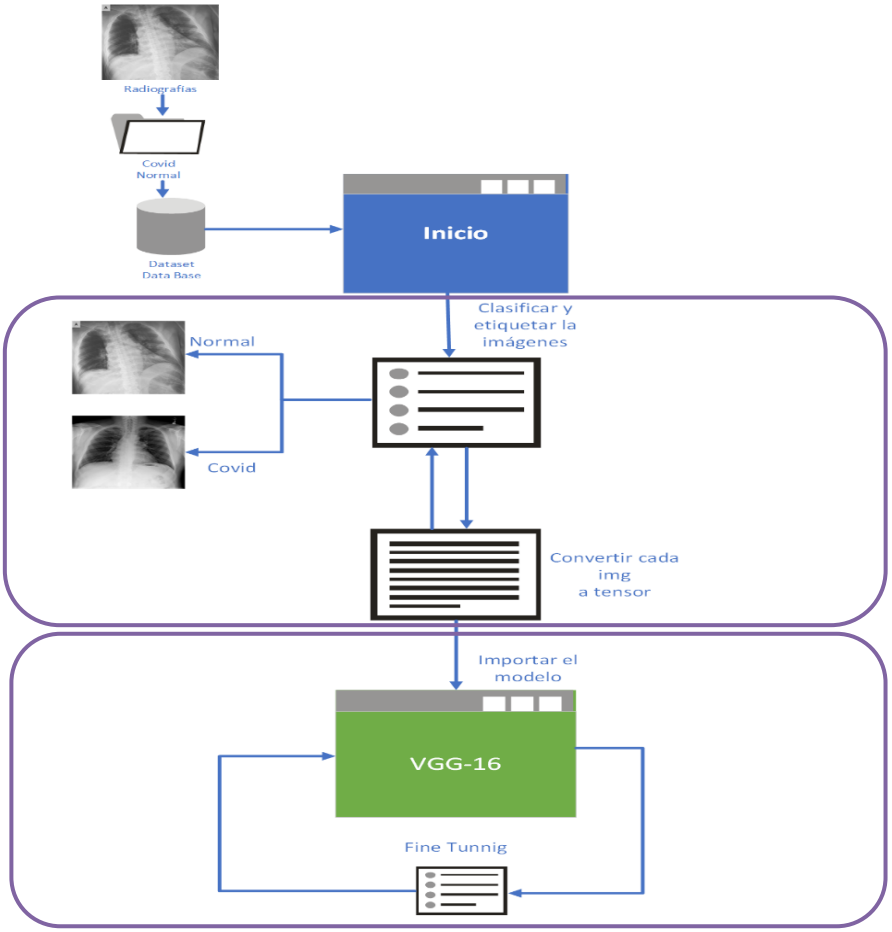


Fig. 28 *Esquema general de la estructura de Diagnos19*

Dicha estructura se divide en tres partes:

- Carga y clasificación de las imágenes (radiografías).
- Conversión de las imágenes a tensores.
- Fine tuning de VGG16.

A continuación, se describe cada una de estas fases, las cuales se han desarrollado considerando el análisis realizado en la sección 6.2.1, en la que se enfatiza que las especificaciones disponibles en la PC local superan las especificaciones disponibles en la nube, por lo que el modelo se ha realizado de forma local, usando JupyterLab como IDE y Github como repositorio del modelo. En cuanto a los requerimientos de software, se ha creado y desplegado un environment (entorno virtual), de tal forma que la instalación y actualización de las librerías y dependencias necesarias para el funcionamiento del modelo sea sustentable y escalable. Las librerías y dependencias mencionadas durante la fase de desarrollo y las especificaciones de los procesos de instalación y configuración se describen de forma detallada en el README de modelo.

6.2.2.1. Carga y clasificación de radiografías

Para cargar las radiografías del datasets TAWSIFUR RAHMAN, se empleó las siguientes librerías:

- **os:** permite utilizar funcionalidades dependientes del sistema operativo, tales como leer o escribir un archivo almacenado localmente o bien, manipular rutas.
- **cv2:** OpenCV, es una librería utilizada para el tratamiento de imágenes mediante algoritmos de IA orientados a la visión por computadora o ML. (véase la sección 4.2.1 OpenCV)
- **tqdm:** permite mostrar el estado de progreso de cualquier variable iterable.

Este proceso se esquematiza en la **Fig. 29**, en el cual, dentro del conjunto de radiografías, se aisló los directorios con las imágenes de interés, y se realizó el proceso de lectura (Read), para ello se utilizó las librerías `os y cv2.` La primera de ellas, obtiene la ruta local de cada una de las radiografías alojadas en ambos directorios. Estas rutas, fueron utilizadas posteriormente para vincular cada una de las imágenes con sus respectivas etiquetas, “Normal” y “Covid-19”, debido a que su identificación en las fases de entrenamiento y evaluación, es crucial para el aprendizaje del modelo.

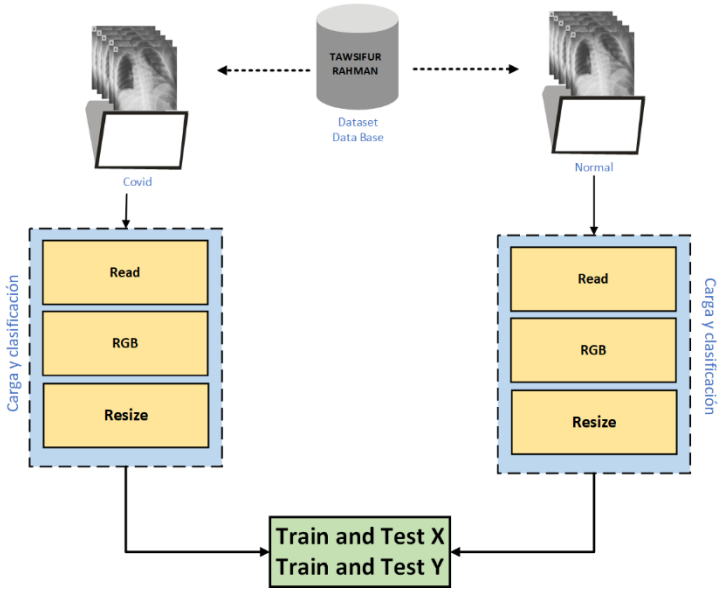


Fig. 29 *Esquema de funcionamiento para cargar, leer y redimensionar las radiografías.*

Por otra parte, la segunda librería, carga cada una de las imágenes usando como parámetro de entrada las rutas obtenidas anteriormente. Una vez realizada la lectura y carga de archivos, se procedió a intercambiar el orden de los canales Blue, Green y Red

(BGR) a Red, Green, Blue (RGB), puesto que la mayoría de las librerías de procesamiento de imágenes (`opencv` y `matplotlib`) usan la composición RGB. Aunque OpenCV procese la configuración BGR sin ningún tipo de problemas, no sucede lo mismo con `matplotlib`.

Por último, fue necesario redimensionar el tamaño de cada una de las imágenes (Resize) con un ajuste de 224 x 224 pixeles, este valor, extraerá un parche de tamaño (224, 224) en la imagen entrante de forma aleatoria, es decir, el parche podría elegir este camino de arriba hacia abajo, de izquierda a derecha, abajo derecha o en cualquier lugar intermedio, garantizando el aumento de los datos en la sección, tal como se muestra en la **Fig. 30**.

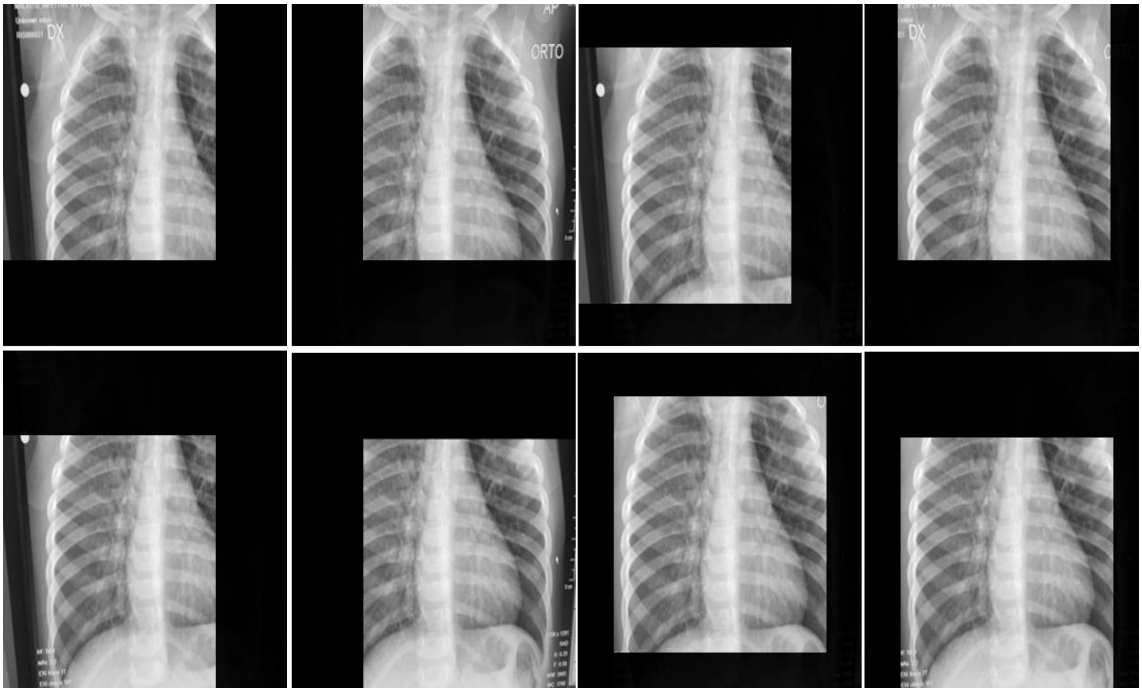


Fig. 30 *Posibles reajustes aleatorios de una radiografía a la cual se aplicó un parche de dimensión a 224 x 224 de alto y ancho respectivamente.*

6.2.2.2. Conversión de imágenes a tensores

Los tensores son un tipo especial de matrices que permiten a las CNN etiquetar y clasificar cualquier tipo de entrada, inclusive los datos no estructurados como las radiografías (imágenes). En este caso en particular, se empleó un tensor de tres dimensiones o 3D, haciendo uso de la librería TensorFlow (TF), quién es uno de los parámetros del modelo VGG16, el cual se ha configurado más adelante.

Retomando el esquema de la **Fig. 30**, el bloque final comprende la creación de cuatro variables (véase la **Fig. 31**), las cuales contienen tanto las matrices que se usaron en la fase de entrenamiento como en la fase de prueba.

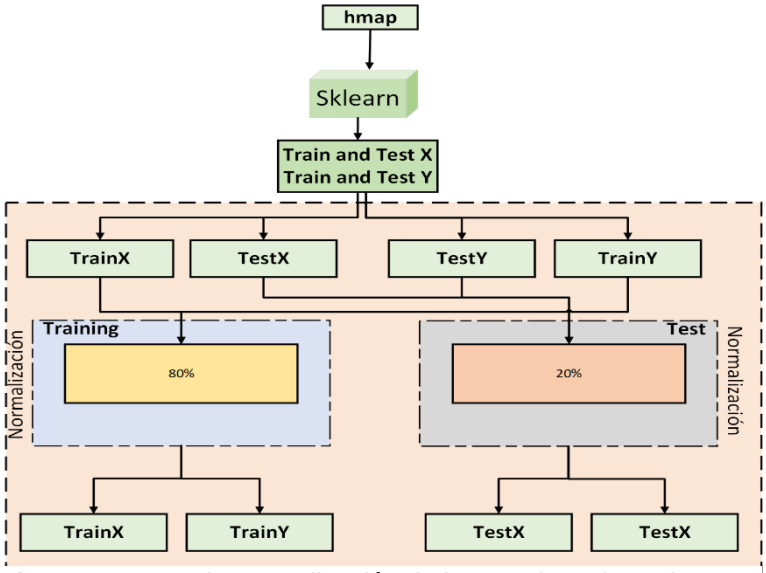


Fig. 31 *Proceso de normalización de las matrices de Train y test previo a al uso de tensores*

Dichas variables se dividen en dos grupos, train (entrenamiento) y test (prueba), donde la distribución de carga es igual al 80% en la fase de entrenamiento y al 20% para la fase de prueba. Por lo que la información de cada una de las variables, se compone de la siguiente manera:

- **TrainX**: contine el 80% del total de las matrices obtenidas a partir de las imágenes cargadas desde el conjunto de datos tras haber realizado el proceso de carga y clasificación. Estas matrices fueron usadas en la fase de entrenamiento del modelo. A continuación, se presenta la matriz en el índice 0 del arreglo:

```
[[107, 107, 107],
[108, 108, 108],
[111, 111, 111],
...,
[ 75,  75,  75],
[ 76,  76,  76],
[ 77,  77,  77]],
```

- **TrainY:** Contiene la información de las etiquetas de clase en forma de matriz. En la **Fig. 31** (parte superior), se presenta un bloque llamado `hmap`, que hace referencia a un HashMap, los cuales se utilizan para almacenar datos clave/valor, lo que en Python se lo conoce como diccionarios. Para este TT, los

valores definidos en la variable hmap, corresponden al nombre del directorio y a la clase a la cual pertenecen, es decir, la clave es el nombre de la carpeta en la que están alojadas las radiografías (Normal, COVID) y su valor, es la clase en la que se encuentra alojada dentro del modelo (`normal, covid-19`). Al definir el `hmap`, la conversión, clave/valor a matriz se realizó con la librería sklearn, obteniendo los siguientes resultados:

```
[[1. 0.]
 [0. 1.]
 [1. 0.]
 ...
 [1. 0.]
 [0. 1.]
 [1. 0.]]
```

Donde:

- [1. 0.] corresponde a una sola radiografía, por lo que habrá tantas como número de matrices haya en TrainX.
- Si, el valor de la matriz en la posición 0 es igual a 1, y la posición 1 es 0, es decir [1. 0.], entonces dicha matriz corresponde a una radiografía de la clase **covid-19**.
- Si, el valor de la matriz en la posición 0 es igual a 0, y la posición 1 es 1, es decir [0. 1.], entonces dicha matriz corresponde a una radiografía de la clase **normal**.

- **TestX:** Está conformado por las matrices de las imágenes que conforman el 20% restante del total de radiografías empleadas en el modelo.
- **TestY:** Contiene el hmap de cada una de las matrices de TestX.

6.2.2.3. Fine tuning de VGG16

VGG16, es un modelo de CNN, sobre el cual se ha basado la construcción de Diagnos19. El modelo como tal, se obtuvo mediante Keras, la cual cuenta con una API que permite desarrollar y entrenar modelos de aprendizaje profundo de forma rápida y flexible. No obstante, tal como se muestra en la **Fig. 32**, el modelo requiere del ajuste de sus hiperparámetros, los cuales determinarán la precisión del modelo.

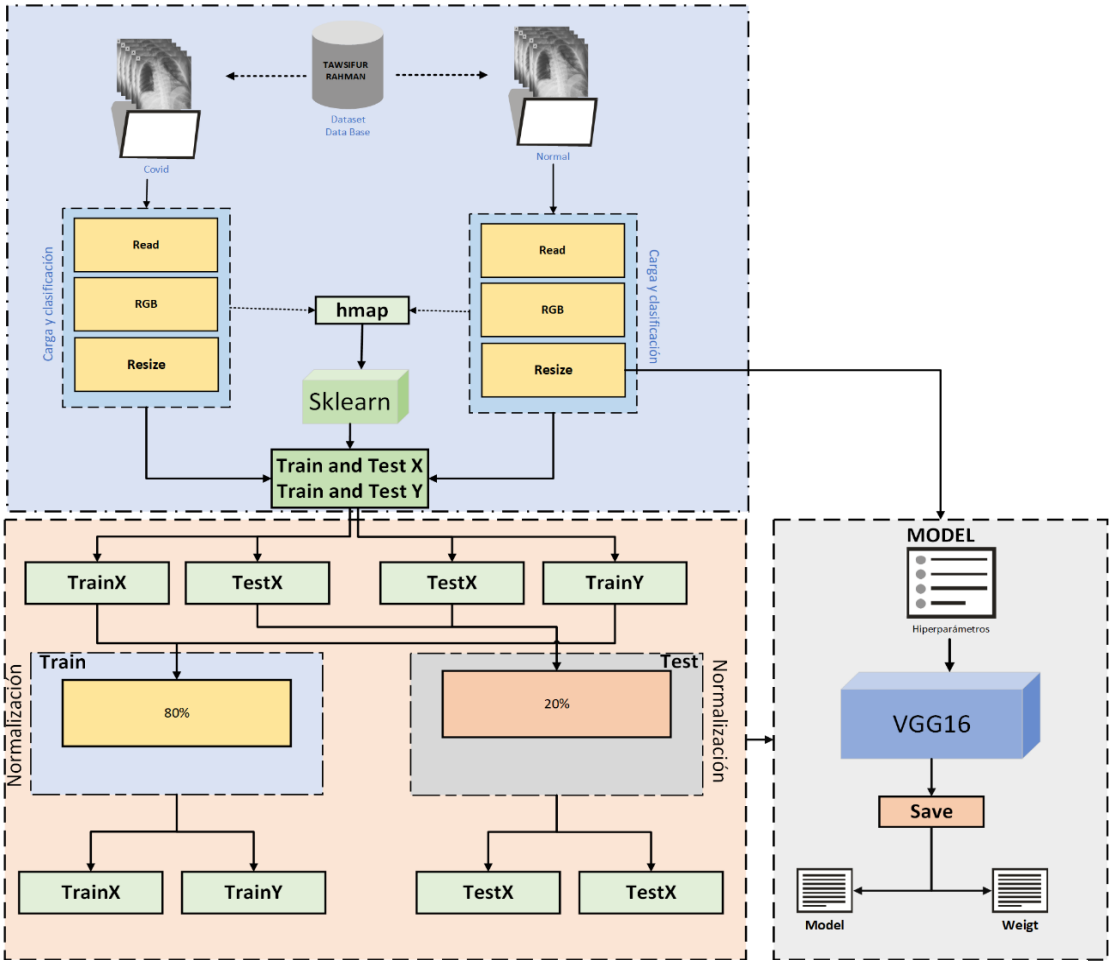


Fig. 32 *Esquema general de la arquitectura del modelo Diagnos19*

Los hiperparámetros necesarios para el buen funcionamiento del modelo se explican a detalle en la sección 4.2.3 del presente TT, por lo que a continuación se describe únicamente su configuración, la cual se ha definido en el siguiente segmento de código (el código completo utilizado en la construcción del modelo Diagnos19, se encuentra disponible en el Anexo 4, sección 1, CD 2):

```
baseModel = VGG16(weights="imagenet", include_top=False, input_t
ensor=Input(shape=(imgSize, imgSize, 3)))

headModel = baseModel.output
headModel = Flatten(name="flatten")(headModel)
headModel = Dense(64, activation="relu")(headModel)
headModel = Dropout(0.5)(headModel)
headModel = Dense(2, activation="softmax")(headModel)
model = Model(inputs=baseModel.input, outputs=headModel)
```

Donde:

- **`baseModel`:** Es una variable que contine una instancia del modelo VGG16, quien, además de los parámetros requeridos en su documentación, necesita

algunas condiciones previas, como el tamaño de las entradas establecidas en (224x224) y la composición de colores RGB. En cuanto a los parámetros:

- **`weights:_`** se ha establecido el valor 'imagenet' (entrenamiento previo en ImageNet) debido a la ausencia de los pesos previos al entrenamiento (no están disponibles para este modelo).
- **`include_top:_`** el valor 'False' indica que no se deben incluir las 3 capas completamente conectadas en la parte superior de la red, pues el output de cada una de las capas será el input de cada una de las capas adyacentes.
- **`input_tensor:_`** se estableció el tensor 3D, el cual será utilizado como entrada para el modelo. La composición de dicho tensor está dada por el tamaño de las imágenes definido en la sección anterior y el total de canales de colores establecido para la normalización de las imágenes, por lo que el tensor final está definido de la siguiente manera: `[224, 224,3].`

- **`headModel:`** En esta variable se ha realizado un conjunto de operaciones las cuales se describen a continuación:
  - ***Flatten***: permite aplanar las dimensiones de una matriz $n\,x\,m$, a una matriz $1\,x\,m$, esta capa se encarga de redimensionar las matrices de entrada y proporcionar una matriz de dimensión $1\,x\,m$, a las capas adyacentes.
  - ***`Dense`*:** permite crear y entrelazar capas de tal forma que el recorrido de la red sea desde la entrada hasta la salida. En este caso, la entrada y su forma "`shape`", se especifica en la variable `baseModel` y las salidas en la propia variable `headModel,` de forma recurrente. Así mismo, las operaciones con la variable de activación `activation="softmax"`, añaden dos capas adicionales (véase **Fig. 33**), que retornan como resultado un softmaxed con la misma forma que los inputs, es decir [1. 0.] o [0. 1.] dependiendo de la entrada.`[98]`

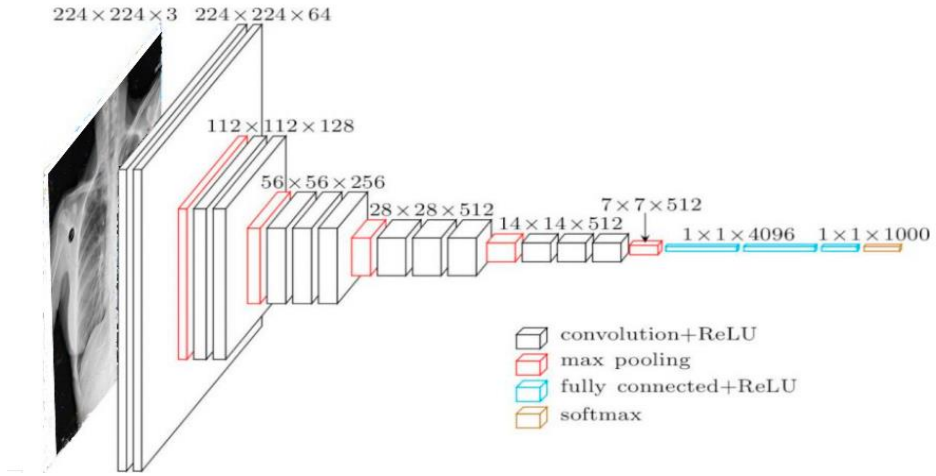


Fig. 33 *Representación gráfica del recorrido por cada capa en el modelo VGG16.*

- ***model*:** En esta variable se ha creado el modelo, tomando en cuenta las configuraciones creadas anteriormente, mismas que se usarán como entradas (baseModel) y salidas (headModel) de VGG16. Además, esta variable permite guardar el resultado del modelo y sus pesos una vez haya sido entrenado, para ello se usa las siguientes funciones:

```
model.save('model.h5')
model.save_weights('modelWeight.h5')
```

Para la representación del modelo resultante almacenado en la variable `model`, se optó por la elaboración de un diagrama de capas (ver la **Fig. 34**) para representar y visualizar el modelo junto con todas sus configuraciones. Otra forma de visualizar esta información es mediante la propiedad `summary()`, proporcionado por keras, la cual retorna una tabla con la información de cada una de las capas y los parámetros totales disponibles para ser entrenados. A continuación, se presenta el número total de parámetros entrenables y no entrenables, obtenidos con esta función:

- Total params: 16,320,514
- Trainable params: 1,605,826
- Non-trainable params: 14,714,688

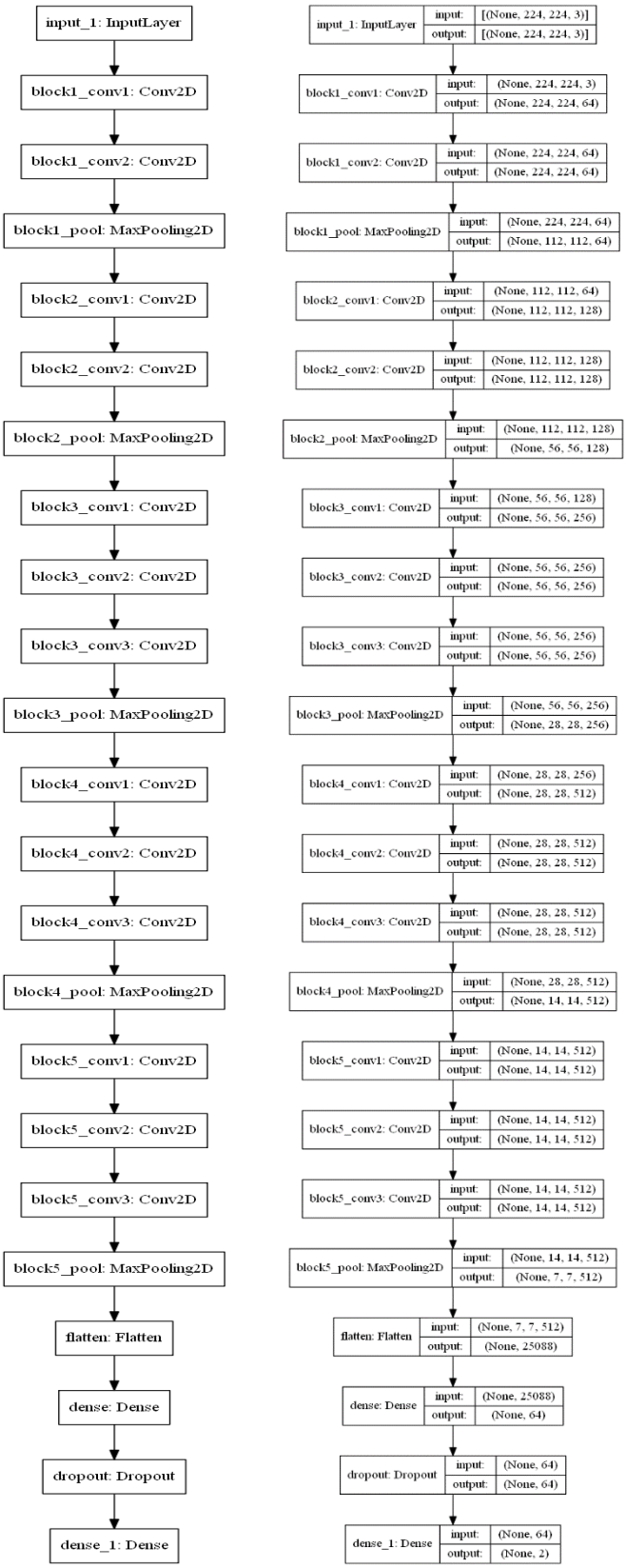


Fig. 34 *Gráfico de capas del modelo VGG16. En la parte izquierda se encuentra el trazado de todas las capas convolucionales del modelo, y la parte derecha se presentan las capas con sus respectivas entradas y salidas.*

6.2.3. Tarea 3: Entrenar el modelo utilizando los datos de prueba.

Para la fase de entrenamiento, se usaron las variables resultantes de la fase de construcción del modelo (TrainX/Y & TestX/Y), las cuales albergan los datos normalizados de las radiografías obtenidas en el dataset TAWSIFUR RAHMAN. Este dataset está conformado por cuatro directorios que contienen cuatro tipos diferentes de radiografías, entre ellos NORMAL, con 1092 radiografías y COVID, con 3615 radiografías (`[normal, covid-19]`), distribuidas en cargas proporcionales del 8:2, es decir el 80% del total de las imágenes se usaron en la fase de entrenamiento y el 20% restante, se utilizaron en la fase de test del modelo (véase la **Fig. 32**). De las 13807 imágenes médicas disponibles, se han establecido varias cargas para las dos fases, dichas cargas se detallan en la **TABLA VII**, en la que se establece la nomenclatura para guardar el modelo (su peso tendrá el mismo nombre con la palabra «peso “weight”» como distintivo entre los archivos), el objetivo de esta subdivisión es medir la precisión, sensibilidad y especificidad, obtenidas en cada uno de los modelos resultantes, los cuales difieren en el número de radiografías (usando la misma proporcionalidad 8:2).

TABLA VII *Segmentación de las radiografías que se utilizaron en la fase de entrenamiento subdivididas de acuerdo a la proporcion 8:2, junto a su respectivo modelo resultante.*

Nombre	Número de imágenes		Total de imágenes	Proporción	
	Covid	Normal		Training(80%)	Test(20%)
modelVGG16Settings	250	250	500	400	100
modelVGG16to1k	500	500	1000	800	200
modelVGG16to2k	1000	1000	2000	1600	400
modelVGG16to4k	2000	2000	4000	3200	800
modelVGG16to6k	3000	3000	6000	4800	1200
modelVGG16to7k	3500	3500	7000	5600	1400

En cuanto al proceso de entrenamiento, se utilizó la metodología Fine Tuning para realizar algunos ajustes y modificaciones a la tasa de aprendizaje, Batch Size y Epoch, los cuales son hiperparámetros que deben ser ajustados para obtener resultados mucho más óptimos en lo que a precisión se refiere. Estos ajustes han sido basados en los trabajos relacionados TR01, TR28 y TR29, los cuales sirvieron de base para el desarrollo del modelo VGG16, descrito en la sección anterior.

A continuación, se presenta el proceso de fine tuning de los hiperparámetros, no sin

antes recalcar que las pruebas se han realizado considerando la distribución mínima de carga de la **TABLA VII**, las distribuciones restantes de esta tabla se tomaron en cuenta para la fase de entrenamiento, una vez se determinó el valor para cada uno de los hiperparámetros.

- **Learning Rate (LR)**: El Learning Rate o Tasa de Aprendizaje, es uno de los hiperparámetros más importantes, ya que este valor determina si la curva de aprendizaje se propaga hacia atrás, es decir indica la tendencia hacia los mínimos, mientras más pequeña sea la tasa de aprendizaje, hará que el modelo converja más lentamente. Por otra parte, si el valor es mucho mayor al que debería, provocará que el modelo diverja. De modo que, si el valor utilizado dentro de esta variable no es el correcto, provocará que el modelo no logre alcanzar los niveles de precisión máximos posibles. Según los TR relacionados, se recomienda el uso de los siguientes valores $1e^{-4}, 3e^{-4}$ y $3e^{-5}$.
- **EPOCHS:** Conocidos también por el nombre de épocas, este valor se encarga de definir el número de veces que una entrada (input) pasa por la red. Por lo general, llegados a un determinado momento, en el que el peso converge en cierto valor, cualquier aumento a partir de este, no aumentará la eficiencia del modelo, sino más bien, limitará su rendimiento. Según los TR, se recomienda el uso de los siguientes valores 20, 50 y 100 para una posible optimización del modelo.
- **Batch Size (BS)**: Es el número de ejemplos que se introducen en la red para que se entrenen a la vez. Si el número es pequeño, significa que la red tiene en memoria poca cantidad de datos, por lo que el tiempo de entrenamiento será mucho menor, dependiendo del total de datos usados en la fase de entrenamiento, sin embargo, es posible que no aprenda ciertas características y detalles que pueden ser significativos en la predicción. Por otra parte, si el valor de BS es mayor, ocurre lo contrario: es más probable que tenga en cuenta los casos más importantes a la hora de aprender, pero el tiempo de entrenamiento es mucho más lento, aunque en gran medida, estos tiempos dependen de los recursos empleados para este fin. Según los TR se recomienda el uso de los siguientes valores 32, 64 y 128.

El resultado del proceso de Fine tuning se presenta en la **TABLA VIII**.

TABLA VIII *Combinación de los hiperparámetros para el proceso de fine tuning.*
Hiperparámetros*: Valor de cada uno de los hiperparámetros, tomando en cuenta los TR.*
Resultados: *Resultados en cuanto al nivel de precisión y pérdida de la fase de test del entrenamiento.*
Gráfico*: Referencia a las gráficas obtenidas de la fase de entrenamiento.*

#	Hiperparámetros			Resultados		
	INIT_LR	**EPOCHS**	**BS**	**Precisión**	**Pérdida**	**Gráfico**
1	$1e^{-4}$	20	32	0.98	0.02	**Fig. 51**
2	$1e^{-4}$	50	64	0.98	0.02	**Fig. 52**
3	$1e^{-4}$	100	128	-	-	-
4	$3e^{-4}$	20	32	0.98	0.02	**Fig. 53**
5	$3e^{-4}$	50	64	0.99	0.01	**Fig. 35**
6	$3e^{-4}$	100	128	-	-	-
7	$3e^{-5}$	20	32	0.96	0.04	**Fig. 54**
8	$3e^{-5}$	50	64	0.96	0.04	**Fig. 55**
9	$3e^{-5}$	100	128	-	-	-

Acorde a los resultados de la **TABLA VIII**, las configuraciones 3, 6 y 9, no presenta ningún tipo de resultado, ya que su configuración sobrepasaba los requisitos computacionales disponibles a la fecha en el que este trabajo se desarrollaba, por lo que no se pudo llevar a cabo las pruebas de fine tuning con tales valores. En cuanto a las configuraciones 7 y 8, su aplicación resulta inviable, debido a que presentan el nivel de precisión más bajo en comparación al resto de configuraciones, tal como se evidencia en la **TABLA VIII** y en sus respectivas gráficas (véase la **Fig. 54** y **Fig. 55**, en la sección de 1).

Mientras que las configuraciones 1, 2 y 4, pueden ser aplicadas sin ningún tipo de inconveniente como posibles valores de los hiperparámetros en el modelo, en vista de su nivel de precisión. No obstante, si se observa con detenimiento las gráficas de la **Fig. 51**, **Fig. 52** y **Fig. 53**, en la sección 0, se puede observar que estos modelos podrían alcanzar valores mucho más óptimos, ya que la tendencia entre la precisión de entrenamiento (Train Accuracy) y la precisión de TEST (Validation Accuracy), varía constantemente, conforme aumenta el número de épocas, lo cual sugiere que un afinamiento en la configuración de los hiperparámetros mejoraría la curva de precisión en ambos valores.

Por último, la configuración número 5 de la **TABLA VIII**, presenta el valor de precisión más elevado de entre todas las configuraciones, y sus gráficas indican que el modelo ha alcanzado su punto máximo de entrenamiento, tal como se evidencia en la gráfica

de la parte derecha en la **Fig. 35**, la línea denotada de color rojo, representa la precisión de entrenamiento general del modelo, en contraste con la línea de color azul, la cual representa la precisión de evaluación. Ambas líneas, a pesar de nunca llegar a intersecarse, presentan un punto de equilibrio, en el caso de la precisión de entrenamiento, su valor tiende a 1, llegada a la época 48, y en el caso de la precisión de la evaluación, su valor se normaliza en 0.99 en la época 23. De ahí en adelante, ambas líneas muestran una tendencia a estabilizarse, garantizando así, una precisión de entrenamiento óptima.

En cuanto a la pérdida general del modelo, en la gráfica derecha de la **Fig. 35,** la línea de color rojo y azul representan la pérdida en la fase de entrenamiento y evaluación respectivamente. Partiendo de la consigna, cuanto menor sea la pérdida, mejor será el modelo, en la gráfica, se puede comprobar que el valor de pérdida producida en la fase de entrenamiento con la configuración 5 de la TABLA VIII, inicia en 0.3 y tiende a 0, estabilizándose a un valor promedio del 0.02 de pérdida.

En cambio, en la fase de TEST, la pérdida se reduce de 60% al 12%, logrando un punto

Fig. 35 *Gráficas de precisión y pérdidas referentes a la configuración 5. A la izquierda, se encuentra el grafico de precisión y a la derecha el gráfico de pérdidas.*

de equilibrio cercano al 15%. Por lo tanto, esta configuración de hiperparámetros garantiza una predicción real del 85% en el peor de los casos, y en el mejor de ellos, el 100%. Esta estimación es mucho más evidente en la **Fig. 36**, donde en una sola gráfica se representa los valores de precisión y pérdida.

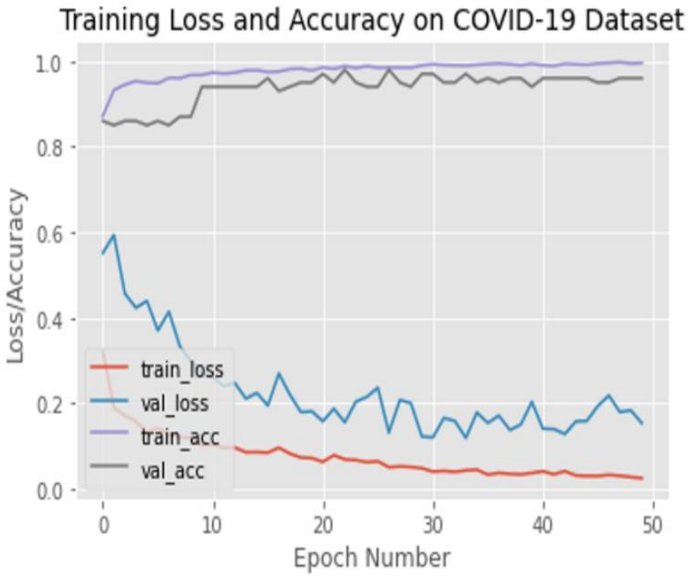


Fig. 36 *Gráfica de precisión y pérdidas referentes a la configuración 5.*

Otra herramienta útil para evaluar la precisión del modelo, es la matriz de confusión (Confusion Matrix), la cual se muestra en la **Fig. 37**. Esta matriz, se compone por la intersección entre las clases que conforman el modelo, siendo la diagonal principal, la cantidad exacta de predicciones correctas (positivas) y la diagonal secundaria, el número exacto de los errores de clasificación del modelo, es decir, la cantidad de falsos negativos y falsos positivos presentes en la predicción.

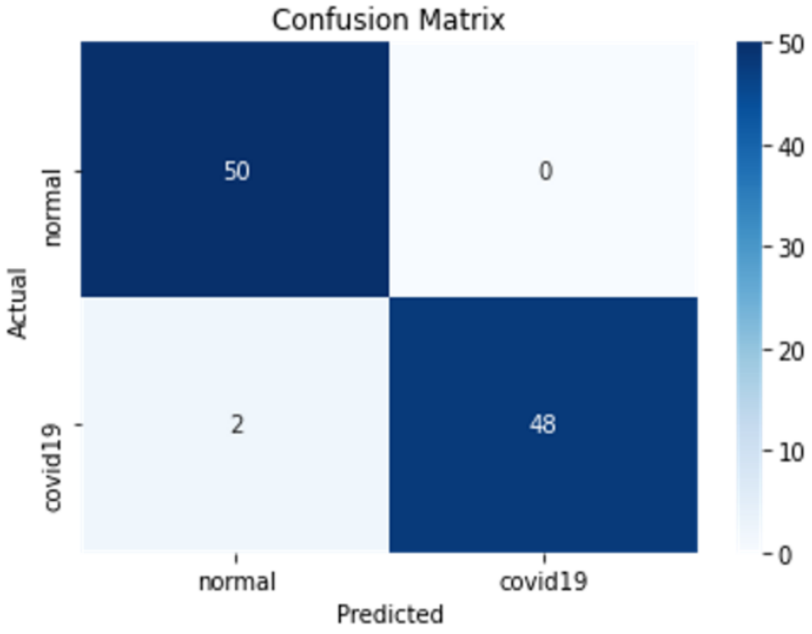


Fig. 37 *Matriz de confusión referente a la configuración 5*

El análisis de la matriz de confusión, se puede abordar de la siguiente manera:

De las 50 imágenes correspondientes a la clase “Normal”, todas fueron clasificadas correctamente, por lo que no existen casos de falsos positivos para Covid-19. En el caso de las radiografías de pacientes con diagnóstico positivo para Covid-19, 2 de las 50 radiografías utilizadas para su evaluación se han clasificado como falso negativo para la enfermedad, estableciendo una relación de falsos negativos aproximada de 1:30, es decir, por cada 30 radiografías de pacientes diagnosticados con Covid-19, una de ella será clasificada como normal (no porta la enfermedad).

En vista de los resultados obtenidos con la configuración 5 (INIT_LR=$3e^{-4}$, EPONCHS=50 & BS=64), y tras haber realizado un análisis minucioso de los resultados, se ha optado por utilizar esta configuración de hiperparámetros para realizar la fase de entrenamiento de los modelos, en base a la carga establecida en la **TABLA VII**.

Puesto que la configuración 5 de los hiperparámetros de la **TABLA VIII** es la más óptima para la fase de entrenamiento, se procedió a establecer dichos valores en la sección de código que se muestra a continuación (el código completo usado en la fase de entrenamiento se encuentra disponible en la sección 2 del Anexo 4, CD 2):

```
INIT_LR = 3e-4
EPOCHS = 50
BS = 64

opt = Adam(lr=INIT_LR, decay=INIT_LR / EPOCHS)

model.compile(loss="categorical_crossentropy", optimizer=opt, me
trics=["accuracy"])
H = model.fit(
    trainAug.flow(trainX, trainY, batch_size=BS),
    steps_per_epoch=len(trainX) // BS,
    validation_data=(testX, testY),
    validation_steps=len(testX) // BS,
    callbacks=[LambdaCallback(on_epoch_end=saveModel),],
    epochs=EPOCHS)
```

Usando la misma metodología para la fase de Fine Tuning, se procedió a entrenar los modelos finales, tomando como referencia la segmentación y distribución de cargas, conforme a la proporción 8:2 de la **TABLA VII**, a partir del segundo valor. Los resultados de los modelos entrenados y validados, junto con sus respectivas gráficas de precisión, pérdida y matrices de confusión, se muestran en el Anexo 2. Conforme a estos

resultados, se optó por el modelo entrenado con 6000 radiografías, mismo que obtuvo los mejores resultados de entrenamiento, tanto en la precisión, sensibilidad, especificidad y promedio ponderado de cada uno de estos parámetros.

En la **TABLA IX**, se detallan los resultados del proceso de entrenamiento, el cual duró cerca de 2 horas, 3 minutos y 43 segundos aproximadamente (usando la GPU para acortar el tiempo de entrenamiento), obteniendo un nivel de precisión global de 99.173% en la fase de entrenamiento. Estos porcentajes son el resultado de la suma de los parámetros de precisión, sensibilidad y especificidad, de cada una de las clases que conforman el modelo denominado, "`modelVGG16to6k`", mismo que ha sido entrenado con 6000 radiografías, divididas en partes iguales para cada clase, es decir 3000 radiografías con diagnóstico positivo para Covid-19, y 3000 imágenes de pacientes sin ningún tipo de anomalía pulmonar.

TABLA IX *Resultados de la fase de entrenamiento del modelo diagnos19, con una carga total de 6000 radiografías.*

	Parámetros	Precisión	Sensibilidad	Especificidad	Accuracy	Support	Time
Training	**Covid-19**	0.99832	0.99208	0.99519	0.99173	16800	2h 3m 43s
	Normal	0.94691	0.98833	0.96718		2400	
	Promedio ponderado	0.99190	0.99161	0.99169		19200	

En esta fase se alcanzó una precisión del 99.832% para los casos de "Covid-19" y el 94.691% para la clase "Normal", siendo la sensibilidad, es decir, la capacidad del modelo para clasificar las imágenes acorde a la clase a la que pertenece, de 99.161%, distribuidas entre el 99.208% para "Covid-19", y 98.833% para la clase "Normal". Otro de los parámetros utilizados para determinar la calidad del modelo, es la especificidad, la cual determina el porcentaje que posee el modelo para determinar los casos que no pertenecen a la clase en cuestión, en este caso, el porcentaje total de especificidad es de 99.169%, distribuidos ente el 99.519% para "Covid-19" y el 96.718% para la clase "Normal". En la gráfica derecha de la **Fig. 38**, se representa mediante la una línea roja, el valor de precisión (Accuracy) total del modelo en la fase de entrenamiento.

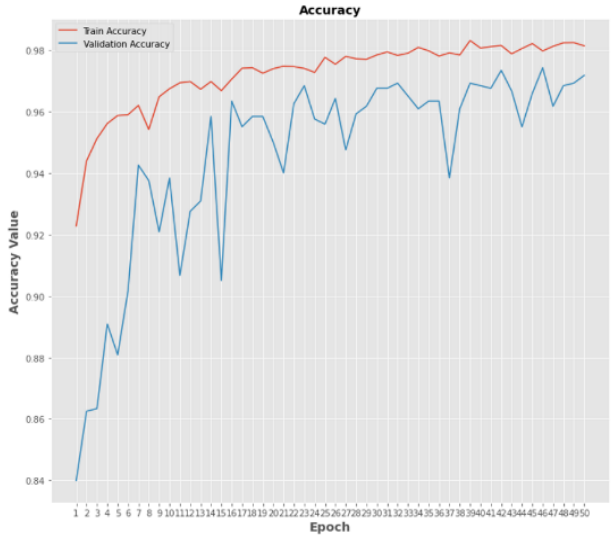


Fig. 38 *Gráficas de precisión correspondientes a la fase de entrenamiento con 6 000 radiografías.*

6.3. Objetivo 3: Evaluar la precisión y sensibilidad del modelo utilizando datos de prueba

6.3.1. Realizar pruebas del modelo empleando Zero Shot Learning

Para corroborar la precisión del modelo Diagnos19, se optó por usar la técnica de Zero Shot Learning (ZSL), la cual ha sido descrita en la sección 5.3.4 del presente TT. El uso de esta técnica ha permitido determinar la precisión del modelo al recibir como dato de entrada una radiografía que no ha sido utilizada en la fase de entrenamiento, simulando así, el comportamiento del modelo en condiciones de uso normal, puesto que las radiografías usadas por los profesionales varían en cuanto al tipo de máquina que las genera y con ello, el tipo de resolución y tamaño. Por tal razón, la implementación de esta técnica consta de dos fases las cuales se describen a continuación:

6.3.1.1. Pruebas Zero Shot Learning fase 1.

En lo que respecta a la primera fase de evaluación utilizando ZSL, se han utilizado un total de 1200 radiografías, distribuidas en partes iguales entre las clases que conforman el modelo. Esta distribución corresponde al 20% del total de radiografías utilizadas para la fase de entrenamiento y validación, según se especificó en la **TABLA VII**.

TABLA X *Resultados de la fase de entrenamiento y evaluación del modelo Diagnos19.*

	Parámetros	Precisión	Sensibilidad	Especificidad	Accuracy	Support	Time
Training	**Covid-19**	0.99832	0.99208	0.99519	0.99173	16800	2h 3m 43s
	Normal	0.94691	0.98833	0.96718		2400	
	Promedio ponderado	0.99190	0.99161	0.99169		19200	
Test	**Covid-19**	0.99003	0.99333	0.99168	**0.99167**	600	
	Normal	0.99331	0.99000	0.99165		600	
	Promedio ponderado	0.99167	0.99167	0.99167		**1200**	

Los resultados obtenidos en esta sección, se presentan en la **TABLA X**, en contraste con los resultados obtenidos en la fase de entrenamiento, concerniente al modelo Diagnos19, cuyo nivel de precisión general es del 99.167%, siendo el resultado de la suma de los promedios ponderados de cada uno de los parámetros que determinan la calidad del modelo. Donde, la precisión de la clase “Covid-19”, fue de 99.003% y del 94.691% para la clase “Normal”, proporcionando un promedio ponderado de 99.167%, solamente 0.023% por debajo del valor obtenido en la fase de entrenamiento. En términos de sensibilidad, el porcentaje alcanzado por la clase “Covid-19”, fue del 99.333% y 99.000% para la clase “Normal”, con un promedio ponderado de 99.167%, superando en 0.006% al valor obtenido en la fase de entrenamiento. Por último, el valor de especificidad alcanzado por la clase Covid-19 y Normal, fue de 99.168% y 99.165% respectivamente, logrando un promedio ponderado de 99.167%, superando en 0.002% al valor conseguido en la fase de entrenamiento. La representación de estos valores se muestra a la derecha de la **Fig. 38**, con una línea de color azul.

Acorde a los resultados de la **TABLA XI**, obtenidos en la primera fase de ZSL, el modelo `modelVGG16to6k` (evaluado con 1200 radiografías)**,** presentó los mejores resultados en esta sección, superando en todos los parámetros que indican la calidad de clasificación (precisión, sensibilidad, especificidad) en comparación a todos los modelos evaluados de acuerdo con el 20% de las cargas establecidas en la **TABLA VII**.

TABLA XI *Resultados de la fase de evaluación de todos los modelos entrenados de acuerdo con las cargas establecidas (1K, 2K, 4K, 6K y 7K).*

Carga	Fase	Parámetros	Precisión	Sensibilidad	Especificidad	Accuracy
1k	**Test**	**Covid-19**	0.98039	1.00000	0.99010	0.99000
		Normal	1.00000	0.98000	0.98990	
		Promedio ponderado	0.99020	0.99000	0.99000	
2k	**Test**	**Covid-19**	0.98039	1.00000	0.99010	0.99000
		Normal	1.00000	0.98000	0.98990	
		Promedio ponderado	0.99020	0.99000	0.99000	
4k	**Test**	**Covid-19**	0.96098	0.98500	0.97215	0.97250
		Normal	0.98462	0.96000	0.97215	
		Promedio ponderado	0.97280	0.97250	0.97250	
6k	**Test**	**Covid-19**	0.99003	0.99333	0.99168	**0.99167**
		Normal	0.99331	0.99000	0.99165	
		Promedio ponderado	**0.99167**	**0.99167**	**0.99167**	
7k	**Test**	**Covid-19**	0.98091	0.95429	0.96741	0.96786
		Normal	0.95549	0.98143	0.96829	
		Promedio ponderado	0.96820	0.96786	0.96785	

Los resultados del modelo 6k (modelVGG16to6k) se justifican y explican en la matriz de confusión de la **Fig. 39**.

De las 600 radiografías correspondientes a la clase “Normal”, 596 fueron clasificadas correctamente, lo que equivale al 99.331% de precisión para identificar radiografías de este tipo. Por consiguiente, 4 de las 600 radiografías fueron clasificadas como positivas para Covid-19, constituyendo el 0.669% de falsos positivos (radiografías normales diagnosticadas como Covid-19).

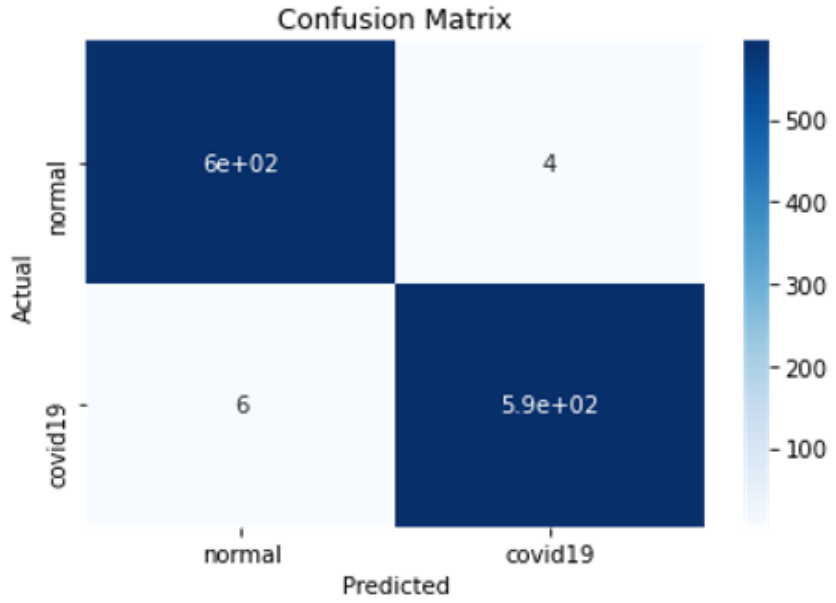


Fig. 39 *Matriz de confusión correspondiente a la fase de entrenamiento del modelo con 6000 radiografías*

En el caso de las radiografías de pacientes con diagnóstico positivo para Covid-19, 6 de las 600 radiografías fueron clasificadas como falsos negativos para la enfermedad, estableciendo una relación aproximada de 1:100, es decir, por cada 100 radiografías ingresadas de pacientes diagnosticados con Covid-19, una de ella será clasificada como normal (no porta la enfermedad), constituyendo así el 0.997% de falsos negativos y el 99.003% de precisión para identificar radiografías que en realidad presenten anomalías provocadas por Covid-19.

6.3.1.2. Pruebas Zero Shot Learning fase 2.

En lo que respecta a la segunda fase de la prueba de ZSL, se usaron 40 radiografías obtenidas de fuentes totalmente distintas a la que se utilizó en la fase de entrenamiento y a las pruebas ZSL de la fase 1, estos dataset fueron “Larxel” e “IEEE8023” (véase la **TABLA II**). Las 40 radiografías fueron distribuidas como se muestra en la **Fig. 40**, y están disponibles en un directorio denominado ZSL[8]

[8] Radiografías ZSL fase 2

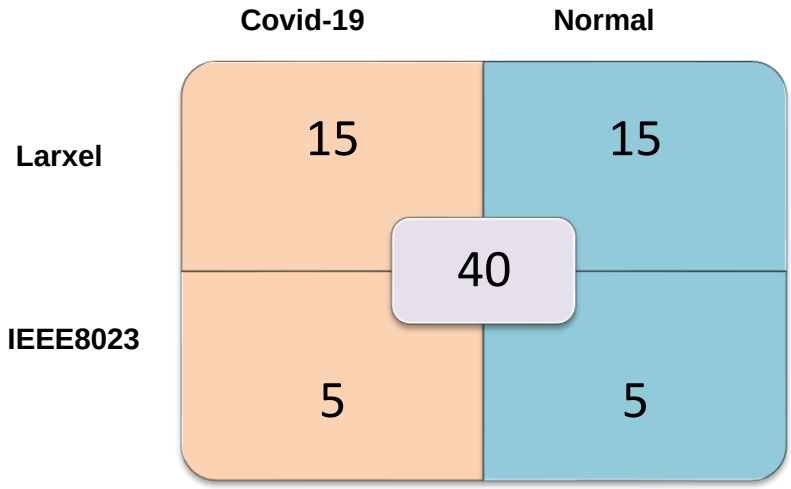


Fig. 40 *Distribución de las radiografías para las pruebas de ZSL*

En cuanto a las pruebas ZSL concernientes a la fase 2, fueron ejecutadas usando el IDE PyCharm (véase la **Fig. 60**, del Anexo 2) con lo cual se obtuvieron los siguientes resultados:

TABLA XII *Diagnósticos del modelo aplicando la técnica de ZSL.*

Clase	File	Accuracy	Diagnos19	Clase	File	Accuracy	Diagnos16
Covid-19	1	99,910	Covid19	Normal	30	92,530	Normal
	2	97,300	Covid19		32	98,580	Normal
	3	99,980	Covid19		33	99,160	Normal
	4	94,400	Covid19		41	99,920	Normal
	5	100,000	Covid19		45	99.850	Normal
	6	97,520	Covid19		51	98,140	Normal
	7	99,950	Covid19		52	77,710	Normal
	8	100,000	Covid19		56	99,710	Normal
	9	100,000	Covid19		59	99,880	Normal
	10	100,000	Covid19		60	100,000	Normal
	11	99,930	Covid19		66	91,180	Normal
	12	100,000	Covid19		72	85,580	Normal
	13	100,000	Covid19		73	82,260	Normal
	14	99,760	Covid19		79	92,810	Normal
	15	99,670	Covid19		81	99.990	Normal
	3591	100,000	Covid19		10188	99.930	Normal

	3594	100,000	Covid19		10189	99,910	Normal
	3596	100,000	Covid19		10190	99,500	Covid19
	3605	77,290	Covid19		10191	96,600	Normal
	3613	99.990	Covid19		10192	100.000	Normal

Los resultados anteriores, permitieron calcular el nivel de precisión del modelo al procesar imágenes totalmente nuevas, obtenidas de fuentes distintas a la seleccionada para el presente TT. Además, los resultados de la tabla se han sintetizado en la matriz de confusión de la **Fig. 41**.

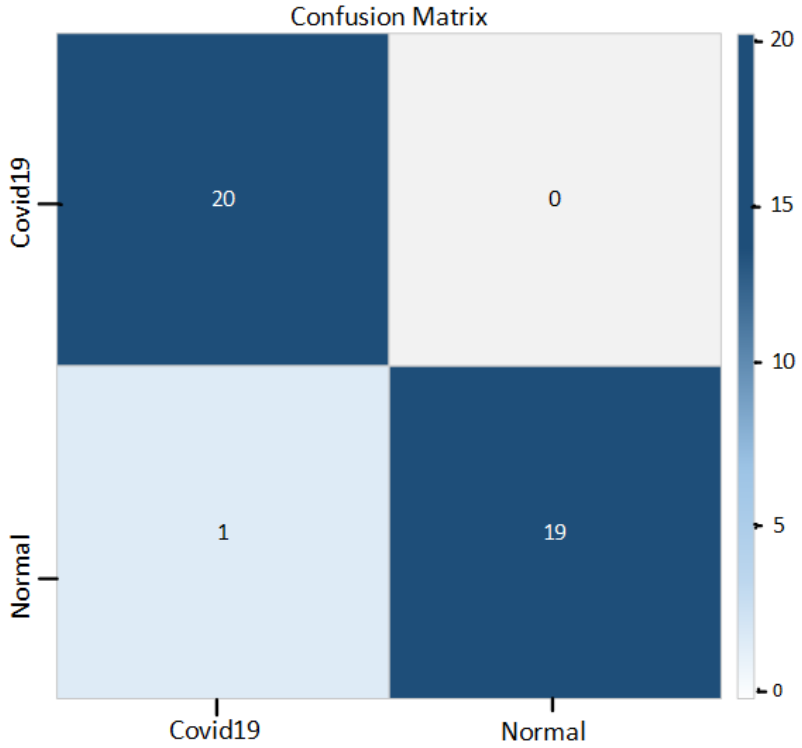


Fig. 41 *Matriz de confusión de los resultados obtenidos aplicando la técnica de ZSL.*

Con los datos expuestos en la matriz de confusión, se procedió con el cálculo de los parámetros, mencionados en secciones anteriores. Por consiguiente, el cálculo de la precisión, la sensibilidad y la especificidad (conforme a las ecuaciones de la sección 4.1) para ambas clases que conforman el modelo, se presentan en la **TABLA XIII**, y su desarrollo se muestra a continuación:

Covid **Normal**

Accuracy:

$$Accuracy = \frac{VP + VN}{VP + FP + FN + VN} = \frac{20 + 19}{20 + 0 + 1 + 19} = \frac{39}{40} = 0{,}97500$$

Precisión:

$$Presición = \frac{VP}{VP + FP}$$

$$Presición = \frac{20}{20 + 0} = 1{,}00000 \qquad Presición = \frac{19}{20 + 1} = 0{,}90476$$

Sensibilidad:

$$Sensibilidad = \frac{VP}{VP + FN}$$

$$Sensibilidad = \frac{20}{20 + 0} = 1{,}00000 \qquad Sensibilidad = \frac{19}{20 + 0} = 0{,}95000$$

Especificidad:

$$Especificidad = \frac{VN}{VN + FP}$$

$$Especificidad = \frac{19}{20 + 1} = 0{,}90476 \qquad Especificidad = \frac{20}{20 + 0} = 1{,}00000$$

TABLA XIII *Resultados de la evaluación del modelo usando la técnica de SZL.*

	Parámetros	Precisión	Sensibilidad	Especificidad	Accuracy	Support
ZSL	**Covid-19**	1,00000	1,00000	0,90476	0,97500	20
	Normal	0,90476	0,95000	1,00000		20
	Promedio ponderado	0,95238	0,97500	0,975000		40

Al obtener los valores de cada uno de los parámetros, se procedió a calcular una métrica llamada *F1 Score*, que permite evaluar en función de la precisión y la sensibilidad del modelo, la capacidad que posee para estimar resultados en base a sus clases. Conforme a este valor, mientras tienda a 1, mejor será la capacidad que posee el modelo sobre dicha clase, no obstante, mientras este valor tienda a 0, menor será su capacidad para manejar dicha clase. Obviamente, estas proporciones están relacionadas con la

precisión y sensibilidad del modelo, por lo que [48], ha establecido una serie de posibles resultados en base a las proporciones de los valores antes mencionados, estas proporciones son:

- **Alta precisión y alta sensibilidad**: el modelo está en la capacidad de manejar perfectamente esa clase.
- **Alta precisión y baja sensibilidad**: el modelo no detecta la clase de forma eficiente, pero cuando lo hace, es altamente confiable.
- **Baja precisión y alta sensibilidad**: el modelo detecta bien la clase, pero también incluye muestras de otras clases.
- **Baja precisión y baja sensibilidad**: el modelo no logra clasificar la clase correctamente.

En base a las cuatro proporciones anteriores, y basados en los resultados obtenidos en la **TABLA VIII**, el modelo "modelVGG16to6k", entrenado con 6000 radiografías, presenta un alto índice de precisión y sensibilidad, por lo que concuerda con el primer caso de los enunciados establecidos por [48], de modo que, nuestro modelo es capaz de clasificar satisfactoriamente los casos presentados por cada una de las clases (Normal y Covid-19), aun cuando el modelo interactúa con radiografías con las cuales no ha sido entrenado previamente. Para sustentar los argumentos que sustentan el presente TT, se obtuvo el valor de F1 Score, para lo cual se emplea la siguiente ecuación:

Ecuación 5

$$F1Score = 2\left(\frac{Sensibilidad * Presicion}{Sensibilidad + Presicion}\right)$$

Los resultados obtenidos con la Ecuación 5, y su desarrollo se presenta a continuación:

	Covid	**Normal**
F1 Score:		
	$Sensibilidad = 1{,}00000$	$Sensibilidad = 0{,}95000$
	$Precisión = 1{,}00000$	$Precisión = 0{,}90476$

$$F1\ Score = 2\left(\frac{1{,}00000\ *1{,}00000}{1{,}00000\ +1{,}00000}\right) \qquad F1\ Score = 2\left(\frac{0{,}95000\ *0{,}90476}{0{,}95000+0{,}90476}\right)$$

$$F1\ Score = 2\left(\frac{1{,}00000}{2{,}00000}\right) \qquad F1\ Score = 2\left(\frac{0{,}85952}{1{,}85476}\right)$$

$$F1\ Score = 2(0{,}50000) \qquad F1\ Score = 2(0{,}46223)$$

$$F1\ Score = 1{,}00000 \qquad F1\ Score = 1{,}92446$$

Los resultados obtenidos tras la aplicación de las pruebas de ZSL, se presentan en la **TABLA XIV**, la cual está compuesta por la precisión del modelo, la sensibilidad y la métrica de F1-Score, Estos parámetros se repiten por cada una de las clases que constituyen el modelo, además se presenta el promedio ponderado de todos los parámetros y su exactitud global.

TABLA XIV *Resultados de la fase de ZSL aplicado al modelo Diagnos19, entrenado con una carga total de 6000 radiografías.*

	Parámetros	Precisión	Sensibilidad	F1-score	Accuracy	Support
ZSL	**Covid-19**	1,00000	1,00000	1,00000	0,97500	20
	Normal	0,90476	0,95000	0,92446		20
	Promedio ponderado	0,95238	0,97500	0,96223		40

En conclusión, el modelo Diagnos19, presentó un alto nivel de precisión y sensibilidad, que corroboran la excelente capacidad que posee para clasificar imágenes con las cuales no ha trabajado anteriormente, siendo el valor de F1-score, la métrica que sustenta dichos resultados, con un promedio ponderado de 96,223% de eficiencia, distribuidas entre el 100% y 92,446% para las clases Covid-19 y Normal respectivamente, constituyendo así un nivel de clasificación totalmente confiable.

6.3.2. Aporte a la investigación.

Para mejorar el proceso de evaluación del modelo Diagnos19, y con ello cumplir con las tareas establecidas en el objetivo 3, se desarrolló una interfaz gráfica de usuario (IU), cuyo fin es facilitar la ejecución de las pruebas, tanto con la metodología Zero Shot Learning, concerniente a la primera actividad del presente objetivo, así como la de contrastar los resultados de nuestro modelo con el diagnóstico realizado por un experto en radiología, concerniente a la segunda actividad. Además, dicha IU, servirá para trabajos futuros relacionados con el presente TT.

Para el desarrollo de la interfaz, se tomó como punto de partida una plantilla de código abierto (open source), con un diseño orientado a la temática de Covid-19, para lo cual se utilizó la siguiente cadena de búsqueda:

'Free' AND 'Coronavirus' AND 'HTML' AND 'Template'

De los resultados obtenidos, se optó por la plantilla de html Desing, llamada Covido, misma que cuenta con una licencia Creative Commons 3.0 (para uso gratuito tanto personal y comercial), 3800 descargas, y es totalmente personalizable. Por lo que, se realizó modificaciones en su parte visual y en su parte estructural, adaptándola a un diseño sencillo e intuitivo, que permite cargar las imágenes médicas del paciente y realizar la valoración respectiva, mostrando como resultado el diagnóstico y la precisión con la que dicho diagnóstico fue emitido, véase la **Fig. 42**, donde se muestra la interfaz con los dos posibles resultados, diagnóstico negativo y diagnóstico positivo para Covid-19, respectivamente.

Fig. 42 *Interfaz gráfica de usuario, en la que se presentan los dos posibles diagnósticos que el modelo Diagnos19 puede arrojar.*

Cabe recalcar que para una mejor escalabilidad de la interfaz en trabajos futuros, se adaptó la plantilla anterior usando un framework de desarrollo llamado Django, con el que se creó una aplicación web, integrando la interfaz gráfica y el modelo desarrollado anteriormente (el código de esta conexión se encuentra en la sección 4 de Anexo 4 disponible en el CD 2), de tal forma que el usuario puede ingresar una radiografía usando la IU para su diagnóstico, dicha radiografía es enviada mediante un método POST a un servicio de alojamiento de imágenes llamada Cloudinary (ver sección 3 del Anexo 4, disponible en el CD 2), este servicio crea una URL pública que será enviada al modelo como dato de entrada. Una vez el modelo haya recibido el parámetro de entrada, procede a realizar todas las operaciones para convertir la radiografía entrante en tensores, que a su vez se convierte en parámetro de entrada para el modelo VGG16 entrenado en la sección 6.2.3. Este modelo junto con su peso, retorna un valor de predicción, mismo que se envía mediante un método POST a la UI donde el usuario podrá visualizar el diagnóstico.

El método POST usado para enviar el valor de la predicción, junto con su valor de precisión, se creó usando un framework de API Rest llamado FastAPI (ver sección 4 del Anexo 4, disponible en el CD 2), que a su vez utiliza el servidor Uvicorn (servidor provisto por FastAPI) para levantar el servicio de forma local, tal como se muestra en la **Fig. 43**.

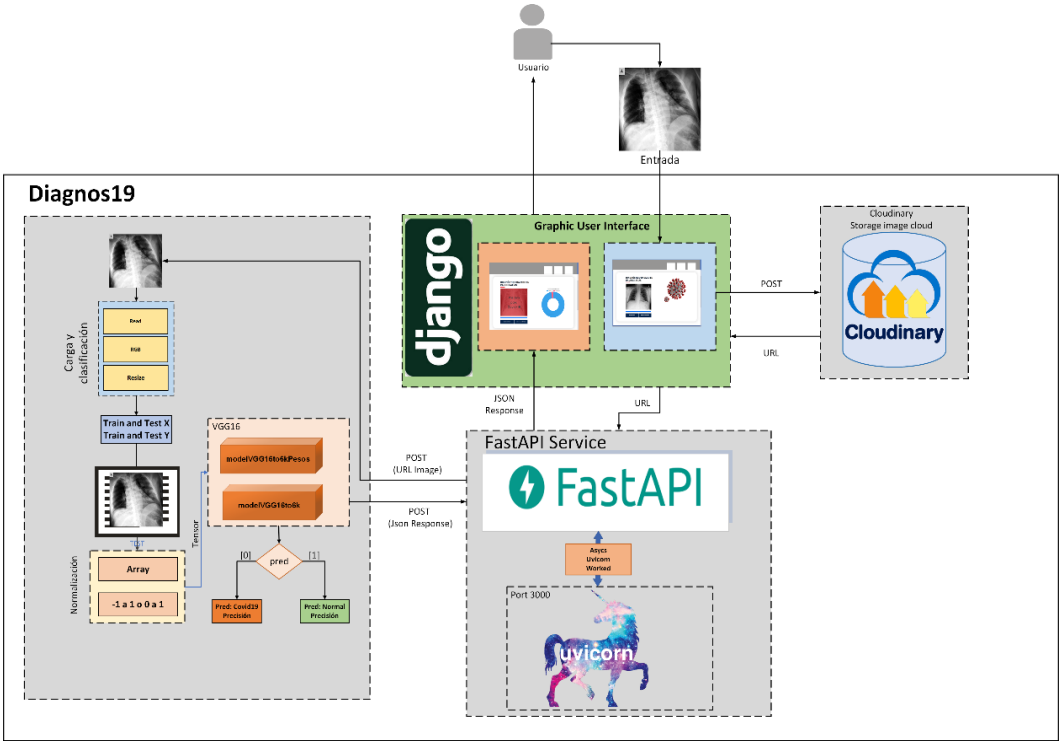


Fig. 43 *Esquema general del funcionamiento del modelo y la integración de la interfaz de Usuario*

Todo este proceso sumado a todos los requisitos y detalles más puntuales, se describen el README del repositorio Diagnos19[9], en el cual se encuentra alojado el proyecto, además, el código para la creación del método POST usando FastAPI, se encuentra en el CD 2, Anexo 4, sección 3.

Es importante mencionar, que esta IU será usada únicamente para facilitar la fase de evaluación del modelo, y no para un uso comercial o producción, puesto que no se realizado un estudio previo para la especificación de requisitos y la ingeniería que involucra la creación de un sistema destinado al uso profesional, no obstante, la IU desarrollada, cuenta con una estructura la cual ha permita implementar mejoras en caso de ser oportuno, y posiblemente se añadan funcionalidades extra según se especificó anteriormente.

6.3.3. Contrastar los resultados con un experto en radiografías.

Con la finalidad de comprobar la fiabilidad de la clasificación del modelo obtenido al aplicar la técnica de ZSL, se ha realizado una comparación entre los resultados obtenidos por nuestro modelo y el diagnóstico realizado por un especialista, el cual trabaja en el hospital básico de la ciudad de Amaluza, que por razones personales ha decidido permanecer en el anonimato.

En cuanto a las técnicas, criterios y conocimientos utilizados por el radiólogo para realizar el diagnóstico de pacientes a partir de radiografías pulmonares, en la entrevista realizada hacia su persona (ver Anexo 3), especificó que, el proceso de diagnóstico visual a partir de radiografías pulmonares, sobre todo en aquellas radiografías en las que exista una definición de baja calidad, las áreas de evaluación son poco visibles, por lo que se requiere de una vasta experiencia para diferenciar los patrones y actividad del virus en las zonas afectadas. Estas diferencias son sutiles entre una radiografía y otra, por lo tanto, la apreciación y reconocimiento de estas características dependen de la experiencia, paciencia y tiempo por parte de los radiólogos, lo cual resulta costoso en términos de trabajo, sobre todo en oleadas donde el número de contagios aumenta.

En cuanto al análisis empleado por el especialista, se basa en determinar aspectos característicos de una posible alteración pulmonar, evidentes en consolidaciones pulmonares como engrosamiento de septos interlobulillares y engrosamiento pleural, presentes en la opacidad en vidrio esmerilado. Debido al uso de términos técnicos, se ha creído necesario ampliar la descripción de cada uno de ellos para una mejor comprensión del diagnóstico emitido:

[9] Diagnos19

- **Consolidaciones pulmonares:** se refiere a la ocupación del espacio aéreo por productos patológicos como pus, agua, sangre, etc. La consolidación pulmonar aparece como un aumento homogéneo de la densidad (parenquimatosa pulmonar) que oculta los márgenes de los vasos y las paredes de las vías respiratorias, siendo visible en forma de luces bronquiales con aire en el seno de una opacidad parenquimatosa pulmonar (conocida como broncograma aéreo), lo cual implica la permeabilidad de las vías respiratorias [47].
- **Opacidad en vidrio esmerilado**: tal como se explicó en la sección 4.1.10, la opacidad en vidrio esmerilado describe la opacificación parenquimatosa pulmonar, producida por el aumento de la atenuación menor respecto a la consolidación, producto de la acción de propagación de un virus, en este caso SARS-CoV-2.

Tomando en cuenta los criterios que el radiólogo ha empleado para emitir un diagnóstico a partir del análisis de radiografías pulmonares, en la **TABLA XV**, se presentan los resultados obtenidos tras el análisis de veinte radiografías distribuidas en partes iguales entre Normal y Covid, dichas radiografías fueron entregadas al especialista de forma aleatoria para su posterior análisis y emisión de diagnóstico. De igual forma, las mismas radiografías fueron ingresadas al modelo Diagnos19 para contrastar ambos diagnósticos.

Las radiografías utilizadas para esta fase de prueba, se encuentran en una carpeta compartida de Google Drive[10], y fueron obtenidas de "TAWSIFUR RAHMAN" (aquellas que no fueron usadas en ninguna de las fases anteriores).

TABLA XV *Comparación de los resultados obtenidos por el modelo Diagnos19 con el diagnóstico emitido por el radiólogo.*

Clase	File	Diagnos19		Especialista	
		Accuracy	Diagnos19	Valoración	Diagnóstico
Covid-19	Radiografía2	99,860	Covid-19	Presencia de opacidad en vidrio esmerilado.	Covid-19
	Radiografía4	100,000	Covid-19	Presencia de opacidad en vidrio esmerilado.	Covid-19
	Radiografía6	100,000	Covid-19	Presencia de opacidad en vidrio esmerilado.	Covid-19
	Radiografía8	100,000	Covid-19	Posible caso de Neumonía	Normal

[10] Radiografías para pruebas con el especialista

	Radiografía10	100,000	Covid-19	Presencia de opacidad en vidrio esmerilado.	Covid-19
	Radiografía12	100,000	Covid-19	No se pude emitir un diagnóstico debido que la radiología esta distorsionada.	Normal
	Radiografía14	75,240	Covid-19	Presencia de opacidad en vidrio esmerilado.	Covid-19
	Radiografía16	100,000	Covid-19	Presencia de opacidad en vidrio esmerilado.	Covid-19
	Radiografía18	100,000	Covid-19	Presencia de opacidad en vidrio esmerilado.	Covid-19
	Radiografía20	100,000	Covid-19	Presencia de opacidad en vidrio esmerilado.	Covid-19
Normal	Radiografía1	98,460	Normal	Ausencia de opacidad en vidrio esmerilado.	Normal
	Radiografía3	77,710	Normal	Ausencia de opacidad en vidrio esmerilado.	Normal
	Radiografía5	97,320	Normal	Ausencia de opacidad en vidrio esmerilado.	Normal
	Radiografía7	77,710	Normal	Ausencia de opacidad en vidrio esmerilado.	Normal
	Radiografía9	77,710	Normal	Ausencia de opacidad en vidrio esmerilado.	Normal
	Radiografía11	99,880	Normal	Ausencia de opacidad en vidrio esmerilado.	Normal
	Radiografía13	77,710	Normal	Ausencia de opacidad en vidrio esmerilado.	Normal
	Radiografía15	99,110	Normal	Ausencia de opacidad en vidrio esmerilado.	Normal
	Radiografía17	99,990	Normal	Ausencia de opacidad en vidrio esmerilado.	Normal
	Radiografía19	73,430	Normal	Presencia de opacidad en vidrio esmerilado.	Covid-19

7. Discusión

Una vez finalizado el proceso de diseño y construcción del modelo basado en redes neuronales convolucionales (CNN) para el diagnóstico de covid-19 mediante el análisis de radiografías pulmonares, y haber sido sometido a un conjunto de pruebas que garantizaron su precisión de diagnóstico, se procede a dar contestación a la pregunta de investigación, la cual versa de la siguiente manera, ¿La creación de un modelo de visión por computadora basado en Redes Neuronales Convolucionales (CNN) para el análisis de radiografías pulmonares diagnosticará el Covid-19?. En base a los resultados obtenidos en los tres objetivos planteados, específicamente aquellos obtenidos en el objetivo número tres, se ratifica que la creación de un modelo basado en CNN para el análisis de radiografías, puede diagnosticar si un paciente es o no portador de Covid 19 con un nivel de precisión del 99.167%, si se utiliza como dato de entrada radiografías de buena resolución, y con 95,238% de precisión en radiografías de resolución baja, además, en base a la comparación con los resultados obtenidos en los TR, sobre el cual se basa este TT, el nivel de precisión obtenido por Diagnos19 es igual e incluso superior (en el mejor de los casos), a los modelos de los TR, considerando nuestra configuración y dataset con el cual el modelo resultante fue entrenado y evaluado. Como dato adicional, el diagnóstico obtenido con Diagnos19 fue verificado y validado con un especialista, al cual se superó en la precisión y en los tiempos empleados para el diagnóstico.

Toda la discusión de estos resultados se desglosa por cada uno de los objetivos planteados, de los cuales se detallan los puntos más importantes en cada proceso, sus limitaciones, mejoras y los aportes más significativos tras el desarrollo del presente TT, en relación a otros trabajos relacionados (TR).

7.1. Objetivo 1: Recopilar información de datasets que contengan imágenes de rayos x, que permitan entrenar y evaluar el modelo para el diagnóstico de Covid-19.

En lo que respecta al cumplimiento del objetivo uno, el cual consistió en recopilar información de los datasets que contengan radiografías pulmonares, fue necesario generar un protocolo de búsqueda con el fin de aumentar la calidad de la información, ya que existían varias fuentes, pero no todas ellas eran accesibles o simplemente, la calidad de sus datos no era óptima para maximizar la precisión del modelo a desarrollar. Es por ello que se optó por elegir algún conjunto de datos con el

cual los TR analizados hayan obtenido buenos resultados, no obstante, los trabajos relacionados provenientes de la RSL diferían en cuanto a sus fuentes, por lo que fue necesario establecer criterios de aceptación que permitieran optar por la mejor opción disponible a la fecha. Como resultado, se obtuvo un protocolo de búsqueda dividido en tres etapas, las cuales se detallan a continuación:

- En primer lugar, se realizó un análisis de cada uno de los TR, para seleccionar trabajos que hicieran referencia a algún datasets, determinando así las posibles fuentes de información secundarias a utilizar. Como resultado se obtuvieron doce TR, mismas que hacían referencia a una fuente de información con las características deseadas.
- En segundo lugar, de los doce resultados obtenidos previamente, se realizó una selección de las fuentes de información, basados en criterios de inclusión y exclusión establecidos sobre el tipo de acceso, tipo de contenido y el número mínimo de datos que cada fuente debía contener. Dando como resultado la preselección de cuatro de las doce fuentes disponibles.
- Finalmente, se eligió al dataset TAWSIFUR RAHMAN, la cual cumplió con todos los criterios de selección establecidos, siendo el dataset que más radiografías disponía para su uso y, además, muchas de las radiografías de las fuentes restantes se encontraban disponibles en este dataset, constituyendo una fuente de información diversa en cuanto a tipo de material y disponibilidad.

La implementación de este protocolo de búsqueda, para el cumplimiento del primer objetivo, permitió optar por el mejor conjunto de radiografías disponibles, considerando los parámetros de accesibilidad, tipo de contenido y el número de radiografías. Pese a que datasets como IEEE8023, fue utilizado en 8 de los 12 TR (TR03, TR05, TR13, TR15, TR22, TR24, TR32 y TR37) y cumplía con todos los requisitos al igual de TAWSIFUR RAHMAN (Kaggle), no fue utilizado en nuestro TT de forma directa, ya que gran parte de sus radiografías se encontraban dentro del dataset que TR40 utilizó. Además, optar por este conjunto de datos abierto, mismo que se encuentra en constante actualización y mejora dentro de la plataforma Kaggle, ayudó a minimizar el trabajo respecto al tratamiento de las radiografías, puesto que, internamente se encontraban clasificadas acorde al tipo de caso al que pertenecían, por consiguiente, bastó con establecer las cargas para cada una de las fases al momento de entrenar y evaluar el modelo.

7.2. Objetivo 2: Crear un modelo para diagnosticar Covid-19 utilizando la arquitectura de CNN.

Concerniente al cumplimiento del objetivo 2, de los 41 TR analizados, trece de ellos hacen referencia al uso de modelos preentrenados con un nivel de precisión que oscila, entre el 83.3% y el 98.5% en el mejor de los casos (véase la **TABLA IV**). De los trece TR, la precisión fue establecida como un criterio de aceptación para los modelos, con la finalidad de optar por el mejor entre todos ellos, por tal razón, se estableció un valor mínimo del 90% de precisión para seleccionar los modelos preentrenados y realizar un análisis minucioso de aquellos que cumplieran con dicho criterio. Como resultado de este análisis, se concluyó que el modelo VGG-16, con un nivel de precisión de 98.5%, fue el que mejores especificaciones presentó, frente a los modelos ResNet-50, con 95.6% e Inception-v3 con 90% respectivamente, además el uso de recursos de este modelo (RAM, CPU, GPU) era similar a los recursos que se encontraban disponibles, sobre todo en cuanto a drivers y librerías que permiten el uso de la GPU (aceleración por hardware). Una vez se definió el modelo base, se procedió con la fase de codificación, detallada con precisión durante el desarrollo del objetivo 2 (véase la sección 6.2.2), no obstante, uno de los principales aspectos a recalcar, es la configuración de los hiperparámetros, Learning Rate (LR), epochs y banch size (tasa de aprendizaje, épocas y tamaño de lote, respectivamente) ya que de esta configuración dependía en gran parte la precisión de clasificación del modelo, por tanto determinar la mejor configuración posible un fue verdadero reto en el desarrollo de este objetivo, debido a que todos los TR usados como referencia utilizaban diferentes configuraciones, por ejemplo, TR12 y TR01, establecían que el uso de los hiperparámetros debía ser establecido de acuerdo los recursos disponibles y el tipo de arquitectura a utilizar, por tal razón, se establecieron un total de 9 pruebas con 9 variaciones distintas de los hiperparámetros, todo esto con el fin de optar por la mejor configuración posible. Dichas pruebas fueron llevadas a cabo con 500 radiografías, distribuidas en partes iguales para cada una de las clases, de tal forma que fuese posible utilizar la mejor configuración en la fase de entrenamiento. En la **Fig. 44**, se presentan los resultados obtenidos por cada una de las configuraciones de los hiperparámetros, siendo la configuración 5 (INIT_LR = $3e^{-4}$, EPOCHS=50, BS=64) la configuración que mejores resultados obtuvo.

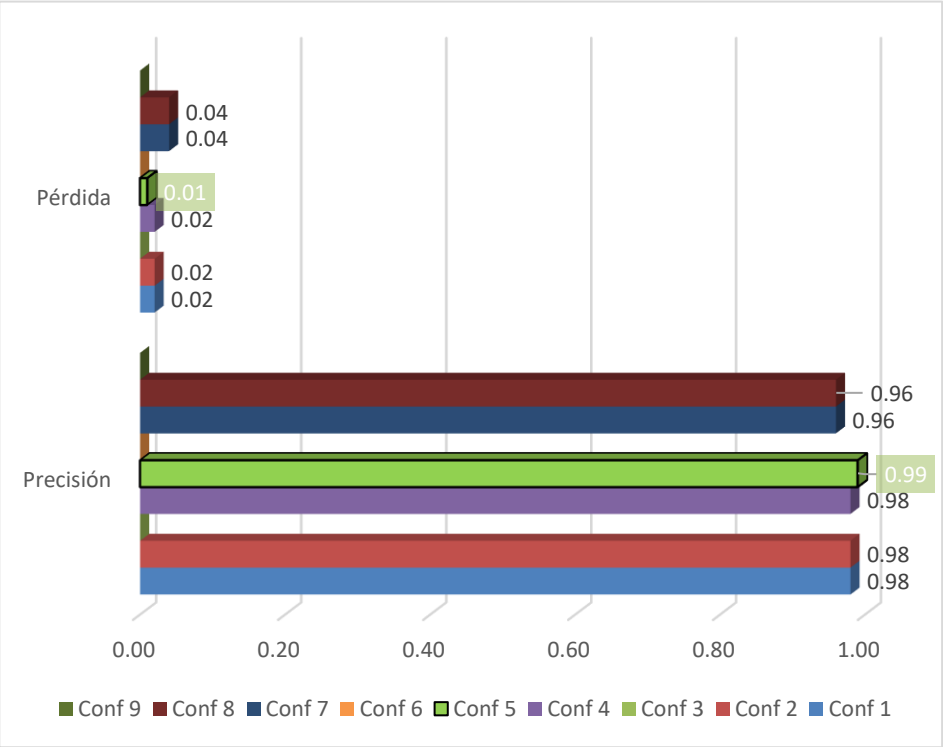


Fig. 44 *Gráfica de barras en la que se presentan los valores de precisión y pérdida en el proceso de selección de los hiperparámetros.*

Posteriormente, tras haber seleccionado la configuración de los hiperparámetros con los mejores resultados, se procedió a entrenar el modelo conforme a las cargas establecidas (1k, 2k, 4k, 6k y 7k), cabe recalcar que el uso de la GPU para el desarrollo de esta fase fue crucial, puesto que se logró reducir drásticamente los tiempos y el uso de recursos durante esta fase, tal como se muestra en la gráfica superior de la **Fig. 45**, en la que se aprecia claramente que el modelo entrenado sin usar la GPU (modelVGG16to3kCPU) tardo aproximadamente 5099 segundos o 1h 24m 59 segundos por época, este tiempo multiplicado por 50, que es el número total de épocas que conforman el modelo, da como resultado una cantidad abrupta de tiempo equivalente a 254954 segundos o lo que es lo mismo, 70h 49m 14s.

Mientras que, con el uso de la GPU, tal como se muestra en la gráfica inferior de la **Fig. 45**, tomando como referencia el modelo “modelVGG16to7kGPU”, entrenado con 7000 radiografías, tarda 8250 segundos o lo que es lo mismo 2h 17m 30s. No hace falta enfatizar la diferencia y eficiencia de la GPU al momento de trabajar con modelos que requieren el procesamiento de grandes cantidades de imágenes, ya que existe una relación directamente proporcional, a más imágenes más recursos utiliza y por ende, más tiempo requiere para completar la fase de entrenamiento, aunque el valor de BS también influye en acortar el tiempo empleado en la fase de entrenamiento, no obstante,

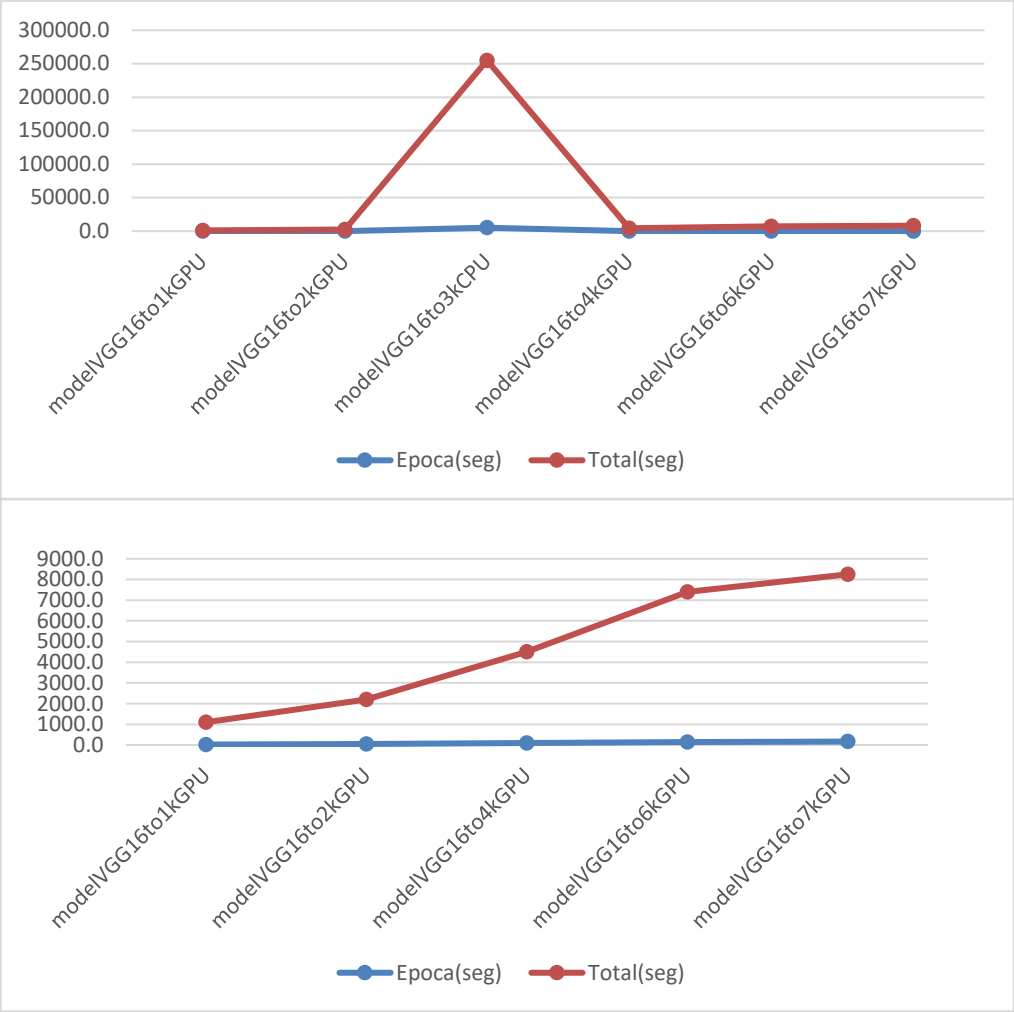


Fig. 45 *Gráfica en la que se representa el tiempo aproximado de entrenamiento para cada uno de los modelos.*

en nuestro caso, se ha limitado el a 64 (BS = 64) como máximo, ya que al superar este valor, la capacidad de procesamiento computacional disponible fue sobrepasado, y por tal razón, en la **Fig. 44**, las configuraciones 1, 4 y 7 con BS=128 y EPOCH>50, no muestran ningún resultado, por lo que si se desea reducir aún más el tiempo de entrenamiento se recomienda utilizar más potencia en hardware, sobre todo en la RAM, que es la que se suele desbordar.

Finalmente, los resultados de la fase de entrenamiento por cada uno de los modelos se muestran en la **Fig. 46**, donde destaca la configuración del modelo “modelVGG16to6k”, entrenada con 6000 radiografías. Estos valores hacen referencia a los resultados obtenidos en la fase de test que el propio modelo requiere como parámetro, siendo la precisión, sensibilidad y especificidad, los parámetros que determinan la calidad del modelo al momento de clasificar las imágenes y, además, sirven como punto de comparación frente a otros modelos desarrollados y presentados en los TR. Esta

comparación, ha permitido formular nuestras propias conclusiones y dar respuesta a la pregunta de investigación sobre el cual se ha basado el presente TT, por consiguiente, en la **TABLA XVI**, se presentan los resultados obtenidos por nuestro modelo frente a los conseguidos en los TR.

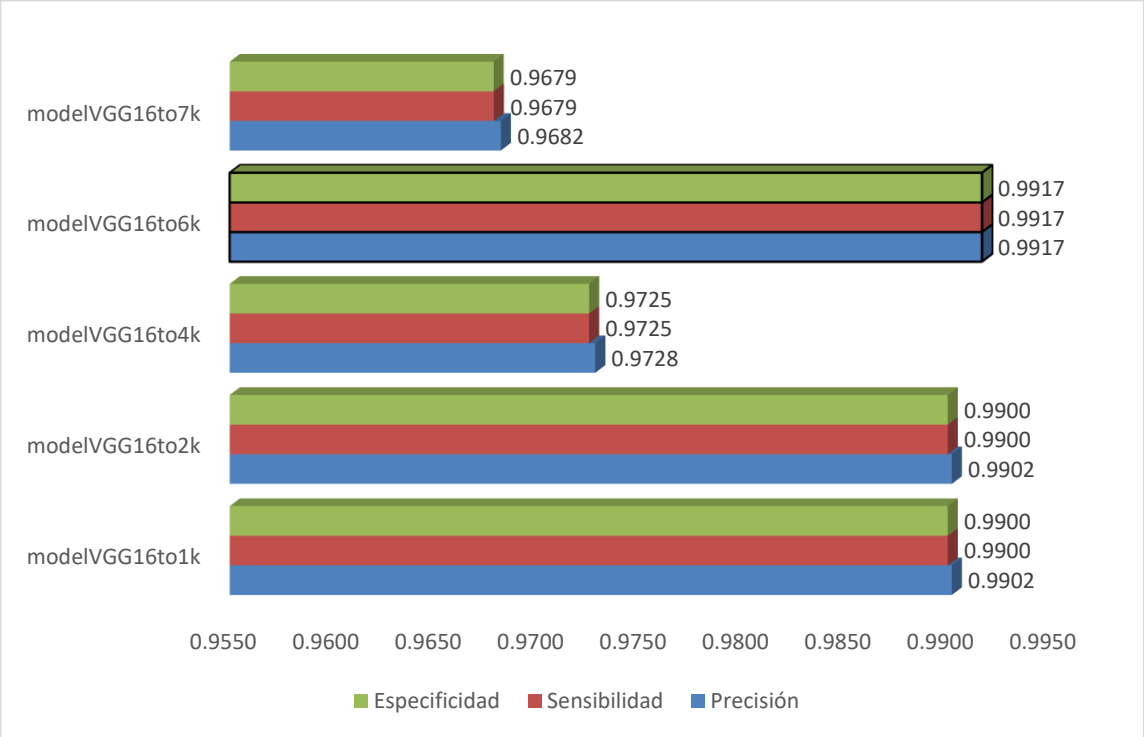


Fig. 46 *Gráfica de barras de los resultados obtenidos en la fase de entrenamiento*

TABLA XVI *Comparación de los resultados obtenidos por nuestro modelo "Diagnos19", y los resultados obtenidos en los TR utilizados en la sección 6.2.1 para la selección del modelo base, correspondiente al cumplimiento del segundo objetivo.*

Estudio	Tipo de imagen	Número de clases	Método usado	Precisión (%)	Sensibilidad (%)	Especificidad (%)
TR01	RX	100 Covid-19 100 Normal	VGG-16	**97.5**	92.3	98.8
TR02	TC	120 Covid-19 120 Normal	COVID19-Net	**87.5**	91.7	86.4
TR07	TC	174 Covid-19 174 Normal	Lasso	**83.9**	86.8	73.3
TR14	RX	172 Covid-19 172 Normal	VGG-16	**98.27**	98.93	80.5
TR16	RX	310 COVID-19(+)	VGG-16+CCBlock	**95.51**	98.47	98.98

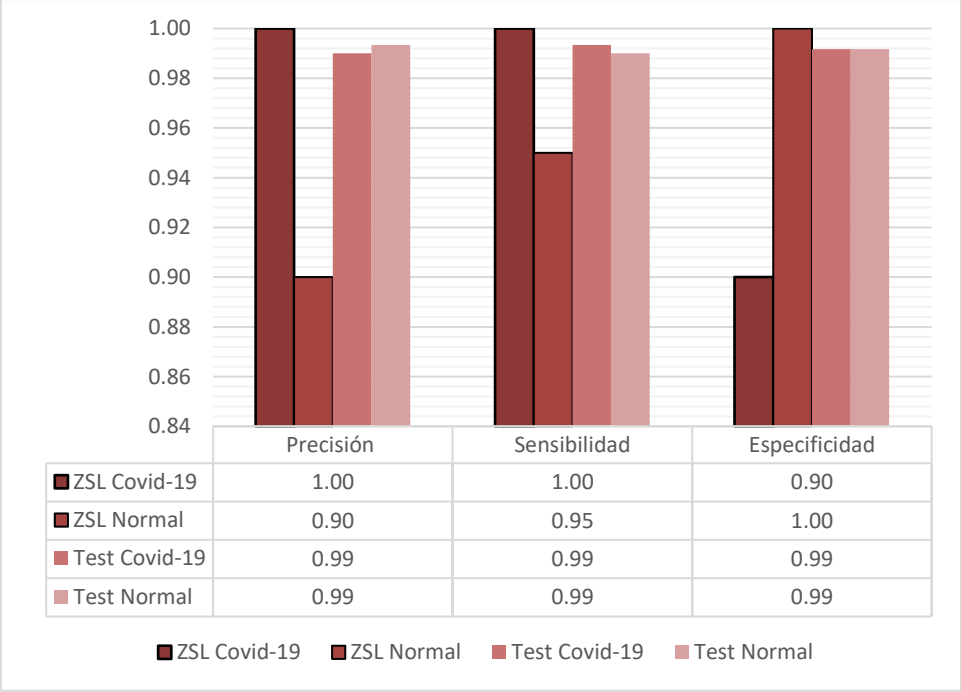
		654 Healthy 864 Pneumonia (virus &bacteria)				
TR25	TC	25 moderate 47 severe 27 critical	U-Net	**83.3**	0.812	0.854
TR26	TC	1551 Covid-19 1883 Normal	ResNet-50	**95.6**	84.4	93.3
TR29	TC	448 Covid-19 308 Normal	V-Net	**85.5**	86.6	93.2
TR40	RX	81 Covid-19 114 viral pneumonia 114 healthy	Inception-V3	**90.0**	87.7	68.0
TT Diagnos19	RX	**600 Covid-19 600 Normal**	**VGG-16**	**99.17**	**99.17**	**99.17**

Tras comparar los resultados obtenidos, se puede afirmar con bases sólidas que el nivel de precisión establecido en la pregunta de investigación equivalente al 90 % fue superado, de igual forma, se logró superar los niveles de precisión alcanzados por aquellos modelos basados en la arquitectura VGG-16 de los trabajos relacionados TR01 y TR14. En cuanto a los modelos basados en otro tipo de arquitectura, sea o no que utilicen radiografías pulmonares como datos de entrada, también fueron superados, aunque cabe recalcar, que nuestro modelo está diseñado para reconocer casos positivos de Covid19, sea cual fuere el estado de progresión y desarrollo de la enfermedad, y descartar cualquier otro tipo de afección que no sea provocada por el virus SARS-CoV-2, ya que Diagnos19, se pensó para ser utilizado como soporte de las pruebas de Reacción en Cadena de la Polimerasa (PCR) y PCR en tiempo real (RT-PCR), mismas que son usadas por el sistema de salud para diagnosticar Covid-19, que dicho sea de paso, superando el umbral de precisión de este tipo de pruebas, el cual oscila entre el 70% y 80%, en el mejor de los casos, según TR09, TR33 y TR35.

7.3. Objetivo 3: Evaluar la precisión y sensibilidad del modelo utilizando datos de prueba.

Finalmente, una de las mayores limitantes para el cumplimiento del objetivo número 3, fue determinar los criterios de evaluación para el modelo, ya que al tratarse de un modelo destinado para uso en la salud, se debía corroborar que los resultados obtenidos por el modelo, fuesen lo más precisos posibles, a la vez que abarcase la mayoría de los escenarios posibles, razón por la cual se optó por utilizar la técnica de Zero Shot Learning (SZL) misma que se subdividió en dos fases con radiografías de resolución variable, y una prueba aún más desafiante, comparar los resultados de nuestro modelo con el diagnóstico realizado por un especialista.

Con el fin de agilizar el proceso de evaluación del modelo con ambas técnicas, se desarrolló un interfaz de usuario (IU, véase la sección Aporte a la investigación.), de tal forma que se visualizase el resultado y fuese amigable tanto para los involucrados en el desarrollo del presente TT, como para los que hagan uso del mismo. Retomando lo mencionado anteriormente, las pruebas de ZSL permitieron medir la eficiencia del modelo al clasificar imágenes con las cuales no ha interactuado en la fase de entrenamiento. Ahora bien, los resultados obtenidos fueron se compararon con los resultados de la fase de test del modelo, tal como se muestran en la **Fig. 47**.



	Precisión	Sensibilidad	Especificidad
ZSL Covid-19	1.00	1.00	0.90
ZSL Normal	0.90	0.95	1.00
Test Covid-19	0.99	0.99	0.99
Test Normal	0.99	0.99	0.99

Fig. 47 *Gráfica que representa los valores de precisión, sensibilidad y especificidad obtenidos por el modelo Diagnos19*

En base a la gráfica, se puede afirmar que el modelo es capaz de clasificar correctamente imágenes de la clase Covid-19, y descartar adecuadamente aquellas radiografías que no presenten ningún tipo de anomalías provocadas por el virus SARS-CoV-2, no obstante, tiende a emitir falsos positivos con mayor frecuencia si procesa una imagen totalmente nueva, sobre todo si la calidad de la mismo no es óptima. Algo similar ocurre al momento de identificar los casos que verdaderamente corresponden a la clase Covid-19, donde los datos obtenidos con las pruebas ZSL superan a los obtenidos en la fase de test, aunque no sea el caso de la clase normal, que se ve superada en ambos parámetros por los valores obtenidos en la fase de evaluación del modelo

Por tal razón, es posible afirmar que Diagnos19 obtuvo mejores resultados globales al momento de tratar con imágenes pertenecientes al mismo dataset con el cual fue entrenado (ZSL fase 1), aunque los resultados obtenidos mediante las pruebas de ZSL con radiografías de resolución variable (ZSL fase 2), se ubica en 0,04 puntos por debajo de los niveles de precisión y especificidad, y tan solo el 0,01 por debajo del valor de precisión obtenidas en la fase de test (véase la **Fig. 48**). Acorde con estos resultados, se puede afirmar que, pese a tratar con imágenes con las cuales nunca ha sido entrenado, Diagnos19 se mantiene muy por encima de los valores mínimos establecidos, superando así, el nivel de precisión de las pruebas tradicionales, lo que permitió dar respuesta a la pregunta de investigación.

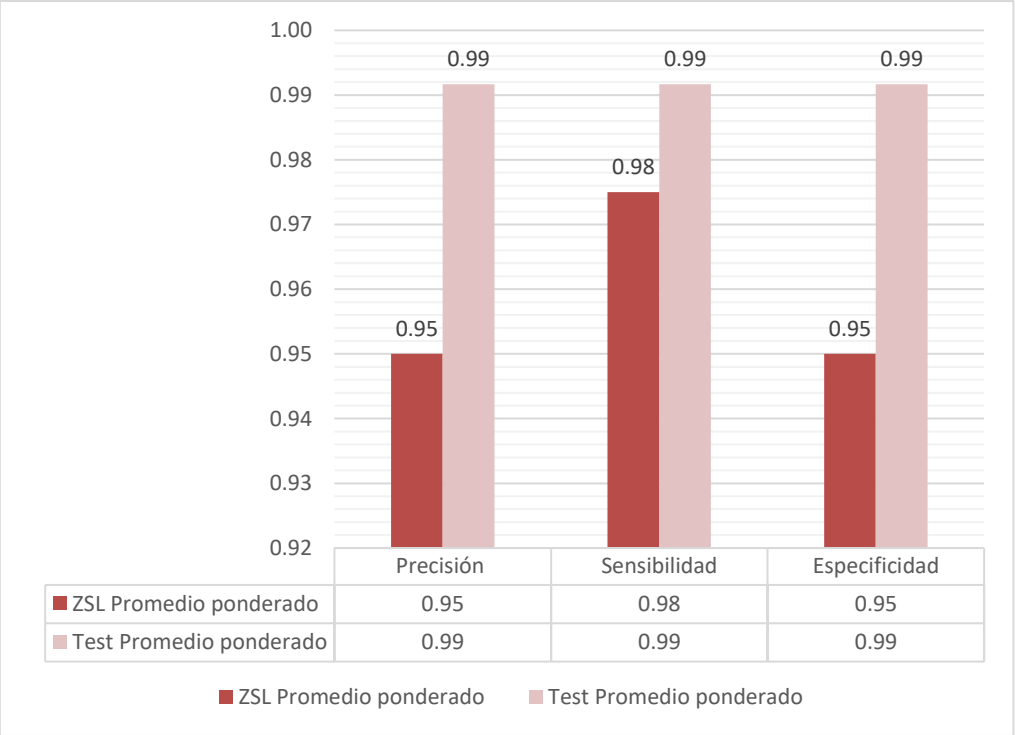


Fig. 48 *Gráfica que representa los resultado de los pormedios ponderados de presición, sensibilidad y especificidad obtenidos por el modelo Diagnos19.*

Por último, tras realizar las pruebas con el especialista, lo más llamativo fue comprobar la velocidad con la cual el experto humano realizaba cada uno de los diagnósticos, y cotejarlos con el tiempo que Diagnos19 tarda en cargar, procesar y emitir el resultado del diagnóstico. Esta comparación permitió determinar la viabilidad del modelo frente al proceso que un humano (experto) realiza para emitir el diagnóstico de Covid-19 a partir del análisis de radiografías pulmonares, y si este es o no capaz de minimizar la carga laboral del profesional de la salud, el cual, en jornadas donde existe un elevado número de contagios, es propenso a reducir su eficiencia laboral. Los resultados de los tiempos empleados en proceso de diagnóstico tanto por el radiólogo y el modelo se presentan en la **Fig. 49**, donde el tiempo promedio empleado por el profesional para realizar el diagnóstico es de 224 segundos (3m 44s). En cambio, el tiempo promedio empleado por Diagnos19 es de 5s, comprobando así, que el modelo es 40% más rápido que un radiólogo, con un nivel de experiencia considerable, ahora bien, este valor podría ser mucho mayor para los casos en el que el radiólogo se encuentre en proceso de formación.

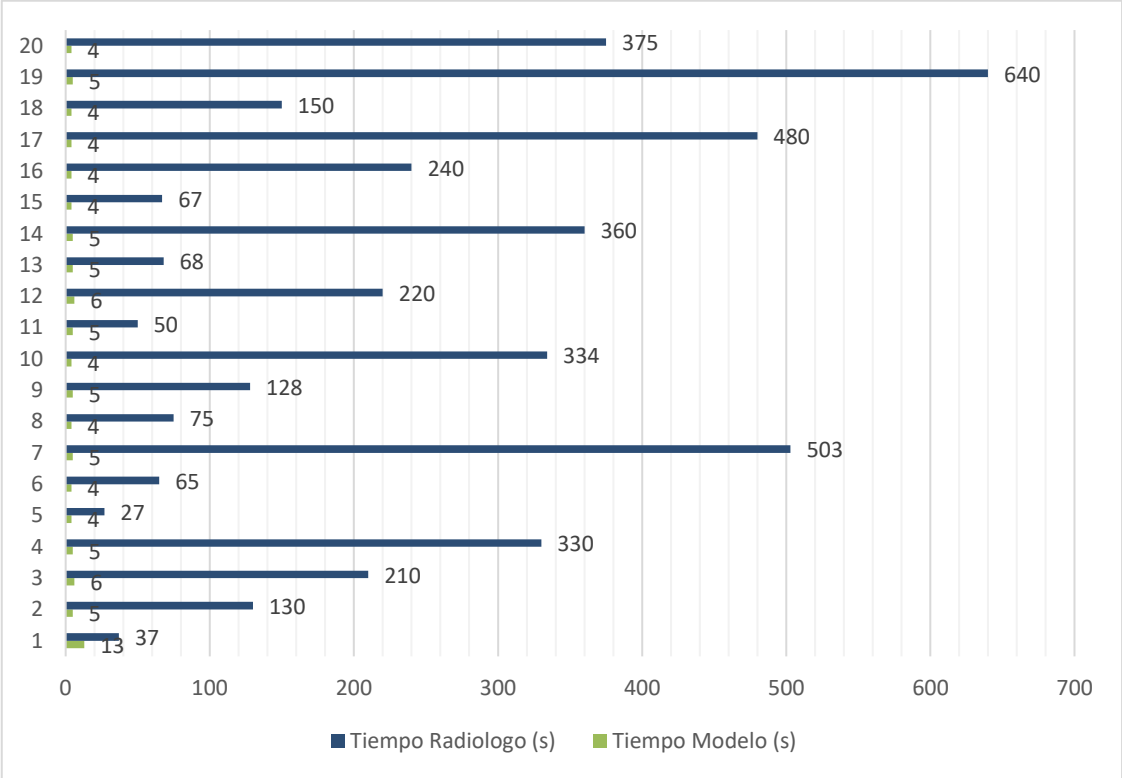


Fig. 49 *Gráfica de barras que representan el tiempo en segundos empleado por el modelo (verde) y el radiólogo (azul)para evaluar y emitir un diagnóstico*

En cuanto a la precisión, en la **Fig. 50**, se grafican los resultados obtenidos por ambas partes, siendo la línea de color rojo, el valor al cual pertenece cada una de las radiografías, por lo que los diagnósticos emitidos correctamente deben seguir la trayectoria de la línea que representa la “Clase” de la radiografía, si la línea difiere de dicha trayectoria, los casos corresponden a falsos negativos o falsos positivos dependiendo del diagnóstico. Acorde a los resultados obtenidos, nuestro modelo presenta un mayor nivel de precisión, tanto para diagnosticar Covid19 como para descartarlo, siendo su promedio de 100% de precisión y sensibilidad para cada una de las clases (línea verde). En cambio, el radiólogo acertó en 17 de las 20 radiografías, lo que equivale al 66.7% de precisión para diagnosticar Covid-19 y 81.8% para diagnosticar casos en los que no existe presencia del virus, como resultado final el radiólogo alcanzó un nivel de precisión global del 73.9%.

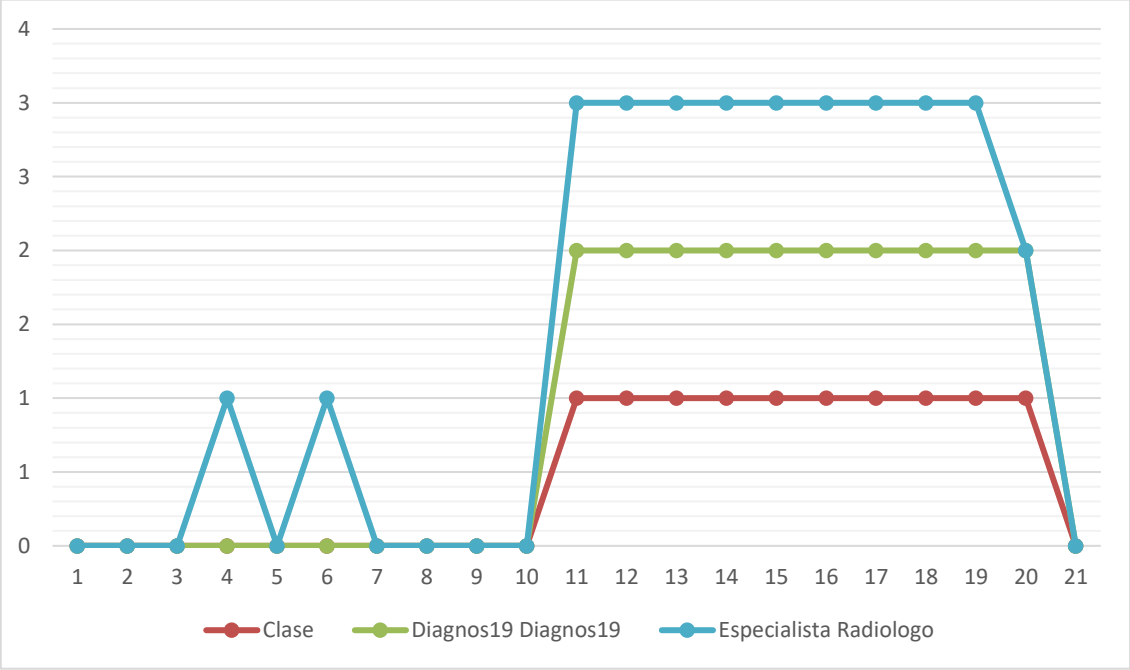


Fig. 50 *Gráfica que representa el diagnóstico emitido por el radiólogo (azul) contrate con el diagnóstico emitido por obtenido por Diagnos19 (verde). Ambos resultados se comparan con el valor real de la clase para cada radiografía analizada (Covid = 0 y Normal = 1)*

Por consiguiente, Diagnos19, superó tanto las pruebas de ZSL y como el nivel de precisión del resultado con un experto humano, acortando significativamente los tiempos de diagnóstico y de igual forma mantuvo un alto nivel de precisión, en ambos casos sobrepasó el 95% de precisión global, ratificando en base a resultados su alto nivel de precisión y confiabilidad.

7.4. Valoración técnica, económica, ambiental y social

7.4.1. Valoración técnica

El presente TT, se valora técnicamente a razón de las múltiples herramientas de hardware y software que fueron utilizadas en la construcción del modelo para el Diagnóstico de Covid-19 Mediante el Análisis de Radiografías Pulmonares Empleando un Modelo Basado en Redes Neuronales Convolucionales (CNN). Librerías como TensorFlow, Scikit-Learn, NumPy, Nvidia CUDA Toolkit, desempeñaron un papel fundamental en la construcción y ajuste fino (Fine Tunning) de los hiperparámetros del modelo preentrenado VGG-16, y la construcción del modelo final denominado Diagnos19. El desarrollo del modelo junto con la interfaz gráfica de usuario (GUI), supuso un enorme esfuerzo y dedicación por parte de los involucrados en el mismo, pues se ha creado un ecosistema en que no solo existe el modelo como tal, sino que la interfaz de usuario, junto con la creación del API Rest que permite su conexión y comunicación con el modelo, supone la aplicación de varias ramas de conocimientos ligados a la ingeniería en sistemas. Así mismo, la preparación del entorno virtual que contiene las herramientas y librerías, y además es donde se despliegan ambos componentes, supone una minuciosa preparación puesto que de ello dependerá el funcionamiento del modelo y su GUI.

7.4.2. Valoración económica

Para la ejecución del presente TT, fueron necesarios ciertos recursos económicos, mismos que se detallan en las **Tabla XVII**, **TABLA XVIII** y **TABLA XIX**:

Tabla XVII *Recursos para talento humano.*

Talento Humano			
Responsable	**Número de horas**	**Costo por hora**	**Costo total**
Tesista	400	$2.50	$1 000.00
Director	40	$10.48	$419.00
Radiólogo	5	$65	$325
TOTAL			**$1 744.00**

TABLA XVIII Recursos técnicos y tecnológicos.

Recursos Técnicos y Tecnológicos			
Recursos de Software			
Nombre		**Costo total**	
Microsoft Office			$0.00
Mendeley Desktop			$0.00
Google Chrome			$0.00
Conda			$0.00
Cuda Toolkit			$0.00
TensorFlow			$0.00
Jupyter Lab			$0.00
Pycharm			$0.00
Python			$0.00
Firma electrónica			$34.00
SUBTOTAL			**$34.00**
Recursos de Hardware			
Nombre	**Cantidad**	**Costo unitario**	**Costo total**
Laptop	1	$2 500.00	$2 500.00
SUBTOTAL			**$2 500.00**
TOTAL			**$2 534.40**

TABLA XIX Recursos para servicios.

Servicios			
Nombre	**Meses**	**Costo unitario**	**Costo total**
Internet	5	$23.43	$117.15
TOTAL			**$117.00**

De acuerdo a todos los recursos económicos anteriormente presentados, se genera la **TABLA XX**, en la que se presenta la sumatoria de cada uno de ellos, con el valor total de los gastos que llevó realizar el presente TT.

TABLA XX Totalidad de los recursos económicos.

Presupuesto General	
Descripción	**Costo total**
Talento Humano	$1 744.00
Recursos Técnicos y Tecnológicos	$2 534.40
Servicios	$117.00
TOTAL	**$4 395.00**

7.4.3. Valoración ambiental

El presente TT, se realizó en su totalidad con recursos tecnológicos y digitales que no tienen un mayor impacto al medio ambiente, además se limitó a lo más mínimo es uso de materiales como papel, plásticos u otro tipo de materiales y elementos que puedan llegar a perjudicar al planeta, inclusive se limitó al de radiografías digitales para reducir el uso de material médico, así mismo, con la posible implementación de Diagnos19 como método alternativo de diagnóstico, será posible reducir el uso de material médico empleado para llevar a cabo las pruebas de diagnóstico tradicional como las pruebas PCR Y RT-PCR.

8. Conclusiones

De acuerdo al trabajo de titulación realizado, se concluye lo siguiente:

- El uso de la arquitectura VGG-16, perteneciente a las redes neuronales convolucionales (CNN), empleada para la construcción de un modelo capas de diagnosticar Covid-19 mediante el análisis de radiografías pulmonares, garantizó un nivel de precisión (Accuracy) del 99.167%, superando así, el valor establecido en la pregunta de investigación y el nivel de precisión de las pruebas PCR.

- El análisis de los trabajos relacionados, permitió crear un protocolo de búsqueda eficiente, mismo que, tras el análisis exhaustivo de sus resultados, se logró obtener un dataset completo, variado y con radiografías pulmonares clasificadas listas para ser utilizadas en el desarrollo del modelo, siendo de gran importancia, no solo por el dataset como tal, sino también ha permitido conocer el estado del arte referente a los modelos usados en cada uno de estos TR, mismos que se usaron para la selección del modelo base y el contraste de los resultados obtenidos.

- La creación de un modelo basado en redes neuronales convolucionales (CNN) para el diagnóstico de Covid-19 utilizando la metodología Fine tuning, permitió obtener resultados los cuales superaron el umbral impuesto en la pregunta problema e incluso, superó los resultados de los modelos implementados en los trabajos relacionados. Por lo tanto, el uso de modelos preentrenados mediante el ajuste de sus hiperparámetros, se puede considerar seriamente como un soporte para médicos clínicos y suplir la falta de insumos médicos para el diagnóstico de Covid-19 y con ello reducir el número de contagios

- En base a los resultados obtenidos en las pruebas realizadas al modelo, se pudo concluir que, diagnos19, está en la capacidad de sustituir las pruebas PCR y RT-PCR, logrando reducir en un 15% el número de falsos negativos, debido a que los resultados obtenidos por el modelo, fueron validados y contrastados por un profesional de la salud, y superó con un nivel de precisión superiores al 95% las pruebas de ZSL.

9. Recomendaciones

De acuerdo a las diferentes observaciones, contratiempos y experiencias obtenidas durante el desarrollo del presente TT, se recomienda lo siguiente:

- Se recomienda utilizar modelos preentrenados para agilizar la construcción del modelo final. Estos modelos difieren en la arquitectura y el tipo de datos con los cuales trabaja, por lo que es indispensable utilizar el modelo acorde al objetivo que se requiera.
- Se recomienda usar la GPU (Graphics Processing Unit) para la fase de entrenamiento, de tal forma que el tiempo de entrenamiento se reduzca y además se utilice de mejor manera los recursos del computador. En caso de no disponer de la GPU, usar los servicios de cloud computing como Google Colab o Azure.
- Se recomienda realizar pruebas con las diferentes combinaciones entre los hiperparámetros (INIT_LR, BS, EPONCH) de tal forma que se elija en base a resultados, la mejor configuración para la fase de entrenamiento. Obteniendo así, un modelo fiable el cual garantice los mejores resultados en la fase de test y uso posterior.
- Se recomienda utilizar un conjunto variado de las fuentes para la obtención de radiografías e imágenes médicas, de ser posible, usar radiografías de resolución y calidad variable, de tal forma que se abarque todos los posibles casos en los cuales el modelo pueda ser utilizado.

9.1. Trabajos futuros

Como trabajos futuros, se determinaron 3 aspectos clave, los cuales mejorarán tanto la eficiencia del diagnóstico y ampliarán su usabilidad en lo que respecta al uso en el ámbito profesional:

- Realizar configuraciones a nivel de código para que el modelo sea capaz de reconocer varias enfermedades que afectan al tracto respiratorio como, neumonía, tuberculosis, influenza etc. y así obtener un sistema completo para el diagnóstico de enfermedades respiratorias a partir del análisis de radiografías pulmonares.
- Mejorar el modelo para que sea capaz de procesar tomografías computacionales TC, como imágenes médicas de entrada al modelo, ya que varios trabajos relacionados tales como TR02, RT07, TR25, TR26 y TR29, aseguran que en las TC de alta resolución es posible reconocer anomalías como OGG (opacidad en vidrio esmerilado) mucho antes que las radiografías pulmonares, por lo que sería posible una detección más oportuna de Covid-19.
- Por último, mejorar la interfaz de usuario, de tal forma que se pueda liberar para el uso de las instituciones involucradas en salvaguardar la bioseguridad de la población, como hospitales, clínicas y centros de salud. Así mismo añadir una sección informativa y de contacto la cual se pueda usar en una posible implementación de una aplicación móvil, de tal forma que sea posible un contacto directo con clínicas y profesionales de la salud dispuestos a ofrecer sus servicios a los usuarios de la App.

10. Bibliografía

[1] H. Yao *et al.*, “Severity Detection for the Coronavirus Disease 2019 (COVID-19) Patients Using a Machine Learning Model Based on the Blood and Urine Tests,” *Front. Cell Dev. Biol.*, vol. 8, 2020, doi: 10.3389/fcell.2020.00683.

[2] I. E. Agbehadji, B. O. Awuzie, A. B. Ngowi, and R. C. Millham, “Review of big data analytics, artificial intelligence and nature-inspired computing models towards accurate detection of COVID-19 pandemic cases and contact tracing,” *Int. J. Environ. Res. Public Health*, vol. 17, no. 15, pp. 1–16, 2020, doi: 10.3390/ijerph17155330.

[3] A. M. Sahan, A. S. Al-Itbi, and J. S. Hameed, “COVID-19 detection based on deep learning and artificial bee colony,” vol. 9, no. 1, pp. 29–36, 2021.

[4] M. Kavitha, T. Jayasankar, P. Maheswara Venkatesh, G. Mani, C. Bharatiraja, and B. Twala, “COVID-19 Disease Diagnosis using Smart Deep Learning Techniques,” *J. Appl. Sci. Eng.*, vol. 24, no. 3, pp. 271–277, doi: 10.6180/jase.202106_24(3).0001.

[5] A. Al-Bawi, K. Ali Al-Kaabi, M. Jeryo, and A. Al-Fatlawi, “CCBlock: An Effective Use of Deep Learning for Automatic Diagnosis of COVID-19 Using X-Ray Images.”

[6] H. S. Maghded, K. Z. Ghafoor, A. S. Sadiq, K. Curran, D. B. Rawat, and K. Rabie, “A Novel AI-enabled Framework to Diagnose Coronavirus COVID-19 using Smartphone Embedded Sensors: Design Study,” in *Proceedings - 2020 IEEE 21st International Conference on Information Reuse and Integration for Data Science, IRI 2020*, Aug. 2020, pp. 180–187, doi: 10.1109/IRI49571.2020.00033.

[7] Google-Noticias, “Coronavirus (COVID-19).” https://bit.ly/3mI3Izq

[8] OMS, “Enfermedad por el coronavirus (COVID-19): Vacunas,” *Onu*, 2021. https://bit.ly/3eDyeGp

[9] S. D. S. Dass, F. Meskaran, and M. Saeedi, “Expert system for early diagnosis of covid-19,” *Int. J. Curr. Res. Rev.*, vol. 12, no. 22, pp. 162–165, Nov. 2020, doi: 10.31782/IJCRR.2020.122227.

[10] J. Born *et al.*, “Accelerating detection of lung pathologies with explainable ultrasound image analysis,” *Appl. Sci.*, vol. 11, no. 2, pp. 1–23, Jan. 2021, doi: 10.3390/app11020672.

[11] Y. Zoabi, S. Deri-Rozov, and N. Shomron, “Machine learning-based prediction of COVID-19 diagnosis based on symptoms,” *npj Digit. Med.*, vol. 4, no. 1, Dec. 2021, doi: 10.1038/s41746-020-00372-6.

[12] C. Chakraborty and A. Abougreen, “Intelligent Internet of Things and Advanced Machine Learning Techniques for COVID-19,” *EAI Endorsed Trans. Pervasive Heal. Technol.*, p. 168505, Jul. 2018, doi: 10.4108/eai.28-1-2021.168505.

[13] S. D. S. Dass, F. Meskaran, and M. Saeedi, “Expert system for early diagnosis of covid-19,” *Int. J. Curr. Res. Rev.*, vol. 12, no. 22, pp. 162–165, 2020, doi: 10.31782/IJCRR.2020.122227.

[14] K. Purohit, A. Kesarwani, D. R. Kisku, and M. Dalui, “COVID-19 detection on chest X-Ray and CT Scan images using multi-image augmented deep learning model,” *bioRxiv*. bioRxiv, Jul. 17, 2020, doi: 10.1101/2020.07.15.205567.

[15] M. Y. Kamil, “A deep learning framework to detect Covid-19 disease via chest X-ray and CT scan images,” *Int. J. Electr. Comput. Eng.*, vol. 11, no. 1, pp. 844–850, Feb. 2021, doi: 10.11591/ijece.v11i1.pp844-850.

[16] S. D. Thepade, S. V. Bang, P. R. Chaudhari, and M. R. Dindorkar, “Covid19 Identification from Chest X-ray Images Using Machine Learning Classifiers with GLCM Features,” *Electron. Lett. Comput. Vis. Image Anal.*, vol. 19, no. 3, pp. 85–97, 2020, doi: 10.5565/REV/ELCVIA.1277.

[17] H. Yao *et al.*, “Severity Detection for the Coronavirus Disease 2019 (COVID-19) Patients Using a Machine Learning Model Based on the Blood and Urine Tests,” *Front. Cell Dev. Biol.*, vol. 8, no. 10, pp. 2776–2786, 2020, doi: 10.3389/fcell.2020.00683.

[18] S. Chattopadhyay, A. Dey, P. K. Singh, Z. W. Geem, and R. Sarkar, “Covid-19 Detection by Optimizing Deep Residual Features with Improved Clustering-Based Golden Ratio Optimizer,” *Diagnostics*, vol. 11, no. 2, p. 315, 2021, doi: 10.3390/diagnostics11020315.

[19] J. D. Arias-Londoño, J. A. Gomez-Garcia, L. Moro-Velazquez, and J. I. Godino-Llorente, “Artificial Intelligence applied to chest X-Ray images for the automatic detection of COVID-19. A thoughtful evaluation approach,” Nov. 2020, [Online]. Available: http://arxiv.org/abs/2011.14259.

[20] M. A. Mohammed *et al.*, “Benchmarking Methodology for Selection of Optimal COVID-19 Diagnostic Model Based on Entropy and TOPSIS Methods,” *IEEE Access*, vol. 8, pp. 99115–99131, 2020, doi: 10.1109/ACCESS.2020.2995597.

[21] S. D. Thepade *et al.*, “Covid19 Identification from Chest X-ray Images Using Machine Learning Classifiers with GLCM Features,” *Electron. Lett. Comput. Vis. Image Anal.*, vol. 19, no. 3, pp. 85–97, 2020, doi: 10.5565/REV/ELCVIA.1277.

[22] M. Taresh, N. Zhu, and T. A. Ali Ali, “Transfer learning to detect COVID-19

automatically from X-ray images, using convolutional neural networks," *medRxiv*. medRxiv, Nov. 07, 2020, doi: 10.1101/2020.08.25.20182170.

[23] O. M. Elzeki, M. Abd Elfattah, H. Salem, A. E. Hassanien, and M. Shams, "A novel perceptual two layer image fusion using deep learning for imbalanced COVID-19 dataset," *PeerJ Comput. Sci.*, vol. 7, p. e364, Feb. 2021, doi: 10.7717/peerj-cs.364.

[24] T. Gao, "Chest X-ray image analysis and classification for COVID-19 pneumonia detection using deep CNN," *medRxiv*. medRxiv, Oct. 12, 2020, doi: 10.1101/2020.08.20.20178913.

[25] P. K. Sethy, S. K. Behera, K. Anitha, C. Pandey, and M. R. Khan, "Computer aid screening of COVID-19 using X-ray and CT scan images: An inner comparison," *J. Xray. Sci. Technol.*, pp. 1–14, Jan. 2021, doi: 10.3233/xst-200784.

[26] Y. KUTLU and Y. CAMGÖZLÜ, "Detection of coronavirus disease (COVID-19) from X-ray images using deep convolutional neural networks," *Nat. Eng. Sci.*, vol. 6, no. 1, pp. 60–74, 2021, doi: 10.28978/nesciences.868087.

[27] M. I. Malik, "CLOUD COMPUTING-TECHNOLOGIES," *Int. J. Adv. Res. Comput. Sci.*, vol. 9, no. 2, pp. 379–384, Apr. 2018, doi: 10.26483/ijarcs.v9i2.5760.

[28] Iaar, "Visión por computadora - Libro online de IAAR," 2020. https://bit.ly/32El8pP (accessed Mar. 23, 2021).

[29] M. G. Arnal and I. D. S. Audiovisuales, "Estudio y aplicación de las redes neuronales convolucionales 3D," 2018.

[30] E. Castillo and A. S. Hadi, "Sistemas Expertos y Modelos de Redes Probabilísticas," p. 639, 1997.

[31] J. M. Pignani, "Sistemas Expertos (Expert System) Orientación I : Informática aplicada a la ingeniería de procesos 1," *Univ. Tecnológica Nac. Fac. Reg. Rosario*, 2011.

[32] J. Li *et al.*, "Multi-task contrastive learning for automatic CT and X-ray diagnosis of COVID-19," *Pattern Recognit.*, vol. 114, p. 107848, Jun. 2021, doi: 10.1016/j.patcog.2021.107848.

[33] J. I. Bagnato, "Convolutional Neural Networks: La Teoría explicada en Español," 2018. https://bit.ly/3JrQH6U (accessed Jun. 26, 2021).

[34] J. M. Díaz, "Artificial intelligence and big data as solutions to COVID-19 [Intel·ligència artificial i big data per fer front a la COVID-19] [Inteligencia artificial y big data como soluciones frente a la COVID-19]," *Rev. Bioet. y Derecho*, no. 50, pp. 315–331, 2020, [Online]. Available:

https://www.scopus.com/inward/record.uri?eid=2-s2.0-85095446979&partnerID=40&md5=5640a8a2af709fde14d2037195f5b3b2.

[35] F. Díaz and A. Toro, “SARS-CoV-2/covid-19: el virus, la enfermedad y la pandemia.,” *Med. y Lab.*, vol. 24, no. 3, pp. 183–205, 2020.

[36] M. de S. P. del Ecuador, “Coronavirus COVID-19 – Ministerio de Salud Pública,” 2020. https://www.salud.gob.ec/coronavirus-covid-19/ (accessed Mar. 21, 2021).

[37] MedlinePlus, “Síndrome respiratorio de Oriente Medio (MERS),” 2020. https://bit.ly/3eyF3c3 (accessed Mar. 21, 2021).

[38] J. María and M. García, “SARS-CoV-2,” 2020.

[39] D. P. Fan *et al.*, “Inf-Net: Automatic COVID-19 Lung Infection Segmentation from CT Images,” *IEEE Trans. Med. Imaging*, vol. 39, no. 8, pp. 2626–2637, Aug. 2020, doi: 10.1109/TMI.2020.2996645.

[40] L. Meng *et al.*, “A Deep Learning Prognosis Model Help Alert for COVID-19 Patients at High-Risk of Death: A Multi-Center Study,” *IEEE J. Biomed. Heal. Informatics*, vol. 24, no. 12, pp. 3576–3584, Dec. 2020, doi: 10.1109/JBHI.2020.3034296.

[41] P. Meseguer and R. L. de M. Badia, *¿Qué sabemos de? Inteligencia Artificial*. 2017.

[42] A. González, “¿Qué es Machine Learning? – Cleverdata,” *Cleverdata*. https://bit.ly/3eCdRJy (accessed Mar. 21, 2021).

[43] Isciii, “Pruebas de diagnóstico del coronavirus: ¿qué es la PCR?, ¿qué son los test rápidos? ¿en qué se diferencian?,” 2020. https://bit.ly/3pFXt1e (accessed Mar. 21, 2021).

[44] O. I. de E. A. OIEA, “Detección del virus de la COVID-19 mediante la RT-PCR en tiempo real,” 2020. https://bit.ly/32OeMUM (accessed Mar. 21, 2021).

[45] L. N. Smith, “Cyclical learning rates for training neural networks,” in *Proceedings - 2017 IEEE Winter Conference on Applications of Computer Vision, WACV 2017*, 2017, no. April, pp. 464–472, doi: 10.1109/WACV.2017.58.

[46] I. N. del Cáncer, “Definición de radiografía del tórax - Diccionario de cáncer del NCI.” https://bit.ly/3pG8Yp2 (accessed Aug. 29, 2021).

[47] R. Sánchez-Oro, J. Torres Nuez, and G. Martínez-Sanz, “Radiological findings for diagnosis of SARS-CoV-2 pneumonia (COVID-19),” *Med. Clin. (Barc).*, vol. 155, no. 1, pp. 36–40, 2020, doi: 10.1016/j.medcli.2020.03.004.

[48] J. I. Barrios Arce, “La matriz de confusión y sus métricas – Inteligencia Artificial – ,” pp. 1–8, 2019, Accessed: Sep. 19, 2021. [Online]. Available:

https://bit.ly/3qCV32q.

[49] A. Mordvintsev, “OpenCV: Introducción a los tutoriales de OpenCV-Python,” 2013. https://bit.ly/3EG4dQT (accessed Jul. 02, 2021).

[50] OpenCV:, “OpenCV: Conversiones de espacio de color.” https://bit.ly/3mIlcfa (accessed Jul. 05, 2021).

[51] opencv dev team, “Geometric Transformations of Images,” 2014. https://bit.ly/3HnC42F (accessed Jul. 05, 2021).

[52] K. Bonawitz *et al.*, “TensorFlow Federated,” 2020. https://www.tensorflow.org/federated (accessed Jul. 05, 2021).

[53] Tensorflow, “TensorFlow.” https://bit.ly/3qxpKGs (accessed Jul. 06, 2021).

[54] G. Colab, “Intro_a_tensorflow.ipynb - Colaboratory.” https://bit.ly/3ze7V2T (accessed Jul. 09, 2021).

[55] T. Core, “Keras.” https://www.tensorflow.org/guide/keras?hl=es (accessed Jul. 11, 2021).

[56] Keras, “Keras: the Python deep learning API,” *Keras: the Python deep learning API*, 2020. https://keras.io/ (accessed Jul. 12, 2021).

[57] Scikit-learn, “scikit-learn: machine learning in Python — scikit-learn 0.24.2 documentation.” https://scikit-learn.org/stable/ (accessed Jul. 12, 2021).

[58] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.*, pp. 1–14, 2015.

[59] A. Mahendran and A. Vedaldi, “Understanding deep image representations by inverting them,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 07-12-June, pp. 5188–5196, Oct. 2015.

[60] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-Decem, pp. 770–778, 2016, doi: 10.1109/CVPR.2016.90.

[61] MathWorks, “ResNet-50 convolutional neural network - MATLAB resnet50 - MathWorks América Latina,” *MathWorks*, 2020. https://bit.ly/3pBXGSX (accessed Jul. 22, 2021).

[62] Keras, “ResNet y ResNetV2.” https://bit.ly/3sJcwJ7 (accessed Jul. 22, 2021).

[63] G. Cloud, “Guía avanzada de Inception v3 para Cloud TPU,” *Google*, 2020. https://bit.ly/3sIoToU (accessed Jul. 22, 2021).

[64] MathWorks, “Red neuronal convolucional inception-v3 - MATLAB inceptionv3 - MathWorks América Latina.” https://bit.ly/33Yl98o (accessed Jul. 23, 2021).

[65] S. El-bana, A. Al-Kabbany, and M. Sharkas, “A multi-task pipeline with specialized streams for classification and segmentation of infection manifestations in COVID-19 scans,” *PeerJ Comput. Sci.*, vol. 6, p. e303, Oct. 2020, doi: 10.7717/peerj-cs.303.

[66] M. Nour, Z. Cömert, and K. Polat, “A Novel Medical Diagnosis model for COVID-19 infection detection based on Deep Features and Bayesian Optimization,” *Appl. Soft Comput.*, vol. 97, Dec. 2020, doi: 10.1016/j.asoc.2020.106580.

[67] J. Qiu *et al.*, “A Radiomics Signature to Quantitatively Analyze COVID-19-Infected Pulmonary Lesions,” *Interdiscip. Sci. Comput. Life Sci.*, 2021, doi: 10.1007/s12539-020-00410-7.

[68] S. Gazzah, O. Bencharef, and F. Marrakech, “A Survey on how computer vision can response to urgent need to contribute in COVID-19 pandemics,” 2020. [Online]. Available: https://github.com/prajnasb/observations/tree/master/experie.

[69] D. M. H. Nguyen, D. M. Nguyen, H. Vu, B. T. Nguyen, F. Nunnari, and D. Sonntag, “An Attention Mechanism with Multiple Knowledge Sources for COVID-19 Detection from CT Images,” Sep. 2020, [Online]. Available: http://arxiv.org/abs/2009.11008.

[70] A. S. A. S. Adly, A. S. A. S. Adly, and M. S. Adly, “Approaches Based on artificial intelligence and the internet of intelligent things to prevent the spread of COVID-19: Scoping review,” *J. Med. Internet Res.*, vol. 22, no. 8, Aug. 2020, doi: 10.2196/19104.

[71] M. Jamshidi *et al.*, “Artificial Intelligence and COVID-19: Deep Learning Approaches for Diagnosis and Treatment,” *IEEE Access*, vol. 8, pp. 109581–109595, 2020, doi: 10.1109/ACCESS.2020.3001973.

[72] W. Alsharif and A. Qurashi, “Effectiveness of COVID-19 diagnosis and management tools: A review,” *Radiography*, 2020, doi: 10.1016/j.radi.2020.09.010.

[73] M. A. Mohammed *et al.*, “Benchmarking Methodology for Selection of Optimal COVID-19 Diagnostic Model Based on Entropy and TOPSIS Methods,” *IEEE Access*, vol. 8, pp. 99115–99131, 2020, doi: 10.1109/ACCESS.2020.2995597.

[74] S. Yazdani, S. Minaee, R. Kafieh, N. Saeedizadeh, and M. Sonka, “COVID CT-Net: Predicting Covid-19 from chest CT images using attentional convolutional network,” *arXiv*, 2020.

[75] K. V Kavitha, S. R. Deshpande, A. P. Pandit, and A. G. Unnikrishnan, “Application of tele-podiatry in diabetic foot management: A series of illustrative cases,”

Diabetes Metab. Syndr. Clin. Res. Rev., vol. 14, no. 6, pp. 1991–1995, 2020, doi: 10.1016/j.dsx.2020.10.009.

[76] W. Cai *et al.*, “CT Quantification and Machine-learning Models for Assessment of Disease Severity and Prognosis of COVID-19 Patients,” *Acad. Radiol.*, vol. 27, no. 12, pp. 1665–1678, 2020, doi: 10.1016/j.acra.2020.09.004.

[77] D. Javor, H. Kaplan, A. Kaplan, S. B. Puchner, C. Krestan, and P. Baltzer, “Deep learning analysis provides accurate COVID-19 diagnosis on chest computed tomography,” *Eur. J. Radiol.*, vol. 133, Dec. 2020, doi: 10.1016/j.ejrad.2020.109402.

[78] M. Ghaderzadeh and F. Asadi, “Deep Learning in Detection and Diagnosis of Covid-19 using Radiology Modalities: A Systematic Review.”

[79] H. Kang *et al.*, “Diagnosis of Coronavirus Disease 2019 (COVID-19) with Structured Latent Multi-View Representation Learning,” *IEEE Trans. Med. Imaging*, vol. 39, no. 8, pp. 2606–2614, 2020, doi: 10.1109/TMI.2020.2992546.

[80] V. A. de Freitas Barbosa *et al.*, “Heg.IA: an intelligent system to support diagnosis of Covid-19 based on blood tests,” *Res. Biomed. Eng.*, 2021, doi: 10.1007/s42600-020-00112-5.

[81] J. Gisby *et al.*, “Longitudinal proteomic profiling of high-risk patients with COVID-19 reveals markers of severity and predictors of fatal disease,” *medRxiv*, vol. 16, no. 2. medRxiv, p. e0247176, Nov. 06, 2020, doi: 10.1101/2020.11.05.20223289.

[82] D. Silahudin, Henderi, and A. Holidin, “Model expert system for diagnosis of COVID-19 using naïve bayes classifier,” in *IOP Conference Series: Materials Science and Engineering*, Dec. 2020, vol. 1007, no. 1, doi: 10.1088/1757-899X/1007/1/012067.

[83] M. D. Li *et al.*, “Multi-Radiologist User Study for Artificial Intelligence-Guided Grading of COVID-19 Lung Disease Severity on Chest Radiographs,” *Acad. Radiol.*, 2021, doi: 10.1016/j.acra.2021.01.016.

[84] D. Zhang, X. Liu, M. Shao, Y. Sun, Q. Lian, and H. Zhang, “The value of artificial intelligence and imaging diagnosis in the fight against COVID-19,” *Pers. Ubiquitous Comput.*, 2021, doi: 10.1007/s00779-021-01522-7.

[85] W. T. Li *et al.*, “Using machine learning of clinical data to diagnose COVID-19: A systematic review and meta-analysis,” *BMC Med. Inform. Decis. Mak.*, vol. 20, no. 1, Sep. 2020, doi: 10.1186/s12911-020-01266-z.

[86] Y. Castán, “Metodo Cientifico y Sus Etapas,” *Castán, Yolanda. Inst. Aragon. Ciencias La Salud*, vol. 2, pp. 1–6, 2006.

[87] I. B. Gutiérrez, “Aportes de la investigación cualitativa a enfermería,” *Investig. en Enfermería Imagen y Desarro.*, vol. 7, no. 1–2, pp. 8–13, 2005.

[88] Escuela de graduandos y educación continua, “El Metodo De Casos,” *Esc. Grad. y Educ. Contin.*, 2016, [Online]. Available: https://bit.ly/31bkjUX

[89] B. Kitchenham and S. Charters, “Guidelines for performing Systematic Literature eviews in Software Engineering,” 2007.

[90] L. Díaz-Bravo, U. Torruco-García, M. Martínez-Hernández, and M. Varela-Ruiz, “La entrevista, recurso flexible y dinámico,” *Investig. en Educ. Médica*, vol. 2, no. 7, pp. 162–167, 2013, doi: 10.1016/s2007-5057(13)72706-6.

[91] A. Rosebrock, “Ajuste con Keras y Deep Learning - PyImageSearch,” 2019. https://bit.ly/3EG1kQ7 (accessed Jun. 11, 2021).

[92] Y. Xian, C. H. Lampert, B. Schiele, and Z. Akata, “Zero-shot learning-a comprehensive evaluation of the good, the bad and the ugly,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 9, pp. 2251–2265, 2019, doi: 10.1109/TPAMI.2018.2857768.

[93] Python, “Python.” https://www.python.org/ (accessed Jun. 27, 2021).

[94] Anaconda, “The World’s Most Popular Data Science Platform,” *Anaconda*, 2021. https://www.anaconda.com/ (accessed Jun. 27, 2021).

[95] Project Jupyter, “Project Jupyter,” 2019. https://jupyter.org/ (accessed Jun. 29, 2021).

[96] S. R. de A. del N. RSNA, “RSNA Pneumonia Detection Challenge,” *Kaggle*, 2018. https://bit.ly/3mIn0oe (accessed Jun. 18, 2021).

[97] T. Rahman, D. M. Chowdhury, and D. M. Chowdhury, “COVID-19 Radiography Database,” *Kaggle*, 2020. https://bit.ly/3mIf0E6 (accessed Jun. 18, 2021).

[98] “La API funcional.” https://bit.ly/3pBIYeK (accessed Sep. 11, 2021).

[99] M. Petticrew and H. Roberts, *Systematic Reviews in the Social Sciences: A Practical Guide*. Blackwell Publishing Ltd, 2008.

[100] V. A. de Freitas Barbosa *et al.*, “Heg.IA: an intelligent system to support diagnosis of Covid-19 based on blood tests,” *Res. Biomed. Eng.*, 2021, doi: 10.1007/s42600-020-00112-5.

[101] O. M. de la S. OMS, “Zoonosis,” 2020. https://bit.ly/32FwlGC (accessed Oct. 06, 2021).

[102] L. J. Aguilar and Ignacio Zahonero Martínez, *Programación en C, C++, Java y UML*, 2nd ed. México, 1369.

[103] “Ajuste de hiperparámetros de un modelo - Azure Machine Learning | Microsoft

Docs.” https://bit.ly/3mJC5pF (accessed Aug. 21, 2021).

[104] RealPython, “Entornos virtuales de Python: una introducción,” 2018. https://bit.ly/3EJKLCx (accessed Jun. 27, 2021).

11. Anexos

Anexo 1. Revisión sistemática de literatura

La presente RSL ha sido desarrollada en diferentes secciones como: la sección de Metodología, donde se han definido las fases que propone Barbara Kitchenham para realizar revisiones de literatura, posteriormente, se ha detallado el proceso de cada una de estas fases. De igual forma, en la sección de Resultados, se detallan las salidas de los procesos aplicados anteriormente, resaltando sobre todo los trabajos relacionados, los cuales se analizan, explican e interpretan de forma más detallada en la sección de Discusión, y finalmente, se plantean las Conclusiones obtenidas durante la realización de la presente RSL.

1. Materiales y métodos

La realización de la revisión sistemática de literatura (RSL), se basa en el proceso establecido en la metodología de Bárbara Kitchenham [89], dicho proceso consta de tres fases fundamentales:

- Planificar la revisión
    - Especificar preguntas de investigación
    - Desarrollar protocolo de revisión
    - Validad protocolo de revisión
    - Conducir la revisión
- Identificar fuentes/estudios relevantes
    - Seleccionar estudios primaries
    - Evaluar la calidad de los estudios
    - Extraer datos requeridos
    - Sintetizar datos
- Documentar la revisión
    - Escribir informe de revisión
    - Validar informe

Aunque algunas de las tareas de estas fases no son obligatorias, como, por ejemplo:

- Es opcional la puesta en marcha de una RSL, ya que depende de la revisión sistemática que se está haciendo sobre una base comercial.
- Evaluar el protocolo de revisión y la evaluación del informe, son opcionales

y dependen de los procedimientos de garantía de calidad decididas por el encargado de la revisión sistemática.

En vista de los puntos anteriormente expuestos, en la presente RSL, no se ejecutaron todos los pasos propuestos por Kitchenham, por consiguiente, se aprovechó la flexibilidad de la metodología en cuanto a la extensión que brinda al investigador, por tanto, esta depende de la necesidad y el alcance que el investigador requiera, y crea conveniente para la obtención de resultados y posterior publicación. De acuerdo a esto, en la **TABLA XXI**, se presentan las fases y tareas que fueron consideradas:

TABLA XXI *Proceso de RSL basado en la metodología de Bárbara Kitchenham.*

Fases	Tareas
Planificación de la revisión	Identificar la necesidad de una revisión.
	Especificar preguntas de investigación.
	Desarrollar protocolo de revisión.
Conducción de la revisión	Identificar fuentes/estudios relevantes.
	Seleccionar estudios primarios.
	Sintetizar datos
Revisión de informes	Escribir informe de revisión.

2. Resultados

2.1. Planificación de la revisión

a) Identificación de la necesidad de una revisión.

Con la presente RSL, se busca determinar las técnicas y métodos utilizados para la detección de Covid-19 (SARS-CoV-2), de otras enfermedades con sintomatología similar tales como, la gripe y la neumonía pulmonar, mediante la aplicación de técnicas de aprendizaje automático o Machine Learning y de esta manera, determinar la viabilidad de diseño y creación de un sistema capaz de diagnosticar, con un nivel de eficiencia aceptable, si el paciente (usuario) es portador de una de las enfermedades antes mencionadas. Todo esto con la finalidad de optar por el método más eficiente, tomando en cuenta los recursos disponibles y, sobre todo, basarnos en trabajos los cuales han guiado la delimitación de nuestra investigación.

b) Especificación de las preguntas de investigación.

Como un mecanismo para guiar el desarrollo de la RSL, se ha planteado una serie de preguntas de investigación, de tal forma que el presente trabajo se enfoque en dar respuesta a estas preguntas, manteniendo siempre la misma perspectiva y enfoque, estas preguntas se presentan en la **TABLA XXII**.

TABLA XXII *Preguntas de investigación para la RSL*

Código	Preguntas de investigación
P1	¿Cuáles son los enfoques existentes basados en Inteligencia Artificial (IA) o Machine Learning (ML), para la detección de Covid-19 (SARS-CoV-2), gripe o neumonía pulmonar?
P2	¿Cuáles son los métodos, modelos o herramientas más precisos empleados hasta la fecha, para el diagnóstico de Covid-19?
P3	¿Cuáles son los recursos necesarios empleados por los métodos de IA para la detección de Covid-19?

2.2. Desarrollo de protocolo de revisión

a) Estrategia de búsqueda

Según Petticrew y Roberts [99], es aconsejable utilizar métodos que permitan recopilar evidencia de manera estructurada, para ello recomiendan utilizar el método PICOC:

- **P**opulation: ¿Quién?
- **I**ntervention: ¿Qué o cómo?
- **C**omparation: ¿Comparado con qué?
- **O**utcomes: ¿Qué estás tratando de lograr / mejorar?
- **C**ontext: ¿En qué tipo de organización / circunstancias?

Este método, se utiliza para describir una estructura basada en los cinco componentes antes mencionados, de tal forma que han permitido definir la cadena de búsqueda. No obstante, de la nemotécnica PICOC, para la presente RSL no se considera el criterio de Comparación (Silahudin et al., 2020), por lo tanto, se emplearon solo cuatro componentes: Population (P), Intervention (I), Outcomes (O) y Context (C). Así mismo,

para una mejor organización y selección de todos los resultados, se utilizó la herramienta Parsifal, la cual sirve de soporte para realizar RSL en el contexto de la Ingeniería del Software. Gracias a esta herramienta, fue posible dar seguimiento a la RSL en cada una de sus etapas, obteniendo así, resultados mucho más óptimos.

b) Fuentes bibliográficas

En cuanto a las fuentes bibliográficas, se ha seleccionado las siguientes bibliotecas virtuales:

- ACM Digital Library (https://dl.acm.org/)
- IEEE Digital Library (https://www.ieee.org/)
- Scopus (http://www.scopus.com)

c) Definir palabras clave para el problema de investigación

Basándonos en la definición de los criterios PICOC, se ha obtenido las siguientes palabras claves, las cuales han permitido construir las cadenas de búsqueda. Las palabras clave en cuestión son:

- Artificial intelligence
- Covid-19
- Diagnosis
- Machine learning
- Expert system
- Neural network

d) Cadenas de búsqueda

Las cadenas de búsqueda fueron construidas en base a las palabras clave definidas anteriormente. Estas cadenas se aplicaron de acuerdo a cada biblioteca virtual seleccionada: ACM Digital Library, IEEE Digital Library y Scopus. Las cadenas finales se presentan en la:

TABLA XXIII *Cadenas de búsqueda para cada una de las bibliotecas virtuales.*

Bibliotecas virtuales	**Cadenas de búsqueda**
ACM Digital Library	[Publication Title: machine learning" "model" "expert system" "expert systems] AND [Abstract: "covid-19 "] AND [[Publication Title: "diagnosis covid-19"] OR [Publication Title: "diagnose covid-19"] OR [Publication Title: "covid-19"]] AND [Abstract: "diagnosis" "disease" "identification"] AND [[Abstract: "machine learning"] OR [Abstract: "model"] OR [Abstract: "expert system"] OR [Abstract: "expert systems]] AND [Publication

	Date: (01/01/2019 TO 01/31/2021)]
IEEE Digital Library	((((((((((“Document Title”:Covid-19) AND “Abstract”:Machine learning) AND “Abstract”:Model) OR “Abstract”:Artificial intelligence) AND “Abstract”:Diagnos) OR “Abstract”:Identification*) AND “Document Title”:Machine learning) OR “Document Title”:Artificial intelligence AND Co-vid*) OR “Document Title”:Model) AND “Abstract”:Covid-19)
Scopus	TITLE ((“machine learning” OR “artificial intelligence” OR “expert system” OR “intelligence systems”) AND (“covid19” OR “covid-19”) AND (diagnos* OR identification OR disease))

e) Criterios de inclusión

Se consideró como criterios de inclusión, los siguientes:

- Artículos o libros con un año de publicación mayores al 2019.
- Artículos o libros en inglés o español.
- El articulo habla sobre los métodos de IA para identificar o diagnosticar Covid-19.

f) Criterios de exclusión

Se consideró como criterios de exclusión, los siguientes:

- Artículos con un año de publicación menor al 2019.
- El modelo no reconoce o diagnostica Covid-19.
- Métodos y técnicas difusos o con ambigüedad.

2.3. Conducción de la revisión

a) Identificación de fuentes y estudios relevantes

El objetivo de la presente RSL, es dar contestación a las preguntas de investigación, mediante la búsqueda de estudios primarios que permitiesen obtener información relacionada con las herramientas basadas en IA empleadas para diagnosticar Covid-19 o similares, debido a que es un tema relativamente nuevo en lo que respecta al diagnóstico de esta enfermedad usando técnicas basadas en IA.

Razón por la cual, la selección de los estudios primarios, se realizó contrastando los resultados de varias fuentes de tal manera que, la información sea de calidad y se enfoque únicamente en el objetivo de la presente RSL.

Para realizar la selección de los estudios se ha seguido el proceso que se muestra en la **Figure 1**

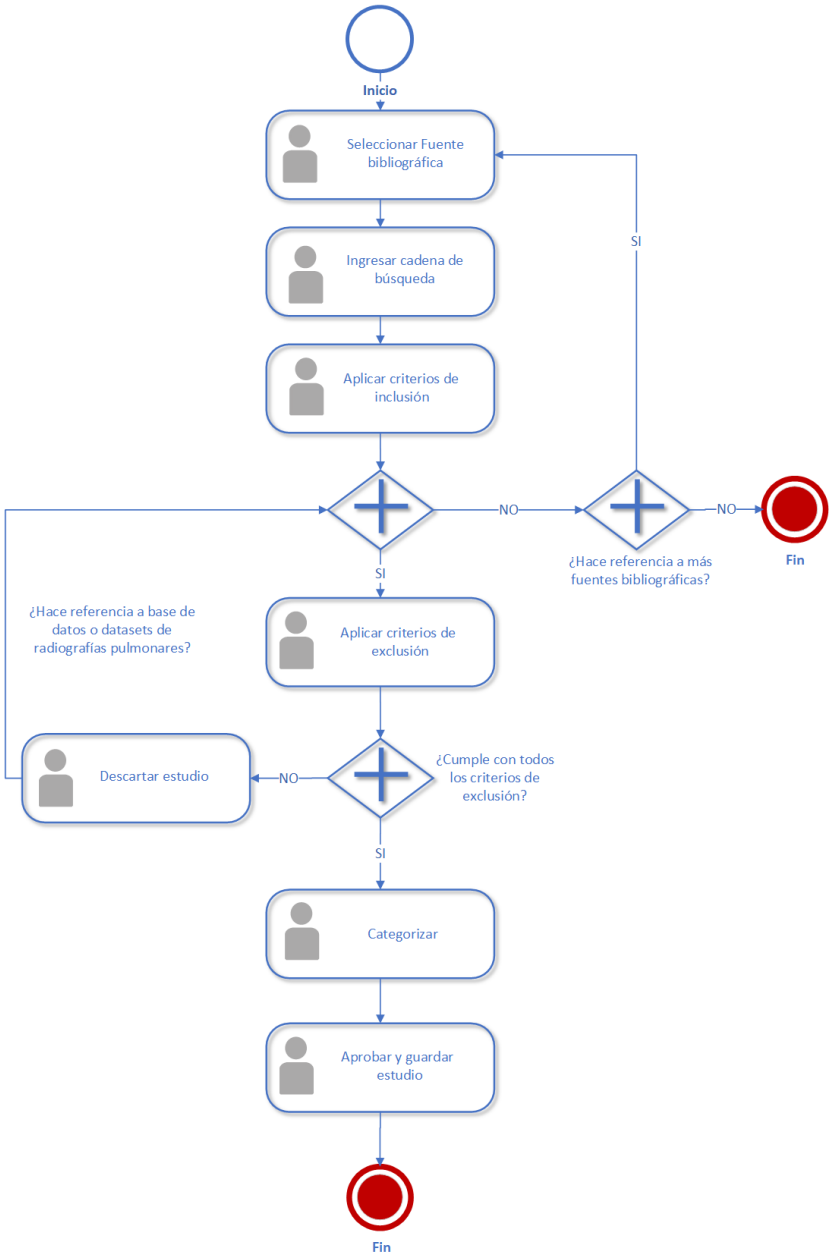


Figure 1 *Diagrama de actividades para el proceso de selección de artículos*

b) *Selección y síntesis de los datos*

Los estudios obtenidos en cada una de las bibliotecas virtuales, corresponden a aquellos trabajos los cuales cumplen con todos los criterios de selección establecidos, y se presentan en la sección 4.4, llamada trabajos relacionados, específicamente en la **TABLA I**, junto con la respuesta a las preguntas de investigación planteadas al inicio de la presente RSL.

Anexo 2. Gráficas y tablas

1. Graficas generadas en el proceso de Fine Tuning del modelo

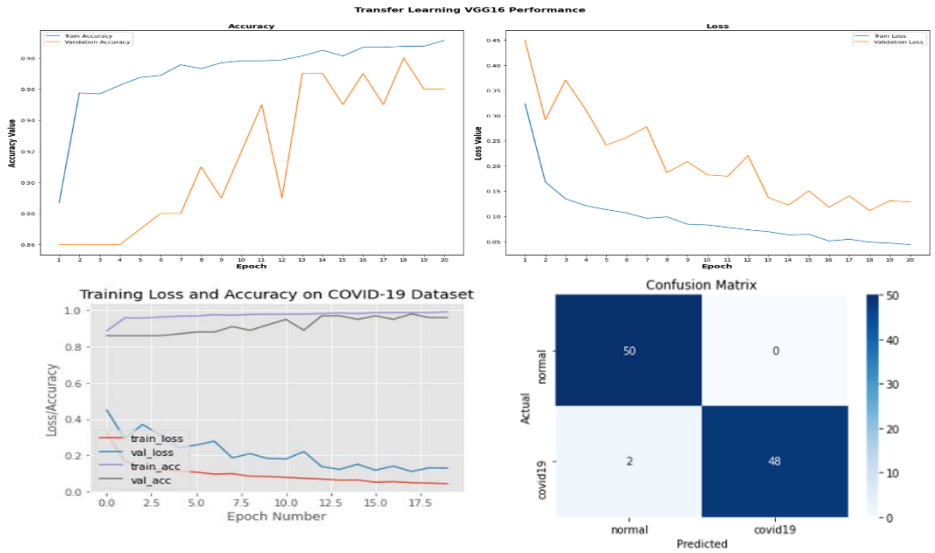


Fig. 51 *Gráficas correspondientes a los resultados de la configuración 1*

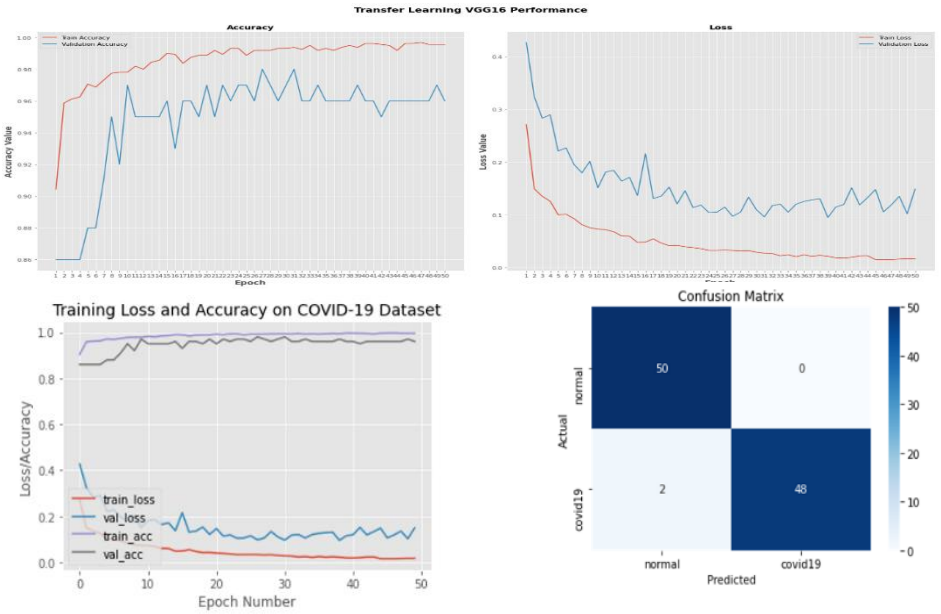


Fig. 52 *Gráficas correspondientes a los resultados de la configuración 2*

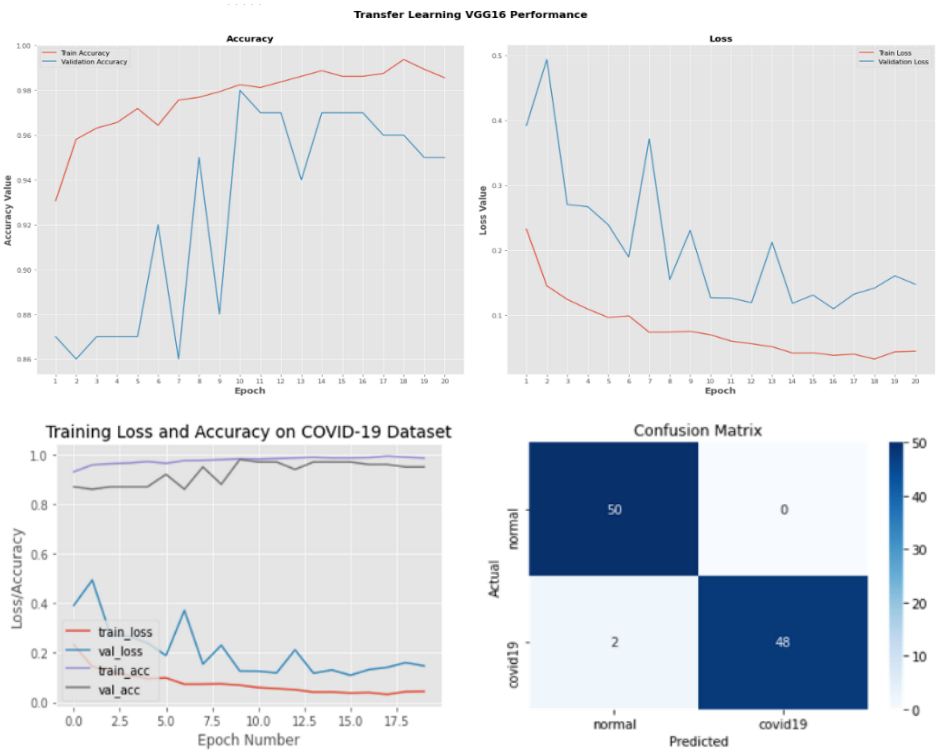


Fig. 53 Gráficas correspondientes a los resultados de la configuración 4

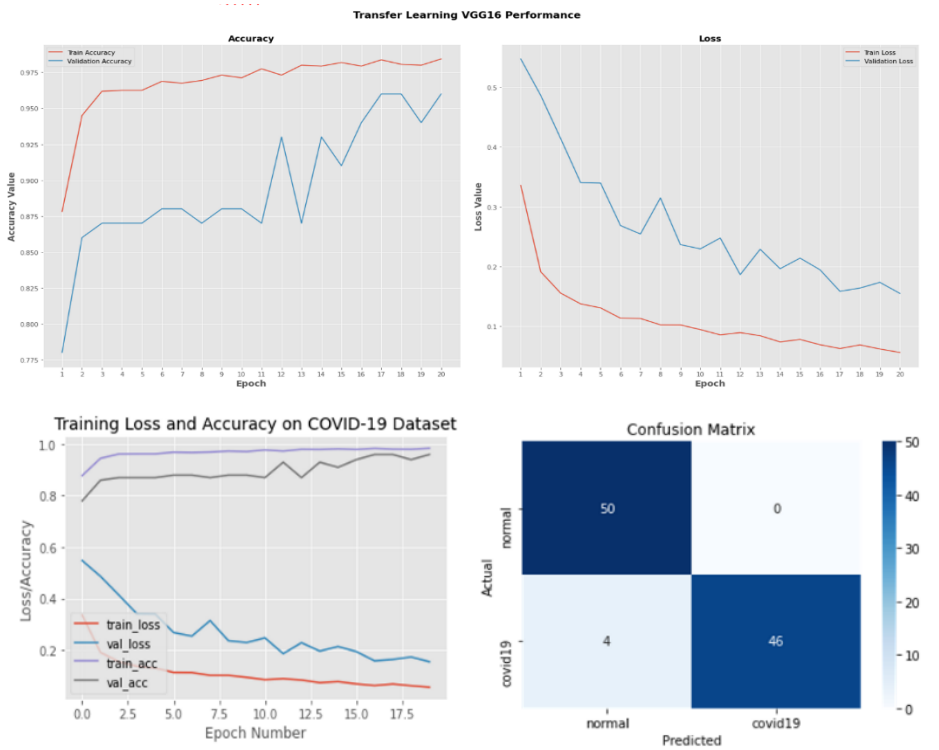


Fig. 54 Gráficas correspondientes a los resultados de la configuración 7

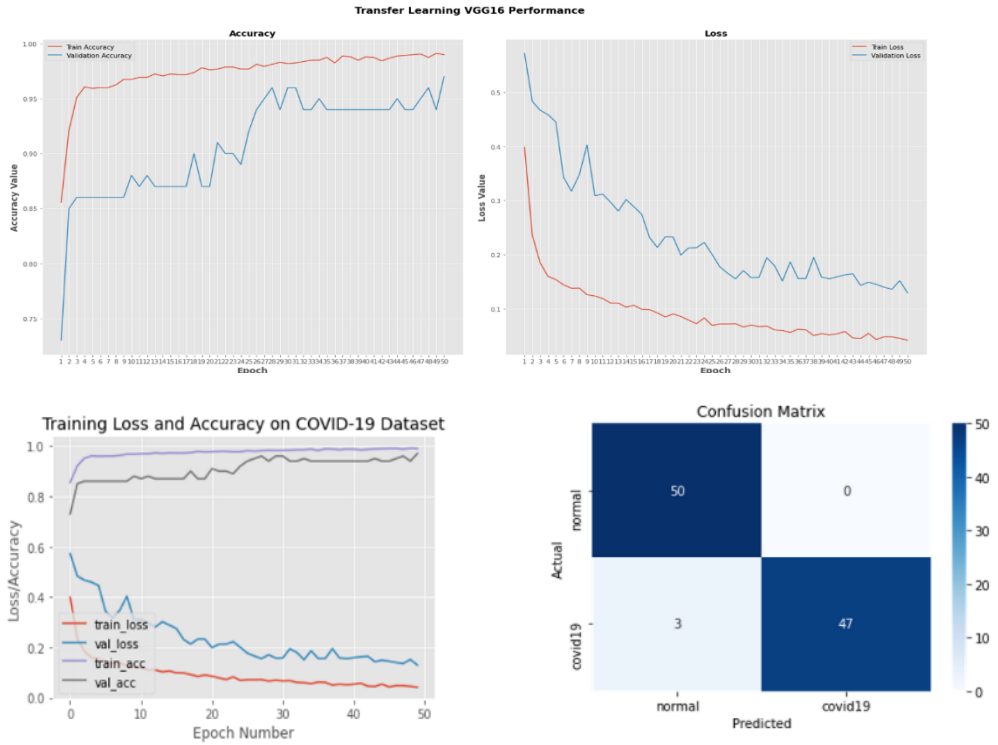


Fig. 55 Gráficas correspondientes a los resultados de la configuración 8

2. Tablas y gráficas generadas en la fase de entrenamiento

2.1. Modelo1: modelVGG16to1k

TABLA XXIV *Resultados de la fase de entrenamiento y test del modelo Diagnos19, para una carga total de 1000 radiografías*

	Parámetros	Precisión	Sensibilidad	Especificidad	Accuracy	Support	Time
Training	**Covid-19**	0.99857	0.99750	0.99803	0.99656	2800	18m 55s
	Normal	0.98263	0.99000	0.98630		400	
	Promedio ponderado	0.99658	0.99656	0.99657		3200	
Test	**Covid-19**	0.98039	1.00000	0.99010	0.99000	100	
	Normal	1.00000	0.98000	0.98990		100	
	Promedio ponderado	0.99020	0.99000	0.99000		200	

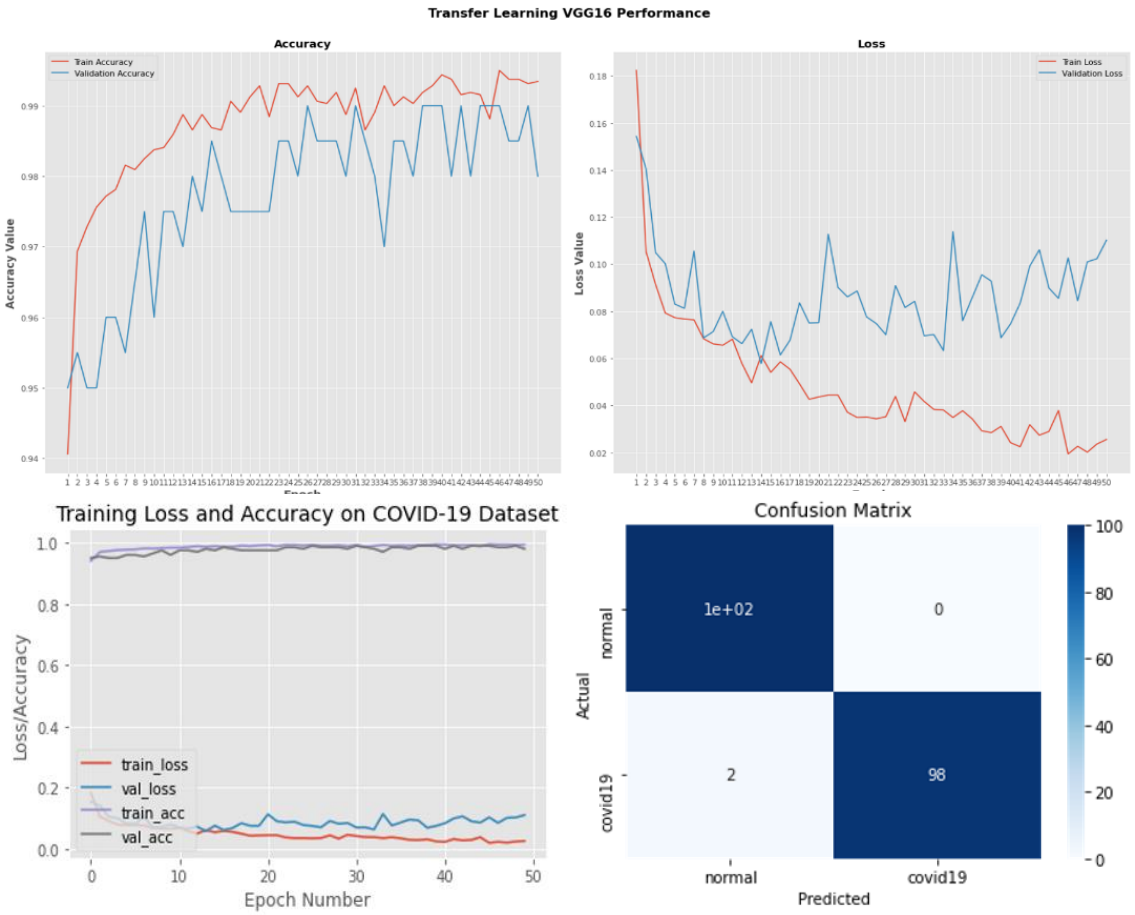


Fig. 56 *Gráficas correspondientes a los resultados del entrenamiento con 1000 radiografías*

2.2. Modelo2: modelVGG16to2k

TABLA XXV *Resultados de la fase de entrenamiento y test del modelo Diagnos19, para una carga total de 2000 radiografías.*

	Parámetros	Precisión	Sensibilidad	Especificidad	Accuracy	Support	Time
Training	**Covid-19**	0.99786	1.0000	0.99813	0.99813	5600	38m 8s
	Normal	1.0000	0.98500	0.99244		800	
	Promedio ponderado	0.99813	0.99813	0.99812		6400	
Test	**Covid-19**	0.98039	1.00000	0.99010	0.99000	200	
	Normal	1.00000	0.98000	0.98990		200	
	Promedio ponderado	0.99020	0.99000	0.99000		400	

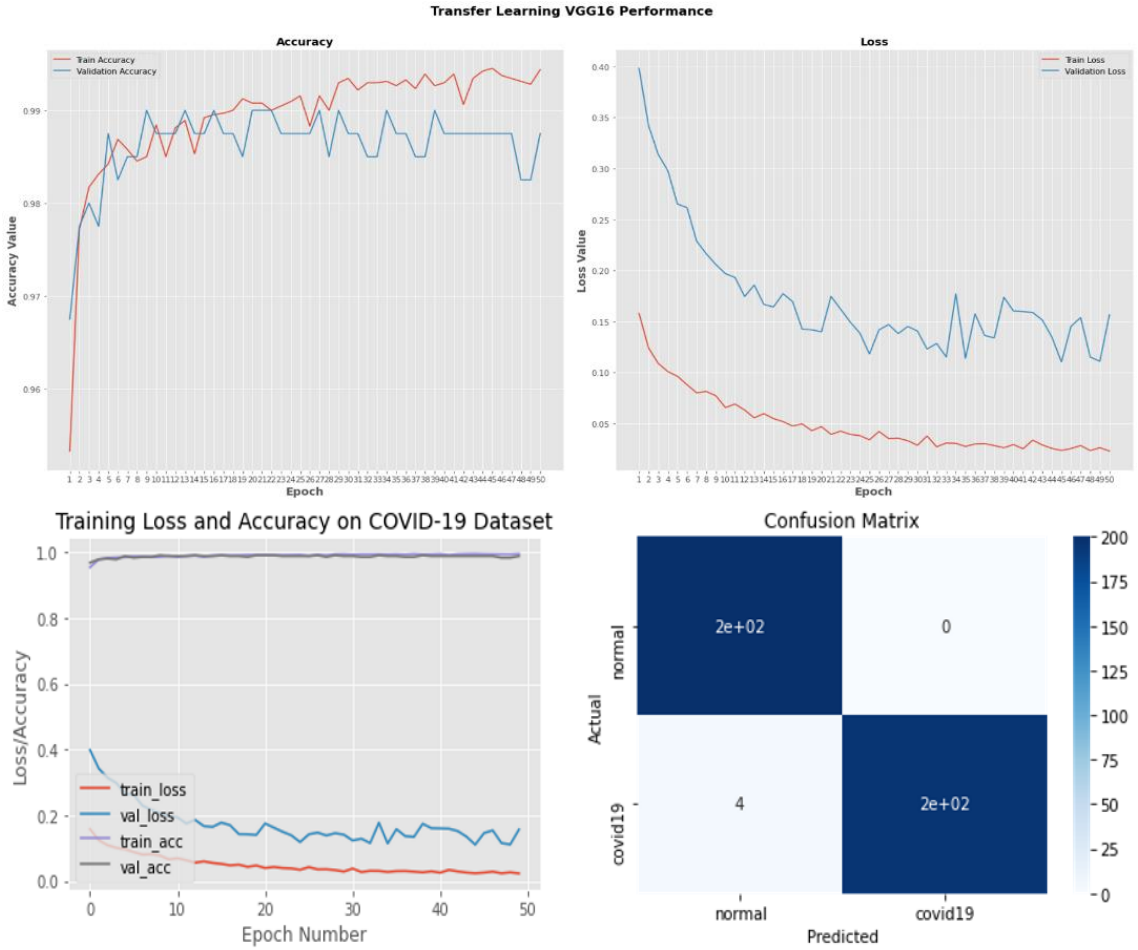


Fig. 57 *Gráficas correspondientes a los resultados del entrenamiento con 2000 radiografías*

2.3. Modelo3: modelVGG16to4k

TABLA XXVI *Resultados de la fase de entrenamiento y test del modelo Diagnos19, para una carga total de 4000 radiografías.*

	Parámetros	Precisión	Sensibilidad	Especificidad	Accuracy	Support	Time
Training	Covid-19	0.99822	0.99875	0.99848	0.99734	11200	1h 15m 48s
	Normal	0.99122	0.98750	0.98936		1600	
	Promedio ponderado	0.99734	0.99734	0.99734		12800	
Test	Covid-19	0.96098	0.98500	0.97215	0.97250	400	
	Normal	0.98462	0.96000	0.97215		400	
	Promedio ponderado	0.97280	0.97250	0.97250		800	

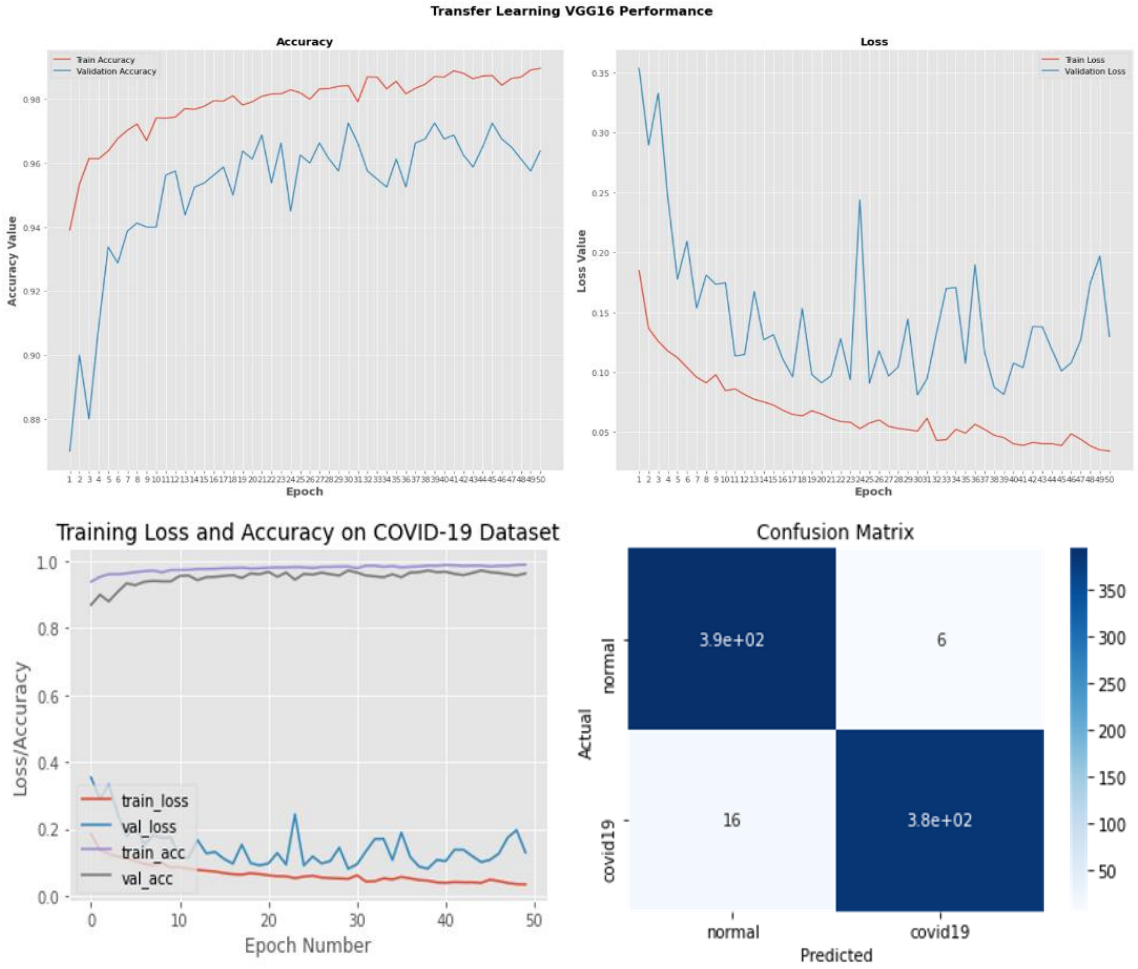


Fig. 58 *Gráficas correspondientes a los resultados del entrenamiento con 4000 radiografías*

2.4. Modelo5: modelVGG16to7k

TABLA XXVII *Resultados de la fase de entrenamiento y test del modelo diagnos19, para una carga total de 1000 radiografías.*

	Parámetros	Precisión	Sensibilidad	Especificidad	Accuracy	Support	Time
Training	Covid-19	0.99938	0.99071	0.99503	0.99134	19600	2h 24m 27s
	Normal	0.93872	0.99571	0.96638		2800	
	Promedio ponderado	0.99180	0.99134	0.99145		22400	
Test	Covid-19	0.98091	0.95429	0.96741	0.96786	700	
	Normal	0.95549	0.98143	0.96829		700	
	Promedio ponderado	0.96820	0.96786	0.96785		1400	

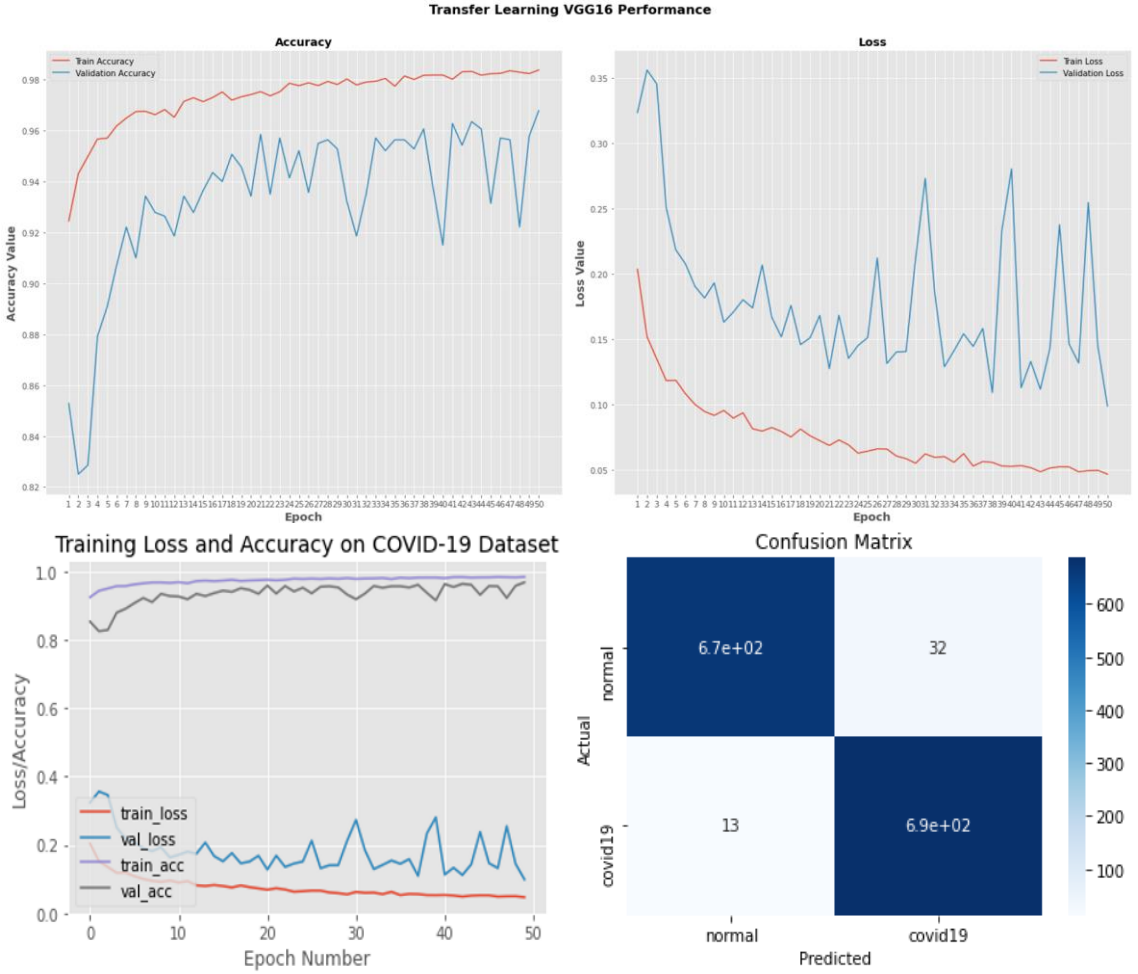


Fig. 59 *Gráficas correspondientes a los resultados del entrenamiento con 7000 radiografías*

3. Ejecución de pruebas ZSL

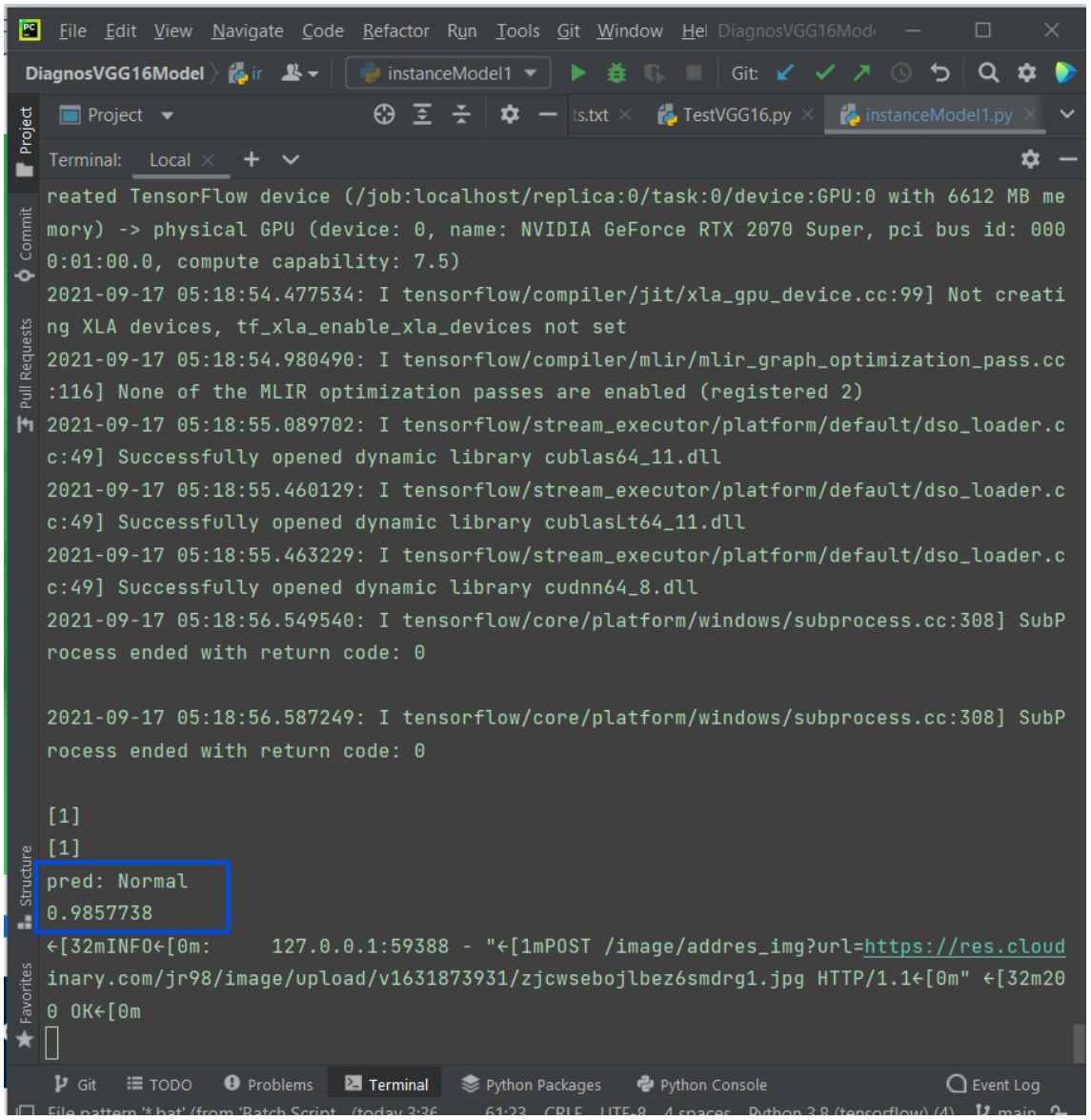


Fig. 60 *Ejecución de pruebas ZSL en la consola de PyCharm. En el recuadro azul se muestra en resultado del diagnóstico y el nivel de precisión alcanzado por la imagen 32 de la* ***TABLA XII***

Anexo 3: Entrevista

La entrevista fue realizada a un especialista en radiología, el cual trabaja en el Hospital Básico Amaluza, correspondiente al segundo nivel de atención de salud del Ministerio de Salud Pública, del distrito 11D05 ubicado en la cabecera cantonal, parroquia Amaluza, específicamente en la calle 27 de abril SN.

La entrevista fue llevada a cabo entre mi persona, como autor del presente TT, y el especialista en radiología, cuya identidad pidió no ser revelada por cuestiones personales. Es por ello que, el tipo de entrevista elegido para llevar a cabo la entrevista fue la “entrevista libre” (referenciada en la sección 5.3.2),

manera informal usando una grabadora de voz para sustentar la evidencia y poder acceder al material posteriormente.

La entrevista completa se encuentra en la siguiente carpeta compartida de Google Drive[11]

[11] Entrevista

Anexo 4: Código del modelo e interfaz de Diagnos19

CD número 2