UNIVERSIDAD ESTATAL DE MILAGRO

FACULTAD CIENCIAS E INGENIERÍA

TRABAJO DE INTEGRACIÓN CURRICULAR

PREVIO A LA OBTENCIÓN DEL TÍTULO DE

INGENIERO EN SISTEMAS COMPUTACIONALES

TEMA: DESARROLLO DE UN SISTEMA WEB UTILIZANDO PROCESAMIENTO DE IMÁGENES DE RAYOS X DE TÓRAX PARA LA DETECCIÓN DE COVID-19 MEDIANTE DEEP LEARNING

Autores:

Sr. Fuentes Peralta Nelson Kevin

Sr. Vera Martínez Samuel Aaron

Tutor:

Mgtr. Vera Paredes Daniel Alexander

Milagro, Diciembre 2021

ECUADOR

DERECHOS DE AUTOR

Ingeniero.

Fabricio Guevara Viejó, PhD.

RECTOR

Universidad Estatal de Milagro

Presente.

Yo, FUENTES PERALTA NELSON KEVIN, en calidad de autor y titular de los derechos morales y patrimoniales del trabajo de integración curricular, modalidad presencial, mediante el presente documento, libre y voluntariamente procedo a hacer entrega de la Cesión de Derecho del Autor, como requisito previo para la obtención de mi Título de Grado, como aporte a la Línea de Investigación Tecnologías de la información y de la comunicación, de conformidad con el Art. 114 del Código Orgánico de la Economía Social de los Conocimientos, Creatividad e Innovación, concedo a favor de la Universidad Estatal de Milagro una licencia gratuita, intransferible y no exclusiva para el uso no comercial de la obra, con fines estrictamente académicos. Conservo a mi favor todos los derechos de autor sobre la obra, establecidos en la normativa citada.

Así mismo, autorizo a la Universidad Estatal de Milagro para que realice la digitalización y publicación de este trabajo de integración curricular en el repositorio virtual, de conformidad a lo dispuesto en el Art. 144 de la Ley Orgánica de Educación Superior.

El autor declara que la obra objeto de la presente autorización es original en su forma de expresión y no infringe el derecho de autor de terceros, asumiendo la responsabilidad por cualquier reclamación que pudiera presentarse por esta causa y liberando a la Universidad de toda responsabilidad.

Milagro, Haga clic aquí para escribir una fecha.

_____

FUENTES PERALTA NELSON KEVIN

Autor 1

CI: 0942234071

DERECHOS DE AUTOR

Ingeniero.

Fabricio Guevara Viejó, PhD.

RECTOR

Universidad Estatal de Milagro

Presente.

Yo, VERA MARTINEZ SAMUEL AARON, en calidad de autor y titular de los derechos morales y patrimoniales del trabajo de integración curricular, modalidad presencial, mediante el presente documento, libre y voluntariamente procedo a hacer entrega de la Cesión de Derecho del Autor, como requisito previo para la obtención de mi Título de Grado, como aporte a la Línea de Investigación Tecnologías de la información y de la comunicación, de conformidad con el Art. 114 del Código Orgánico de la Economía Social de los Conocimientos, Creatividad e Innovación, concedo a favor de la Universidad Estatal de Milagro una licencia gratuita, intransferible y no exclusiva para el uso no comercial de la obra, con fines estrictamente académicos. Conservo a mi favor todos los derechos de autor sobre la obra, establecidos en la normativa citada.

Así mismo, autorizo a la Universidad Estatal de Milagro para que realice la digitalización y publicación de este trabajo de integración curricular en el repositorio virtual, de conformidad a lo dispuesto en el Art. 144 de la Ley Orgánica de Educación Superior.

El autor declara que la obra objeto de la presente autorización es original en su forma de expresión y no infringe el derecho de autor de terceros, asumiendo la responsabilidad por cualquier reclamación que pudiera presentarse por esta causa y liberando a la Universidad de toda responsabilidad.

Milagro, Haga clic aquí para escribir una fecha.

_____

VERA MARTÍNEZ SAMUEL AARON

Autor 2

CI: 0929746493

APROBACIÓN DEL TUTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR

Yo, VERA PAREDES DANIEL ALEXANDER en mi calidad de tutor del trabajo de integración curricular, elaborado por los estudiantes FUENTES PERALTA NELSON KEVIN y VERA MARTÍNEZ SAMUEL AARON, cuyo título es DESARROLLO DE UN SISTEMA WEB UTILIZANDO PROCESAMIENTO DE IMÁGENES DE RAYOS X DE TÓRAX PARA LA DETECCIÓN DE COVID-19 MEDIANTE DEEP LEARNING, que aporta a la Línea de Investigación Tecnologías de la información y de la comunicación previo a la obtención del Título de Grado Ingeniero en Sistemas Computacionales; considero que el mismo reúne los requisitos y méritos necesarios en el campo metodológico y epistemológico, para ser sometido a la evaluación por parte del tribunal calificador que se designe, por lo que lo APRUEBO, a fin de que el trabajo sea habilitado para continuar con el proceso previa culminación de Trabajo de Integración Curricular de la Universidad Estatal de Milagro.

Milagro, Haga clic aquí para escribir una fecha.

VERA PAREDES DANIEL ALEXANDER

Tutor

C.I: 0914192182

APROBACIÓN DEL TRIBUNAL CALIFICADOR

El tribunal calificador constituido por:

Elija un elemento. Haga clic aquí para escribir apellidos y nombres (tutor).

Elija un elemento. Haga clic aquí para escribir apellidos y nombres (Secretario/a).

Elija un elemento. Haga clic aquí para escribir apellidos y nombres (integrante).

Luego de realizar la revisión del Trabajo de Integración Curricular, previo a la obtención del título (o grado académico) de INGENIERO EN SISTEMAS COMPUTACIONALES presentado por el estudiante FUENTES PERALTA NELSON KEVIN.

Con el tema de trabajo de Integración Curricular: DESARROLLO DE UN SISTEMA WEB UTILIZANDO PROCESAMIENTO DE IMÁGENES DE RAYOS X DE TÓRAX PARA LA DETECCIÓN DE COVID-19 MEDIANTE DEEP LEARNING.

Otorga al presente Trabajo de Integración Curricular, las siguientes calificaciones:

Trabajo Integración Curricular		[ ]
Defensa oral		[ ]
Total		[ ]

Emite el siguiente veredicto: (aprobado/reprobado) _____

Fecha: Haga clic aquí para escribir una fecha.

Para constancia de lo actuado firman:

	Nombres y Apellidos	Firma
Presidente	Apellidos y nombres de Presidente.	_____
Secretario /a	Apellidos y nombres de Secretario	_____
Integrante	Apellidos y nombres de Integrante.	_____

APROBACIÓN DEL TRIBUNAL CALIFICADOR

El tribunal calificador constituido por:

Elija un elemento. Haga clic aquí para escribir apellidos y nombres (tutor).

Elija un elemento. Haga clic aquí para escribir apellidos y nombres (Secretario/a).

Elija un elemento. Haga clic aquí para escribir apellidos y nombres (integrante).

Luego de realizar la revisión del Trabajo de Integración Curricular, previo a la obtención del título (o grado académico) de INGENIERO EN SISTEMAS COMPUTACIONALES presentado por el estudiante VERA MARTÍNEZ SAMUEL AARON.

Con el tema de trabajo de Integración Curricular: DESARROLLO DE UN SISTEMA WEB UTILIZANDO PROCESAMIENTO DE IMÁGENES DE RAYOS X DE TÓRAX PARA LA DETECCIÓN DE COVID-19 MEDIANTE DEEP LEARNING.

Otorga al presente Proyecto Integrador, las siguientes calificaciones:

Trabajo de Integración Curricular	[ ]
Defensa oral	[ ]
Total	[ ]

Emite el siguiente veredicto: (aprobado/reprobado) _____

Fecha: Haga clic aquí para escribir una fecha.

Para constancia de lo actuado firman:

	Nombres y Apellidos	Firma
Presidente	Apellidos y nombres de Presidente.	_____
Secretario /a	Apellidos y nombres de Secretario	_____
Integrante	Apellidos y nombres de Integrante.	_____

DEDICATORIA

Dedico está tesis de manera especial a mi madre, quien fue parte fundamental en mi desarrollo personal, sentó en mí las bases de responsabilidad, disciplina y deseos de superación. Además, dedico con mucho cariño a mis familiares, amigos y a todos quienes me apoyaron y me impulsaron a seguir adelante.

También a mis maestros quienes compartieron su conocimiento y sabiduría a lo largo de mi vida estudiantil para que pueda formarme profesionalmente. A todos ellos le dedico mi tesis.

Fuentes Peralta Nelson Kevin

Dedico a mis abuelas, por ser mis pilares fundamentales que me apoyaron e inspiraron a seguir con mi carrera profesional, a mis hermanos para que me tomen como ejemplo de que todo es posible en la vida, si se lo proponen, y a mi mejor amiga por estar presto ayudarme cuando más lo necesitaba.

Samuel Aaron Vera Martínez

AGRADECIMIENTO

Agradezco a Dios por ayudarme a escoger el camino correcto y darme la sabiduría para poder lograr este objetivo en mi vida. Agradezco a mi madre Silvia Judith Peralta Suarez por ser el pilar fundamental en mi vida, siempre brindándome su comprensión y su amor. También, agradezco a aquellas personas que de forma directa o indirecta creyeron firmemente en mis habilidades para llevar adelante este trabajo.

Agradezco inmensamente a todos mis familiares, tíos, primos y a mis abuelos. También quiero agradecer a Marina León por ser la persona que me acompaño en el transcurso de mi carrera y me ayudó a superarme.

Fuentes Peralta Nelson Kevin

Agradezco a Dios, por darme fuerza y sabiduría para seguir adelante, a mis padres por su sincero amor, apoyo y confianza, a mis abuelas por ser mi inspiración, a mis hermanos por ser incondicionales, y a mi mejor amigo por estar a mi lado en el proceso para forjarme como profesional.

También, para mis demás familiares y amigos que no dudaron en ningún momento que lograría concluir con mis estudios de Pregrado, y para los docentes que compartieron sus conocimientos con pasión en cada clase, inspirándonos a ser excelentes profesionales con sed de Justicia.

Samuel Aaron Vera Martínez

ÍNDICE GENERAL

DERECHOS DE AUTOR i
DERECHOS DE AUTOR ii
APROBACIÓN DEL TUTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR iii
APROBACIÓN DEL TRIBUNAL CALIFICADOR iv
APROBACIÓN DEL TRIBUNAL CALIFICADOR v
DEDICATORIA vi
AGRADECIMIENTO vii
ÍNDICE GENERAL viii
ÍNDICE DE FIGURAS ix
ÍNDICE DE TABLAS x
RESUMEN xii
ABSTRACT xiii
CAPÍTULO 1 1
1. INTRODUCCIÓN 1
1.1. Planteamiento del problema 1
1.2. Objetivos 3
1.3. Alcance 3
1.4. Estado del arte 5
CAPÍTULO 2 24
2. METODOLOGÍA 24
2.1. Metodología de Investigación 24
2.2. Metodología de Desarrollo 24
CAPÍTULO 3 50
3. PROPUESTA DE SOLUCIÓN 50
3.1. Análisis de factibilidad 50
3.2. Propuesta 52
3.3. Validación de la propuesta 63
CONCLUSIONES 64

RECOMENDACIONES 65
BIBLIOGRAFÍA 66
ANEXOS 69

ÍNDICE DE FIGURAS

Figura 1 Sistema web de predicción de covid-19 4
Figura 2 Estructura de una red neuronal convolucional. 13
Figura 3 Aplicación de filtros a la convolución. 14
Figura 4 Función de activación Relu. 16
Figura 5 Función de activación Sigmoidal. 16
Figura 6 Arquitectura LeNet 17
Figura 7 Arquitectura VGGNet-19. 18
Figura 8 Proceso de creación y entrenamiento de la red neuronal convolucional. 25
Figura 9 Carga de los datos a nuestro drive 27
Figura 10 Clases de etiquetas, Normal y Covid. 27
Figura 11 Configuración del entorno de trabajo en Google Colab 29
Figura 12 Entrenamiento 1: Variación de la exactitud y pérdida del entrenamiento. 31
Figura 13 Entrenamiento 1: Variación de la entropía cruzada durante el entrenamiento. .. 32
Figura 14 Entrenamiento 2: Variación de la exactitud y pérdida del entrenamiento. 33
Figura 15 Entrenamiento 2: Variación de la entropía cruzada durante el entrenamiento. .. 34
Figura 16 Matriz de confusión de las pruebas realizadas del entrenamiento 1. 35
Figura 17 Matriz de confusión de las pruebas realizadas del entrenamiento 2. 36
Figura 18 Entrenamiento con el aumento de datos: Variación de la exactitud y pérdida del entrenamiento. 39
Figura 19 Entrenamiento con aumento de datos: Variación de la entropía cruzada durante el entrenamiento. 39
Figura 20 Matriz de confusión de las pruebas realizadas del entrenamiento con aumento de datos. 40
Figura 21 Arquitectura de la aplicación web 45
Figura 22 Diagrama de caso de uso de la aplicación web. 46
Figura 23 Diagrama de clases de la aplicación 47

Figura 24 Creación del repositorio en GitHub. 48
Figura 25 Diagrama de la base de datos de la aplicación web. 53
Figura 26 Menú de la aplicación. 55
Figura 27 Menú de la aplicación para el grupo de usuarios médicos. 55
Figura 28 Menú de la aplicación para el grupo de usuarios pacientes. 56
Figura 29 Módulo mis resultados. 56
Figura 30 Módulo Grupos de usuarios. 57
Figura 31 Módulo menú. 57
Figura 32 Módulo accesos a los módulos. 58
Figura 33 Módulo especialidades de médicos. 58
Figura 34 Módulo vacunas. 59
Figura 35 Formulario de registro de pacientes. 59
Figura 36 Formulario de registro de médicos. 60
Figura 37 Formulario de registro de Análisis radiográficos. 60
Figura 38 Módulo mi perfil. 61
Figura 39 Módulo cambiar contraseña. 61
Figura 40 Reportes estadísticos de la aplicación. 62
Figura 41 Login de la aplicación. 62

ÍNDICE DE TABLAS

Tabla 1 Causas y consecuencias del problema. 2
Tabla 2 Tipos de aprendizaje automático. 11
Tabla 3 Matriz de confusión. 20
Tabla 4 Total de imágenes del set de datos. 26
Tabla 5 Modelos VGGNet19. 28
Tabla 6 Configuración de los parámetros para el entrenamiento. 29
Tabla 7 Configuración de los parámetros para el preprocesamiento de las imágenes. 30
Tabla 8 Entrenamiento 1: capas agregadas a la arquitectura VGG19. 30
Tabla 9 Resultados del entrenamiento 1 de la CNN. 31
Tabla 10: Entrenamiento 2: capas agregadas a la arquitectura VGG19. 32
Tabla 11 Resultados del entrenamiento 2 de la CNN. 33

Tabla 12 Prueba de predicción del entrenamiento 1. 34
Tabla 13 Prueba de predicción del entrenamiento 2. 35
Tabla 14 Comparativa entre los entrenamientos de las CNN realizadas. 36
Tabla 15 Enlaces de imágenes de rayos x con covid. 37
Tabla 16 Cantidad de imágenes agregadas a nuestro set de datos. 38
Tabla 17 Resultados del entrenamiento con el aumento de datos de la CNN. 38
Tabla 18 Prueba de predicción del entrenamiento con aumento de datos. 40
Tabla 19 Requerimiento-001 41
Tabla 20 Requerimiento-002 42
Tabla 21 Requerimiento-003 42
Tabla 22 Requerimiento-004 43
Tabla 23 Requerimiento-005 43
Tabla 24 Requerimiento-006 43
Tabla 25 Requerimiento-007 44
Tabla 26 Requerimiento-008 44
Tabla 27 Requerimiento-009 44
Tabla 28 Cronograma de actividades 49
Tabla 29 Recursos de hardware utilizados en el proyecto. 50
Tabla 30 Recursos de software utilizados en el proyecto. 50
Tabla 31 Costo en recursos de hardware. 51
Tabla 32 Costo en recursos de software. 51
Tabla 33 Costo en recurso humano para el desarrollo del proyecto. 51
Tabla 34 Costo total del desarrollo del proyecto. 52
Tabla 35 Funcionamiento de los módulos de la aplicación. 54
Tabla 36 Informe de pruebas de la aplicación web. 63
Tabla 37 Informe final de la aplicación web. 63

Título de Trabajo Integración Curricular: Desarrollo de un sistema web utilizando procesamiento de imágenes de rayos x de tórax para la detección de covid-19 mediante Deep learning.

RESUMEN

El Covid-19 ha desatado una pandemia el cual ha provocado una inestabilidad en el sistema de salud y en el sistema económico a nivel mundial, En una etapa temprana no se suele detectar si la persona presenta síntomas que indiquen si es un caso positivo o negativo de covid-19, por eso se propone desarrollar un sistema donde se gestionan los diferentes casos de covid-19 para la detección temprana del virus, con ayuda de inteligencia artificial, utilizando tecnologías como Python que cuenta con librerías como: Tensor Flow, Keras, pandas, numpy y matplotlib, que son las herramientas necesaria para lograr la construcción de la solución para detectar los casos de covid-19 a través de imágenes de rayos-x. Además del uso de esas librerías, se usa el framework Django el cual nos permite el desarrollo de una aplicación web de manera ágil y rápida.

Por otro lado, el desarrollo del proyecto involucra una serie de pasos previo a la construcción del producto final, en la cual se aplican metodologías de desarrollo de software e inteligencia artificial, que nos permiten optimizar tiempos y recursos. Desde el análisis de requerimientos, hasta el diseño, codificación y pruebas.

PALABRAS CLAVE: Covid-19, Framework, Django, Tensorflow, Aplicación web.

Título de Trabajo Integración Curricular: Development of a web system using chest x-ray image processing for the detection of covid-19 using Deep learning.

ABSTRACT

Covid-19 has unleashed a pandemic which has caused instability in the health system and the economic system worldwide. In an early stage, it is not usually detected if the person has symptoms that indicate whether it is a positive or negative case. of covid-19, that is why it is proposed to develop a system where the different cases of covid-19 are managed for the early detection of the virus, with the help of artificial intelligence, using technologies such as Python that has libraries such as: Tensor Flow, Keras, pandas, numpy and matplotlib, which are the tools necessary to achieve the construction of the solution to detect covid-19 cases through x-ray images. In addition to the use of these libraries, the Django framework is used, which allows us to develop a web application in an agile and fast way. On the other hand, the development of the project involves a series of steps prior to the construction of the final product, in which software development methodologies and artificial intelligence are applied, which allow us to optimize time and resources. From requirements analysis, to design, coding and testing.

KEY WORDS: Covid-19, Framework, Django, Tensorflow, web Application. .

CAPÍTULO 1

1. INTRODUCCIÓN

En la actualidad se presenció una crisis sanitaria por el virus covid-19, que en sus inicios fue catalogado como epidemia y de manera acelerada se convirtió en pandemia a nivel mundial, debido a su fácil transmisión causando graves problemas al sistema respiratorio. Por esto, muchos sistemas de salud colapsaron, motivo a la masiva cantidad de pacientes que se internaban con estado de salud crítico. Por lo tanto, para poder realizar una detección de covi-19 efectiva y accesible, se plantea hacer uso de la Inteligencia Artificial, aplicando Deep Learning, a través del análisis de imágenes de rayos x, específicamente del tórax. Por tal motivo, la presente investigación se enfoca en el desarrollo de una aplicación web que permita gestionar y controlar a los pacientes de un centro de salud, el cual se ha denominado "LabCov". En donde el sistema permitirá implementar el modelo de red neuronal convolucional que previamente se creará, codificará y entrenará para realizar las predicciones de casos de covid-19 mediante imágenes de radiografías de tórax.

De este modo, se analizan conceptos relacionados con inteligencia artificial y redes neuronales convolucionales, así como su estructura, arquitecturas existentes, funcionamiento, etc. Una vez obtenida la red neuronal convolucional, ésta será implementada en nuestra aplicación web, en la cual se utiliza como lenguaje de programación Python junto con el Framework Django.

1.1. Planteamiento del problema

La pandemia por Covid-19 ha provocado que los centros de salud colapsan debido a la gran cantidad de personas que evidencian síntomas de esta enfermedad. De tal manera que no se cuenta con la cantidad de pruebas necesarias para detectar si los pacientes están

contagiados, por consiguiente, al determinar que no se dispone de suficientes pruebas, se plantea desarrollar como alternativa el uso de imágenes de rayos x del tórax para predecir si un paciente tiene Covid o no. Dado que se conoce que el covid-19 es una enfermedad que ataca principalmente a los pulmones.

Por lo tanto, haciendo uso de técnicas de inteligencia artificial mediante la creación y entrenamiento de una red neuronal convolucional, junto con la construcción de un sistema web capaz de gestionar a los pacientes y poder realizar los análisis de radiografías de tórax, se busca optimizar los tiempos y recursos en la realización de las pruebas de covid-19.

1.1.1. Causas y consecuencias del problema

Tabla 1 Causas y consecuencias del problema.

Causas	Consecuencias
Escasez de pruebas.	Colapso en los centros de salud.
Tiempo de entregas tardíos de los resultados de las pruebas realizadas.	No empezar el tratamiento en caso de que el paciente se diagnostique positivo a covid.
Gran cantidad de pacientes aglomerados en los centros de salud.	Procesos de control y gestión deficientes en la administración del centro de salud.
Desconocimiento de la cantidad de casos positivos y casos negativos en el centro de salud.	Falta de control de las pruebas realizadas y de inexistencia de informes con reportes estadísticos.
Procesos manuales de registro.	Desorganización.

Nota: Causas y consecuencias debido a la pandemia por covid-19 en los centros de salud y la falta de pruebas para identificar casos con covid. Fuente y elaboración propia.

1.2. Objetivos

1.2.1. Objetivo General

Desarrollar un sistema web aplicando procesamiento de imágenes de rayos X de tórax para la detección de Covid-19 mediante Deep Learning.

1.2.2. Objetivos Específicos

- Analizar las diferentes arquitecturas de redes neuronales convolucionales existentes para el entrenamiento de redes neuronales.
- Elegir y diseñar el modelo arquitectónico de nuestra red neuronal convolucional.
- Entrenar el modelo convolucional con un set de imágenes de radiografías de tórax con pacientes sanos y con covid-19.
- Desarrollar una aplicación web para la detección y predicción de covid-19.

1.3. Alcance

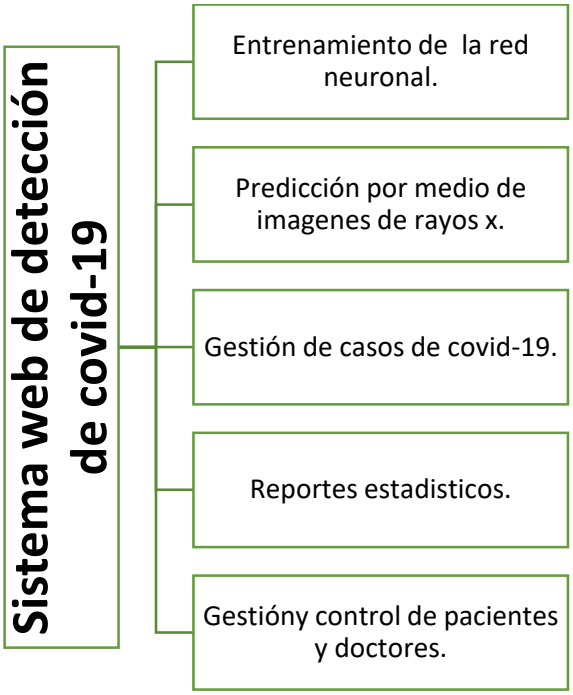
Este proyecto se realiza con el propósito de mitigar la cola de espera en el análisis de pruebas de covid-19, implementando técnicas de IA como viene siendo el Deep Learning, que permite clasificar las imágenes que se ingresan en casos positivos y casos negativos, utilizando redes neuronales convolucionales.

El desarrollo del sistema se divide en partes de construcción, entrenamiento y predicción de la red neuronal convolucional, la cual se construirá utilizando tecnologías como Python, Tensor Flow, Keras, numpy, pandas, etc. Librerías optimizadas para el procesamiento de datos. Estas herramientas mencionadas son las que permitirán lograr el desarrollo de un sistema capaz de predecir si la imagen de rayos x es positiva o negativa a covid-19.

Dado que es necesario tener un set de datos previamente clasificado en conjuntos de imágenes de rayos x con covid y normal; se recopila la mayor cantidad de imágenes las cuales serán utilizadas para entrenar la red neuronal convolucional. Además, se desarrollará un sistema web que sea capaz de implementar el modelo de red neuronal

convolucional diseñado. Para que a través de un módulo de la aplicación se pueda realizar el análisis de una radiografía de tórax del paciente para detectar si tiene covid o no. Cabe recalcar que el paciente debe realizarse una radiografía previamente. También, el sistema será capaz de gestionar a los médicos y pacientes del centro de salud, permitiendo observar reportes estadísticos de los casos analizados.

Figura 1 Sistema web de predicción de covid-19



Nota: En esta figura se puede observar lo que se va a desarrollar en el proyecto. Fuente y elaboración propia.

1.4. Estado del arte

1.4.1. Antecedentes del estudio

En el Hospital Italiano de Buenos Aires (HIBA) en Argentina en la época de pandemia ha optado por implementar una herramienta que permite el diagnóstico de covid-19 por medio de imágenes que se fundamenta en técnicas de inteligencia artificial. El diagnóstico por imágenes corresponde a una radiología simple y de ellas un 40% son de radiografías del tórax (RxTx), la creciente demanda de RxTx se debe a que es un estudio veloz y económico y que se encuentran en distintos centros de salud (HIBA).

La empresa Thirona situada en países bajos, se dedica a la creación de herramientas que con la implementación de técnicas de IA ayuden a predecir ciertas enfermedades con dos tipos de entradas uno sería la de tomografía computarizada (CT) y otro el rayos x, al presentarse la emergencia sanitaria esta empresa se fundamentó en su solución existente, LungQ que se enfoca en la detección de tuberculosis y se encuentra activo en más de 40 países en el mundo y que procesa datos de más de 350 hospitales en el día. A partir de esa solución se creó CAD4COVID que tiene dos versiones una para CT y otra para rayos x. Además, estos se encuentran de forma libre en soporte a la crisis sanitaria (Thirona, 2021).

En el Hospital Royal Bolton ubicado en el Reino Unido, el radiólogo Rizwam Malik, ha tenido un interés en la IA, dado que ve un gran potencial en el área médica para facilitar el trabajo, él logró encontrar una herramienta cuyo nombre es qXR de la empresa Qure.ai que tiene como propósito el diagnóstico de imágenes de rayos x del tórax, y en semanas se realizó la modificación para que pudiera detectar neumonía enfocada en covid-19, para que ayudara a ejecutar lecturas y así disminuir el tiempo de análisis de las imágenes de rayos x (Hao, 2020).

1.4.2. Fundamentación teórica

Metodologías de Desarrollo de Software. Las metodologías de desarrollo son una agrupación de procesos, que se siguen para lograr obtener un producto de software de alta calidad. Actualmente existen diferentes tipos de metodologías, la cual se divide en dos categorías principales, metodologías tradicionales y ágiles.

Metodología Ágil. Las metodologías de desarrollo ágiles son las más usadas, debido a su proceso incremental, esto quiere decir que fija bloques de avances, donde en cada bloque se presenta un parte del proyecto hasta donde se haya establecido. Permitiendo ser flexible en cada interacción con proyectos de mediano y corto plazo, debido a que se puede realizar modificaciones dependiendo si se cumple con lo establecido o el cliente requiere alguna modificación en ese punto.

De manera que este tipo de metodologías no tienden a tener problemas en la fecha de entrega del proyecto, dado que se realiza un seguimiento que involucra a todos los participantes.

Metodología Tradicional. Esta metodología se opone al cambio, lo cual conlleva a tener problemas futuros, dado que es una metodología en la cual se define las etapas previamente. Por consiguiente, no se puede continuar a la siguiente etapa sin antes haber culminado la etapa actual y una vez avanzado a la siguiente etapa no se podrá regresar a la etapa anterior.

Sin embargo, es conveniente en muchos proyectos, debido a que es un modelo ya empleado desde hace mucho tiempo, haciendo lo robusto. Además, de que está orientado a obtener resultados y estos resultados serán de una alta confianza, por el motivo que se promueve a realizar un trabajo efectivo, donde se define los requerimientos antes de empezar a diseñar, y se diseña antes que empezar a codificar.

Metodología en Cascada. Consiste en una metodología robusta, que se divide en 7 etapas generales.

1. Análisis de requerimientos. En esta etapa se indica de manera global, lo que se va a desarrollar en el software, sus funcionalidades y requerimientos.

2. Diseño de sistema. Esta etapa consiste en la maquetación de las ideas sobre cómo estará implementado estructuralmente el producto final.

3. Diseño de programa. Se modela las características y el funcionamiento del producto con la finalidad de cumplir con las necesidades del usuario final.

4. Codificación. En esta parte se implementa el código fuente en algún lenguaje de programación para hacer posible la construcción del sistema.

5. Pruebas. Se ensambla todas las partes del sistema y se realiza la comprobación de que no tenga fallas. Por lo general se suele realizar pruebas unitarias, para la comprobación de cada módulo del sistema.

6. Verificación. El usuario finalmente ejecutará el programa y observará que este cumpla con sus expectativas.

7. Mantenimiento. Se realizan mejoras o correcciones de problemas que se generan en el transcurso del tiempo.

Covid-19. A finales del 2019 el mundo se vio afectado debido a una enfermedad que provocó múltiples muertes a causa de un brote que provoca neumonía severa. Teniendo su inicio en china en la ciudad de Wuhan (Díaz & Toro, 2020) este virus denominado SARS-CoV-2 mayormente conocido como COVID-19, es un síndrome respiratorio agudo severo de tipo 2 que ataca directamente a los pulmones, sé ha

evidenciado que este virus genera síntomas en ciertas personas, en la cual existen pacientes asintomáticos que no presentan indicios de estar contagiados con la enfermedad.

Por otro lado, los pacientes sintomáticos evidencian síntomas leves de fiebre, tos seca, dolor de cabeza, mareos, dolor abdominal, diarrea, náusea y vómito. Además, puede provocar infecciones severas desencadenando insuficiencia renal aguda y síndrome de dificultad de respirar provocando la muerte (Melián, Calcumil, Boin, & Carrasco, 2020).

De acuerdo a distintos estudios el COVID-19 se transmite directa o indirectamente de persona a persona por medio del contacto. Entre las técnicas para diagnosticar si una persona está contagiada por el virus, están las pruebas rápidas PCR en la cual este tipo de prueba lo que hace es detectar el material genético del virus en la persona, por lo general no siempre da un pronóstico eficiente, dando falsos resultados en donde si existen evidencia de síntomas, en esos casos es recomendable realizar otra prueba (Melián, Calcumil, Boin, & Carrasco, 2020).

Procesamiento de imágenes médicas. En las últimas décadas el procesamiento de imágenes ha sido de gran ayuda en el área de la medicina, facilitando a los profesionales de la salud realizar diagnósticos de distintos tipos de patologías por medio de imágenes digitalizadas.

Entre los distintos tipos de imágenes médicas se tiene: radiografías, ultrasonidos, tomografía axial computarizada (TAC), imágenes de resonancia magnética (MRI), fotografía convencional, resonancia magnética, entre otros.

Rayos X. Las radiografías de rayos X nos permiten observar la parte interna de nuestro cuerpo a través de ondas electromagnéticas, por lo general este tipo de radiografías nos muestra una imagen en tono blanco y negro. Su utilización es muy variada ya que se

pueden realizar radiografías de distintas partes de nuestro cuerpo, como son radiografías abdominales, radiografías de huesos, tórax, dientes, extremidad inferior o superior, de la mano, pelvis, cráneo, cuello, entre otros (Moore, Dailey, & Agur, 2013).

La realización de este tipo de pruebas es rápida e indolora, sin embargo, no cualquiera puede interpretar los resultados mostrados en las imágenes de rayos x, para esto existen profesionales capaces de interpretar y dar un diagnóstico correcto.

Radiografías de Tórax. Las radiografías de tórax o torácica permiten realizar diagnósticos de los pulmones, en donde se detecta si existen infecciones o afecciones pulmonares como son la presencia de neumonía, tuberculosis o cáncer a los pulmones, incluso por la pandemia mundial por COVID-19 las radiografías de tórax también están siendo utilizadas para detectar si una persona presenta esta enfermedad (Maguiña, Murguía, Verona, Gomero, & Véliz, 2021).

Inteligencia artificial. Cuando se habla de inteligencia artificial se refiere a la capacidad de las máquinas de imitar al comportamiento humano, de acuerdo a (Takeyas, 2007) “La IA es una rama de las ciencias computacionales encargada de estudiar modelos de cómputo capaces de realizar actividades propias de los seres humanos en base a dos de sus características primordiales: el razonamiento y la conducta”.

Por otro lado, también se dice que la inteligencia artificial “tiene como objeto que los ordenadores hagan la misma clase de cosas que puede hacer la mente” (Boden, 2016). En otras palabras, la inteligencia artificial busca solucionar procedimientos repetitivos, en los que una máquina lo haga por nosotros.

En la actualidad la inteligencia artificial está presente en la mayor parte de las industrias implementando tecnologías como el uso de la visión artificial, reconocimiento de voz, procesamiento del lenguaje natural, sistemas expertos, robots y machine learning.

Inteligencia artificial en la Medicina. La medicina en los últimos años ha logrado grandes avances en los cuales se aprovechan las distintas técnicas de inteligencia artificial existentes, "formadas por una serie de algoritmos lógicos suficientemente entrenados a partir de los cuales las máquinas son capaces de tomar decisiones para casos concretos a partir de normas generales" (Avila, Mayer, & Quesada, 2020).

Esto quiere decir que la inteligencia artificial ofrece grandes beneficios en el área de la medicina ya que a través de ella se puede realizar diagnósticos precisos y rápidos, mediante el procesamiento de datos médicos se han automatizado procesos como la detección de patologías y realizar tratamientos con anticipación a los pacientes con un índice de error mínimo (Lugo Reyes, Guadalupe, & Murata, 2014).

Algo importante de recalcar es que, el proceso de diagnósticos, involucra una serie de conocimientos previos, capacitaciones y experiencias para reconocer patrones e identificar la enfermedad para evaluar la condición del paciente. Por lo tanto, los sistemas basados en inteligencia artificial deben ser robustos y poseer una gran cantidad de información de entrada para su correcto funcionamiento.

Aprendizaje automático. Una rama de la inteligencia artificial es el machine learning o en español aprendizaje automático, donde las computadoras son capaces de aprender algo sin la necesidad de estar programadas para ello. Por lo general esta clase de aprendizaje utiliza algoritmos de reconocimiento de patrones (Lasse Petteri , 2018). En donde se le proporcionan datos y en base a estos datos de entrada el sistema lo que hace es

generar sus propias decisiones. Esto quiere decir que con el pasar del tiempo dichos sistemas mejoran automáticamente.

Tipos de aprendizaje automático. En machine learning existen distintos tipos de técnicas de aprendizaje en donde cada una se caracteriza por resolver distintos tipos de problemas de acuerdo al contexto que se lo desee aplicar, entre ellos existe el aprendizaje supervisado, no supervisado y por refuerzo (Álvarez, Quirós, & Cortés, 2020).

Tabla 2 Tipos de aprendizaje automático.

Aprendizaje supervisado	Aprendizaje no supervisado	Aprendizaje por refuerzo
• Necesitan la intervención de un ente externo. • Se basa previamente en el etiquetado de los datos por los humanos.	• No necesita la intervención de un ente externo. • No necesita etiquetar los datos.	• Interactúa con el entorno

Nota: En esta tabla se puede observar los tipos de aprendizajes automáticos en inteligencia artificial. Fuente y elaboración propia.

Redes neuronales. De acuerdo a (Matich, 2001) las redes neuronales "son redes interconectadas masivamente en paralelo de elementos simples (usualmente adaptativos) y con organización jerárquica, las cuales intentan interactuar con los objetos del mundo real del mismo modo que lo hace el sistema nervioso biológico". Por lo tanto, se puede decir que las redes neuronales artificiales emulan el comportamiento de las redes neuronales humanas.

Redes neuronales convolucionales. Este tipo de redes neuronales se caracterizan por tener un alto potencial de procesamiento de datos. En donde la principal característica que la diferencia de las demás redes neuronales es que los datos de entradas son imágenes,

siendo muy eficientes en la clasificación, capaces de extraer características y reconocer patrones.

El funcionamiento de las redes neuronales convolucionales se asemeja a la forma en que los seres humanos perciben el mundo a través de los ojos. En donde “si vemos una cara, la reconocemos porque tiene orejas, ojos, una nariz, cabello, etc. Entonces, para decidir si algo es una cara, lo hacemos como si tuviéramos unas casillas mentales de verificación de las características que vamos marcando” (Torres, 2018).

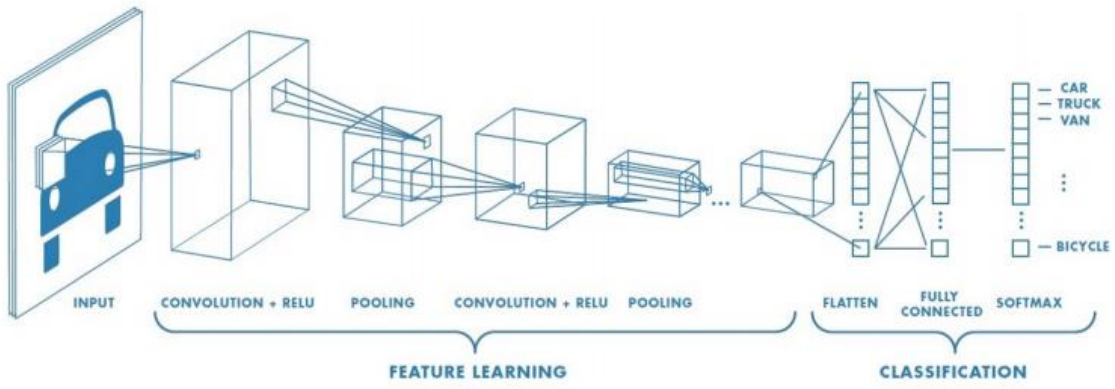
La forma en la que se representan las imágenes es de forma matricial tridimensional obteniendo sus propiedades como son el ancho, alto, profundidad y color, identificados por valores numéricos de píxeles (Sánchez Agustino, 2016). Además, las redes neuronales convolucionales se las denominan full connected, en otras palabras, que cada capa oculta de la red neuronal convolucional está conectada con las capas que la preceden y suceden.

Las redes neuronales convolucionales a pesar que su aprendizaje es automático, es considerado como un tipo de aprendizaje supervisado ya que previamente se le proporciona los datos etiquetados para su entrenamiento (Sánchez Martínez, 2018). Sin embargo, para que la red neuronal pueda identificar las características únicas de cada objeto y a su vez generalizarlo es necesario que la red se entrene y reconozca una gran cantidad de imágenes.

Estructura de una red neuronal convolucional. Este tipo de red se caracteriza por “realizar operaciones de convolución entre una capa de entrada y un filtro o kernel, dando como resultado un mapa de características (feature map). Este proceso se repite con varias capas intermedias, con el objetivo de extraer características de las imágenes con distintos niveles de extracción” (Sánchez Martínez, 2018).

Estas redes se encuentran constituidas por capas de convolución y agrupamiento alternadamente, junto con la capa de entrada, además de una capa de clasificación.

Figura 2 Estructura de una red neuronal convolucional.



Nota: En esta figura se observa que en una red neuronal convolucional los datos de entradas son imágenes y luego existen distintas capas que se encargan de extraer las características de la imagen para posteriormente ser clasificadas. Fuente: (Sánchez Martínez, 2018).

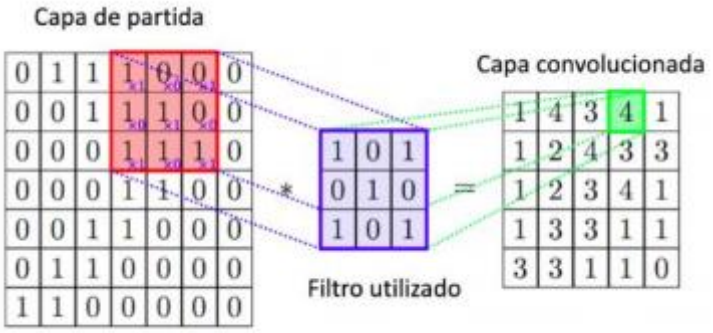
Capa de entrada. Esta capa se encarga de tomar como entrada los píxeles de una imagen, si las imágenes son en escalas grises y el alto y ancho de la imagen es de 50x50 píxeles, habrá un total de 2.500 neuronas.

Por otro lado, si la imagen es a color existiría un total de 7.500 neuronas para la entrada de la red, ya que se utiliza el sistema de color RGB. Por otra parte, se conoce que una imagen se conforma por píxeles que oscilan entre 0 y 255, es necesario transformar dichos valores en 0 y 1. De tal manera que se parte el valor del píxel entre 255 para normalizar los datos de entrada para la red (Artola Moreno, 2019).

***Capas convolucionales*.** Las capas convolucionales le dan origen al nombre de la red neuronal, en ella se recibe como entrada la imagen a analizar, internamente lo que realiza es extraer mediante filtros las características de la imagen como son los bordes y colores de un objeto, realizando operaciones de productos y sumas entre la imagen de

entrada y un filtro kernel (Wejéus, 2014). Como resultado la capa convolucional disminuye el número de elementos que conforman la red generando mapas de características a partir de cada convolución, este proceso se lo conoce como la capa de aplicación de filtros.

Figura 3 Aplicación de filtros a la convolución.



Nota: En esta figura se puede observar los filtros que se aplican en cada capa de convolución. Fuente: (Artola Moreno, 2019).

Capas de reducción o Pooling. Esta capa se encuentra situada después de las capas de convolución, se encarga de ir agrupando y reduciendo las dimensiones de entrada para la siguiente capa de convolución (Sánchez Martínez, 2018) En donde su objetivo es "disminuir aún más la carga computacional del sistema y, al mismo tiempo, ayudar con la caracterización de la imagen obteniendo y localizando los rasgos predominantes en ella" (Durán Suárez, 2017). Por lo general en esta capa se aplican las denominadas funciones de agregación que se describen más adelante.

Capas totalmente conectadas o full connected. Por último, se tiene la capa encargada de realizar la clasificación, "la capa totalmente conectada se encuentra justo a continuación de la última capa de pooling. Está compuesta por un número de neuronas igual al número de clases" (Durán Suárez, 2017). En donde se posee:

- **Capa flatten:** Encargada de convertir los datos que recibe en datos estructurados en un vector, el cual es tratado por la capa full connected.

- **Capa softmax:** Es una capa en el estado final de la red neuronal la cual con su función de activación normaliza los datos de salida del full connected, y este valor se comprende que no debe ser mayor a 1.

Funciones de agregación. Es donde se establece el método en cómo se agregarán las señales de entrada, por lo general se suele realizar una suma ponderada de las señales de entrada que usa el peso relacionado a cada sinapsis. Su expresión matemática es la siguiente:

$$\sum_{i=1}^{n} E_i * W_i$$

Donde $\boldsymbol{E}$ representa las entradas y $\boldsymbol{W}$ el peso.

Funciones de activación. Permite realizar el cálculo para la señal de salida de la red neuronal, para elegir una función de activación se tendrá en cuenta el tipo de datos que se está usando, la organización de la red y el algoritmo de entrenamiento (Navarro Moreno, 2020).

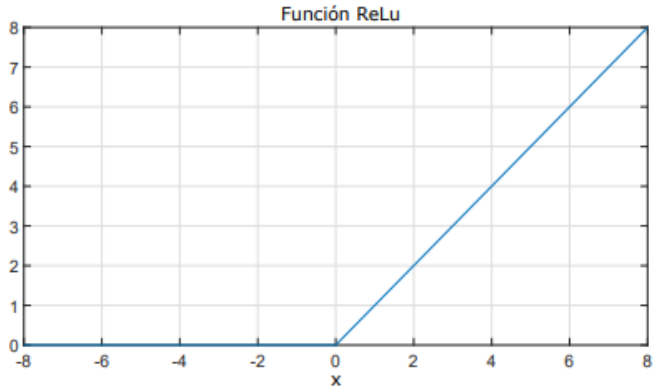
Existen distintas funciones de activación entre las más utilizadas se encuentran las siguientes:

función ReLU (Rectified Linear Units). Su expresión matemática es la siguiente:

$$f(x) = max(0, x)$$

Gráficamente se la representa así:

Figura 4 Función de activación Relu.



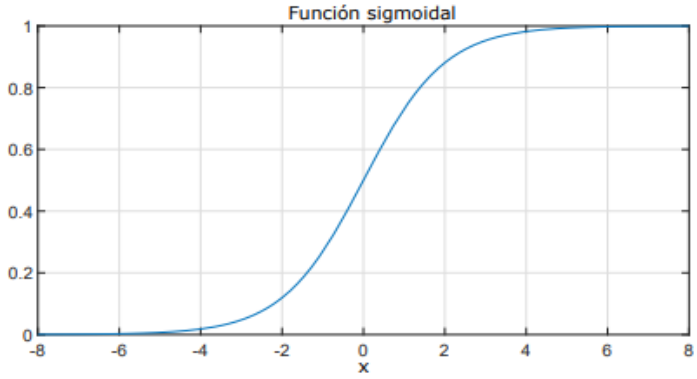
Nota: En esta figura se puede observar gráficamente la función de activación ReLU. Fuente y elaboración propia.

Función sigmoidal. Su expresión matemática es la siguiente:

$$f(x) = \frac{1}{1 + e^{-x}}$$

Gráficamente se la representa así:

Figura 5 Función de activación Sigmoidal.



Nota: En esta figura se observa gráficamente la función de activación Sigmoidal. Fuente y elaboración propia.

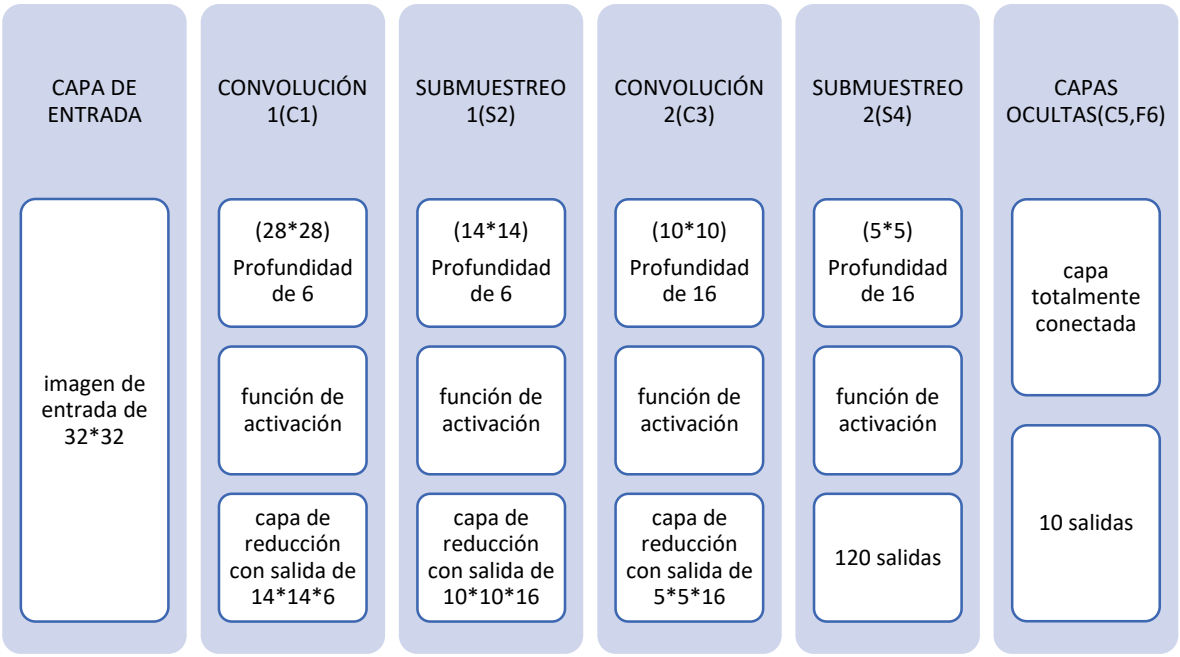
Arquitectura de redes neuronales convolucionales. Las arquitecturas de redes neuronales surgieron con el fin de ser reutilizables y permitir que los expertos en inteligencia artificial puedan optimizar el tiempo de desarrollo de sus proyectos,

estableciendo patrones y estructuras estándar (García, 2019). Entre las arquitecturas más conocidas y utilizadas se tiene las siguiente:

Arquitectura LeNet. Una de las primeras arquitecturas de redes neuronales convolucionales aplicada con éxito, su principal característica es que fue diseñada para reconocer e interpretar caracteres, ya sean números o letras.

La arquitectura LeNet se encuentra conformada por dos capas convolucionales y por capas de activación, a continuación, se representará su estructura:

Figura 6 Arquitectura LeNet.

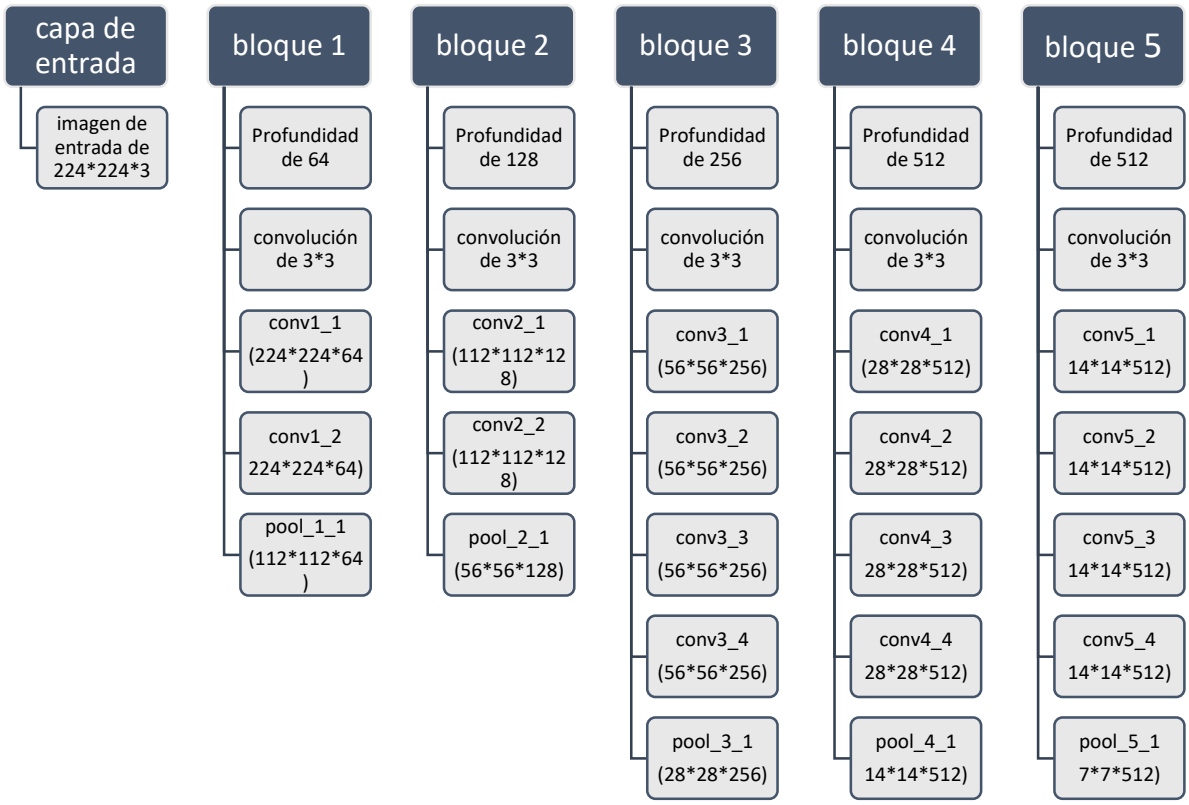


Nota: En esta figura se ve la arquitectura LeNet. Fuente y elaboración propia.

Arquitectura VGGNet. Esta arquitectura se caracteriza por su simplicidad y profundidad de la red neuronal convolucional, se encuentra conformada por pequeños filtros de $3 * 3$. En cada una de sus capas convolucionales y en las de submuestreo o agrupamiento se utiliza filtros de $2 * 2$.

Aunque existen distintas variaciones de esta arquitectura como son la VGGNet-16 y VGGNet-19, se ha demostrado que tienen un buen desempeño, además, que es posible implementarla en cualquier proyecto de IA con redes neuronales convolucionales. A continuación, se mostrará como se encuentra conformada la arquitectura VGGNet-19:

Figura 7 Arquitectura VGGNet-19.



Nota: En esta figura se puede ver la arquitectura VGGNet con 19 capas de profundidad. Fuente y elaboración propia.

Ajustes de modelos. Los ajustes de modelos a la hora de entrenar nuestra red neuronal convolucional nos permiten optimizar los parámetros de configuración del modelo.

Desajuste. Ocurre cuando el modelo planteado es demasiado sencillo que no realiza la correcta clasificación, incluso si se realizan las pruebas con los mismos datos de entrenamiento utilizados.

Ajuste adecuado. Es cuando los datos se encuentran en un punto de equilibrio, permitiendo que el modelo sea correcto, minimizando los errores y clasificando correctamente los datos.

Sobreajuste. Sucede cuando se le proporciona una gran cantidad de datos a la hora de realizar el entrenamiento, generando un sobreajuste en la clasificación del modelo. Causando predicciones incorrectas.

Entrenamiento de las redes neuronales convolucionales. Este paso es uno de los más importante a la hora de entrenar una red neuronal convolucional, en esta se tiene que configurar ciertos parámetros como son los siguientes (García, 2019):

Batch size. Consiste en configurar la cantidad de datos que el algoritmo va a analizar como muestra, del total de set de entrenamiento. Es decir, si se tiene un Batch size de 10 con un lote de 100 muestras de entrenamiento, se seleccionará las 10 primeras muestras del 1 al 10 del conjunto de datos de entrenamiento, luego, toma las segundas 10 muestras del 11 al 20 y vuelve a formar la red.

Epoch. La función de este parámetro es la de regular cuantas veces el algoritmo recorrerá todos los datos, es decir, el algoritmo tiene que recorrer todo el lote de imágenes del entrenamiento para completar una época.

Iteración. Este parámetro consiste en ajustar la cantidad de veces que los datos pasan por el algoritmo.

Matriz de confusión. La matriz de confusión resulta útil al momento de medir que tan bien se desempeña el modelo entrenado (Domínguez Pavón, 2019), consiste en sumar los valores de la diagonal y dividirlos por el total.

Tabla 3 Matriz de confusión.

		Predicción	
		Positivos	Negativos
Observaciones	Positivos	Positivos verdaderos (VP)	Negativos falsos (FN)
	Negativos	Positivos falsos (FP)	Negativos verdaderos (VN)

Nota: Matriz de confusión de 2 x 2. Fuente y elaboración propia.

En donde:

VP = Es la cantidad de valores positivos que fueron clasificados correctamente como positivos.

VN= Es la cantidad de valores negativos que fueron clasificados correctamente como negativos.

FN=Es la cantidad de valores negativos que fueron clasificados incorrectamente como negativos.

FP= Es la cantidad de valores positivos que fueron clasificados incorrectamente como positivos.

Accuracy. Nos permite identificar la exactitud con la que predice el modelo de la red neuronal creada, es decir que tan cerca está de ofrecer un resultado verdadero.

La fórmula matemática es la siguiente:

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

Recall. Métrica que a partir del total de casos positivos me permite conocer qué porcentaje de casos positivos fueron predichos correctamente.

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

Precision. Permite conocer el porcentaje de los valores positivos entre todos los positivos obtenidos.

Su fórmula matemática es la siguiente:

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

1.4.3. Definiciones

Uml. Conocido como el lenguaje unificado de modelado, permite crear modelos de diagramas de clases, diagrama de casos de usos, diagramas entidad-relación, algoritmos, etc.

Python. Lenguaje de programación flexible, tiene una línea de aprendizaje muy versátil, útil a la hora de empezar a programar, consta con una amplia comunidad y distintas librerías que se pueden implementar en cualquier proyecto. Ideal para la ciencia de datos y aprendizaje automático.

Django. El framework Django es una herramienta para facilitar el desarrollo del Back-end de aplicaciones web, gracias a su simplicidad y eficiencia es recomendable para el desarrollo a gran escala y con rapidez de distintas clases de proyectos.

Google Colab. Potente herramienta que se encuentra en la nube, ideal para el desarrollo de proyectos de aprendizaje automático, por defecto viene con librerías instaladas para la ciencia de datos, gracias a que se encuentra en la nube se consigue aprovechar los beneficios que tiene, como es la de poder utilizar una unidad de procesamiento de datos (GPU) de manera remota en nuestro equipo local, si este no cuenta con los recursos necesario, espacio en disco duro, memoria RAM, etc. El servicio de google colab nos permite resolver más rápido los algoritmos de entrenamiento de los modelos en menos tiempo.

Kaggle. Es una plataforma Web que nos proporciona recursos útiles para desarrollar prácticas de machine learning, muy utilizado en el análisis de ciencia de datos, análisis predictivo, etc. Cuenta con miles de bases de datos de distintos campos de aplicación. Ideal para principiantes que buscan extender su conocimiento en inteligencia artificial.

Imagenet. Imagenet consiste en una herramienta de base de datos que contiene millones de imágenes con miles de categorías disponibles, la cual ha permitido grandes avances en el mundo de la visión por computadora. Útil a la hora de entrenar el modelo de red neuronal convolucional.

Tensor Flow. Librería de código abierto más utilizada en el área del aprendizaje automático, contiene herramientas para la ciencia de datos, computación numérica, ideal para la construcción de redes neuronales aprovechando la flexibilidad y arquitectura que ofrece.

Transfer Learning. Técnica de aprendizaje profundo muy utilizada en modelos de redes neuronales pre-entrenadas, consiste en modificar la estructura original de la red neuronal pre-entrenada, agregando nuevas capas. de las cuales solo esas se van a entrenar y las demás capas no, ya que se prefiere conservar los pesos que se obtuvieron al ya ser pre-entrenadas.

Learning Rate. Parámetro que se utiliza en el entrenamiento de redes neuronales que permite ajustar la tasa de aprendizaje.

Data Augmentation. Técnica que permite aumentar la cantidad de datos a partir de la modificación del brillo de la imagen, el zoom, orientación, rotación, entre otros.

Data Balanceada. Cuando se refiere a Data balanceada se hace referencia a que cada una de las clases que se tengan deben tener la misma cantidad de datos para realizar la clasificación.

CAPÍTULO 2

2. METODOLOGÍA

2.1. Metodología de Investigación

En el desarrollo del proyecto se contará con metodologías de investigación que permitan identificar características y procesos para la construcción del producto de software.

2.1.1. Investigación descriptiva

La investigación de tipo descriptiva nos permite describir a partir de datos cuantitativos los resultados, enfocándose en la extracción de características de un conjunto de datos de imágenes que son transformados para posteriormente usarlos como datos de entrada para la red neuronal.

2.1.2. Investigación diagnóstica

La investigación de tipo diagnóstica se aplicó para recolectar información relacionada con el tema de estudio, permitiendo dar una solución mediante la observación, clasificación y agrupación de datos.

2.1.1. Investigación experimental

La investigación de tipo experimental se aplicó en la creación y entrenamiento de la red neuronal convolucional, haciendo uso del método heurístico ensayo y error, en donde se manipulan los datos y se observan los resultados obtenidos.

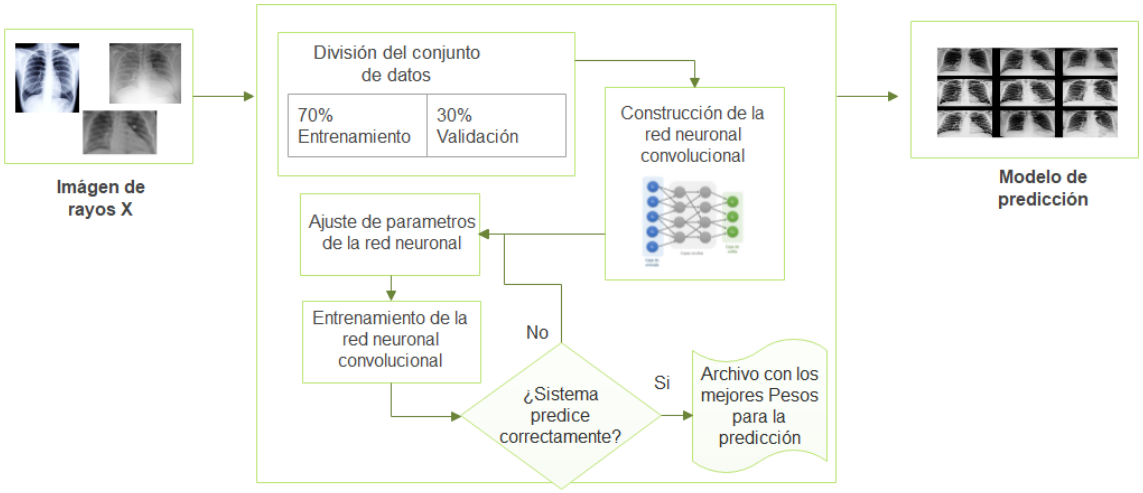
2.2. Metodología de Desarrollo

2.2.1. Red neuronal convolucional

En esta sección se explicarán los pasos que se aplicó para el diseño, construcción, entrenamiento y evaluación del modelo de red neuronal convolucional de detección de covid-19, por medio de imágenes de rayos x del tórax.

Proceso para la construcción de la CNN. De manera general se puede observar en la figura 8 los procesos que se realizan para entrenar una CNN.

Figura 8 Proceso de creación y entrenamiento de la red neuronal convolucional.



Nota: En esta figura se puede observar que la entrada del sistema son imágenes de rayos x de tórax proporcionadas por Kaggle. De las cuales es necesario dividir del total de conjunto de imágenes Normales y con Covid el 70% para el entrenamiento y 30% para la validación. Mediante los ajustes de parámetros se busca obtener los mejores pesos para el sistema de predicción. Obteniendo así el modelo CNN que se encargará de identificar los casos de pacientes que se encuentren sanos o con Covid. Fuente y elaboración propia.

A continuación, se narrarán los pasos a seguir para el entrenamiento de la red neuronal convolucional.

Paso 1 - Selección del conjunto de datos. En este paso se define el dataset que se va a utilizar para el entrenamiento, validación y prueba de la red neuronal convolucional, para ello se recurre a la plataforma de Kaggle que contiene una gran cantidad de bases de datos gratuitas que permiten resolver problemas relacionados con ciencia de datos y machine learning.

El dataset seleccionado consta de imágenes de rayos x de los tórax clasificados en radiografías con casos positivos de covid-19, junto con radiografías de neumonía y normal.

En donde para el entrenamiento de la red neuronal, solo se usará las radiografías de casos positivos a covid y normal, en el siguiente enlace se encontrará el data set en Kaggle: https://www.kaggle.com/tawsifurrahman/covid19-radiography-database.

Paso 2 - Descarga y extracción del set de datos. Una vez descargado el set de datos, se procede a extraer y seleccionar las imágenes de covid y normal.

Tabla 4 Total de imágenes del set de datos.

Total imágenes covid	Total imágenes normales
3.616 imágenes	10.192 imágenes

Nota: En esta tabla se ve la cantidad de imágenes que se tiene a nuestra disposición para el desarrollo de la red neuronal convolucional. Fuente y elaboración propia.

Paso 3 – División y carga de los datos. Para realizar el entrenamiento de la CNN se divide el set de datos en tres subconjuntos:

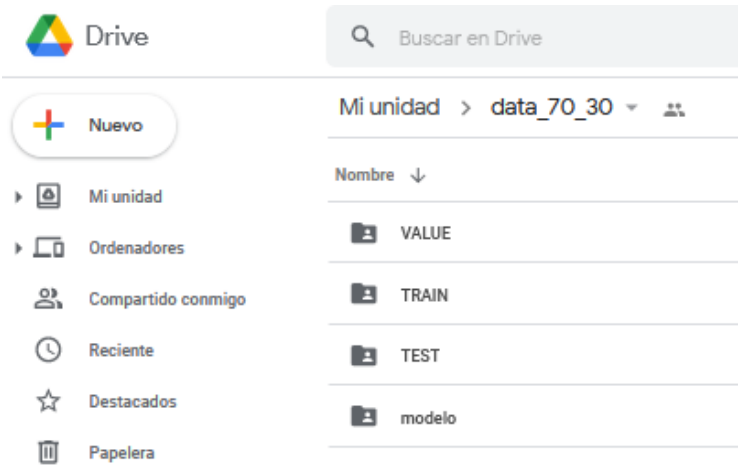
Subconjunto de prueba. Aquí se encontrarán las imágenes que se utilizarán para evaluar el desempeño del modelo entrenado, dividido en 600 imágenes de covid y 600 imágenes normales. De modo que quedan 3.016 imágenes con covid y 9.592 imágenes normales de las cuales serán divididas 70% para entrenamiento y 30 % para validación.

Subconjunto de entrenamiento. Aquí se encontrarán las imágenes que se utilizarán para entrenar el modelo, dividido en 2.609 imágenes de covid y 6.714 imágenes normales.

Subconjunto de validación. Aquí se encontrarán las imágenes que se utilizarán para validar el modelo, dividido en 887 imágenes de covid y 2.878 imágenes normales.

Una vez se han clasificado las imágenes se procede a subir las carpetas a nuestro Drive para ser utilizadas en Google Colab.

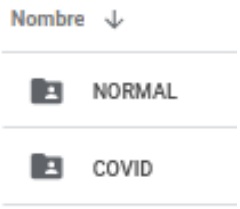
Figura 9 Carga de los datos a nuestro drive.



Nota: En esta tabla se observa que los datos se han cargado a nuestro Drive. En donde la carpeta "Train" contiene las imágenes para el entrenamiento, la carpeta "Value" las imágenes para la validación y la carpeta "Test" las de pruebas. Aparte se creó una carpeta modelo, donde se almacenarán los pesos y modelo entrenado. Fuente y elaboración propia.

En la figura 10 se puede observar que se ha dividido en dos clases una carpeta con la etiqueta "NORMAL" y otra con la etiqueta "COVID" en todos los subconjuntos de datos.

Figura 10 Clases de etiquetas, Normal y Covid.



Nota: En esta figura se observa que tiene dos clases de imágenes las normales y con covid. Fuente y elaboración propia.

Paso 4 – Definición del modelo. En este paso se ha utilizado el modelo VGGNet, de tal manera que se parte de un modelo pre-entrenado. Para ello se ha seleccionado el modelo "VGG-19", que consta de 19 capas de profundidad, usando Transfer Learning.

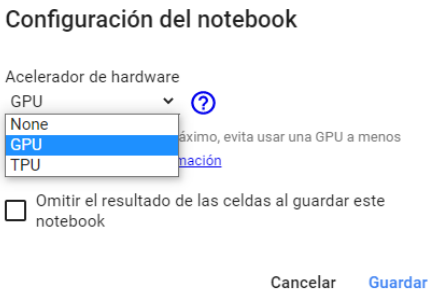
Tabla 5 Modelos VGGNet19.

Capa		Mapa de características	Tamaño
Entrada	Imagen	1	224 × 224 × 3
1	2 × Conv	64	224 × 224 × 64
	MaxPool	64	112 × 112 × 64
3	2 × Conv	128	112 × 112 × 128
	MaxPool	128	56 × 56 × 128
5	4 × Conv	256	56 × 56 × 256
	MaxPool	256	28 × 28 × 256
9	4 × Conv	512	28 × 28 × 512
	MaxPool	512	14 × 14 × 512
13	4 × Conv	512	14 × 14 × 512
	MaxPool	512	7 × 7 × 512
17	FC1(Dense)	-	4096
18	FC2(Dense)	-	1000
19	Output	-	2

Nota: En esta tabla se observa las capas de la red VGGNet. Fuente y elaboración propia.

Paso 5 – Configuración del entorno de trabajo. Aquí se va a utilizar Google Colab para codificar y entrenar la red neuronal convolucional, ya que ofrece grandes beneficios como es la utilización gratuita de la GPU en la nube permitiéndonos entrenar en menos tiempo la red neuronal convolucional.

Figura 11 Configuración del entorno de trabajo en Google Colab.



Nota: En esta figura se observa cómo se configura el entorno de Google Colab con GPU. Fuente y elaboración propia.

Paso 6 – Creación de la red neuronal con Python. En este paso se crea el algoritmo con el cual se va a entrenar la red neuronal convolucional creada.

Configuración de los parámetros de entrenamiento. Para el entrenamiento del modelo de red neuronal convolucional, se ha utilizado los siguientes parámetros:

Tabla 6 Configuración de los parámetros para el entrenamiento.

Parámetros	Descripción	Valor
Longitud y altura	Tamaño de las imágenes de entrada.	224, 224
Épocas	Cantidad de veces que se entrena la red.	40
Batch_size	Cantidad de imágenes a analizar por lote.	64
Clases	Clases de imágenes en este caso: covid y normal.	2

Nota: En esta tabla se observa los parámetros que se usaron en todos los entrenamientos de la red neuronal convolucional. Fuente y elaboración propia.

Procesamiento de las imágenes de entrenamiento. Aquí se realiza un proceso de "aumento de datos", que consiste en aumentar las imágenes durante el entrenamiento, rotando la posición de la imagen, acercando, alejando y cambiando el brillo de la imagen, para esto se utiliza la función "ImageData Generator" de Tensor Flow (TensorFlow, 2021).

A continuación, se detalla los parámetros que se utilizaron para realizar el aumento de datos.

Tabla 7 Configuración de los parámetros para el preprocesamiento de las imágenes.

Parámetros	Descripción
Rescale	Cambiar el rango de píxeles, normalizándolos de 0 a 1.
Shear_range	Intensidad de inclinación.
Zoom_range	Realiza zoom a la imagen.
Horizontal_flip	Voltear horizontalmente la imagen.

Nota: En esta tabla se observa los parámetros que se usaron para procesar y manipular las imágenes. Estos parámetros nos permiten redimensionar y cambiar el estado de la imagen para que así la red pueda reconocer mejor sus características. Fuente y elaboración propia.

Paso 7 – Entrenamiento de la red neuronal convolucional. Una vez definidos los pasos anteriores, empieza el entrenamiento de la red neuronal convolucional.

Entrenamiento 1. Para este entrenamiento se utilizó la estructura del modelo VGG19. En donde se modificó y se agregaron las siguientes capas:

Tabla 8 Entrenamiento 1: capas agregadas a la arquitectura VGG19.

Capa	valor	Función de activación
Fc1	4096	Relu
Fc1	1000	Relu

Nota: En esta tabla se aprecia que se ha agregado dos capas para ser entrenadas con nuestro set de datos. Fuente y elaboración propia.

Una vez se completó el proceso de entrenamiento de la red neuronal se obtuvieron los siguientes resultados.

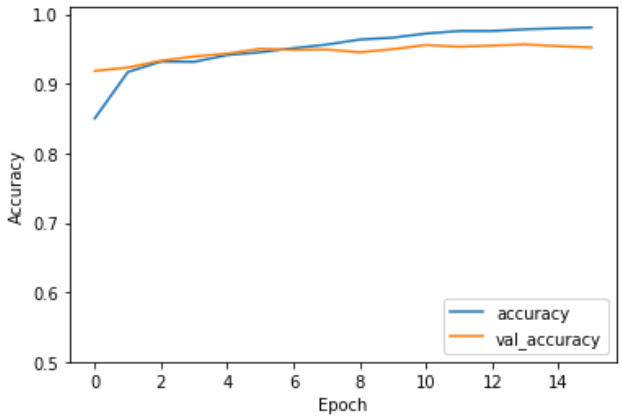
Tabla 9 Resultados del entrenamiento 1 de la CNN.

Métrica	Entrenamiento	Validación
Loss	0.1363	0.1457
Accuracy	0.9454	0.9504

Nota: En este primer entrenamiento se tuvo resultados aceptables con una exactitud de 0.9454 en los datos de entrenamiento y 0.9504 en los datos de validación. Fuente y elaboración propia.

En la figura 12 se puede observar cómo varía la precisión del entrenamiento y de la validación en cada época.

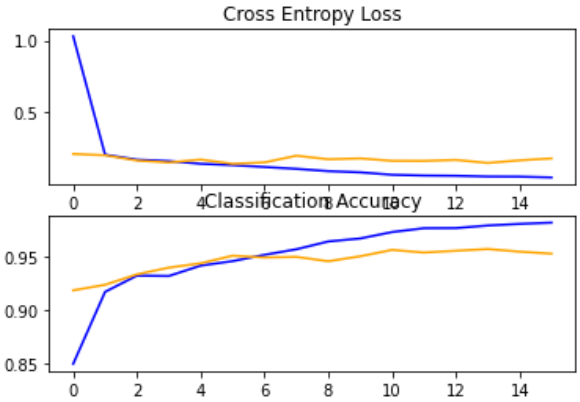
Figura 12 Entrenamiento 1: Variación de la exactitud y pérdida del entrenamiento.



Nota: En esta figura se aprecia como la exactitud del entrenamiento 1 varía en cada época. Fuente y elaboración propia.

También se detalla en la figura 13 la pérdida de entropía cruzada durante el entrenamiento. Las líneas de color azul representan el Loss y Accuracy del entrenamiento y las naranjas de la validación.

Figura 13 Entrenamiento 1: Variación de la entropía cruzada durante el entrenamiento.



Nota: En esta figura se puede observar como la red neuronal durante el entrenamiento 1 va mejorando su exactitud y disminuyendo la pérdida en cada época. Fuente y elaboración propia.

Entrenamiento 2. Para este entrenamiento se hicieron cambios en el modelo VGG19. En donde se modificó y se agregaron las siguientes capas:

Tabla 10: Entrenamiento 2: capas agregadas a la arquitectura VGG19.

Capa	valor	Función de activación
Fc1	1024	Relu
Fc2	256	Relu
Fc3	32	Relu

Nota: En esta tabla se observa que se ha agregado tres capas para ser entrenadas con nuestro set de datos. Fuente y elaboración propia.

Una vez se completó el proceso de entrenamiento de la segunda red neuronal se obtuvieron los siguientes resultados.

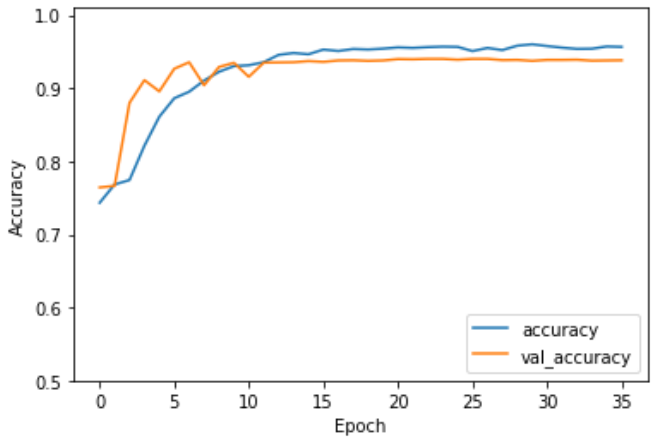
Tabla 11 Resultados del entrenamiento 2 de la CNN.

Métrica	Entrenamiento	Validación
Loss	0.1312	0.1687
Accuracy	0.9509	0.9402

Nota: En este segundo entrenamiento se tuvo resultados aceptables, con una exactitud de 0.9509 en los datos de entrenamiento y 0.9404 en los datos de validación, pero no se obtuvo una mejora respecto al entrenamiento 1. Fuente y elaboración propia.

En la figura 14 se puede observar cómo varía la precisión del entrenamiento y de la validación en cada época.

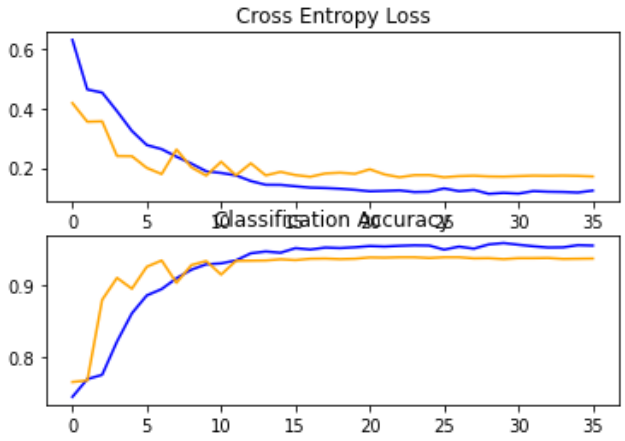
Figura 14 Entrenamiento 2: Variación de la exactitud y pérdida del entrenamiento.



Nota: En esta figura se observa como la exactitud del entrenamiento 2 varía en cada época. Fuente y elaboración propia.

También se detalla en la figura 15 la pérdida de entropía cruzada durante el entrenamiento. Las líneas de color azul representan el Loss y Accuracy del entrenamiento y las naranjas de la validación.

Figura 15 Entrenamiento 2: Variación de la entropía cruzada durante el entrenamiento.



Nota: En esta figura se puede observar como la red neuronal del entrenamiento 2 va mejorando su exactitud y disminuyendo la pérdida en cada época. Fuente y elaboración propia.

Paso 8 – Prueba de red neuronal convolucional. Para verificar que tan eficiente es el modelo CNN creado, se realizan pruebas para identificar la precisión con la que predice la CNN entrenada. Para todas las pruebas se utilizaron 600 imágenes con covid y 600 imágenes normales.

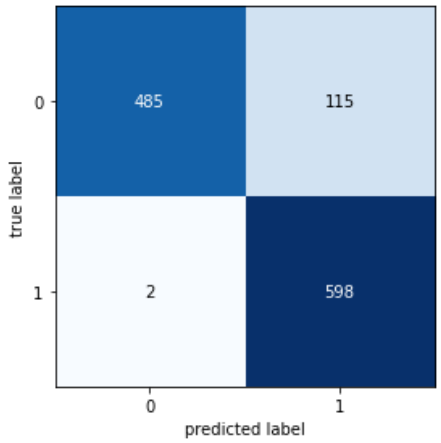
Prueba Entrenamiento 1. En la primera prueba se obtiene como resultado una exactitud del 0.9025.

Tabla 12 Prueba de predicción del entrenamiento 1.

Métrica	**Covid (0)**	**Normal (1)**
Precision	0.9959	0.8387
Recall	0.8083	0.9967
f1-score	0.8924	0.9109

Nota: En esta tabla se puede observar que tan bien predice la red neuronal del entrenamiento 1, a partir de las métricas de Precision, Recall y f1-score. Fuente y elaboración propia.

Figura 16 Matriz de confusión de las pruebas realizadas del entrenamiento 1.



Nota: En esta figura se puede observar que de las 600 imágenes de pruebas con covid la red predijo 485 correctamente y 115 incorrectamente y de las 600 imágenes normal la red predijo 598 correctamente y 2 incorrectamente. Fuente y elaboración propia.

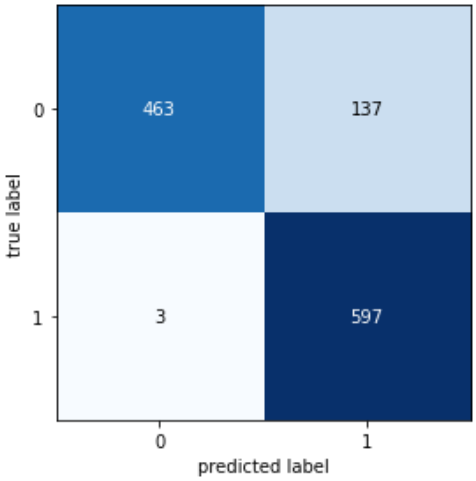
Prueba entrenamiento 2. En la prueba 2 se obtiene como resultado una exactitud del 0.8833.

Tabla 13 Prueba de predicción del entrenamiento 2.

Métrica	**Covid (0)**	**Normal (1)**
Precision	0.9936	0.8134
Recall	0.7717	0.9950
f1-score	0.8687	0.8951

Nota: En esta tabla se puede observar que tan bien predice la red neuronal del entrenamiento 2, a partir de las métricas de Precision, Recall y f1-score. Pero los resultados obtenidos no son mejores que los del entrenamiento 1. Fuente y elaboración propia.

Figura 17 Matriz de confusión de las pruebas realizadas del entrenamiento 2.



Nota: En esta figura se puede observar que de las 600 imágenes de pruebas con covid la red predijo 463 correctamente y 137 incorrectamente y de las 600 imágenes normal la red predijo 597 correctamente y 3 incorrectamente. Fuente y elaboración propia.

Resultados de las pruebas. Aquí se observa los resultados que se obtienen de las CNN entrenadas.

Tabla 14 Comparativa entre los entrenamientos de las CNN realizadas.

Red neuronal entrenada	Exactitud
Entrenamiento CNN 1	0.9025
Entrenamiento CNN 2	0.8833

Nota: En esta tabla se puede observar que el entrenamiento 1 tiene una exactitud del 0.9025 y la del entrenamiento 2 tiene una exactitud del 0.8833. Fuente y elaboración propia.

Entre todas la que mejor realizó las predicciones fue la del primer entrenamiento como un 0.9025 de exactitud, pero no predice correctamente las imágenes que son positivos a Covid, por lo tanto, se ha identificado que los datos no se encuentran debidamente balanceados, es decir hay más imágenes normales que de Covid, causando un sobreajuste en el entrenamiento de la red neuronal convolucional.

Por consiguiente, para nivelar está diferencia de datos, se procede a aumentar la cantidad de imágenes de covid, de las cuales se han recopilado de diferentes sitios que contienen imágenes de rayos X de tórax con covid.

A continuación, se incluyen los enlaces donde se encuentran alojadas las imágenes:

Tabla 15 Enlaces de imágenes de rayos x con covid.

Nombre Data Set	Enlace
Chest X-ray (Covid-19 & Pneumonia)	https://www.kaggle.com/prashant268/chest-xray-covid19-pneumonia/
COVID-19 Detection X-Ray Dataset	https://www.kaggle.com/darshan1504/covid19-detection-xray-dataset
COVID-19 X rays	https://www.kaggle.com/andrewmvd/convid19-x-rays?select=X+rays
Covid-19 Image Dataset	https://www.kaggle.com/pranavraikokte/covid19-image-dataset?select=Covid19-dataset
COVID-19 Chest X-ray Image Dataset	https://www.kaggle.com/alifrahman/covid19-chest-xray-image-dataset
Chest Xray for covid-19 detection	https://www.kaggle.com/fusicfenta/chest-xray-for-covid19-detection

Nota: En esta tabla se proporciona los enlaces de donde se obtuvo las demás imágenes de covid-19 para corregir el balance de datos y sobreajuste. Fuente y elaboración propia.

Por último, se utiliza “una base de datos de código abierto mantenida por el Dr. Joseph Paul Cohen” (Roy, 2020). En donde se adjunta el enlace de las imágenes:

https://drive.google.com/drive/folders/1bPRw1C7JtL47fbMtKtwLuJZP4pDvcF6B

Aumento de datos y corrección de sobreajuste. Como bien se menciona, se necesitan más imágenes para poder balancear los datos y evitar el sobreajuste. Por lo tanto, se suben al drive todas las imágenes de covid que se han recolectado. En la tabla 16 se observa la cantidad de imágenes agregadas.

Tabla 16 Cantidad de imágenes agregadas a nuestro set de datos.

Imágenes de covid para entrenamiento	**Imágenes de covid para validación**
980 imágenes	422 imágenes

Nota: En esta tabla se puede observar que se ha aumentado 980 imágenes para el entrenamiento y 422 imágenes para la validación. Por lo tanto, nuestro set de imágenes de covid para el entrenamiento aumenta a 3.589 y para la validación en 3.300. Fuente y elaboración propia.

A continuación, se describen los resultados del entrenamiento con el aumento de imágenes en la tabla 17.

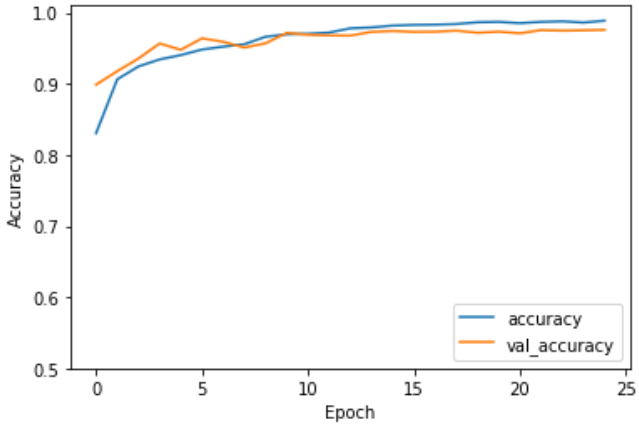
Tabla 17 Resultados del entrenamiento con el aumento de datos de la CNN.

Métrica	**Entrenamiento**	**Validación**
Loss	0.0486	0.9743
Accuracy	0.9819	0.0794

Nota: En la época 15 se obtuvo una exactitud de 0.9819 en los datos de entrenamiento y 0.9743 en los datos de validación, por lo cual este modelo se guardó automáticamente como el mejor del entrenamiento. Fuente y elaboración propia.

En la figura 18 se observa cómo varía la precisión del entrenamiento y de la validación en cada época aplicado el aumento de datos.

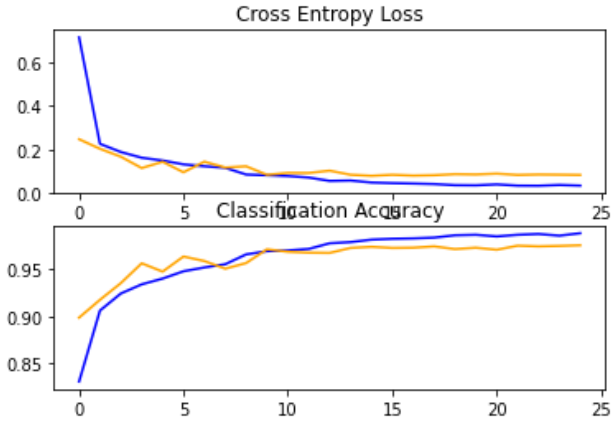
Figura 18 Entrenamiento con el aumento de datos: Variación de la exactitud y pérdida del entrenamiento.



Nota: En esta figura se observa como la exactitud del entrenamiento con el aumento de datos varía en cada época. Fuente y elaboración propia.

También se detalla en la figura 19 la pérdida de entropía cruzada durante el entrenamiento con el aumento de datos. Las líneas de color azul representan el Loss y Accuracy del entrenamiento y las naranjas de la validación.

Figura 19 Entrenamiento con aumento de datos: Variación de la entropía cruzada durante el entrenamiento.



Nota: En esta figura se puede observar como la red neuronal durante el entrenamiento con aumento de datos va mejorando su exactitud y disminuyendo la pérdida en cada época. Fuente y elaboración propia.

Prueba Entrenamiento con aumento de datos. En la prueba que se realizó se obtiene como resultado una exactitud del 0.9683.

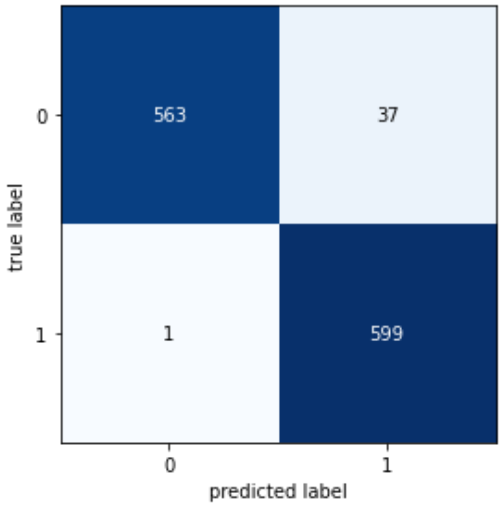
Tabla 18 Prueba de predicción del entrenamiento con aumento de datos.

Métrica	Covid (0)	Normal (1)
Precision	0.9982	0.9418
Recall	0.9383	0.9983
f1-score	0.9674	0.9693

Nota: En esta tabla se puede observar que tan bien predice la red neuronal del entrenamiento con el aumento de datos, a partir de las métricas de Precision, Recall y f1-score. Fuente y elaboración propia.

A continuación, se puede observar en la figura 20 los resultados que se obtienen a partir de la matriz de confusión.

Figura 20 Matriz de confusión de las pruebas realizadas del entrenamiento con aumento de datos.



Nota: En esta figura se puede observar que de las 600 imágenes de pruebas con covid la red predijo 563 correctamente y 37 incorrectamente y de las 600 imágenes normal la red predijo 599 correctamente y 1 incorrectamente, notando una gran mejora a diferencia de las pruebas anteriores. Fuente y elaboración propia.

2.2.2. Aplicación web

El desarrollo de software por lo general se encuentra relacionado con metodologías de desarrollo que permiten optimizar tiempo y recursos. Permitiendo que el o los integrantes de un equipo sean más productivos, por consiguiente, adoptar metodologías resulta útil a la hora de desarrollar un proyecto.

Para el desarrollo de la aplicación web se utilizará la metodología de cascada la cual consiste en ir avanzando fase por fase, es decir, mientras no se haya culminado una fase no se puede saltar a la otra fase si la predecesora no se ha culminado.

Fase 1 – **Análisis de requerimientos.** El presente proyecto tiene la necesidad de cubrir la problemática sobre la detección temprana de covid-19, por lo cual se debe implementar una aplicación web que permita el ingreso de imágenes de rayos-x para el módulo de predicción de covid-19.

Requerimientos funcionales.

Tabla 19 Requerimiento-001

Nombre:	Inicio de sesión.	Fecha:	5 de octubre del 2021
Identificación:	REQ-001	Tipo:	Funcional
Descripción:			
La aplicación web permitirá el inicio de sesión a partir de credenciales de acceso como son un nombre de usuario y clave. Al crear un médico o paciente el usuario será el correo y la clave su número de cédula.			

Nota: En esta tabla se establece el requerimiento del usuario. Fuente y elaboración propia.

Tabla 20 Requerimiento-002

Nombre:	Módulos de la aplicación	Fecha:	5 de octubre del 2021
Identificación:	REQ-002	Tipo:	Funcional
Descripción:			
La aplicación web debe contar con los siguientes módulos: Grupos. – Permite gestionar los grupos de usuarios Menú. – Permite gestionar los menús de opción de la aplicación Control de acceso al menú. – Permite activar o desactivar menús de la aplicación de acuerdo al grupo de usuario que pertenezca. Usuarios. – Permite la gestión de usuarios. Pacientes. -Permite la gestión de usuarios pertenecientes al grupo paciente. Médicos. - Permite la gestión de usuarios pertenecientes al grupo médico. Especialidades médicas. - Permite la gestión de las especialidades de los médicos. Reportes. – Permite visualizar gráficamente los reportes de casos de covid-19. Análisis radiográfico. – Permite gestionar y realizar el análisis de las radiografías de rayos x. Mis resultados. – Permite visualizar a los pacientes los resultados de su análisis realizado. vacunas. – Permite gestionar las distintas vacunas existentes contra el covid.			

Nota: En esta tabla se establece el requerimiento del usuario. Fuente y elaboración propia.

Tabla 21 Requerimiento-003

Nombre:	Grupos y permisos de usuarios.	Fecha:	5 de octubre del 2021
Identificación:	REQ-003	Tipo:	Funcional
Descripción:			
La aplicación web debe contar con 3 tipos de grupos de usuario: Administrador, Paciente y Médico. A cada uno de ellos se podrá asignar los permisos respectivos. Administrador. – Tendrá permiso a todos los módulos de la aplicación y acciones de agregar, editar, ver y eliminar. Médico. – Tendrá permiso a los módulos de pacientes, reportes y análisis radiográfico. Paciente. – Sólo tendrá permiso al módulo de mis resultados.			

Nota: En esta tabla se establece el requerimiento del usuario. Fuente y elaboración propia.

Tabla 22 Requerimiento-004

Nombre:	Registro de pacientes	Fecha:	5 de octubre del 2021
Identificación:	REQ-004	Tipo:	Funcional
Descripción:			
Al momento de registrar un paciente en la aplicación se debe ingresar los datos personales como: apellidos, nombre, cédula, dirección, género, fecha de nacimiento, correo electrónico, si se encuentra vacunado contra covid-19 si es de ser así que permita seleccionar el tipo de vacuna que se aplicó.			

Nota: En esta tabla se establece el requerimiento del usuario. Fuente y elaboración propia.

Tabla 23 Requerimiento-005

Nombre:	Registro de médicos	Fecha:	5 de octubre del 2021
Identificación:	REQ-005	Tipo:	Funcional
Descripción:			
Al momento de registrar un médico en la aplicación se debe ingresar los datos personales como: apellidos, nombre, cédula, dirección, género, fecha de nacimiento, correo electrónico y su especialidad.			

Nota: En esta tabla se establece el requerimiento del usuario. Fuente y elaboración propia.

Tabla 24 Requerimiento-006

Nombre:	Reportes estadísticos	Fecha:	5 de octubre del 2021
Identificación:	REQ-006	Tipo:	Funcional
Descripción:			
La aplicación web en el módulo de reportes debe mostrar gráficos estadísticos totales y por años. También, el porcentaje de casos positivos y negativos de covid, total de casos por mes, por género y por edad.			

Nota: En esta tabla se establece el requerimiento del usuario. Fuente y elaboración propia.

Tabla 25 Requerimiento-007

Nombre:	Predicción de covid por medio de radiografías de tórax.	Fecha:	5 de octubre del 2021
Identificación:	REQ-007	Tipo:	Funcional
Descripción:			
La aplicación web debe contar con un módulo que permita a los médicos realizar las pruebas de covid por medio de radiografías de tórax, en donde se pueda subir la imagen digitalizada de la radiografía y guardar los resultados de la prueba.			

Nota: En esta tabla se establece el requerimiento del usuario. Fuente y elaboración propia.

Requerimientos no funcionales.

Tabla 26 Requerimiento-008

Nombre:	Interfaz gráfica de usuario	Fecha:	5 de octubre del 2021
Identificación:	REQ-008	Tipo:	No funcional
Descripción:			
La aplicación debe ser responsive para cualquier tipo de dispositivo, intuitiva y de fácil uso.			

Nota: En esta tabla se establece el requerimiento del usuario. Fuente y elaboración propia.

Tabla 27 Requerimiento-009

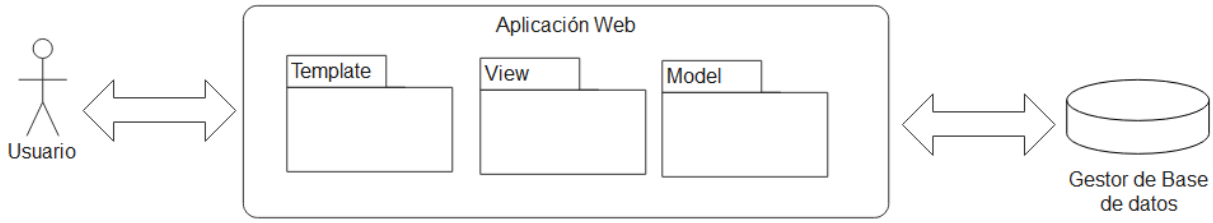
Nombre:	Tiempo de respuesta	Fecha:	5 de octubre del 2021
Identificación:	REQ-009	Tipo:	No funcional
Descripción:			
Los tiempos de respuesta de la aplicación web deben ser de inmediato.			

Nota: En esta tabla se establece el requerimiento del usuario. Fuente y elaboración propia.

Fase 2 – **Diseño.** En esta fase se diseña el modelo lógico de la aplicación web.

Arquitectura de la aplicación. La aplicación web al ser construida con el framework Django sigue la arquitectura modelo, vista, template.

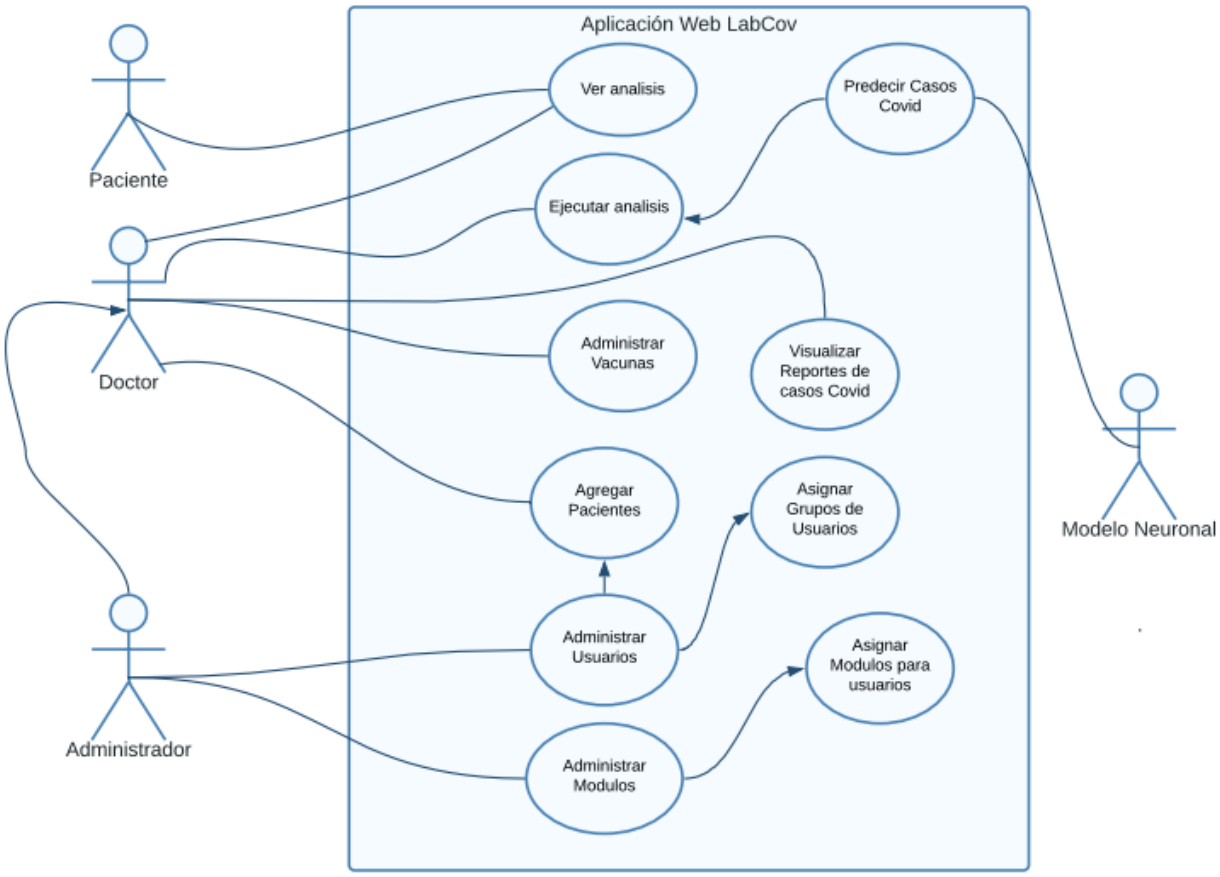
Figura 21 Arquitectura de la aplicación web



Nota: En esta figura se muestra como es la arquitectura de la aplicación web, en dónde los usuarios interactúan con la aplicación, que se encuentra construido siguiendo la arquitectura modelo, vista, template de Django. Fuente y elaboración propia.

Diagrama de casos de uso. En base a los requerimientos funcionales que se plantearon en la fase anterior, se diseña mediante diagrama de casos de usos el funcionamiento de la aplicación web.

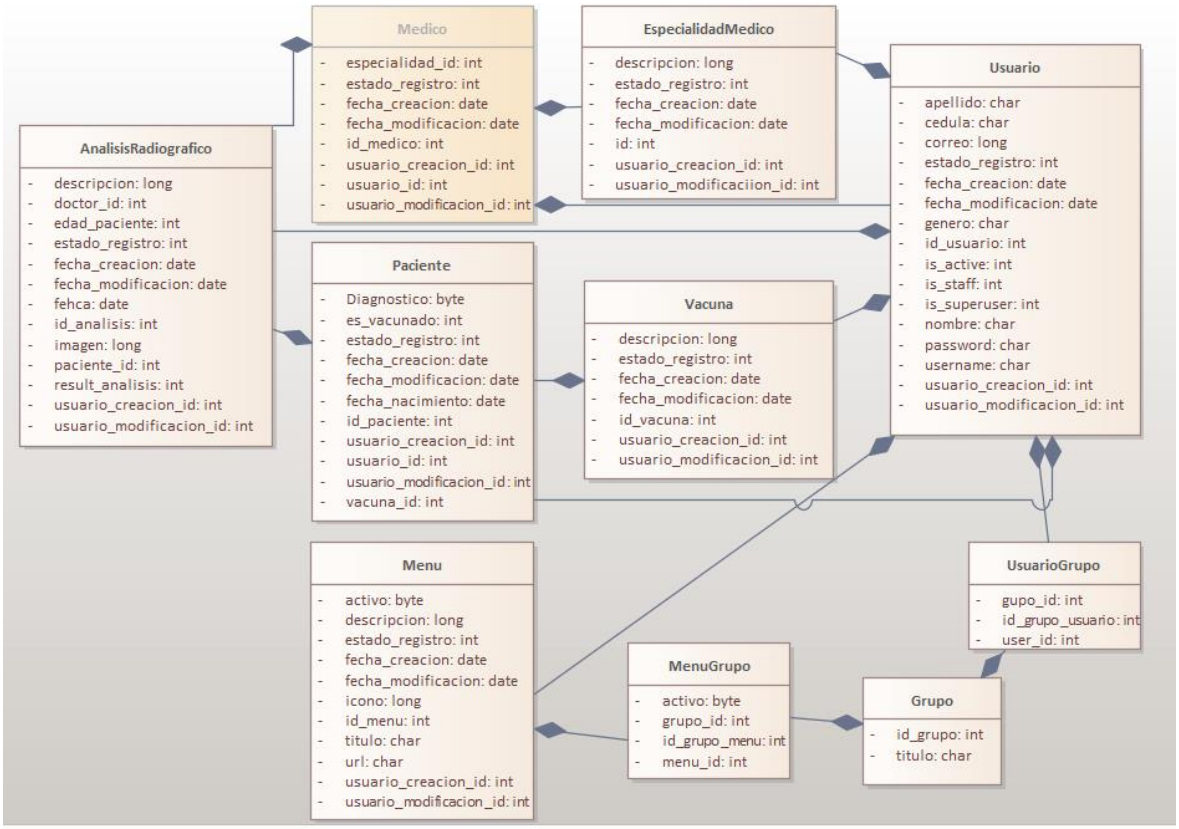
Figura 22 Diagrama de caso de uso de la aplicación web.



Nota: En esta figura se puede observar que los pacientes solo pueden acceder al módulo de análisis para visualizar los pruebas de covid que se han realizado, Los médicos pueden acceder a los módulos de ejecutar análisis y módulo de vacunas, pueden agregar pacientes y por último visualizar los reportes de los casos de covid. Por otro lado, el administrador del sistema podrá administrar a los usuarios y los módulos o menú de la aplicación. Fuente y elaboración propia.

Diagrama de clases. Con la finalidad de modelar la estructura de los datos se diseña el diagrama de clases de la aplicación.

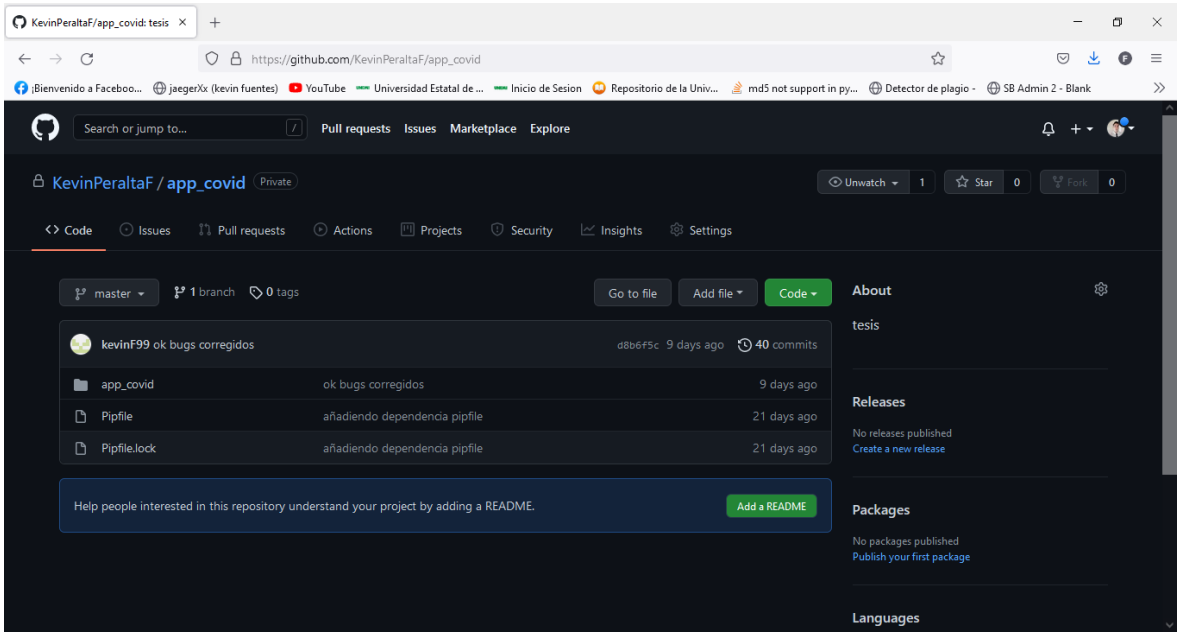
Figura 23 Diagrama de clases de la aplicación



Nota: En esta figura se muestra el diagrama de clases de la aplicación web. Fuente y elaboración propia.

Fase 3 – **Programación.** En esta fase se configura inicialmente el entorno de trabajo, la conexión con la base de datos y se crean los directorios en donde se alojarán los templates de la aplicación. Además de la creación del repositorio en GitHub para llevar el historial de cambios y poder trabajar colaborativamente.

Figura 24 Creación del repositorio en GitHub.



Nota: En esta figura se puede observar la creación del repositorio en GitHub con la finalidad de poder gestionar las distintas versiones del proyecto. Fuente y elaboración propia.

Fase 4 – **Pruebas y validación.** Luego de la etapa de implementación se realizarán las pruebas de funcionamiento del sistema con el fin que este cumpla con los requerimientos funcionales y no funcionales. Con la finalidad de detectar posibles Bugs en el sistema y poder corregirlos para que el sistema funcione correctamente.

Cronograma de actividades

Tabla 28 Cronograma de actividades

Actividades	Meses (Semana)											
	Octubre				Noviembre				Diciembre			
	Semana 1	Semana 2	Semana 3	Semana 4	Semana 1	Semana 2	Semana 3	Semana 4	Semana 1	Semana 2	Semana 3	Semana 4
Análisis de requerimientos												
Diseño de la aplicación web												
Creación de la red neuronal convolucional												
Entrenamiento de la red neuronal convolucional												
Programación de la aplicación web												
Pruebas												

Nota: En esta tabla se puede observar las actividades realizadas a lo largo del desarrollo del proyecto. Fuente y elaboración propia.

CAPÍTULO 3

3. PROPUESTA DE SOLUCIÓN

3.1. Análisis de factibilidad

A continuación, se analizará la factibilidad técnica del proyecto en donde se especificarán los recursos, herramientas, costos y tecnologías utilizadas.

3.1.1. Factibilidad técnica

Hardware

Tabla 29 Recursos de hardware utilizados en el proyecto.

Hardware	Características
Laptop HP	• Intel(R) Core (TM) i3-1005G1 CPU @ 1.20GHz • 8,00 GB RAM • 240 GB SSD

Nota: En esta tabla se puede observar los recursos de hardware que se utilizaron para el desarrollo del proyecto. Fuente y elaboración propia.

Software

Tabla 30 Recursos de software utilizados en el proyecto.

Software	Versión
Python	3.7
Django	3.2.9
Bootstrap	4.2.1
PostgreSQL	13
Google Colab	-
Tensor Flow	2.7
Visual Code	1.62.3

Nota: En esta tabla se puede observar los recursos de software que se utilizaron para el desarrollo del proyecto. Fuente y elaboración propia.

3.1.2. Factibilidad económica

Hardware

Tabla 31 Costo en recursos de hardware.

Hardware	Costo
Laptop HP	$750

Nota: En esta tabla se puede observar los costos de los recursos de hardware que se utilizaron para el desarrollo del proyecto. Fuente y elaboración propia.

Software

Tabla 32 Costo en recursos de software.

Software	Costo
Python	$0
Django	$0
Bootstrap	$0
PostgreSQL	$0
Google Colab	$0
Tensor Flow	$0
Visual Code	$0

Nota: En esta tabla se puede observar los costos en recursos de software que se utilizaron para el desarrollo del proyecto. Fuente y elaboración propia.

Recurso Humano

Tabla 33 Costo en recurso humano para el desarrollo del proyecto.

Recurso Humano	Cantidad	Tiempo	Costo
Programador	1	3 meses	$1800
Ingeniero en aprendizaje automático	1	3 meses	$2400

Nota: En esta tabla se puede observar los recursos humanos que se utilizaron para el desarrollo del proyecto. Fuente y elaboración propia.

Costo total

Tabla 34 Costo total del desarrollo del proyecto.

DETALLE	COSTO
Hardware	$750
Software	$0
Recurso Humano	$4.200
Total	**$4.950**

Nota: En esta tabla se puede observar el costo total del desarrollo del proyecto. Fuente y elaboración propia.

Análisis de costo y beneficios

El desarrollo del proyecto tendrá un costo total de $4.950 dólares. En donde la aplicación web permitirá automatizar los procesos de detección de covid-19 en los pacientes a través de imágenes de radiografías de tórax.

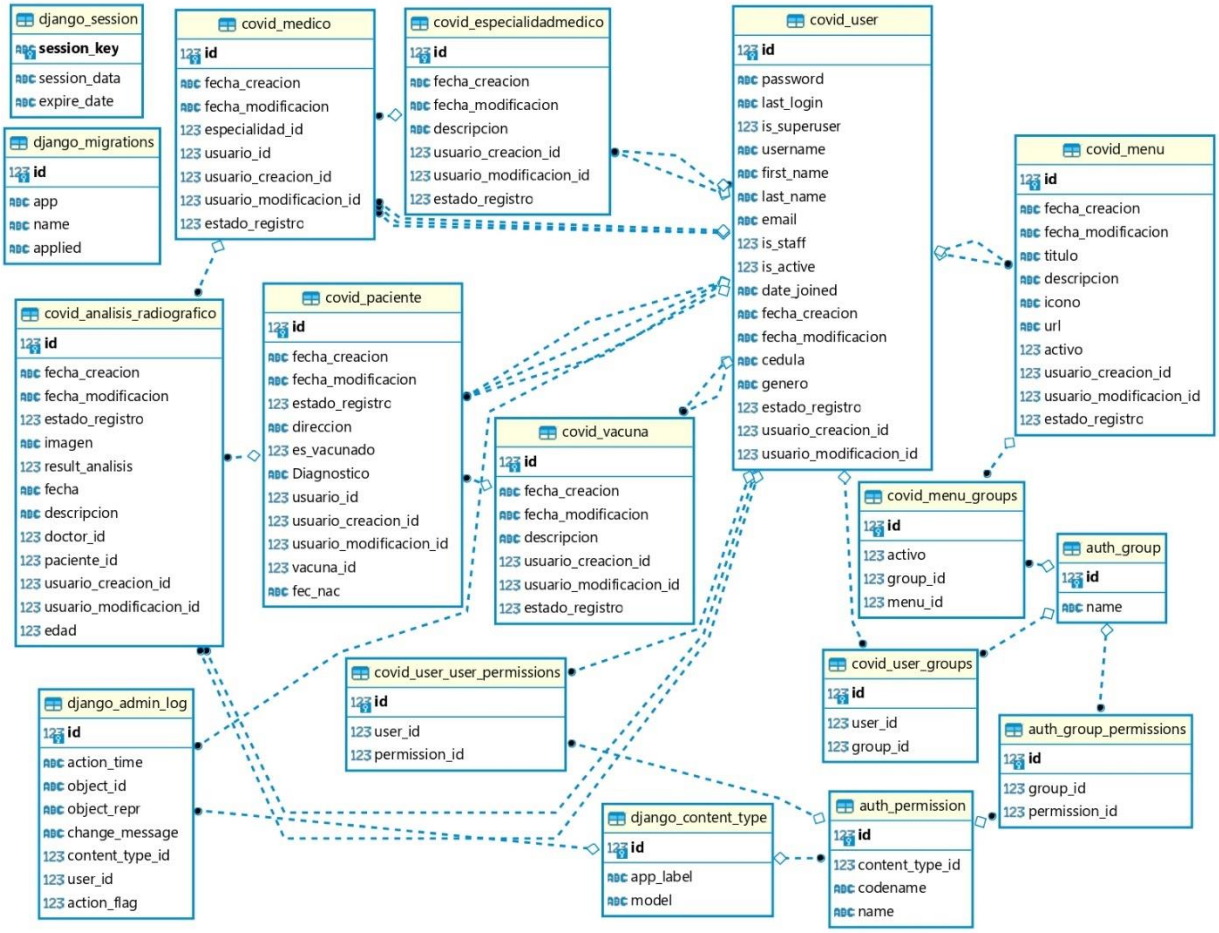
Entre los beneficios del desarrollo del proyecto está que se utilizan tecnologías y herramientas open source, que no necesitan de ninguna licencia para operar y no generarían gastos adicionales.

Por otro lado, el sistema también permite obtener informes de los casos de covid que se han confirmado con la finalidad de observar el comportamiento del virus, logrando identificar el número de pacientes confirmados con covid anualmente, mensualmente, por género y edad.

3.2. Propuesta

3.2.1. Base de datos de la aplicación

Figura 25 Diagrama de la base de datos de la aplicación web.



Nota: En esta figura se puede observar el diagrama de entidad relación de la aplicación web. Fuente y elaboración propia.

3.2.2. Funcionamiento de la aplicación

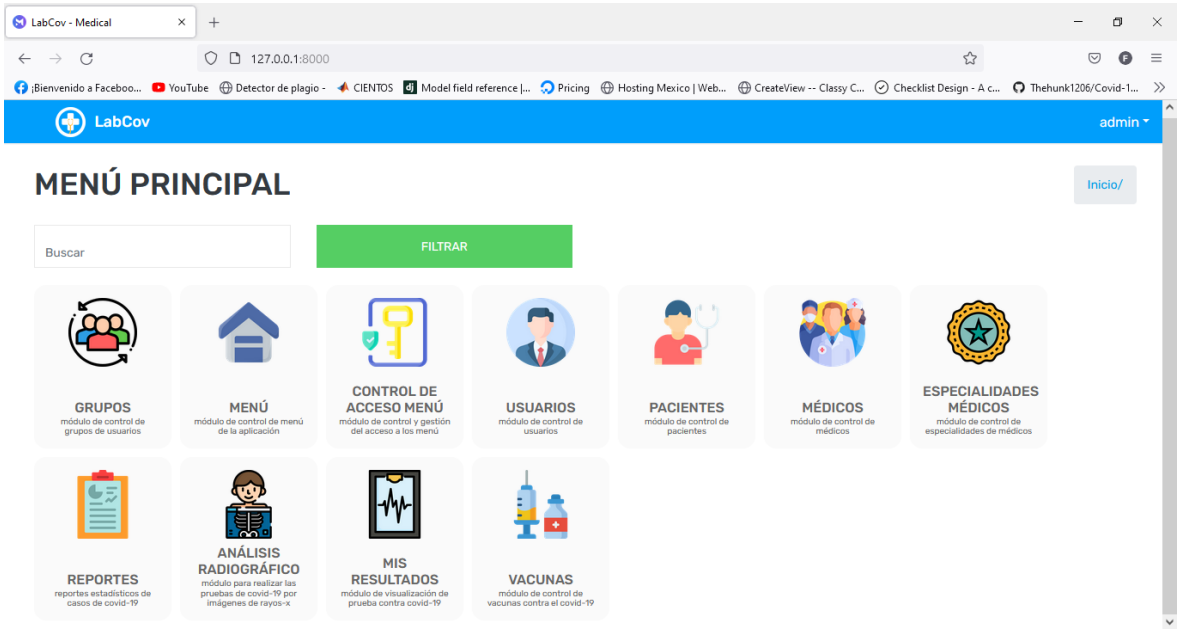
A continuación, en la tabla 35 se describen las funcionalidades de los módulos de la aplicación.

Tabla 35 Funcionamiento de los módulos de la aplicación.

Módulo de la aplicación	**Descripción**	**Grupo de usuario**
Grupos	En este módulo se crean los grupos de usuarios, es decir los roles de cada usuario.	Administrador.
Menú	En este módulo se gestionan y crean los menús de la aplicación para cada grupo de usuarios.	Administrador.
Control de acceso menú	En este módulo se puede activar o desactivar los menús.	Administrador.
Usuarios	En este módulo se gestionan y crea usuarios para la aplicación	Administrador.
Pacientes	En este módulo se gestionan y crean pacientes.	Médico.
Médicos	En este módulo se gestionan y crean médicos de la aplicación.	Administrador.
Especialidades Médicos	En este módulo se gestiona y crean las especialidades de los médicos.	Administrador.
Reportes	En este módulo se puede visualizar los reportes estadísticos.	Médico.
Análisis Radiográficos	En este módulo se suben la radiografía de tórax para analizar si tiene covid.	Médico.
Mis resultados	En este módulo los pacientes podrán ver los resultados de sus análisis.	Paciente.
Vacunas	En este módulo se gestionan y crean las vacunas que existen contra el covid-19.	Médico.
Perfil	En esta opción se puede ver información del usuario que se encuentre con sesión iniciada en la aplicación.	Todos los usuarios.
Cambiar contraseña	En esta opción se podrá cambiar de contraseña.	Todos los usuarios.

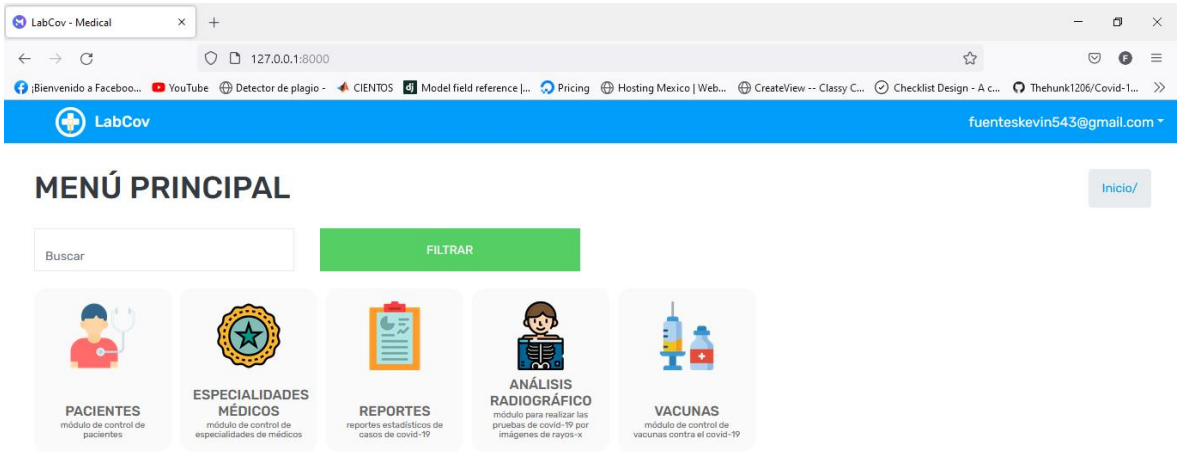
Nota: En esta tabla se describe el funcionamiento de los módulos de la aplicación web desarrolladas. Fuente y elaboración propia.

Figura 26 Menú de la aplicación.



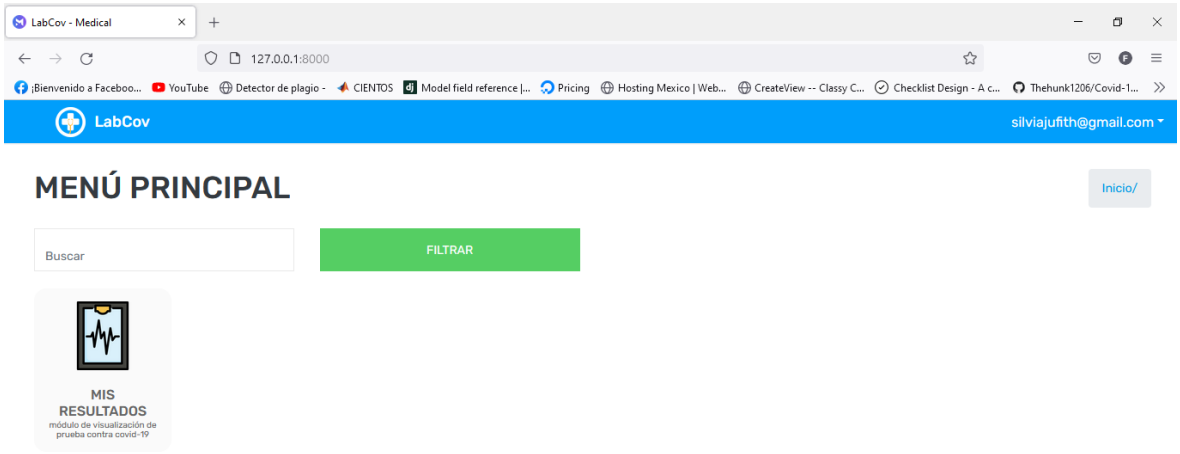
Nota: En esta figura se muestran los menús de la aplicación. Fuente y elaboración propia.

Figura 27 Menú de la aplicación para el grupo de usuarios médicos.



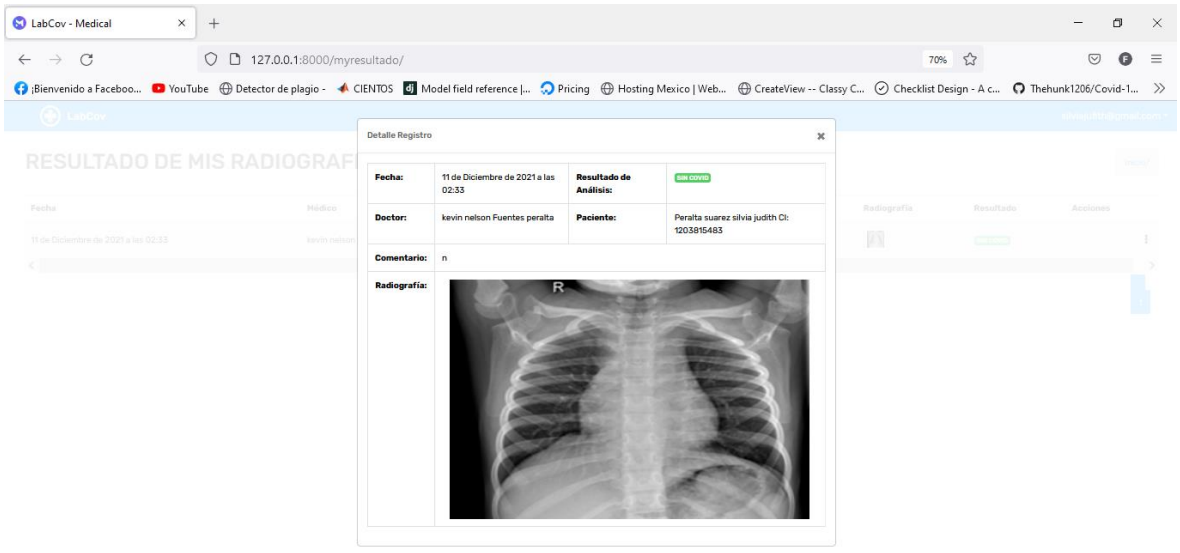
Nota: En esta figura se muestran los menús de la aplicación a los que pueden acceder los médicos. Fuente y elaboración propia.

Figura 28 Menú de la aplicación para el grupo de usuarios pacientes.



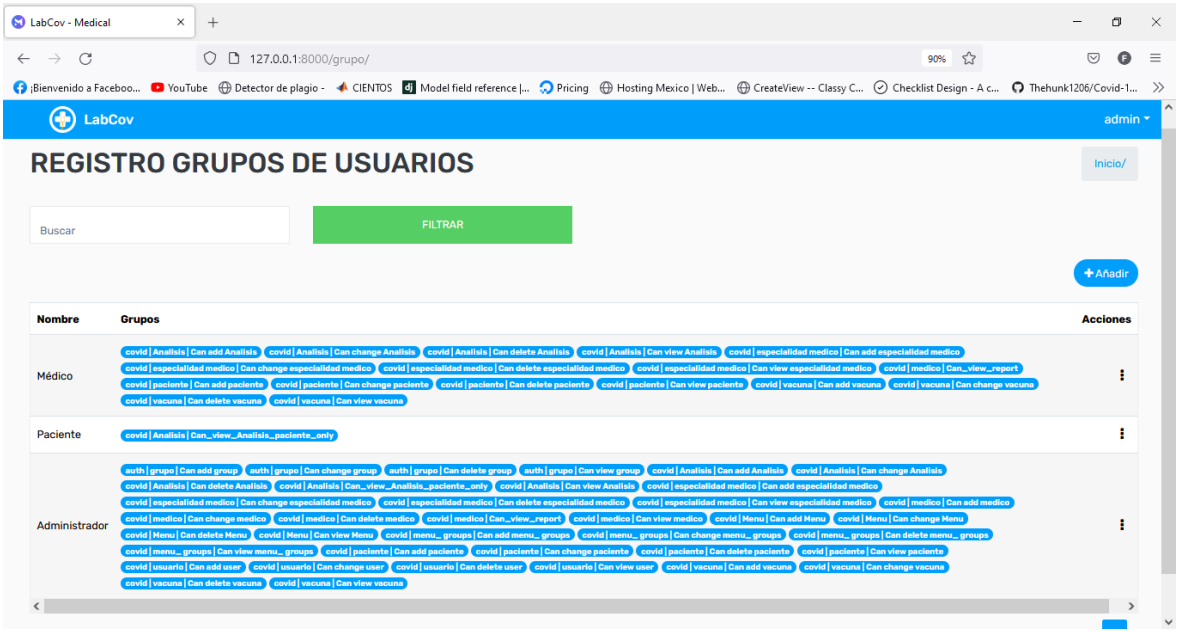
Nota: En esta figura se muestran los menús de la aplicación a los que pueden acceder los pacientes. Fuente y elaboración propia.

Figura 29 Módulo mis resultados.



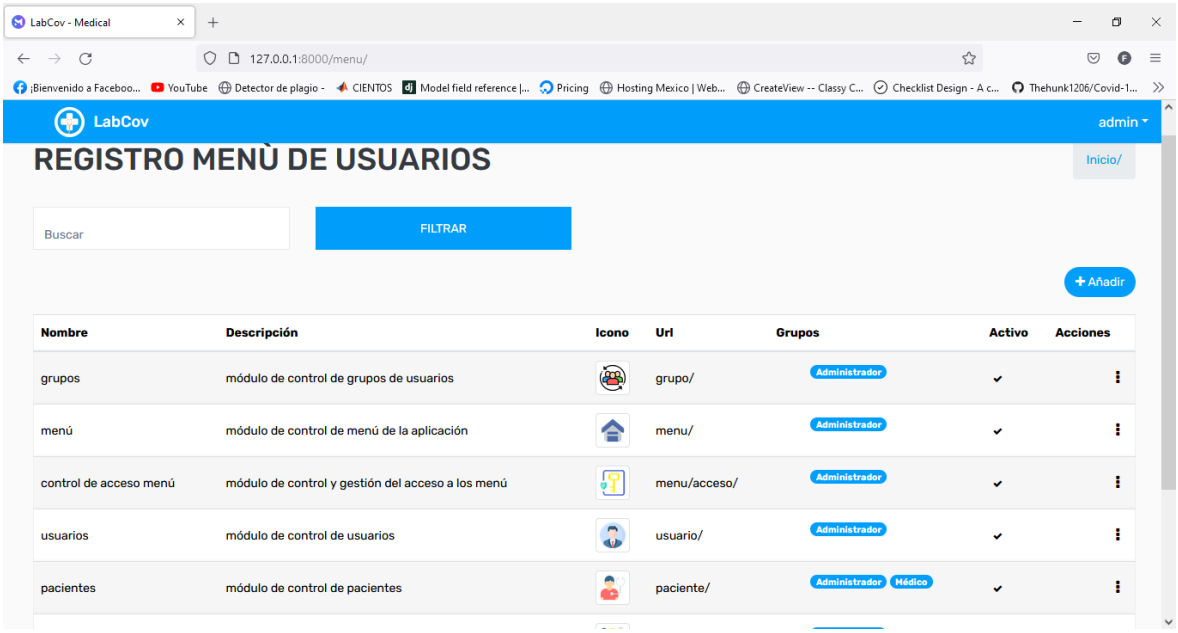
Nota: En esta figura se muestran los resultados de los análisis que el paciente se ha realizado, sólo él puede acceder a este módulo. Fuente y elaboración propia.

Figura 30 Módulo Grupos de usuarios.



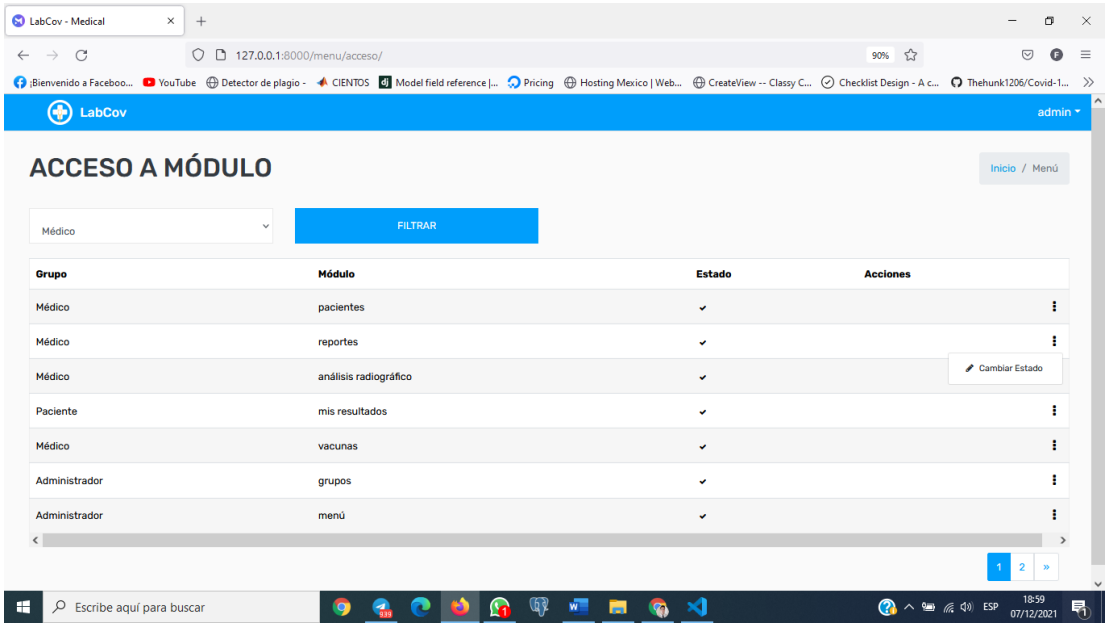
Nota: En esta figura se muestra el módulo de control y gestión de grupos de usuarios, aquí se puede crear los grupos y asignar los permisos a los que se tendrá acceso. Fuente y elaboración propia.

Figura 31 Módulo menú.



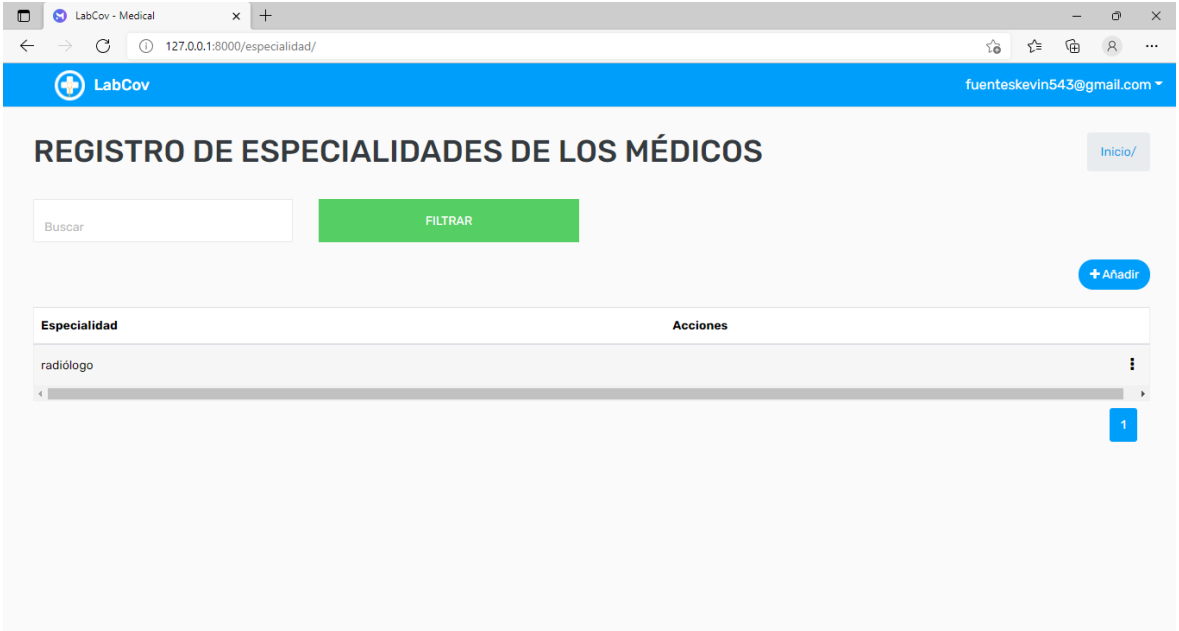
Nota: En esta figura se muestra el módulo menú, aquí se crean los menús y se define al grupo que pertenece cada menú. Fuente y elaboración propia.

Figura 32 Módulo accesos a los módulos.



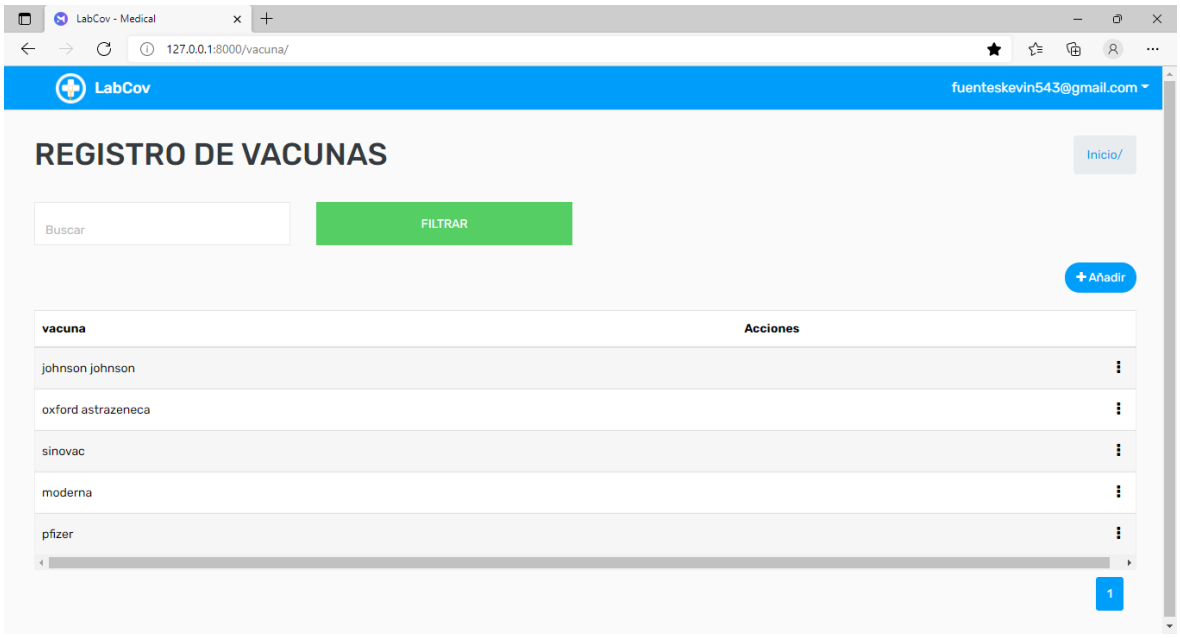
Nota: En esta figura se muestra el módulo acceso a módulo, Aquí se puede activar o desactivar la visibilidad del menú de cada grupo de usuario. Fuente y elaboración propia.

Figura 33 Módulo especialidades de médicos.



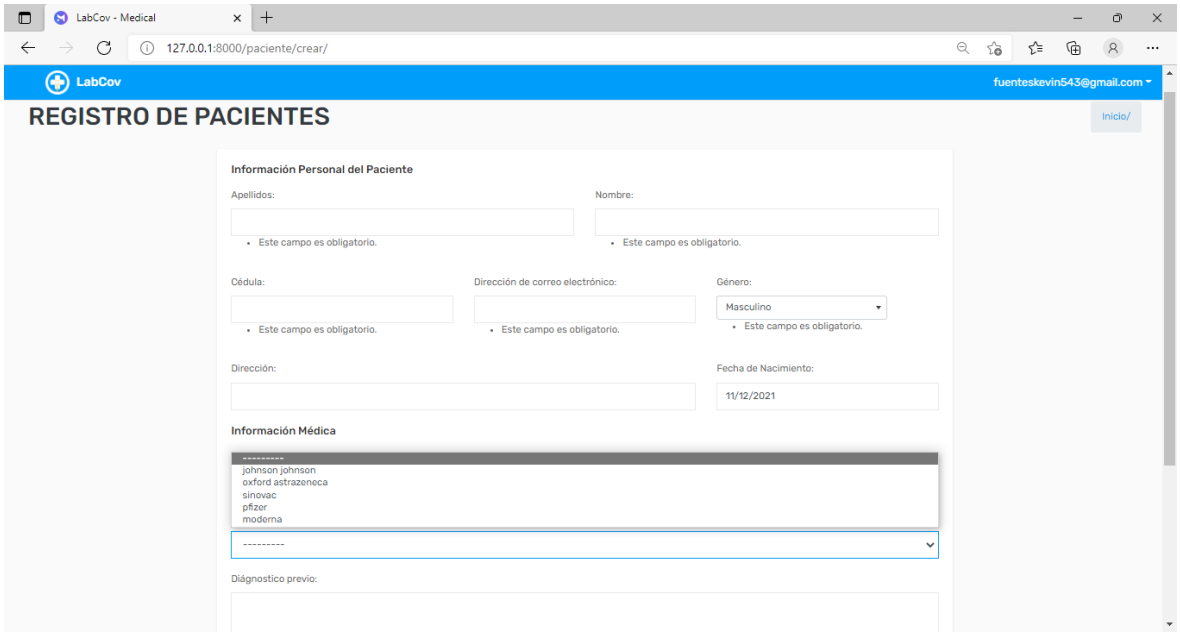
Nota: En esta figura se muestra el módulo de especialidades de los médicos. Fuente y elaboración propia.

Figura 34 Módulo vacunas.



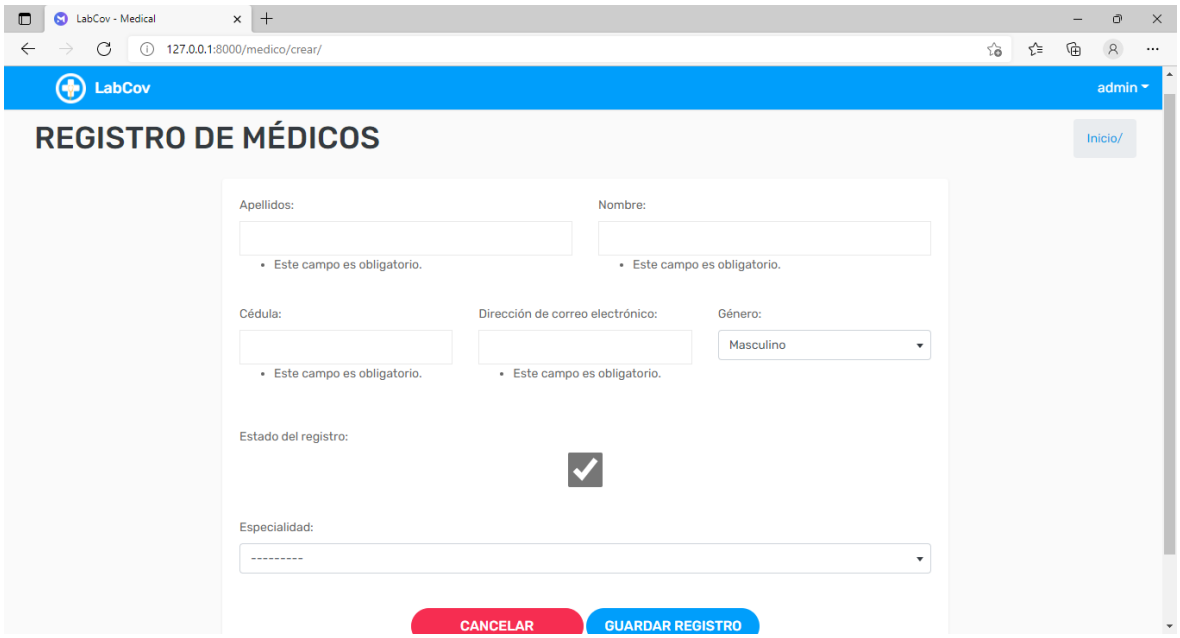
Nota: En esta figura se muestra el módulo de vacunas. Fuente y elaboración propia.

Figura 35 Formulario de registro de pacientes.



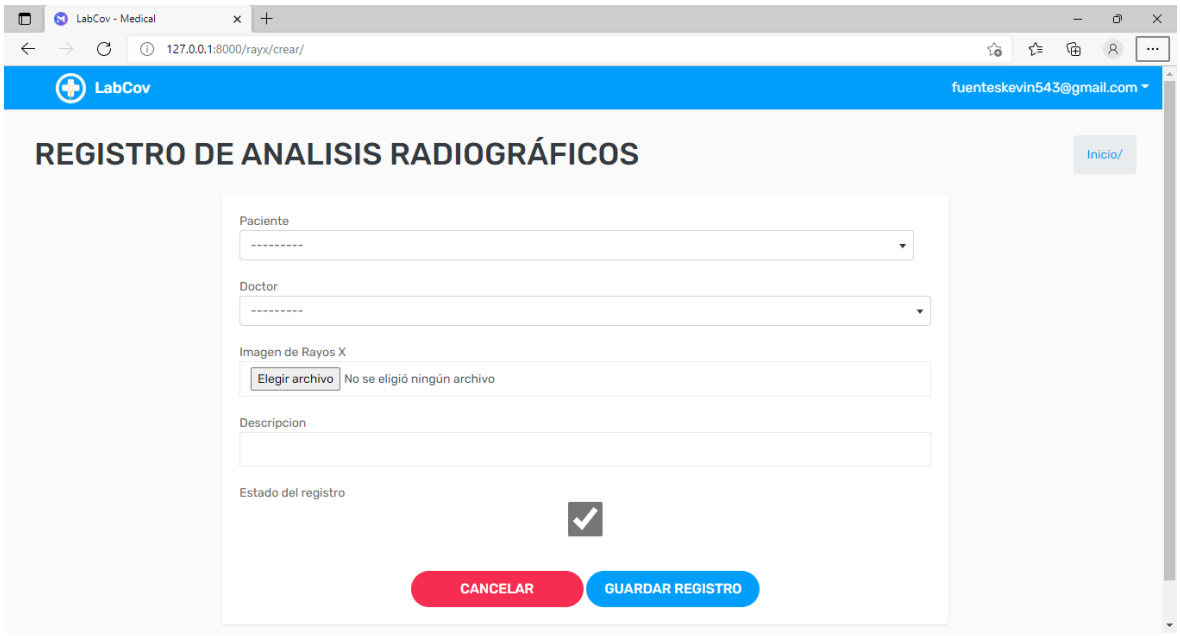
Nota: En esta figura se muestra el formulario de registro de pacientes de la aplicación web. Fuente y elaboración propia.

Figura 36 Formulario de registro de médicos.



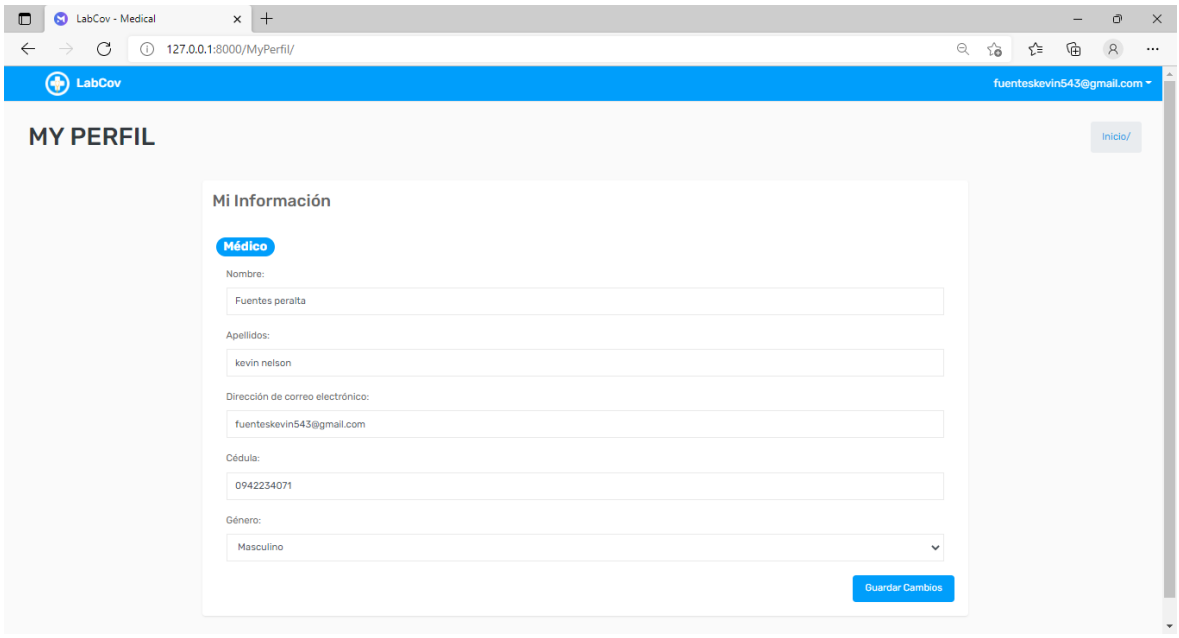
Nota: En esta figura se muestra el formulario de registro de médicos de la aplicación. Fuente y elaboración propia.

Figura 37 Formulario de registro de Análisis radiográficos.



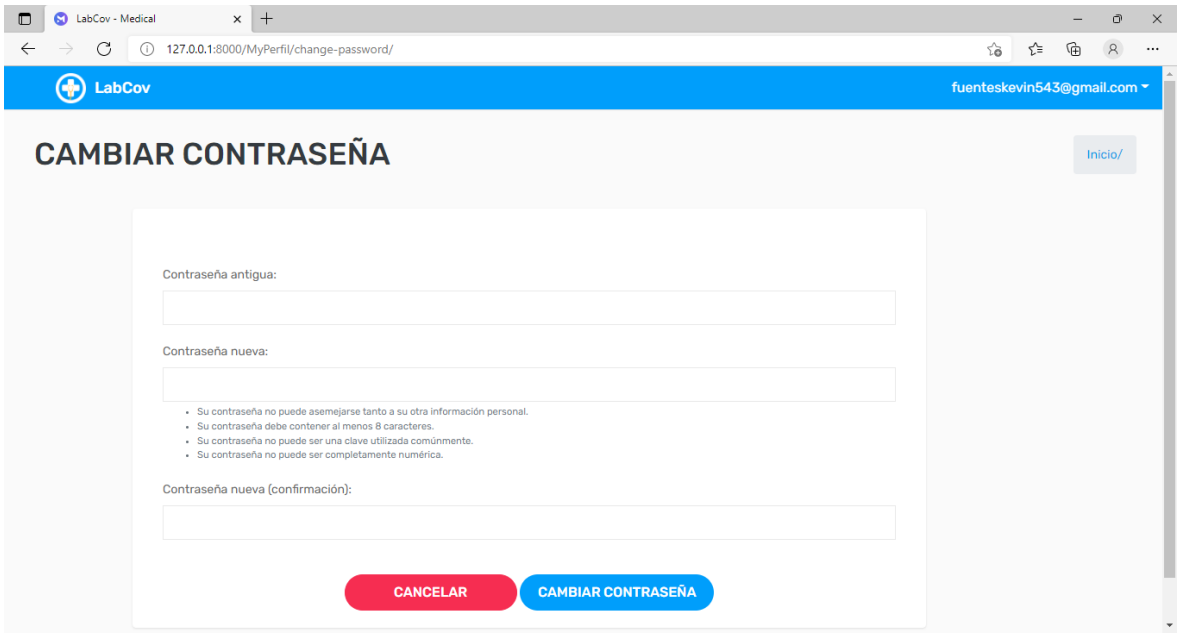
Nota: En esta figura se muestra el formulario de registro de Análisis radiográfico de la aplicación, aquí se sube la imagen de rayos x del paciente y se realiza la predicción para saber si tiene covid o no. Fuente y elaboración propia.

Figura 38 Módulo mi perfil.



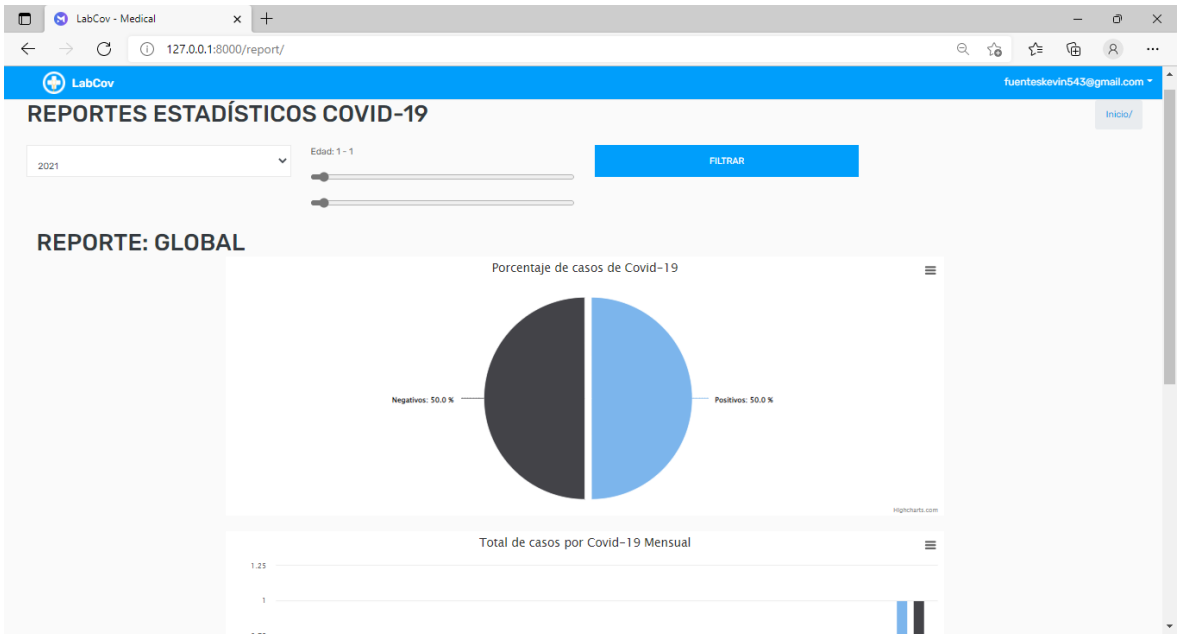
Nota: En esta figura se muestra el módulo mi perfil, aquí se puede editar nuestro perfil. Fuente y elaboración propia.

Figura 39 Módulo cambiar contraseña.



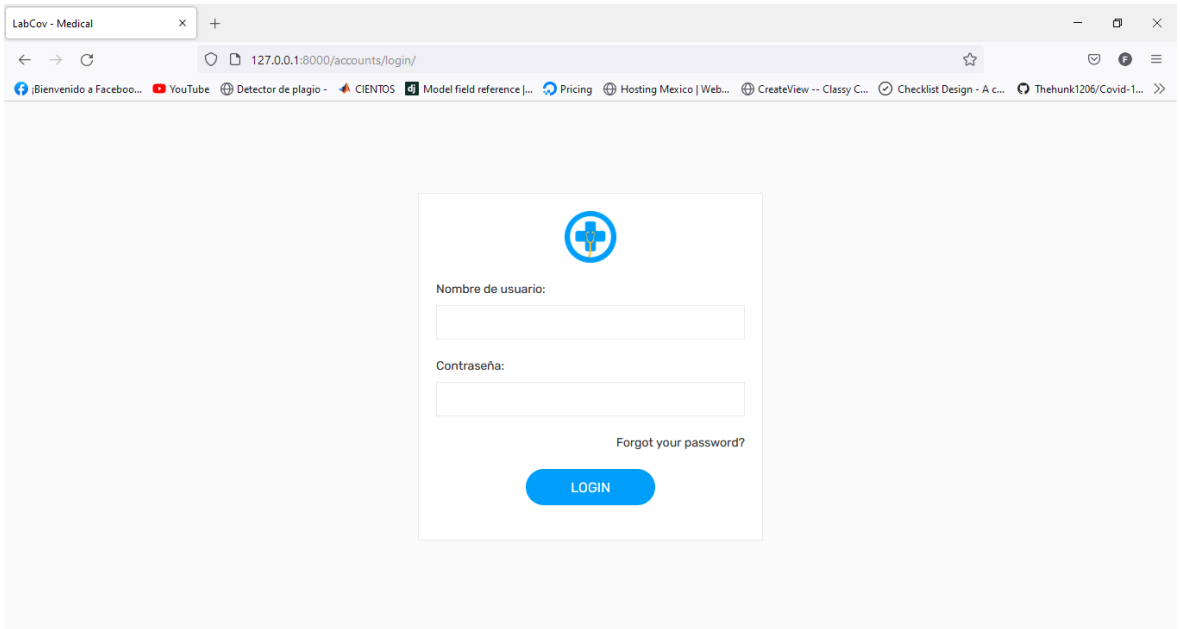
Nota: En esta figura se muestra el módulo cambiar contraseña. Fuente y elaboración propia.

Figura 40 Reportes estadísticos de la aplicación.



Nota: En esta figura se muestra el módulo de reportes estadísticos de la aplicación web, aquí se puede observar el porcentaje de casos con covid y sin covid global, por año o por mes, además de reportes por edad y género. Fuente y elaboración propia.

Figura 41 Login de la aplicación.



Nota: En esta figura se muestra el login de la aplicación. Fuente y elaboración propia.

3.3. Validación de la propuesta

3.3.1. Informe de prueba

Tabla 36 Informe de pruebas de la aplicación web.

Criterio	Cumple
El tiempo de respuesta de la aplicación web es rápido.	Si
La aplicación web es adaptable a todos los dispositivos.	Si
La aplicación web muestra alertas cuando se realiza una reacción de guardado, eliminado y modificado.	Si
La aplicación web es fácil de usar.	Si
Se cumple con los requerimientos establecidos.	Si
Los mantenimientos y módulos de la aplicación funcionan correctamente	Si

Nota: En esta tabla se puede observar el informe de pruebas de la aplicación web. Fuente y elaboración propia.

3.3.2. Informe final

Tabla 37 Informe final de la aplicación web.

Criterio	Cumple
Rendimiento	Si
Funcionalidad	Si
Usabilidad	Si

Nota: En esta tabla se puede observar el informe final de la aplicación web. Fuente y elaboración propia.

CONCLUSIONES

La arquitectura VGG19 resultó ser muy robusta y generó buenos resultados al realizar el entrenamiento con el set de datos de imágenes de rayos x recolectado. Al ser un modelo pre-entrenado, no es necesario entrenar todas las capas sino solo las capas agregadas para aprovechar los pesos de las capas inferiores que ya fueron pre-entrenadas. A esta técnica se la denomina transfer learning.

Una vez elegida la arquitectura de la red neuronal convolucional el desempeño que se obtiene con los parámetros iniciales fue prometedor, el punto clave para que el modelo de red neuronal convolucional realice mejor las predicciones, son la cantidad de imágenes que se utiliza para realizar el entrenamiento y la validación de los datos, por lo general se debe trabajar con un gran número de imágenes, para que la red reconozca la mayor cantidad de características de las imágenes y así pueda identificar mejor a qué clase pertenece, “covid” o ”normal”.

La aplicación web en donde se implementó la CNN creada cuenta con distintos módulos que permiten optimizar los procesos que se pueden llevar a cabo dentro de un centro de salud, como es el análisis de radiografías para detectar covid.

La utilización de arquitecturas de redes neuronales convolucionales, permite optimizar el tiempo de entrenamiento de un modelo convolucional con diferentes conjuntos de datos ya que permite reutilizar los pesos al ser un modelo pre-entrenado.

RECOMENDACIONES

Se recomienda para próximos trabajos relacionados con la creación y entrenamiento de redes neuronales convolucionales se realice el entrenamiento con miles de imágenes ya que para lograr una predicción más exacta se necesita que la red se entrene con muchas imágenes. Cuando se entrena una red neuronal convolucional por lo general los datos deben estar debidamente balanceados, es decir, la cantidad de imágenes entre las distintas clases deben ser iguales o semejantes, ya que se puede producir un sobreajuste si se tiene más imágenes de una clase que de la otra, y así mismo, cuando se tienen pocas imágenes puede ocurrir un desajuste, debido a que no se tiene las suficientes imágenes para el entrenamiento.

Por otro lado, también se recomienda que no solo se trabaje con dos clases de imágenes de rayos x, “covid” y “normal”, ya que existen otras enfermedades que afectan a los pulmones como son la neumonía, opacidad pulmonar y neumonía viral, permitiendo tener un diagnóstico más preciso del estado del paciente.

Se recomienda utilizar Google Colab sino se cuentan con un equipo de hardware con especificaciones técnicas altas, para realizar proyectos de inteligencia artificial ya que nos ofrecen un alto rendimiento en procesamiento de datos, gracias a que nos proporcionan recurso gratis como acceso a la GPU, TPU en la nube, realizando los procesos mucho más rápido y sin realizar instalaciones de librerías o configuraciones previa ya que Google Colab nos ofrece todas las herramientas ya integradas en su servidor , además, si la versión gratuita no es suficiente se puede compra alguno de los planes que tiene disponible.

Por último, se recomienda implementar mejoras a la aplicación web, ya que su objetivo es gestionar los procesos de un centro de salud, por lo tanto, sería factible la implementación de un módulo que permita gestionar el historial clínico de los pacientes.

BIBLIOGRAFÍA

Artola Moreno, Á. (2019). Clasificación de imágenes usando redes neuronales convolucionales en Python.

Boden, M. (2016). *Inteligencia Artificial.* Madrid: AI. Its Nature and future.

Álvarez, V. M., Quirós, M. L., & Cortés, B. M. (2020). Inteligencia artificial y aprendizaje automático en medicina. *Revista Médica Sinergia*.

Avila, T. J., Mayer, P. M., & Quesada, V. V. (2020). La inteligencia artificial y sus aplicaciones en medicinaI: introducción antecedentes a la IA y robótica. *Elseiver*.

Díaz, C., & Toro, M. A. (2020). SARCoV-2/COVID-19: The virus, the disease and the pandemic. *Medicina y Laboratorio,* 183-205.

Domínguez Pavón, S. (2019). Identificación del modelo de cámara mediante Redes Neuronales Convolucionales.

Durán Suárez, J. (2017). Redes neuronales convolucionales en R: Reconocimiento de caracteres escritos a mano. *Universidad de Sevilla.*

García, A. R. (2019). Sistema de percepción de elementos viarios usando técnicas de visión por computador para aplicación en conducción autónoma. *Escuela Técnica Superior de Ingeniería Industrial*.

Hao, K. (13 de 05 de 2020). *MIT Technology Review*. Obtenido de technologyreview.es: https://www.technologyreview.es/s/12171/la-covid-19-acelera-el-uso-de-la-ia-medica-con-riesgos-y-beneficios

HIBA. (s.f.). *Hospital Italiano de Buenos Aires.* Obtenido de hospitalitaliano.org.ar: https://www1.hospitalitaliano.org.ar/?utm_source=email_marketing&utm_admin=110784&utm_medium=email&utm_campaign=Departamento_de_Informtica_en_Salud_Webinar_IA#!/home/infomed/noticia/108402

ichi.pro. (07 de 12 de 2021). *Determinar el ajuste perfecto para su modelo ML. .* Obtenido de Ichi.pro: https://ichi.pro/es/determinar-el-ajuste-perfecto-para-su-modelo-ml-56712257009264

Lasse Petteri , R. (2018). *Inteligencia artificial 101 cosas que debes saber sobre nuestro futuro.* España: Editorial Planeta, S.A.

Lugo Reyes, S. O., Guadalupe, M. C., & Murata, C. (2014). Inteligencia artificial para asistir el diagnóstico clínico en medicina. *Revista Alergia México*.

Maguiña, Q. J., Murguía, L., Verona, M., Gomero, C. R., & Véliz, J. (2021). Factores asociados a hallazgos anormales en radiografías digitales de tórax en trabajadores asintomáticos en Lima, Perú. *acta médica peruana*.

Matich, D. J. (2001). Redes Neuronales: Conceptos Básicos y Aplicaciones. *Universidad Tecnológica Nacional, México.*

Melián, R. A., Calcumil, H. P., Boin, B. C., & Carrasco, S. ,. (2020). Detección de COVID -19 (SARS-CoV-2) mediante la saliva: Una alternativa diagnóstica poco invasiva. *International journal of odontostomatology*, 316-320.

Moore, K., Dailey, A., & Agur, A. (2013). *Anatomía con orientación clínica.* EE.UU.: Editorial médica panamericana.

Navarro Moreno, A. (2020). Introducción a las Redes Neuronales aplicadas al Aprendizaje Supervisado. *Journal contribution*.

Roy, A. (23 de Julio de 2020). *Detección de Covid-19 a partir de rayos X mediante aprendizaje profundo*. Obtenido de Medium: https://medium.com/swlh/covid-19-detection-from-x-ray-using-deep-learning-a8b8046fc077

Sánchez Agustino, F. (2016). *Diseño de un sistema de reconocimiento automático de matrículas de vehículos mediante una red neuronal convolucional.* Obtenido de http://hdl.handle.net/10609/52222

Sánchez Martínez, M. (2018). Detección de personas mediante técnicas de aprendizaje automático: SVM y CNN.

Takeyas, B. L. (2007). Introducción a la inteligencia artificial. *Instituto Tecnológico de Nuevo Laredo*.

TensorFlow. (03 de diciembre de 2021). *TensorFlow Core v2.7.0*. Obtenido de TensorFlow:

https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/image/ImageDataGenerator

Thirona. (2021). *Thirona*. Obtenido de thirona.eu: https://thirona.eu/cad4covid/

Torres, J. (2018). *Deep learning Introducción práctica con Keras.* Barcelona: Watch this space.

Wejéus, S. (2014). A Neural Network Approach to Arbitrary SymbolRecognition on Modern Smartphones. *kth computer science and communication.*

ANEXOS

Anexo 1 – Arquitectura de la red neuronal convolucional utilizada.

```
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 224, 224, 3)]     0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 224, 224, 64)      1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 224, 224, 64)      36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 112, 112, 64)      0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 112, 112, 128)     73856
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 112, 112, 128)     147584
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 56, 56, 128)       0
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 56, 56, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_conv4 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 28, 28, 256)       0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 28, 28, 512)       1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_conv4 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 14, 14, 512)       0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv4 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 7, 7, 512)         0
_________________________________________________________________
flatten (Flatten)            (None, 25088)             0
_________________________________________________________________
fc1 (Dense)                  (None, 4096)              102764544
_________________________________________________________________
predictions (Dense)          (None, 1000)              4097000
_________________________________________________________________
output (Dense)               (None, 2)                 2002
```

```
=================================================================
Total params: 126,887,930
Trainable params: 126,887,930
Non-trainable params: 0
_________________________________________________________________
```

Anexo 2 – Mejor entrenamiento de la red neuronal convolucional.

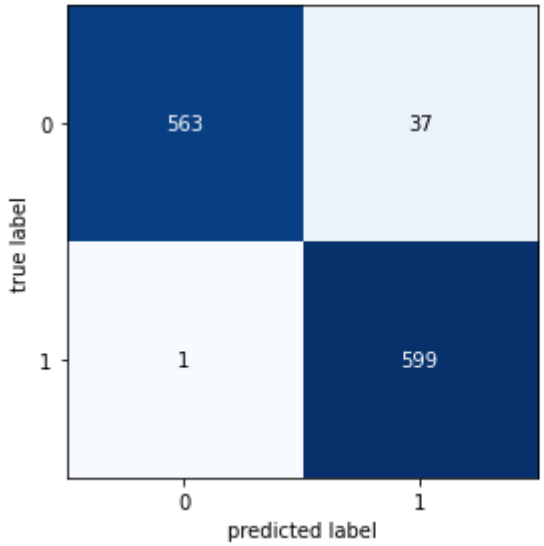
```
/usr/local/lib/python3.7/dist-
packages/tensorflow/python/keras/engine/training.py:1963: UserWarning:
`Model.fit_generator` is deprecated and will be removed in a future
version. Please use `Model.fit`, which supports generators.
  warnings.warn('`Model.fit_generator` is deprecated and '
Epoch 1/40
151/151 [==============================] - 5723s 38s/step - loss: 0.7150
- accuracy: 0.8306 - val_loss: 0.2473 - val_accuracy: 0.8988

Epoch 00001: val_loss improved from inf to 0.24730, saving model to
/content/drive/MyDrive/data_70_30/modelo/modeloVGG19_aumento_3_epoc.h5
Epoch 2/40
151/151 [==============================] - 233s 2s/step - loss: 0.2262 -
accuracy: 0.9063 - val_loss: 0.2034 - val_accuracy: 0.9178

Epoch 00002: val_loss improved from 0.24730 to 0.20341, saving model to
/content/drive/MyDrive/data_70_30/modelo/modeloVGG19_aumento_3_epoc.h5
Epoch 3/40
151/151 [==============================] - 235s 2s/step - loss: 0.1887 -
accuracy: 0.9245 - val_loss: 0.1670 - val_accuracy: 0.9356

Epoch 00003: val_loss improved from 0.20341 to 0.16697, saving model to
/content/drive/MyDrive/data_70_30/modelo/modeloVGG19_aumento_3_epoc.h5
Epoch 4/40
151/151 [==============================] - 231s 2s/step - loss: 0.1623 -
accuracy: 0.9343 - val_loss: 0.1148 - val_accuracy: 0.9567

Epoch 00004: val_loss improved from 0.16697 to 0.11484, saving model to
/content/drive/MyDrive/data_70_30/modelo/modeloVGG19_aumento_3_epoc.h5
Epoch 5/40
151/151 [==============================] - 231s 2s/step - loss: 0.1495 -
accuracy: 0.9404 - val_loss: 0.1437 - val_accuracy: 0.9478

Epoch 00005: val_loss did not improve from 0.11484
Epoch 6/40
151/151 [==============================] - 235s 2s/step - loss: 0.1319 -
accuracy: 0.9482 - val_loss: 0.0954 - val_accuracy: 0.9639

Epoch 00006: val_loss improved from 0.11484 to 0.09544, saving model to
/content/drive/MyDrive/data_70_30/modelo/modeloVGG19_aumento_3_epoc.h5
Epoch 7/40
151/151 [==============================] - 238s 2s/step - loss: 0.1232 -
accuracy: 0.9523 - val_loss: 0.1441 - val_accuracy: 0.9591

Epoch 00007: val_loss did not improve from 0.09544
Epoch 8/40
151/151 [==============================] - 235s 2s/step - loss: 0.1161 -
accuracy: 0.9557 - val_loss: 0.1173 - val_accuracy: 0.9510
```

```
Epoch 00008: ReduceLROnPlateau reducing learning rate to
0.0005000000237487257.

Epoch 00008: val_loss did not improve from 0.09544
Epoch 9/40
151/151 [==============================] - 233s 2s/step - loss: 0.0852 -
accuracy: 0.9662 - val_loss: 0.1230 - val_accuracy: 0.9570

Epoch 00009: val_loss did not improve from 0.09544
Epoch 10/40
151/151 [==============================] - 233s 2s/step - loss: 0.0817 -
accuracy: 0.9696 - val_loss: 0.0839 - val_accuracy: 0.9716

Epoch 00010: val_loss improved from 0.09544 to 0.08390, saving model to
/content/drive/MyDrive/data_70_30/modelo/modeloVGG19_aumento_3_epoc.h5
Epoch 11/40
151/151 [==============================] - 233s 2s/step - loss: 0.0783 -
accuracy: 0.9704 - val_loss: 0.0926 - val_accuracy: 0.9690

Epoch 00011: val_loss did not improve from 0.08390
Epoch 12/40
151/151 [==============================] - 230s 2s/step - loss: 0.0713 -
accuracy: 0.9720 - val_loss: 0.0924 - val_accuracy: 0.9680

Epoch 00012: ReduceLROnPlateau reducing learning rate to
0.0002500000118743628.

Epoch 00012: val_loss did not improve from 0.08390
Epoch 13/40
151/151 [==============================] - 230s 2s/step - loss: 0.0559 -
accuracy: 0.9781 - val_loss: 0.1028 - val_accuracy: 0.9678

Epoch 00013: val_loss did not improve from 0.08390
Epoch 14/40
151/151 [==============================] - 227s 2s/step - loss: 0.0577 -
accuracy: 0.9793 - val_loss: 0.0841 - val_accuracy: 0.9731

Epoch 00014: ReduceLROnPlateau reducing learning rate to
0.0001250000059371814.

Epoch 00014: val_loss did not improve from 0.08390
Epoch 15/40
151/151 [==============================] - 224s 1s/step - loss: 0.0486 -
accuracy: 0.9819 - val_loss: 0.0794 - val_accuracy: 0.9743

Epoch 00015: val_loss improved from 0.08390 to 0.07940, saving model to
/content/drive/MyDrive/data_70_30/modelo/modeloVGG19_aumento_3_epoc.h5
Epoch 16/40
151/151 [==============================] - 227s 2s/step - loss: 0.0457 -
accuracy: 0.9828 - val_loss: 0.0840 - val_accuracy: 0.9731

Epoch 00016: val_loss did not improve from 0.07940
Epoch 17/40
151/151 [==============================] - 227s 2s/step - loss: 0.0439 -
accuracy: 0.9833 - val_loss: 0.0803 - val_accuracy: 0.9733

Epoch 00017: ReduceLROnPlateau reducing learning rate to
6.25000029685907e-05.

Epoch 00017: val_loss did not improve from 0.07940
```

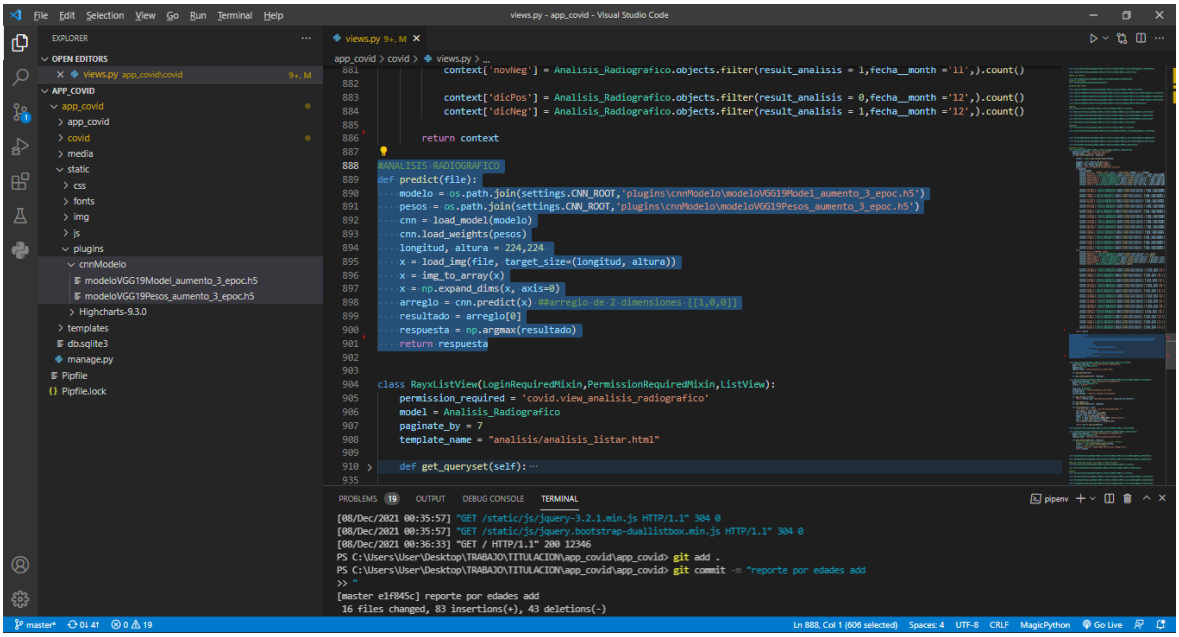
```
Epoch 18/40
151/151 [==============================] - 228s 2s/step - loss: 0.0415 -
accuracy: 0.9841 - val_loss: 0.0820 - val_accuracy: 0.9748

Epoch 00018: val_loss did not improve from 0.07940
Epoch 19/40
151/151 [==============================] - 225s 1s/step - loss: 0.0363 -
accuracy: 0.9866 - val_loss: 0.0869 - val_accuracy: 0.9719

Epoch 00019: ReduceLROnPlateau reducing learning rate to
3.125000148429535e-05.

Epoch 00019: val_loss did not improve from 0.07940
Epoch 20/40
151/151 [==============================] - 225s 1s/step - loss: 0.0356 -
accuracy: 0.9871 - val_loss: 0.0856 - val_accuracy: 0.9733

Epoch 00020: val_loss did not improve from 0.07940
Epoch 21/40
151/151 [==============================] - 225s 1s/step - loss: 0.0399 -
accuracy: 0.9853 - val_loss: 0.0893 - val_accuracy: 0.9712

Epoch 00021: ReduceLROnPlateau reducing learning rate to
1.5625000742147677e-05.

Epoch 00021: val_loss did not improve from 0.07940
Epoch 22/40
151/151 [==============================] - 226s 1s/step - loss: 0.0342 -
accuracy: 0.9871 - val_loss: 0.0834 - val_accuracy: 0.9755

Epoch 00022: val_loss did not improve from 0.07940
Epoch 23/40
151/151 [==============================] - 225s 1s/step - loss: 0.0338 -
accuracy: 0.9879 - val_loss: 0.0853 - val_accuracy: 0.9748

Epoch 00023: ReduceLROnPlateau reducing learning rate to
7.812500371073838e-06.

Epoch 00023: val_loss did not improve from 0.07940
Epoch 24/40
151/151 [==============================] - 226s 1s/step - loss: 0.0372 -
accuracy: 0.9862 - val_loss: 0.0845 - val_accuracy: 0.9752

Epoch 00024: val_loss did not improve from 0.07940
Epoch 25/40
151/151 [==============================] - 222s 1s/step - loss: 0.0343 -
accuracy: 0.9888 - val_loss: 0.0833 - val_accuracy: 0.9760

Epoch 00025: ReduceLROnPlateau reducing learning rate to
3.906250185536919e-06.

Epoch 00025: val_loss did not improve from 0.07940
Epoch 00025: early stopping
```

Anexo 3 – Mejor resultado de las pruebas de la red neuronal convolucional.

	precision	recall	f1-score	support
0	0.9982	0.9383	0.9674	600
1	0.9418	0.9983	0.9693	600
accuracy			0.9683	1200
macro avg	0.9700	0.9683	0.9683	1200
weighted avg	0.9700	0.9683	0.9683	1200



Anexo 4 – Implementación de la red neuronal creada en la aplicación web.



Anexo 5 – Análisis radiográfico con imágenes de rayos x.