

UNIVERSIDAD AUTÓNOMA DE CIUDAD JUÁREZ

División Multidisciplinaria en Ciudad Universitaria

Instituto de Ingeniería y Tecnología

Departamento de Ingeniería Eléctrica y Computación



ALGORITMO DE CLASIFICACION MULTI-ETIQUETA PARA LA
IDENTIFICACIÓN DE RETINOPATIAS EN IMAGENES DE LA
RETINA

Reporte Técnico de Investigación presentado por:

Héctor Gerardo Sánchez Quiroga

169840

Requisito para la obtención del título de

INGENIERO DE SOFTWARE

Asesor:

Dr. Rogelio Florencia Juárez

Dra. Julia Patricia Sánchez Solís

Ciudad Juárez, Chihuahua a 22 de noviembre de 2022

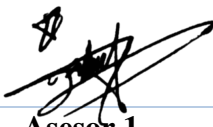
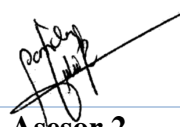
Asunto: Liberación de Asesoría

Mtro. Ismael Canales Valdiviezo
Jefe del Departamento de Ingeniería
Eléctrica y Computación
Presente. -

Por medio de la presente me permito comunicarle que después de haber realizado las asesorías correspondientes al reporte técnico **Algoritmo de Clasificación Multi-Etiqueta para la Identificación de Retinopatías en Imágenes de la Retina**, del alumno Héctor Gerardo Sánchez Quiroga de la Licenciatura en Ingeniería de Software, consideramos que lo ha concluido satisfactoriamente, por lo que puede continuar con los trámites de titulación intracurricular.

Sin más por el momento, reciba un cordial saludo.

Atentamente

 Asesor 1	 Asesor 2
Dr. Rogelio Florencia Juárez	Dra. Julia Patricia Sánchez Solís

Ccp. Mtro. David Absalón Uruchurtu Moreno
Héctor Gerardo Sánchez Quiroga
Archivo

Ciudad Juárez, Chihuahua a 22 de noviembre de 2022

Asunto: Autorización de impresión

C. Héctor Gerardo Sánchez Quiroga

Presente. -

En virtud de que cumple satisfactoriamente los requisitos solicitados, informo a usted que se autoriza la impresión del proyecto de **Algoritmo de Clasificación Multi-Etiqueta para la Identificación de Retinopatías en Imágenes de la Retina**, para presentar los resultados del proyecto de titulación con el propósito de obtener el título de Licenciado en Ingeniería de Software.

Sin otro particular, reciba un cordial saludo.

Dra. Julia Patricia Sánchez Solís
Profesora de Seminario de Titulación II

Declaración de Originalidad

Yo, Héctor Gerardo Sánchez Quiroga declaro que el material contenido en esta publicación fue generado con la revisión de los documentos que se mencionan en la sección de referencias y que la solución desarrollada es original y no ha sido copiada de ninguna otra fuente, ni ha sido usada para obtener otro título o reconocimiento en otra Institución de Educación Superior.

Sánchez Héctor

Héctor Gerardo Sánchez Quiroga

Agradecimientos

El proceso para la realización de este proyecto fue muy arduo y largo, y al momento de terminarlo solo puedo estar agradecido por todo el apoyo que mis padres me han otorgado en los momentos más difíciles y que siempre estuvieron a mi lado.

También agradeciendo a mis compañeros de la carrera que entre todos nos esforzamos para completar cada uno de nuestros sueños.

Por último, agradeciendo a cualquier persona que en un futuro pueda utilizar este trabajo de titulación para apoyo en cualquier forma académicamente.

Gracias

Dedicatoria

Todo el esfuerzo que se realizó en este proyecto está dedicado a mi madre porque fue la persona que siempre me apoyo en todo el transcurso de mi educación. Además de mi padre que siempre estuvo a su lado dando apoyo cuando era importante y de enseñarme como es que uno no tiene que rendirse y siempre seguir adelante, aunque existan diversas adversidades.

Gracias por convertirme en la persona que soy el día de hoy, con todos estos diferentes conocimientos y habilidades son lo que me vuelven a mí.

Índice de contenido

Declaración de Originalidad	iv
Agradecimientos	v
Dedicatoria.....	vi
Índice de contenido	vii
Índice de figuras.....	ix
Índice de tablas	x
Resumen.....	xi
Introducción	1
I. Planteamiento del problema	2
1.1 Antecedentes	2
1.2 Definición del problema	3
1.3 Objetivos	4
1.4 Pregunta (s) de Investigación y/o Hipótesis.....	4
1.5 Justificación	4
II. Marco Referencial	6
2.1 Marco teórico	6
2.1.1 Retinopatías.....	6
2.1.2 Aprendizaje Automático	7
2.1.3 Algoritmos de Clasificación	9
2.1.4 Aprendizaje Profundo	10
2.1.5 Redes Neuronales Artificiales.....	11
2.1.6 Redes Neuronales Convolucionales (CNN).....	12
2.1.7 Técnicas y Métricas de Evaluación.....	14
2.2 Marco tecnológico	16
2.2.1 Python	16
2.2.2 Anaconda	16
2.2.3 Jupyter Notebook	17
2.2.4 Tensorflow	17
	vii

2.2.5 Keras	17
III. Desarrollo del proyecto	18
3.1 Producto Esperado	18
3.2 Entendimiento de la problemática	21
3.2.1 Determinar objetivos del proyecto de Investigación	22
3.2.2 Determinar requerimientos del proyecto de investigación	22
3.3 Entendimiento de los datos	22
3.3.1 Obtención de los datos a utilizar en el proyecto	22
3.3.2 Exploración de los datos obtenidos.....	22
3.3.3 Verificar la calidad de los datos.....	24
3.4 Preparación de los datos.....	27
3.4.1 Instalación de herramientas.....	27
3.5 Modelado	28
3.5.1 Configuración de la red convolucional	28
3.5.2 Entrenamiento del modelo	28
IV. Resultados y Discusiones	32
4.1 Formas de validación y Obtención de Resultados	32
4.2 Resultados	32
4.1.2 Evaluación y Graficación de los resultados	34
4.1.3 Evaluación de los resultados.....	38
4.2 Discusiones	39
V. Conclusiones	41
Referencias.....	43
Apéndices.....	47

Índice de figuras

Figura 1: Tipos de Aprendizajes	7
Figura 2: Arquitectura de Red Neuronal Convolucional	13
Figura 3: Operación en una capa convolucional	13
Figura 4: Operación de capa Max Pooling	14
Figura 5: Diagrama de la arquitectura	18
Figura 6: Fases de la metodología Crisp-DM	20
Figura 7: Gráfica de Cantidad de Enfermedades	23
Figura 8: Gráfica de las dimensiones de las imágenes del dataset Training	25
Figura 9: Gráfica de las dimensiones de las imágenes del dataset Test	26
Figura 10: Gráfica de las dimensiones de las imágenes del dataset Validation	27
Figura 11: Librerías VGG16	29
Figura 12: ImageDataGenerator	29
Figura 13: Transfer Learning	29
Ilustración 14: Resumen de modelo VGG16	30
Figura 15: Código de Ajuste y Compilación	30
Figura 16: Carga de ResNet50	31
Figura 17: Matriz de Confusión VGG16 DR	35
Figura 18: Matriz de Confusión VGG16 MH	35
Figura 19: Matriz de Confusión VGG16 ODC	36
Figura 20: Matriz de Confusión VGG16 TSLN	36
Figura 21: Matriz de Confusión ResNet50 DR	37
Figura 22: Matriz de Confusión ResNet50 MH	37
Figura 23: Matriz de Confusión ResNet50 ODC	38
Figura 24: Matriz de Confusión ResNet50 TSLN	38

Índice de tablas

Tabla 1: Matriz de Confusión	15
Tabla 2: Cantidad de imágenes por retinopatía.....	23
Tabla 3: Retinopatías más comunes en el dataset	24
Tabla 4: Cantidad de dimensiones en Dataset Training	24
Tabla 5: Cantidad de dimensiones en Dataset Test	25
Tabla 6: Cantidad de dimensiones en Dataset Test	26
Tabla 7: Métricas de matrices de confusión	39

Resumen

Las redes neuronales convolucionales (CNN) han exhibido beneficios en las áreas de clasificación de imágenes, generalmente los problemas de investigación incursionan en la clasificación binaria, pero en el mundo real los problemas generalmente aparecen en un enfoque de múltiples etiquetas. El uso de CNN dentro de un diagnóstico médico puede ayudar a los médicos a hacer un diagnóstico más rápido y ayudar a los estudiantes a tener una referencia en el momento en que hacen un diagnóstico. La solución propuesta es una clasificación de múltiples etiquetas entrenada y evaluada en un conjunto de datos proporcionado por la Organización Mundial de la Salud. Las diferentes configuraciones muestran que el modelo se puede entrenar y obtener un mejor resultado si se entrena con más información de la proporcionada.

Palabras clave: Deep Learning, Redes Convolucionales, Transfer Learning, Retinopatía, Clasificación Multi-etiqueta.

Convolutional Neural Networks (CNN) have exhibit benefits in the areas of image classification, usually research problems dabble in binary classification but in the real world the problems usually appear in a multilabel approach. Using CNN for a medical diagnosis can help doctors to made faster diagnosis and help students have a reference the moment they make a diagnosis. The proposed solution is a multi-label classification trained and evaluated on a dataset provided by the World Health Organization. The different configurations show the model can be trained and obtain a better result if trained with more information than the one provided.

Important Terms: Deep Learning, Convolutional Neural Network, Transfer Learning, Retinopathy, Multi-label Classification

Introducción

La vista es el principal sentido que los humanos utilizan para obtener información del medio natural. La ceguera es la etapa final en la mayoría de las enfermedades de la vista, entonces la rápida detección y prevención de enfermedades que afecten la visión pueden cambiar la vida completa a una persona. La retina es un área de gran importancia dentro del ojo humano para realizar el proceso de la visión, entonces cualquier enfermedad que ocurra en esta área puede afectar a todo el ojo. Las consultas médicas pueden ser realizadas por médicos que tengan experiencia en el área de la enfermedad o es posible utilizar un sistema que pueda apoyar al momento de realizar una decisión. En el área de las retinopatías es de suma importancia detectarla a tiempo para poder realizar un diagnóstico y tratamiento porque existen una gran cantidad de personas que pierden la vista por culpa de estas enfermedades. Se desarrollo un clasificador multi-etiqueta de retinopatías, utilizando un dataset ofrecido por la Organización Mundial de la Salud (OMS), además de que se realizaron diversas pruebas para conocer las configuraciones que tuvieran mejor resultado para clasificar. La forma en la que se desarrolla el clasificador es por medio de redes neuronales convolucionales que usara solamente el dataset ofrecido por la OMS.

I. Planteamiento del problema

1.1 Antecedentes

La retina es el área donde comienza la visión, dentro de la retina se encuentran dos tipos de capas: *epitelio pigmentario* y el *neuroepitelio*. El proceso de la visión es el siguiente: primero, la luz entra en el ojo, después atraviesa cada una de las capas que se encuentran dentro de la retina hasta llegar al epitelio pigmentario, donde la luz es reflejada y luego es percibida por los fotorreceptores y transmite la señal de los fotorreceptores por medio de las células bipolares y ganglionares hacia el sistema nervioso central [1].

El principal receptor de los estímulos visuales es la retina, por lo tanto, las complicaciones que puedan afectar a este tejido pueden producir ceguera en el peor de los casos. En [2] se comentará acerca de las retinopatías más comunes en el mundo. En el primer lugar se encuentra la retinopatía diabética la cual es la principal causa de ceguera en personas de 25 a 74 años. Otra retinopatía es el desprendimiento de la retina, que como su nombre menciona, se define como la separación en la retina de los fotorreceptores y el pigmento retinal epitelio. Existe también la degeneración macular por la edad, donde a las personas de edad avanzada empiezan a manifestar ciertas lesiones maculares que podrían ser visibles por manchas amarillas. Existen más enfermedades como la manifestación retinal del Sida o la retinopatía neumática.

La retina igual que otros tejidos u órganos de nuestro cuerpo puede tener ciertas complicaciones que la afecten. En [3] se habló acerca de que veinte años después de que un paciente se le detectará diabetes, la mayoría de los pacientes con diabetes tipo 1 y el 60% de los pacientes con diabetes tipo 2 tendrá un cierto grado de retinopatía. El tratamiento puede prevenir la ceguera en la mayoría de los casos así que es esencial identificar a los pacientes que pudieran tener retinopatía antes de desarrollar algún tipo de complicaciones en la visión.

Trabajos Preliminares:

En el siguiente estudio [4] se tuvo a la retinopatía diabética como una de las principales retinopatías de estudio. La retinopatía diabética (RD) es una de las causas de pérdidas de la visión más altas del mundo. Alrededor de un décimo de la población que vive con diabetes presenta un grado de posibilidad de desarrollar esta enfermedad, entonces es

importante detectar lo más rápido posible e intervenir. Una de las técnicas más comunes para detectar RD es la fotografía del fondo de ojo de la retina, entonces el proceso para detectar manualmente esta problemática es muy extenso, entonces se optó por utilizar una serie de algoritmos de diagnóstico automáticos de medicina para detectar; que serían tres tipos de redes neuronales convolucionales DenseNet, ResNet y VGG16; y clasificar las fotografías de la retina en un enfoque de multi-etiquetas.

Una de las partes más importantes y esenciales dentro del área clínica de la oftalmología es realizar diagnósticos utilizando imágenes de Topografía de Coherencia Optica (OCT). Por tal motivo, en [5] se empezó a utilizar un sistema totalmente automático de diagnóstico que estuviera basado en aprendizaje profundo para detectar trastornos de la retina, en especial la degeneración macular de Drusen, y el edema macular diabético. Cabe mencionar que si estos trastornos no son diagnosticados y tratados a tiempo pueden causar una pérdida de la visión.

En [6] se utilizó una red neuronal convolucional de aprendizaje profundo para poder expandir un *dataset* acerca de imágenes de la retina. El *dataset* contenía fotografías en 10 categorías, una categoría conformada por imágenes de una retina normal y nueve categorías de retinopatías. Se empleó MatConvNet que es un *toolbox* de MATLAB para implementar redes neuronales convolucionales para una detección automatizada de las múltiples retinopatías además de que se utilizó la base de datos de *Structured Analysis of The Retina* (STARE) que es un proyecto donde se tienen 400 imágenes con diferentes retinopatías que ha sido creado para beneficiar a estudios clínicos y entrenamiento de internos médicos. Se utilizaron dos métodos de *Transfer Learning* (VGG19-TL-RF y VGG19-TL-SVM) que superaron a los modelos de aprendizaje profundo (VG19 y AlexNet).

1.2 Definición del problema

Las retinopatías son de las enfermedades más propensas a desarrollar ceguera en el mundo, principalmente la retinopatía diabética [3]. Esta enfermedad es solamente una de varias patologías que se pueden desarrollar en la retina, como también podría ser el desprendimiento de la retina o la degeneración macular de Drusen.

Al utilizar herramientas como las redes neuronales podría apoyar al médico y a los pacientes porque cuando se realiza la consulta oftalmológica y se observa la retina, lo más

común es que se pidan otros estudios o se mande el estudio como referencia hacia otro médico u hospital. Entonces el sistema propuesto disminuiría el tiempo en el que un médico tomará una decisión hacia cual es la mejor forma de enfrentar la retinopatía. Además de que serviría para apoyar a internos en su entrenamiento al momento de estar diagnosticando. Porque existe evidencia de que la detección temprana en casos de ciertas retinopatías es una de las piezas claves para prevenir la pérdida permanente de la agudeza visual [7].

1.3 Objetivos

Desarrollar un clasificador multi-etiqueta de retino patologías, utilizando un *dataset* ofrecido por la Organización Mundial de la Salud.

Objetivos Específicos:

- Analizar las imágenes del *dataset* de retinopatías
- Realizar un preprocesamiento de las imágenes del *dataset* de retinopatías
- Entrenar una red convolucional con el *dataset* de retinopatías
- Evaluar diferentes configuraciones de la red convolucional

1.4 Pregunta (s) de Investigación y/o Hipótesis

¿Cuál es la configuración que tuvo mayor nivel de Exactitud al momento de clasificar las retinopatías?

1.5 Justificación

El sistema propuesto apoyaría a una gran cantidad de persona, porque una de las retinopatías que se puede detectar es la retinopatía diabética que es la complicación por la diabetes más común. Además de que no solamente clasificaría acerca de esta retinopatía sino de otras 44 diferentes retinopatías no tan comunes pero que afectan a un grado igual o mayor que la retinopatía diabética [8].

La creación del sistema beneficiaría en el proceso de detección de retinopatías porque disminuiría el tiempo en el que un personal médico podría tomar una decisión hacia cual es la mejor forma de enfrentarse hacia una retinopatía, además de que podría advertir de posibles retinopatías no tan comunes.

II. Marco Referencial

2.1 Marco teórico

La siguiente sección se especificarán los conceptos básicos que serán utilizados en el desarrollo de este proyecto. Los conceptos que serán definidos serán: Retinopatías, Aprendizaje Automático, Algoritmos de Clasificación, Clasificación Multi-etiqueta, Aprendizaje Profundo, Redes Neuronales Artificiales, Redes Neuronales Convolucionales, y por ultimo las Técnicas y Métricas de Evaluación a usar en el proyecto.

2.1.1 Retinopatías

Una retinopatía es definida como una enfermedad de la retina según [9]. Con esta definición tan simple se pueden llevar a un tema más específico como podrían ser las diversas patologías que pueden afectar a la retina. Dentro de la base de datos RfMiD [10], que será la que se utilizará en este proyecto se encuentran diversas retinopatías, a continuación, se explicarán unas cuantas de ellas para dar a entender más a acerca de cuáles tipos de retinopatías serán abarcadas en este proyecto.

La retinopatía diabética es la causa más frecuente en casos más actuales de ceguera en la población desde los 20 años hasta los 74 años [11]. Para entender la retinopatía diabética se necesita comprender como se conforma la enfermedad, esta enfermedad es una complicación microvascular de la diabetes mellitus, que empieza desde anomalías leves hasta llegar a la retinopatía diabética proliferativa moderada y grave [11] [10].

Otra enfermedad que se encuentra en la base de datos sería el desprendimiento de retina, la enfermedad consiste en el desprendimiento de la retina neurosensorial del epitelio pigmentario de la retina subyacente. Lo que ocurre es que las fuerzas que unen a la retina neurosensorial y la retina subyacente se vuelven débiles, y produce un desprendimiento de retina. Existen 4 diversos tipos de desprendimiento de retina: Regmatógeno, Tracción, Seroso, y Combinado tracción-regmatógeno [10] [12].

Además de las enfermedades anteriores, otra enfermedad que se encuentra en la base de datos es la Degeneración Macular relacionada con la edad que es la causa más común de ceguera irreversible en personas mayores de 50 años en países desarrollados. Esta es una

enfermedad progresiva crónica de la retina central que usualmente se desarrolla por la edad avanzada de las personas [10] [13].

2.1.2 Aprendizaje Automático

Las retinopatías es una de las enfermedades que afectan a una gran área de la población mundial de diversos rangos de edad. Entonces se ha estado observando en los últimos años un gran incremento de diversas técnicas para prever que la enfermedad se vuelva más grave. Una de estas técnicas ha sido el aprendizaje automático y cómo se ha utilizado en diversas áreas.

El aprendizaje automático es una forma de estadística aplicada con mayor énfasis en el uso de las computadoras para realizar estimaciones de funciones que serían estadísticamente complicadas y con un menor énfasis. Un algoritmo de aprendizaje automático debe tener la habilidad de aprender sobre los mismos datos. Entonces el aprendizaje automático enfrenta problemas que serían muy difíciles de resolver utilizando los programas ya definidos por los seres humanos [14].

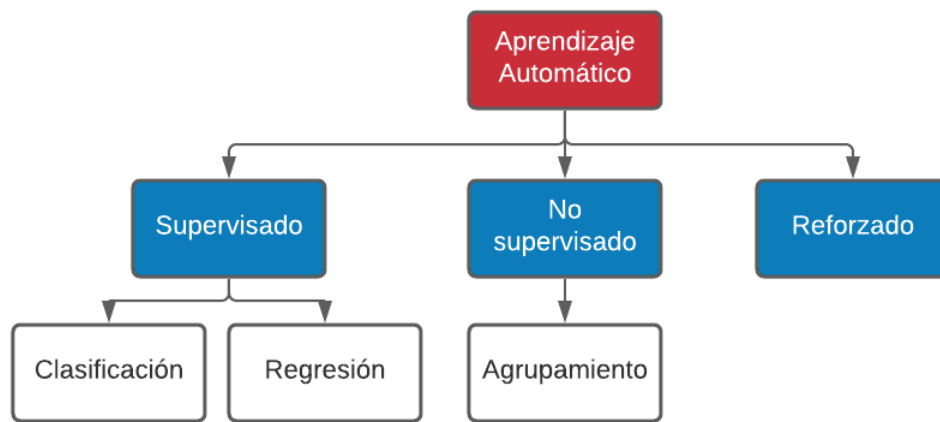


Figura 1: Tipos de Aprendizajes

Como se observa en la Figura 1, el aprendizaje automático puede tomar 3 grandes áreas de importancia que serían Supervisado, No Supervisado y Reforzado [15].

El aprendizaje supervisado es un modelo que al momento de que le hagan entradas no conocidas el modelo puede realizar predicciones acerca de estos valores. Para que el algoritmo aprenda debe tener valores de entrada y sus respuestas de salida para aprender

cómo es que el modelo estaría funcionando. Se recomienda utilizar este algoritmo cuando hay datos que están categorizados, etiquetados, o separados en grupos o clases diferentes [15]. La forma que esto ocurre es que los datos vienen en pares donde hay un objeto de entrada y otro objeto de salida deseado (llamado señal de supervisión). El algoritmo producirá una función de acuerdo con los datos que se le dieron, para que sea utilizado en nuevos valores [16].

El aprendizaje no supervisado tiene como objetivo que el modelo pueda encontrar los patrones para que las entradas lleguen hacia los valores de salida porque existen ciertos patrones que ocurren con mayor regularidad que otros, además de poder encontrar patrones que aún no han sido descubiertos y comprendidos. Entonces un supervisor estaría revisando cuáles son los valores de salida correctos. Usualmente es usado en reconocimiento de voz, o compresión de imágenes entre otros [15].

El aprendizaje reforzado es donde un agente autónomo elige decisiones óptimas para poder alcanzar la meta que se especificó. La forma en que ocurre esto, es porque se gratifica a las acciones de acuerdo con las consecuencias de sus acciones y con esto se deciden los mejores pasos a seguir [15]. Los agentes desean conseguir la mayor recompensa posible, entonces los agentes pueden elegir cualquier acción que sea óptima, aun tomando en cuenta las consecuencias a largo plazo de acuerdo con sus acciones actuales [16].

Como los tipos de problemas más comunes en los que se utiliza el aprendizaje automático serían [14]:

- Clasificación: Este tipo de problema se genera cuando es necesario que un programa especifique en cuáles de las k categorías puede pertenecer un valor.
- Regresión: Este tipo de problema se genera cuando se requiere que se prediga un valor numérico de acuerdo con otro valor de entrada que se le dio al sistema.
- Transcripción: El sistema de aprendizaje automático tiene que observar datos no estructurados y transcribir la información de estos datos en una forma textual discreta. Por ejemplo, utilizando un sistema de reconocimiento de caracteres que leyera fotografías para poder convertirlas en una secuencia de caracteres.
- Traducción máquina: Una de las problemáticas más comunes cuando se está globalizando el planeta es no comprender otros idiomas, así que una de las formas más comunes de utilizar los sistemas de aprendizaje automático sería la traducción

máquina. Los valores de entrada son secuencias de caracteres en un cierto idioma, que serán convertidos a otra secuencia de caracteres en el idioma deseado.

- Salidas estructuradas: Este tipo de problemas son cualquiera donde las salidas son vectores que tengan una unión fuerte entre diversos elementos. La transcripción y la traducción máquina son unas de los diferentes tipos que abarca esta problemática. Un ejemplo podría ser las leyendas de las imágenes donde es necesario que se entienda qué es la imagen para que se pueda dar una descripción breve sobre la imagen en un lenguaje natural como leyenda.
- Detección de anomalías: Una de las problemáticas más utilizadas por los bancos podría ser la detección de anomalías, en el área de la detección de fraudes de tarjetas de crédito o débito. Porque de acuerdo con un conjunto de eventos se puede señalar cuáles son los inusuales permitiendo que las compañías conozcan cuándo existe una compra anómala o algo extraño que podría afectar a sus usuarios.
- Denoising: Este tipo de problemática existe cuando se requiere que se limpien datos corruptos. Como entrada llegarían datos que están corruptos o incompletos, entonces el sistema de acuerdo con diversos procesos podría predecir cuáles serían los valores limpios originales.

2.1.3 Algoritmos de Clasificación

En la sección anterior se observó que una de las problemáticas donde se puede utilizar el aprendizaje automático sería la clasificación, entonces ahora se describirán los diversos tipos de algoritmos que pueden existir. Los algoritmos de clasificación se pueden dividir en dos tipos de clasificaciones que serían la clasificación de etiqueta única, y la clasificación multi-etiqueta.

En la clasificación de etiqueta única se ligan los valores de entrada con una cierta etiqueta de un conjunto de etiquetas q , además se puede dividir en [17]:

- Clasificación Binaria: Este tipo de clasificación es cuando los valores de entrada solo pueden pertenecer a dos tipos de categorías. La clasificación binaria es la más básica y es de donde proviene la idea más básica de qué es una clasificación.
- Clasificación Multiclase: Este tipo de clasificación es cuando los valores de entrada pueden pertenecer a un de más de dos categorías. Uno de los métodos de

clasificación multiclase es la clasificación jerárquica donde las clases tienen una forma de árbol, y cada nodo hoja es una clase, y las ramas son subconjuntos de otras clases.

La clasificación multi-etiqueta es un poco diferente de la clasificación de una sola etiqueta, un valor de entrada puede ser definido por diversas etiquetas y no este fijo a una sola etiqueta. El número de etiquetas a utilizar es dinámico y puede cambiar de acuerdo con la situación en la que se requiere utilizarlo. Existen tres tipos de técnicas para utilizar esta clasificación [17]:

- Métodos de adaptación de algoritmos: En este método el algoritmo tiene que adaptarse para la clasificación multi-etiqueta. Algunas variantes serían: Boosting, kNN, árboles de decisión, redes neuronales, y Máquinas de Soporte de Vectores.
- Métodos de transformación de problemas: El método hace que la clasificación multi-etiqueta sea convertida en varios problemas de clasificación binaria o multiclase.
- Métodos de ensamblaje: El método usa un conjunto de métodos de transformación y adaptación de algoritmos donde se combinan los resultados para realizar una clasificación multi-etiqueta.

2.1.4 Aprendizaje Profundo

El aprendizaje profundo es un subcampo del aprendizaje automático que acerca de cómo un sistema aprende de las abstracciones de los datos al utilizar diversas capas de procesamiento. Es una tecnología que en los últimos años ha estado creciendo en las áreas de procesamiento de lenguaje natural, visión computacional. Una de las razones principales por las que se utiliza en las tareas de visión computacional se debe a que puede extraer características mientras que hace la discriminación de las características [18] [19].

Los métodos de aprendizaje automático principalmente se dividen en 4 categorías con más uso actualmente [18]:

- Redes Neuronales Convolucionales: Este método es de los más usados actualmente, y es el más común de todos los métodos para cuando se quiera realizar problemas de visión computacional.

- Máquinas de Boltzmann Restringidas (RBM): Es una red neuronal estocástica. Es una variante de la máquina de Boltzmann con la diferencia que existe una restricción que vuelve a las unidades visibles y escondidas a formar una gráfica bipartita [18].
- Codificación automática: Es una clase de modelos que tienen como objetivo el de asignar las entradas hacia un espacio latente y mapearlo al espacio original. Las redes neuronales basadas en codificación automático consisten en dos partes [20]:
 - Codificador: Asigna las entradas a una capa escondida
 - Decodificador: Mapea hacia el espacio original

Al concatenar ambas partes es posible que se pueda utilizar como una forma de entrenar modelos.

- Codificación Escasa: Los vectores de entrada son reconstruidos utilizando una combinación lineal dispersa de vectores básicos. Se utiliza para extraer características de los datos. El proceso suele ser muy lento para aplicaciones como el reconocimiento de patrones en tiempo real [21].

2.1.5 Redes Neuronales Artificiales

Las redes neuronales artificiales son lo más cercano que los seres humanos han alcanzado en recrear una neurona real. Porque la neurona biológica contiene una cantidad muy grande de conexiones entre cada una, además de que las operaciones que realiza son totalmente asíncronas. Pero se logró capturar la esencia de que es una red neuronal utilizando unidades básicas computacionales [22].

Una red neuronal artificial es considerada como un modelo simplificado de la estructura de una red neuronal biológica. Mientras que una RNA son unidades de procesamiento interconectadas. Dentro de la red neuronal se pueden encontrar las siguientes áreas que conforman cada una de las estructuras de una red neuronal [22]:

- Unidad de procesamiento: Consiste en un área sumadora seguida de un área de salida. En el área sumadora se reciben N valores de entrada, se pondera cada valor, y después se calcula una suma ponderada denominada valor de activación. En el área de salidas se produce una señal del valor de activación. Con la señal del peso

para cada entrada se determina si las entradas son excitadoras (pesos positivos) o inhibitorias (pesos negativos).

- **Interconexiones:** Dentro de la unidad de procesamiento, las unidades están interconectadas. Entonces las entradas de la unidad de procesamiento pueden ser las salidas de otras unidades de procesamiento o de fuentes externas. Entonces es posible que las salidas de una unidad puedan ser entradas para otras unidades o para la misma unidad.
- **Operaciones:** Al calcular la suma ponderada de los pesos, la función de activación determina la salida actual de la salida de la función de unidad. Esto genera las dinámicas de activación que determinan los valores de activación de las demás unidades. Entonces, para una red los estados de activación, junto con sus unidades y las interconexiones serían la memoria de corto plazo de una neurona. Una vez que todo el proceso de aprendizaje está completado, el último set de pesos corresponderá a la memoria a largo plazo. El proceso de estar actualizando los pesos sería llamado un algoritmo de aprendizaje.

2.1.6 Redes Neuronales Convolucionales (CNN)

En la sección anterior se mostró uno de los tipos de redes neuronales que se pueden aplicar. Se escogió utilizar las redes neuronales convolucionales porque han dado resultados favorables en la visión computacional.

Las redes neuronales convolucionales funcionan de la siguiente manera, primero está la etapa de avance donde cada imagen debe ser representada con su peso y sesgos actuales en cada una de las capas. Después, la salida de la predicción se utiliza para calcular cuál fue el costo de la pérdida con las etiquetas de verdad. Luego, de acuerdo con estos valores de pérdida, se hacen actualizaciones a los parámetros de peso y sesgo para prepararse para el siguiente calculo. Esta etapa es iterativa hasta que se desee parar el aprendizaje de la red [18].

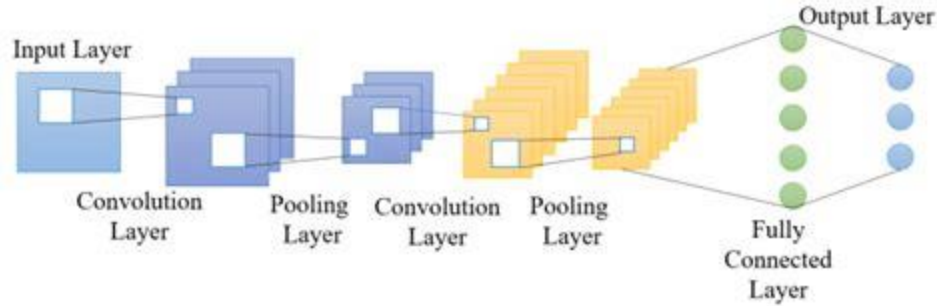


Figura 2: Arquitectura de Red Neuronal Convolutiva

Las redes neuronales convolucionales consisten en tres capas neuronales que son capas convolucionales, capas de agrupación, y capas completamente conectadas como se puede observar en la Figura 2 de manera gráfica. Cada una de estas capas tiene su propia acción, que se explicará a continuación [18]:

- Capas convolucionales: En la Figura 3, se observa cómo una imagen es recibida como valor de entrada para después realizar filtros convolucionales como sería utilizando varios *kernels* para generar mapas de características [19]. En la que cada mapa está conectado por un banco de filtros hacia las capas anteriores por medio de un conjunto de pesos.

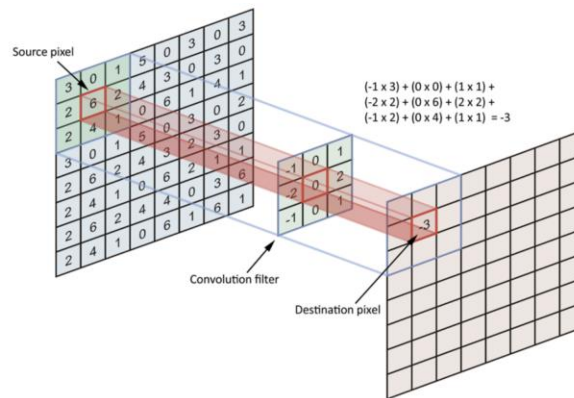


Figura 3: Operación en una capa convolutiva

- Capas de agrupación: Usualmente después de una capa convolutiva sigue una capa de agrupación. Esta capa es utilizada para reducir las dimensiones de los mapas de características y los parámetros de la red. Se observa en la Figura 4, un

ejemplo de cómo es el funcionamiento donde se tiene un mapa de características de 4×4 que después se reduce con ayuda del *max pooling* a una matriz de 2×2 .

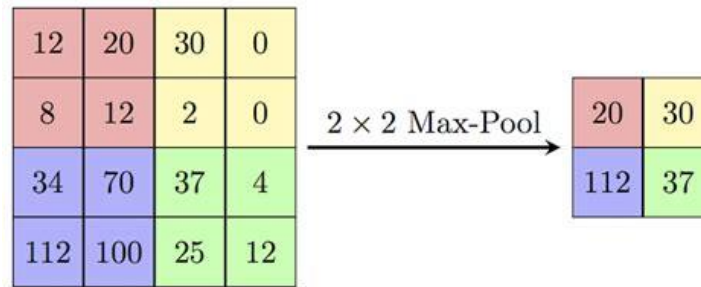


Figura 4: Operación de capa Max Pooling

- Capas completamente conectadas: Recibe una entrada después de la última capa de agrupación. En la que se convierten las matrices de los mapas de características en un vector de características de una dimensión. Contiene alrededor del 90% de los parámetros en una CNN. En esta capa se puede enviar el vector hacia la clasificación de imágenes o dejarlo como un vector de características para procesamiento posterior.

2.1.7 Técnicas y Métricas de Evaluación

Como se había observado en 2.2.3, una de las formas de clasificación que existen es la clasificación multi-etiqueta, que usualmente utiliza una evaluación conocida como matriz de confusión. La matriz de confusión es una de las métricas más comunes utilizadas en problemas de clasificación, porque sirve para problemas de clasificación binaria y clasificación multiclase [23].

Una matriz de confusión puede mostrar el número de casos donde ocurren los valores predichos y actuales. En la Tabla 1 se puede observar una matriz de confusión. En la clase predicha es el valor que se le asignó a la salida, mientras que en la clase verdadera es la clase donde debería de estar el valor. Dentro de la matriz de confusión existen cuatro diferentes tipos de clases [23]:

- Verdaderos Negativos (*True Negatives*, TN): Son el número de ejemplos negativos que fueron clasificados como negativos.
- Falsos Positivos (*False Positives*, FP): Son el número de ejemplos positivos que fueron clasificados como negativos.

- Falsos Negativos (*False Negatives*, FN): Son el número de ejemplos negativos que fueron clasificados como positivos.
- Verdaderos Positivos (*True Positives*, TP): Son el número de ejemplos positivos que fueron clasificados como positivos.

Tabla 1: Matriz de Confusión

Clase Predicha			
Clase Actual		Negativo	Positivo
	Negativo	Verdaderos Negativos (TN)	Falsos Positivos (FP)
	Positivo	Falsos Negativos (FN)	Verdaderos Positivos (TP)

Al ser una matriz de confusión multi-etiqueta se debe crear una matriz de confusión para cada clase. Por medio de los valores obtenidos de la matriz de confusión es posible obtener diferentes métricas para evaluar el modelo de clasificación. Como sería la Exactitud (PRE), una métrica que cuantifica si el modelo no etiqueta como positiva las muestras que son negativas, que se puede observar en Eq 1. Como es clasificación multi-etiqueta se tiene que hacer PRE en cada una de las clases [23] [24].

$$PRE = \frac{TP}{TP + FP} \quad (1)$$

Cobertura (REC) es una métrica que calcula la capacidad en la que el sistema puede encontrar todas las muestras positivas, se puede observar en Eq 2. Esta métrica se tendrá que mantener entre 1 y 0, donde entre un valor más cercano a 1 es mejor. Como es una clasificación multi-etiqueta es necesario hacerlo para cada clase [24].

$$REC = \frac{TP}{TP + FN} \quad (2)$$

F1 score (F1) es una métrica que es interpretada como el promedio entre los promedios de PRE y REC. Al igual que Cobertura entre más cercano a 1 será un mejor modelo. Se puede observar la Eq 3 para entender cómo obtener este valor. Se tiene que realizar el F1 Score para cada clase porque el clasificador es multi-etiqueta [24].

$$F1 = \frac{2 * (PRE * REC)}{PRE + REC} \quad (3)$$

2.2 Marco tecnológico

En la siguiente sección se presentará una descripción de las herramientas tecnológicas que serán necesarias para la realización del proyecto. Que serán las siguientes: Python, Anaconda, Jupyter Notebook, Tensorflow y Keras.

2.2.1 Python

Python es un lenguaje de programación de propósito general de alto nivel. Generalmente se utiliza como un lenguaje de scripting, páginas web, entre otras por la compilación automática a código de bytes que después es ejecutado [25]. Las principales características serían las siguientes [26]:

- Sintaxis muy clara
- Fuertes capacidades de inspección
- Interacción con objetos muy intuitiva
- Expresión natural del código procesal
- Modularidad completa
- Manejo de errores basado con excepciones
- Tipos de datos dinámicos de muy alto nivel
- Extensas bibliotecas estándar y módulos de terceros para la mayoría de las tareas
- Extensiones y módulos escrito fácilmente en C, C++
- Integrable dentro de las aplicaciones como una interfaz de scripting

2.2.2 Anaconda

Anaconda es una distribución de Python que permite realizar ciencia de los datos y aprendizaje automático de una forma más sencilla para los programadores. Anaconda cuenta con diversos paquetes y librerías open source en el área científica. Se tienen dos áreas principales dentro de Anaconda: [27]

- Conda: Un sistema de manejo de paquetes open source para los sistemas de Windows, macOS y Linux [28].
- Navigator: Una interfaz gráfica para que se puedan abrir las diferentes aplicaciones y vuelva más fácil el manejo de los diferentes ambientes y paquetes que Conda pueda manejar.

2.2.3 Jupyter Notebook

Jupyter es un ambiente web en forma de cuadernos. Que hace más eficiente y flexible al momento de estar trabajando en áreas de ciencia de datos, aprendizaje automático, entre otros [29].

2.2.4 Tensorflow

Tensorflow es una plataforma de código abierto para que se pueda implementar aprendizaje automático. Se cuenta con un ecosistema, que contiene diversas librerías y recursos para poder desarrollar e implementar el aprendizaje automático de una manera más sencilla [30].

2.2.5 Keras

Keras es librería open source que sirve como una interfaz para poder construir redes neuronales. Keras soporte ambas la computación CPU y la GPU [31].

III. Desarrollo del proyecto

En la siguiente sección se comentará acerca del producto propuesto, además de las diversas limitantes y delimitaciones que el proyecto tuvo. Por último, se mostrará la metodología de desarrollo que se escogió.

3.1 Producto Esperado

En la Figura 5 se observa se observa el diagrama que fue la base para la arquitectura del proyecto. Se creó un modelo de aprendizaje automático para clasificar distintas patologías de la retina que los seres humanos pueden padecer utilizando fotografías digitales provenientes de la OMS (Organización Mundial de la Salud). Se utilizó una red neuronal convolucional para obtener las etiquetas correspondientes a las diversas patologías que una imagen puede contener.

El modelo se llevó a cabo utilizando el lenguaje de programación Python, además de utilizar la librería de *Pandas* para el análisis de los datos, junto con *Keras* para poder implementar el uso de las redes neuronales convolucionales.

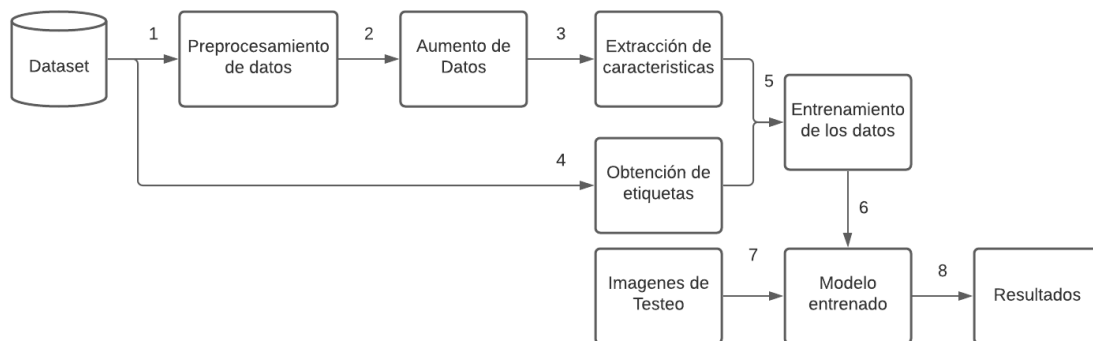


Figura 5: Diagrama de la arquitectura

En la Figura 5, se puede observar de manera gráfica como es que funciona la arquitectura. El dataset está compuesto por 3200 fotos que están organizadas en 3 folders clasificados como *Entrenamiento*, *Validación* y *Testeo*. En el procesamiento de datos se analizan y limpian los datos, en caso de que se tenga una cantidad insuficiente de datos es necesario realizar un aumento de datos. En paralelo se realiza una extracción de las características de cada etiqueta. En este caso las etiquetas son las distintas enfermedades que el dataset posee.

En la última fase, se entrena el modelo y con las imágenes de testeo se evalúa el modelo para obtener los resultados del modelo.

3.1.1 Delimitaciones y limitaciones

En la siguiente sección se mostrarán las diversas delimitación y limitantes que el proyecto tiene. Esto se aplica para mostrar el alcance del proyecto y que áreas no son posibles de acceder.

3.1.1.1 Delimitaciones

Las delimitantes del proyecto son:

- Solo se puede hacer una clasificación dentro de las 46 categorías diferentes (Ojo sano, y 45 patologías distintas).
- Solamente se podrían utilizar imágenes con una resolución igual a las imágenes del *dataset*.
- No poder clasificar imágenes que contengan otro tipo de patologías oculares.

3.1.1.2 Limitaciones

Las limitantes del proyecto son:

- Se cuenta con una computadora de rendimiento bajo para realizar el análisis de los datos.
- Solamente se cuenta con el *dataset* de la Organización Mundial de la Salud.

3.1.2 Metodología de desarrollo

Para el desarrollo de este proyecto de investigación se utilizó la metodología CRISP-DM. Una de las metodologías que son usadas para el procesamiento de los datos, en esta sección se explicará en qué consiste y cuáles fases la conforman.

La metodología de CRISP-DM es un tipo de metodología donde se provee una estructura para que se puedan realizar proyectos de procesamiento de datos. Esta metodología muestra una visión del ciclo de vida de un proyecto de procesamiento de datos, contiene las fases, las tareas de cada fase y las relaciones que existen en cada una de estas fases y tareas. La principal razón por la que se escogió esta metodología es que funciona en proyectos de procesamiento de datos, esto ayudó a conocer cuáles son las fases que hay que tomar en cuenta para el desarrollo del proyecto y la forma eficiente de avanzar con el desarrollo del proyecto. Además de que como esta metodología se adapta a cualquier tipo de proyecto no es necesario que se le vayan a realizar tantos cambios para hacer que funcione de una

manera adecuada con el proyecto [32]. En la Figura 6 se puede observar las fases de la metodología [32]:

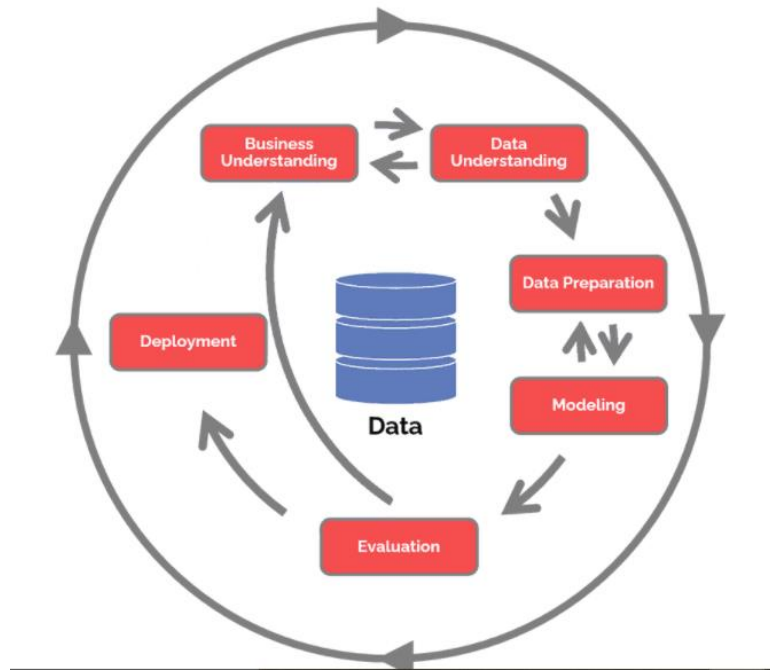


Figura 6: Fases de la metodología Crisp-DM

Una explicación detallada de las fases será explicada a continuación:

1. Entendimiento de la problemática: Originalmente esta etapa sería entendimiento del negocio, pero como no se cuenta con un negocio, se decidió alterar esta fase para que se pueda comprender la problemática y qué factores pueden existir en ella [32]. Las etapas que se realizarán en esta fase serán:
 - a. Determinar objetivos del proyecto de investigación
 - b. Determinar requerimientos del proyecto de investigación
2. Entendimiento de los datos: Esta fase inicia con la obtención de los datos iniciales, después sigue con diferentes actividades para poder comprender los datos y descubrir hechos dentro de los datos [32]. Las actividades que se piensan realizar en esta etapa son las siguientes:
 - a. Obtención de los datos a utilizar en el proyecto
 - b. Exploración de los datos obtenidos

3. Preparación de los datos: Se realizan las actividades necesarias para crear el *dataset* final que será usado durante todo el proyecto [32]. Las actividades que se realizaran en esta etapa son las siguientes:
 - a. Instalación de herramientas
 - b. Limpieza de los datos
4. Modelado: Se elige la selección del modelado y se aplica. Además de obtener los resultados del modelo. Las actividades que se realizaran en esta etapa son las siguientes:
 - a. Configuración de la red convolucional
 - b. Entrenamiento del modelo
 - c. Obtención de resultados
5. Evaluación: Al desarrollar un modelo en la sección anterior es necesario conocer si el modelo en verdad es útil. Entonces se tiene que evaluar para determinar si el modelo llega a un resultado satisfactorio. Las actividades que se realizaran en esta etapa son las siguientes:
 - a. Gráficación de los resultados
 - b. Evaluación de los resultados
6. Despliegue: Como finalización del proyecto es necesario dar un reporte final para encontrar cuáles fueron las áreas que se podía mejorar y dar un mantenimiento final. Las actividades que se realizaran en esta etapa son las siguientes:
 - a. Mantenimiento del proyecto
 - b. Reporte final del proyecto
 - c. Presentar el proyecto

3.2 Entendimiento de la problemática

Se realizaron cambios a esta sección, en un inicio era Entendimiento del negocio, se tomó la decisión de cambiar esta fase a el entendimiento de la problemática, donde se puede dar una idea general de cuál es la razón por la que se realiza el proyecto. Las etapas que se realizaron son las siguientes:

3.2.1 Determinar objetivos del proyecto de Investigación

El objetivo principal es el desarrollo de un clasificador multi-etiqueta de retinopatías, que fue creado por medio de un dataset que la OMS ofreció.

Para más información acerca de los objetivos de la investigación, se pueden encontrar los objetivos específicos en la Sección 1.3.

3.2.2 Determinar requerimientos del proyecto de investigación

Los requerimientos del proyecto son la obtención de valores favorables, como Exactitud, Cobertura o F1 Score al momento de realizar la matriz de confusión que se obtiene al finalizar el Proyecto.

3.3 Entendimiento de los datos

La comprensión de los datos es un área clave para obtener una idea general de la estructura u obtener un cierto conocimiento de la información que será usada en el proyecto. Esto puede darnos diversas nociones o juicios acerca de cómo los datos actúan en conjunto.

3.3.1 Obtención de los datos a utilizar en el proyecto

La obtención de los datos fue mediante la página web conocida como Kaggle, una página donde los usuarios pueden publicar diferentes datasets. El dataset utilizado en este proyecto fue creado por la OMS que originalmente fue obtenido de la publicación creada por Pachade, Porwal, entre otros [10].

3.3.2 Exploración de los datos obtenidos

Se encontraron distintos valores acerca de los datos como principalmente la composición de las enfermedades que se observan en la Figura 7 en donde se puede observar una gráfica de mayor a menor de cada una de las enfermedades que existen en el dataset. Para información más detallada de la composición de la Figura 7, se puede referir a la Tabla 2.

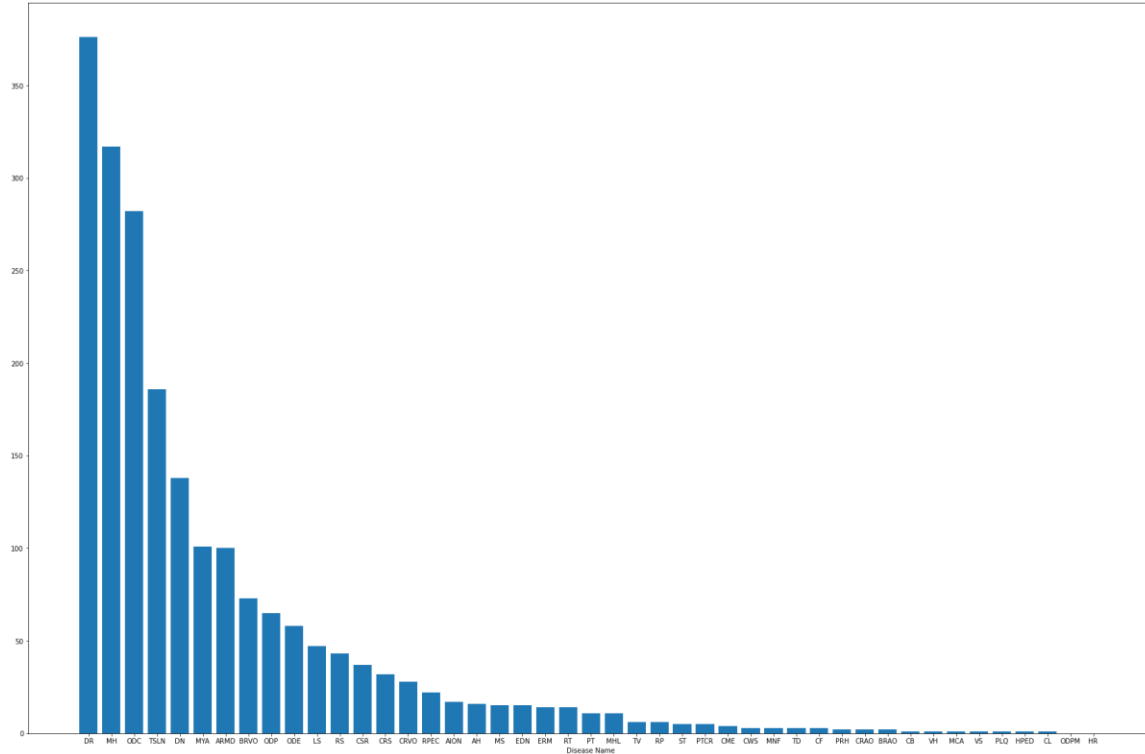


Figura 7: Gráfica de Cantidad de Enfermedades.

Se encontró la cantidad de cada una de las retinopatías que había en el dataset. Con la información recopilada se obtuvo la Tabla 2, que muestra el número de imágenes que cada una de las diferentes enfermedades muestra como positivo.

Tabla 2: Cantidad de imágenes por retinopatía.

R.	#	R.	#	R.	#	R.	#	R	#
DR	376	MS	15	AION	17	CWS	3	PTCR	5
ARMD	100	CSR	37	PT	11	CB	1	CF	3
MH	317	ODC	282	RT	14	ODPM	0	VH	1
DN	138	CRVO	28	RS	43	PRH	2	MCA	1
MYA	101	TV	6	CRS	32	MNF	3	VS	1
BRVO	73	AH	16	EDN	15	HR	0	BRAO	2
TSLN	186	ODP	65	RPEC	22	CRAO	2	PLQ	1
ERM	14	ODE	58	MHL	11	TD	3	HPED	1
LS	47	ST	5	RP	6	CME	4	CL	1

En la Tabla 3 se pueden observar las cinco retinopatías más comunes dentro del dataset.

Tabla 3: Retinopatías más comunes en el dataset

Enfermedad	Cantidad de imágenes con la enfermedad
DR	376
MH	317
ODC	282
TSLN	186
DN	138

3.3.3 Verificar la calidad de los datos

Existe la necesidad de conocer la calidad de los datos con los que se trabajó. Entonces se revisaron distintos valores de las imágenes como sus dimensiones, en qué formato están las imágenes.

Se crearon scripts para conocer las características de las imágenes. Las imágenes están divididas en tres folders. El primer folder es conocido como *Entrenamiento*, el segundo es el folder de *Testeo* y por último está el folder de *Validación*. Cada folder tuvo una función para el modelo, el primero entreno al modelo, el segundo comprobó el modelo además de las diferentes redes que se pensaban utilizar y el último fue usado para validar que el modelo funcione ingresando un nuevo set de imágenes no antes vistas.

El folder de entrenamiento contiene las imágenes en formato PNG, además de un archivo en formato CSV que muestra información acerca de todas las imágenes, esta información son el ID de las imágenes, si la imagen tiene una enfermedad y cuál enfermedad tiene la imagen. El folder guarda 1920 imágenes. Las dimensiones de las imágenes se observan en tres pares de dimensiones. Las dimensiones se pueden observar en la Tabla 4.

Tabla 4: Cantidad de dimensiones en Dataset Training

Ancho	Altura	Cantidad
2048	1536	150
2144	1424	1493
4288	2848	277

La Figura 8 que se observa a continuación es usada para mostrar las diferencias de dimensiones en cada uno de los pares. En el eje Y se encuentra la altura de las imágenes, mientras que en el eje X se observa el ancho de las imágenes.

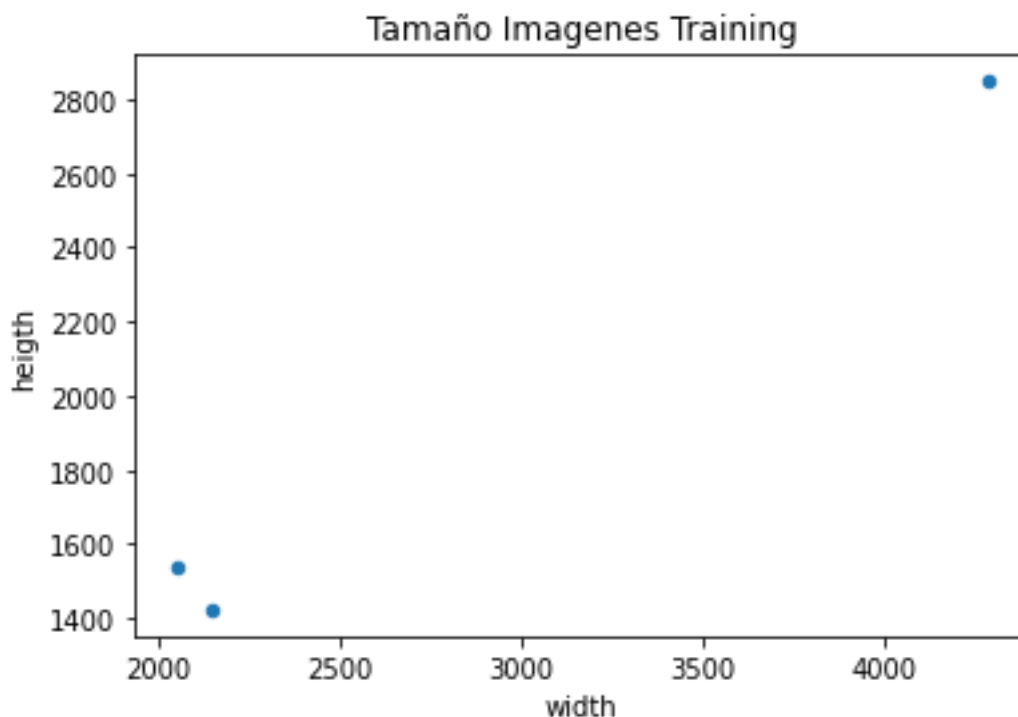


Figura 8: Gráfica de las dimensiones de las imágenes del dataset Training

El folder de test contiene las imágenes en formato PNG, además de un archivo en formato CSV que muestra información acerca de todas las imágenes, esta información son el ID de las imágenes, la retinopatía que la imagen muestre y si la imagen exhibe o no una enfermedad. El folder guarda 640 imágenes. Las dimensiones de las imágenes se observan en tres pares de dimensiones. Las dimensiones se pueden observar en la Tabla 5.

Tabla 5: Cantidad de dimensiones en Dataset Test

Ancho	Altura	Cantidad
2048	1536	73
2144	1424	439
4288	2848	128

La Figura 9 que se observa a continuación es usada para mostrar las diferencias de dimensiones en cada uno de los pares. En el eje Y se encuentra la altura de las imágenes, mientras que en el eje X se observa el ancho de las imágenes.

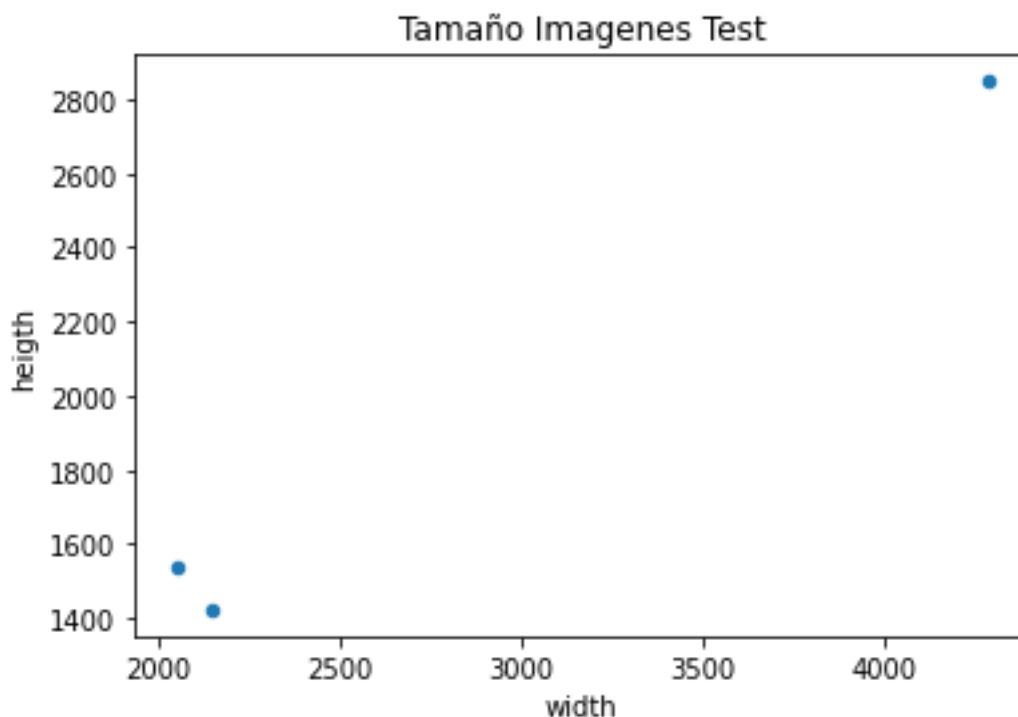


Figura 9: Gráfica de las dimensiones de las imágenes del dataset Test

El folder de validación contiene las imágenes en formato PNG, además de un archivo en formato CSV que muestra información acerca de todas las imágenes: El ID la retinopatía que la imagen muestre y si la imagen exhibe o no una enfermedad. El folder guarda 640 imágenes. Las dimensiones de las imágenes se observan en tres pares de dimensiones. Las dimensiones se pueden observar en la Tabla 6.

Tabla 6: Cantidad de dimensiones en Dataset Test

Ancho	Altura	Cantidad
2048	1536	53
2144	1424	495
4288	2848	92

Con esta información se crea una gráfica para comprender cómo están organizados los valores de altura y anchura de una manera gráfica. La Figura 10 muestra las diferentes dimensiones.

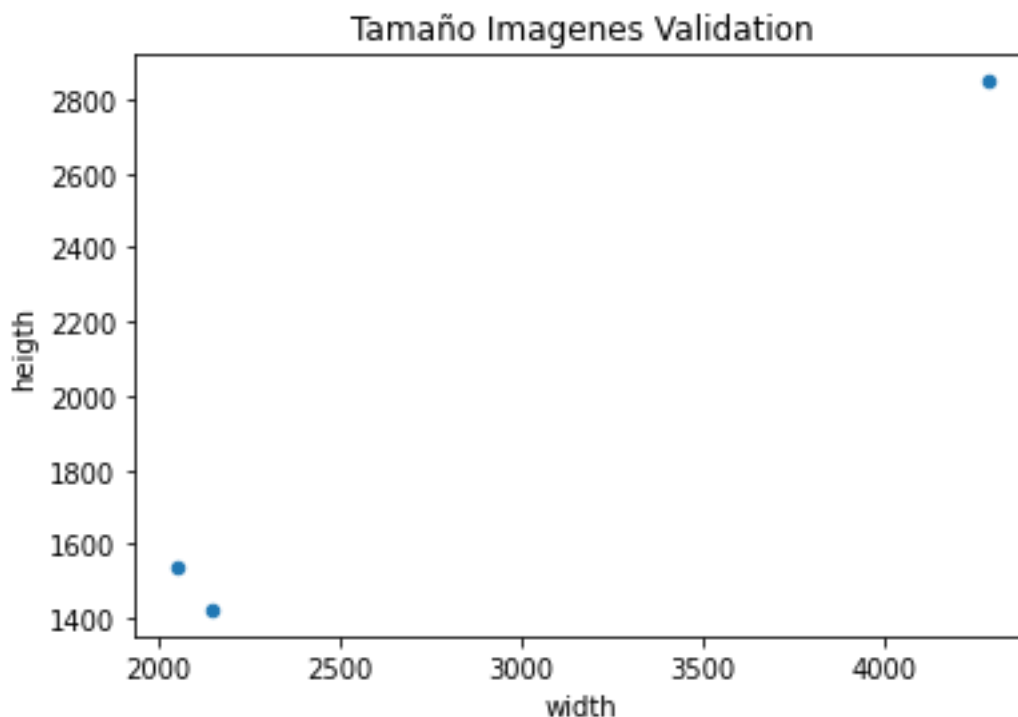


Figura 10: Gráfica de las dimensiones de las imágenes del dataset Validation

3.4 Preparación de los datos

Inicialmente se utilizó una computadora para entrenar el modelo, pero tardó 3 días en terminar. Se optó por utilizar Google Colab, además de que fue necesario separar cada una de las fotografías en sub-folders por cada correspondiente retinopatía. Finalmente fue necesario subir cada uno de los folders en Google Drive para poder utilizarlos al momento de entrenar el modelo.

3.4.1 Instalación de herramientas

La instalación de las diversas herramientas fue un proceso simple y rápido. Por el hecho de utilizar al inicio del proyecto la herramienta de Google Colaboratory. Después se instaló la herramienta de Anaconda en la computadora del presentante de este trabajo. Anaconda tiene una variedad de herramientas, la que se utilizó fue Jupyter Notebook. Un ambiente web en el que se pueden crear notebooks, se escogió por su simpleza, su parecido a Google Colaboratory y que se cuenta con experiencia de esta herramienta.

3.5 Modelado

En la clasificación de imágenes uno de los recursos usados son los modelos de la arquitectura de Redes Neuronales Convolucionales. En [33] se realizó una comparativa de los diferentes modelos para la clasificación convolucional, donde se concluyó que la mejor arquitectura era ResNet50 con 0.9733 de Exactitud, otro modelo con una buena Exactitud será VGG16 con 0.9707 de Exactitud. Estos modelos fueron elegidos para crear el clasificador multi-etiqueta.

3.5.1 Configuración de la red convolucional

El dataset de imágenes solamente cuenta con 1920 imágenes para entrenamiento, entonces se tomó la decisión de hacer uso de una red pre-entrenada en particular ResNet50 o VGG16, para poder configurar la red convolucional y hacer uso de los pesos.

La configuración es por medio secuencial, que es cuando pasa una capa y después otra de una forma secuencial. Dentro de esta configuración existen tres capas principales que son utilizadas. Primero es una Capa de Convolución que es la que procesa las entradas de las neuronas. A continuación, hay una Capa de *Pooling* que reemplaza las salidas de la red con un promedio estadístico de los valores para reducir el tamaño. Y por último se encuentran las Capas Totalmente Conectadas que es donde se realizan las funciones de activación.

Primero, se recreó la estructura del modelo de la red convolucional VGG16. La estructura VGG16, son trece capas convolucionales, cinco capas de *Max Pooling*, y tres capas densas. La suma total es de 21 capas, pero solamente 16 de las capas tienen peso. La última capa que es agregada es una capa densa de predicción que tendrá el número total de categorías en este caso 21.

El modelo de ResNet50 es una red neuronal convolucional con 50 capas. El modelo inicia con una capa Convolucional de 7x7 hasta llegar a la capa final que es una capa *Pooling* que contienen mil elementos y es normalizada para cada mapa de características. Entonces se obtiene como resultado un vector dimensional 1000 que es enviado a la capa de predicción, una capa sigmoidea con la cantidad de categorías que haya [34].

3.5.2 Entrenamiento del modelo

El entrenamiento del modelo primero fue realizado en una computadora portátil, donde estaba todo el dataset. Pero después de varias pruebas se optó por utilizar Google Colab.

Para el entrenamiento primero fue necesario importar las librerías a utilizar que pueden ser observadas en la Figura 11.

```
[8] from tensorflow.keras.applications.vgg16 import VGG16
    from tensorflow.keras.applications.vgg16 import preprocess_input
```

Figura 11: Librerías VGG16

Antes de realizar el entrenamiento del modelo es necesario crear un *ImageDataGenerator* donde van a encontrarse todas las imágenes y las categorías que contienen. Se realizaron experimentos para ver si un aumento de datos sería efectivo, pero hizo que la exactitud fuera menor que si no hubiera aumento de datos. Se puede ver el *ImageDataGenerator* en la Figura 12.

```
train_batches = ImageDataGenerator(
    preprocessing_function=tf.keras.applications.vgg16.preprocess_input).flow_from_directory(directory = train_path,
        target_size = (224,224),
        classes = dirlist2,
        batch_size = 20)

test_batches = ImageDataGenerator(
    preprocessing_function=tf.keras.applications.vgg16.preprocess_input).flow_from_directory(directory=test_path,
        target_size=(224,224),
        classes=dirlist,
        batch_size=10,
        shuffle=False)
```

Figura 12: ImageDataGenerator

Inicialmente solamente se había copiado la arquitectura de VGG16 que se encuentra en [35]. Pero se investigó más acerca de la transferencia del aprendizaje, esto se realizó al obtener los pesos de la red convolucional del proyecto de ImageNet, esto haría que la red convolucional tenga mayor Exactitud porque ya tiene una cierta experiencia o aprendizaje. La forma en la que se transfieren los pesos en el código se observa en la Figura 13.

```
## Loading VGG16 model
base_model = VGG16(weights="imagenet")
base_model.trainable = False ## Not trainable weights
```

Figura 13: Transfer Learning

Como ya se tiene la base del modelo que son las 16 capas que tiene VGG16, es necesario hacer una secuencia de estos para que pueda correr de manera correcta. Algo que hay que notar es en la última función, que es donde se especifica cuántas categorías son las que van

a ser utilizadas en el modelo. En la Figura 14 se puede observar la secuencia del modelo. Para conocer más acerca de cómo ocurre esta secuencia se encuentra en la Sección 3.5.1.

```
base_model.summary()
```

Model: "sequential"		
Layer (type)	Output Shape	Param #
vgg16 (Functional)	(None, 1000)	138357544
flatten (Flatten)	(None, 1000)	0
dense (Dense)	(None, 4096)	4100096
dense_1 (Dense)	(None, 4096)	16781312
dense_2 (Dense)	(None, 7)	28679

=====
Total params: 159,267,631
Trainable params: 20,910,087
Non-trainable params: 138,357,544

Ilustración 14: Resumen de modelo VGG16

Finalmente se realiza la compilación y el ajuste del modelo. El código de esto se puede observar en la Figura 15. Con la compilación se optó por utilizar el optimizador *Adam* por el hecho de que es usado en diferentes publicaciones y en el GitHub oficial de la red VGG16, mientras que en la función de *Loss* se utilizó la *Categorical CrossEntropy* que es la utilizada en clasificación Multi-Etiqueta. Además, se incluyó una función de *Early Stopping* que es utilizada para evitar el sobre ajuste. Por último, se encuentra el ajuste, que es la forma en la que los parámetros son nivelados en el modelo para mejorar la Exactitud. La forma en la que se realiza esto es por medio de un algoritmo sigmoideo que crea un modelo de aprendizaje automática y compara estos valores con valores reales de testeo [36].

```
from tensorflow.keras.callbacks import EarlyStopping

base_model.compile(
    optimizer='adam',
    loss='categorical_crossentropy',
    metrics=['accuracy'],
)

es = EarlyStopping(monitor='val_accuracy', mode='max', patience=5, restore_best_weights=True)

base_model.fit(x = train_batches, validation_data = test_batches, epochs=10, callbacks=[es])
```

Figura 15: Código de Ajuste y Compilación

En la Figura 15 también se puede observar la cantidad de épocas que el modelo ejecuta, en este caso se optaron por 10 épocas porque tarda alrededor de 8 horas realizar el proceso y

por el *Early Stopping* usualmente dejaba de realizarlo hasta la 8va o 9na época de entrenamiento. En el caso de ResNet50 es parecido solo que se tiene que cargar ResNet50 desde la siguiente línea de código que es parecida a la que es utilizada con VGG16, la línea puede ser observada en la Figura 16.

```
from tensorflow.keras.applications.resnet50 import ResNet50
from tensorflow.keras.applications.resnet50 import preprocess_input

## Loading ResNet50 model
res_model = ResNet50(weights="imagenet")
res_model.trainable = False ## Not trainable weights
```

Figura 16: Carga de ResNet50

IV. Resultados y Discusiones

4.1 Formas de validación y Obtención de Resultados

En la Sección 2.1.7, se describió la matriz de confusión además de las diversas métricas que se pueden utilizar para poder evaluar el algoritmo de clasificación que se desarrolló.

El dataset de retinopatías se divide en 3 diferentes sets de datos:

- Training Set: Que contiene 1920 archivos que servirán para el entrenamiento.
- Evaluation Set: Contiene 640 archivos que será utilizada en la evaluación.
- Test Set: Contiene 640 archivos que será utilizada en el testeo.

Se deberá calcular una matriz de confusión para cada una de las etiquetas que se encuentra en el dataset. Además de que se utilizó cada una de las diversas métricas anteriormente explicadas en la Sección 2.1.7 como sería *Exactitud* que se puede observar en la Eq 1, *Cobertura* en Eq 2 y por último el *F1 Score* en Eq 3. Cada una de estas métricas fueron utilizadas para validar la red neuronal.

4.2 Resultados

La siguiente sección dará a conocer los resultados obtenidos por las diversas configuraciones que fueron realizadas con las redes convolucionales, junto con las métricas que serán utilizadas para dar una conclusión al proyecto y una discusión acerca de los resultados provenientes de las configuraciones.

Se realizaron diferentes configuraciones en las redes convolucionales, así como con la cantidad de datos y categorías que se estaban analizando. Inicialmente se empezó a trabajar con la red neuronal VGG16 con 46 diferentes categorías. Pero se tuvieron que eliminar dos categorías al inicio porque no tenían ninguna foto en el dataset. Después fueron eliminadas 8 categorías más porque no existían en el dataset de testeo.

La primera prueba se realizó con 30 épocas, 36 categorías, sin utilizar *transfer learning* y utilizando una computadora *DELL Inspiron 15 con Intel Core i5-7200u, 8GB de RAM e Intel HD Graphics 620*. Se realizaron alrededor de 18 épocas, pero la computadora falló y como última época se obtuvo .2405 de Exactitud. Después de esto el proyecto se movió a Colab y se empezó a utilizar una red pre-entrenada VGG16 con *transfer learning*.

Las siguientes pruebas fueron realizadas con 36 categorías y después con 21 categorías exclusivamente en la red VGG16. Los resultados obtenidos en las pruebas de 36 categorías fue una Exactitud de 0.3252 después de 6 épocas. Originalmente se codificaron 10, pero por el *Early Stopping* detectó *Overfitting* y terminó el entrenamiento. En el caso de las 21 categorías, fue una Exactitud de 0.3964 después de 7 épocas. La razón para elegir inicialmente 36 categorías es que se escogió solamente las categorías mayores a 15 imágenes en el dataset de entrenamiento. Para las 21 categorías se eligieron solamente las que tenían mayor a 30 imágenes en el dataset.

Después de investigar más acerca de diversas redes convolucionales, se encontró información de ResNet50, entonces se decidió experimentar con esta red para conocer cuáles resultados se podrían obtener. La primera prueba con ResNet50 fue con 21 clases, y el resultado obtenido fue una Exactitud de 0.4656 en solamente 10 épocas. Con esta información se decidió experimentar más con ResNet50 que con VGG16.

La siguiente prueba que se realizó fue con 15 categorías, que son las categorías mayores a 50 imágenes. Se iba a realizar la prueba con ambas redes convolucionales, pero cuando la red VGG16 obtuvo una Exactitud menor que cuando se utilizaron 21 clases que fue de 0.3768.

Después de utilizar las 21 categorías, se realizó un aumento de datos para observar cómo es que el modelo lo experimentaría. Se realizaron los siguientes cambios:

- Rango de Rotación: 40 grados.
- Cambio de rango de altura: 200 pixeles (Sería mover la imagen arriba o abajo).
- Cambio de rango de anchura: 200 pixeles (Sería mover la imagen izquierda o derecha).
- Re-escalado: 1/255 Para mantener los pixeles en un rango de 0 a 255.
- Rango de Zoom: Aplicar zoom al azar en una imagen.
- Voltear Horizontalmente: Voltear las imágenes de forma horizontal.
- Fill Mode: Rellenar los pixeles, después de un cambio de ancho/altura.

Después de obtener un resultado de Exactitud de 0.2807 se optó por disminuir aún más la cantidad de clases que el dataset está clasificando.

Las últimas pruebas realizadas fueron con 7 y 4 categorías. En el caso de 7 clases, son las mayores a 100 imágenes en el dataset, mientras que en el caso de las 4 categorías son las

que tuvieran más de 150 imágenes en el dataset. Con la prueba de 7 clases, se obtuvo una Exactitud de 0.4927, utilizando VGG16. En cambio, la red ResNet50 obtuvo una Exactitud de 0.5507 que sería la primera vez que se obtiene un valor mayor a 0.500. La última prueba realizada fue con 4 categorías. Se obtuvo una Exactitud de 0.6357 utilizando ResNet50, y con VGG16 fue una Exactitud de 0.5728.

4.1.2 Evaluación y Graficación de los resultados

El último experimento realizado con la red VGG16 dio como resultados un modelo entrenado que daba una Exactitud de 0.5728. Esto representa una clasificación un poco mayor a la mitad de todos los casos. En el último experimento se utilizaron 4 categorías que son DR, MH, ODC y TSLN. Para conocer los resultados en una prueba se obtuvieron 4 matrices de confusión. Una matriz de confusión es una tabla que tiene 4 combinaciones diferentes de valores predichos y reales.

En la Figura 17, se puede observar la matriz de confusión para la categoría de DR con la red VGG16. Existen 4 cuadrantes:

- Verdaderos Negativos: El primer cuadrante que son Real 0 y Predicho 0, son los valores que no tienen la retinopatía y fueron clasificados como ausentes de la retinopatía.
- Falsos Positivos: El segundo cuadrante que son Real 0 y Predicho 1, son los valores que no tienen la retinopatía, pero son clasificados con ella.
- Falso Negativo: El tercer cuadrante que son Real 1 y Predicho 0, son los valores que tienen la retinopatía, pero no son clasificados como tal.
- Verdaderos Positivos: El cuarto cuadrante que son Real 1 y Predicho 1, son los valores que tienen la retinopatía y son clasificados con ella.

En el dataset de *Test*, hay 124 imágenes clasificadas con DR, entonces el clasificador pudo clasificar 99 de estas imágenes de manera exitosa. Aunque también tuvo errores como clasificar 25 sin retinopatía cuando se padecía y 80 que no padecía la retinopatía, pero fueron clasificados como que la padecían.

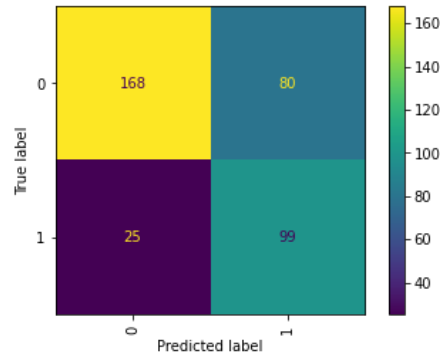


Figura 17: Matriz de Confusión VGG16 DR

En la Figura 18, se observa la matriz de confusión de MH. En el dataset de *Test*, hay 104 imágenes clasificadas con MH, entonces el clasificador pudo clasificar 5 de estas imágenes de manera exitosa. Aunque también tuvo errores como clasificar 86 sin retinopatía cuando se padecía y 62 que no padecía la retinopatía, pero fueron clasificados como que la padecían.

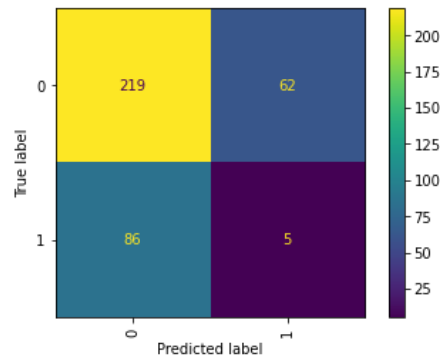


Figura 18: Matriz de Confusión VGG16 MH

En la Figura 19, se observa la matriz de confusión de ODC. En el dataset de *Test*, hay 91 imágenes clasificadas con ODC, entonces el clasificador pudo clasificar 9 de estas imágenes de manera exitosa. Aunque también tuvo errores como clasificar 95 sin retinopatía cuando se padecía y 50 que no padecía la retinopatía, pero fueron clasificados como que la padecían.

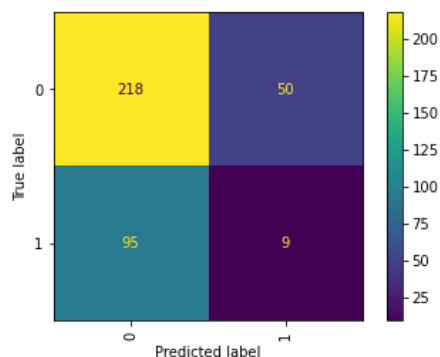


Figura 19: Matriz de Confusión VGG16 ODC

En la Figura 20, se observa la matriz de confusión de TSLN. En el dataset de *Test*, hay 53 imágenes clasificadas con ODC, entonces el clasificador pudo clasificar 15 de estas imágenes de manera exitosa. Aunque también tuvo errores como clasificar 38 sin retinopatía cuando se padecía y 52 que no padecía la retinopatía, pero fueron clasificados como que la padecían.

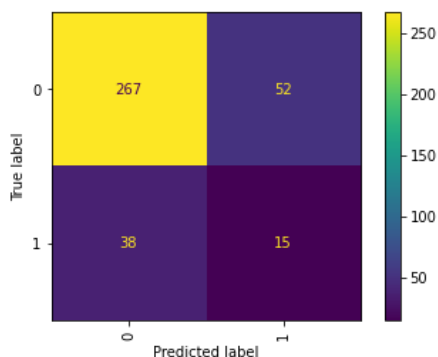


Figura 20: Matriz de Confusión VGG16 TSLN

Las siguientes matrices de confusión son las obtenidas por la red ResNet50. Igual que el experimento con la red VGG16, se utilizaron 4 clases que son: DR, MH, ODC y TSLN. En la Figura 21, se observa la matriz de confusión de DR con la red ResNet50. En el dataset de *Test*, hay 124 imágenes clasificadas con DR, entonces el clasificador pudo clasificar 111 de estas imágenes de manera exitosa. Aunque también tuvo errores como clasificar 13 sin retinopatía cuando se padecía y 113 que no padecía la retinopatía, pero fueron clasificados como que la padecían.

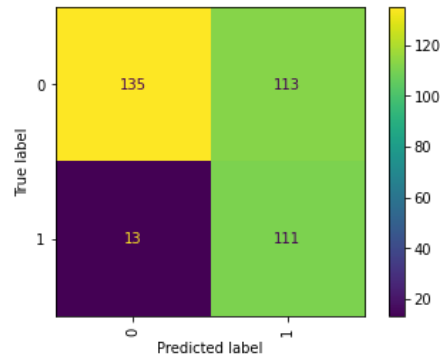


Figura 21: Matriz de Confusión ResNet50 DR

En la Figura 22, se observa la matriz de confusión de MH con la red ResNet50. En el dataset de *Test*, hay 104 imágenes clasificadas con MH, entonces el clasificador pudo clasificar 14 de estas imágenes de manera exitosa. Aunque también tuvo errores como clasificar 77 sin retinopatía cuando se padecía y 79 que no padecía la retinopatía, pero fueron clasificados como que la padecían.

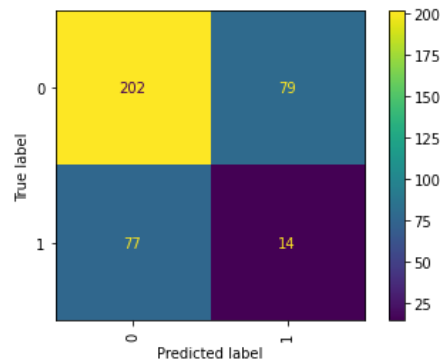


Figura 22: Matriz de Confusión ResNet50 MH

En la Figura 23, se observa la matriz de confusión de ODC. En el dataset de *Test*, hay 91 imágenes clasificadas con ODC, entonces el clasificador pudo clasificar 4 de estas imágenes de manera exitosa. Aunque también tuvo errores como clasificar 100 sin retinopatía cuando se padecía y 42 que no padecía la retinopatía, pero fueron clasificados como que la padecían.

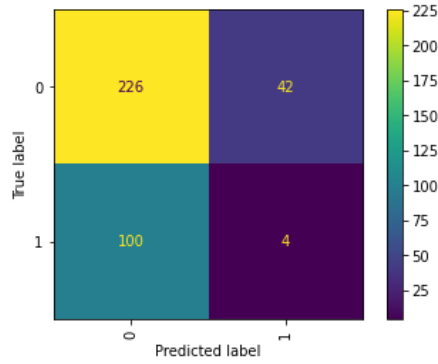


Figura 23: Matriz de Confusión ResNet50 ODC

En la Figura 24, se observa la matriz de confusión de TSLN. En el dataset de *Test*, hay 53 imágenes clasificadas con ODC, entonces el clasificador pudo clasificar 5 de estas imágenes de manera exitosa. Aunque también tuvo errores como clasificar 48 sin retinopatía y 4 que no padecía la retinopatía, pero fueron clasificados como que la padecían.

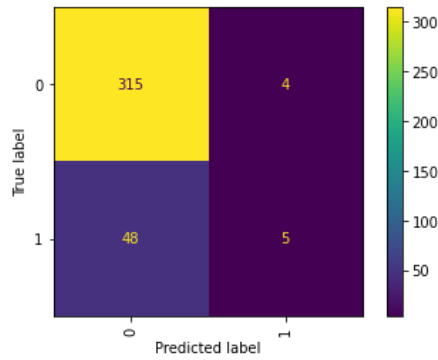


Figura 24: Matriz de Confusión ResNet50 TSLN

4.1.3 Evaluación de los resultados

Las matrices de confusión pueden dar valiosa información, además de que también se pueden realizar diferentes fórmulas como para la Exactitud, *Cobertura* y el *F1 Score* que pueden ser encontrados en las Eq. 1, Eq. 2 y Eq. 3 respectivamente. Para encontrar más información de estas métricas pueden consultar la Sección 2.1.7. La Tabla 7 muestra las 3 métricas utilizadas para evaluar la red convolucional. Con la tabla se pueden ver distintas observaciones como sería: En la clase DR con la red ResNet50 hay un mayor *Cobertura* de 0.89516 en comparación con VGG16 con una Cobertura de 0.79838, pero se tiene una Exactitud más baja. Esto ocurre en una manera parecida con su contraparte de la red VGG16, pero en esta se tiene un mayor grado de Exactitud con 0.55307.

Los otros valores de MH, ODC, y TSLN, usualmente tienen valores bastante parecidos con excepción de TSLN. TSLN es la categoría con menor cantidad de imágenes, pero también demuestra una Exactitud mayor que 0.5, pero no da un Cobertura tan alto para dar un resultado favorable su Exactitud con 0.55556 y Cobertura de 0.09433. Por medio de una evaluación del promedio de cada una de las 4 F1 Scores que cada red tiene, se obtuvo que VGG16, tuvo un mayor valor de F1 Score promedio de 0.269278 en comparación con ResNet50 que fue de 0.25118.

Tabla 7: Métricas de matrices de confusión

Clase	Red	Exactitud	Cobertura	F1 Score
DR	VGG16	0.55307	0.79838	0.6534
MH	VGG16	0.07462	0.05494	0.06329
ODC	VGG16	0.15254	0.08653	0.11042
TSLN	VGG16	0.22388	0.28301	0.25
DR	ResNet50	0.09062	0.89516	0.63793
MH	ResNet50	0.15053	0.15384	0.15217
ODC	ResNet50	0.08695	0.38461	0.05333
TSLN	ResNet50	0.55556	0.09433	0.16129

4.2 Discusiones

Inicialmente, se tenía una red convolucional con 36 diferentes categorías que daba una Exactitud de 0.2405, al finalizar las diferentes configuraciones se pudo encontrar como es que con cada una de ellas se obtenían mejores o peores valores. Como sería con el caso del aumento de datos, donde ocurrió una Exactitud más baja que la original sin ella. También se pudo comprender que la mayoría de los problemas que provenían de la clasificación realizada por la red era el desequilibrio que había entre la cantidad de imágenes que cada una de las clases tenía.

Se puede observar que la categoría con mayor F1 Score es DR, que es la categoría con la mayor cantidad de imágenes en el dataset de entrenamiento y en el dataset de testeo. Esto puede generar un desequilibrio con las otras categorías. Se puede observar que existen valores muy pobres con las clases de MH, ODC y TSLN. En ambas redes dieron valores menores a 0.5 de Exactitud y Cobertura con excepción de TSLN en la Exactitud de ResNet50.

En general las clases tuvieron en cierto modo valores similares, como sería con el caso de ODC y MH, que ambas tuvieron Exactitud y Cobertura bajos. Esto se puede observar en su F1 Score donde ninguno supera los 0.20. El más cercano fue MH, con un valor de Exactitud y Cobertura similar posiblemente con una mayor cantidad de imágenes se hubiera podido incrementar estos valores. Pero en el caso de DR en ambas redes, se puede ver con un Cobertura alto, significando que muchas imágenes son clasificadas con esta retinopatía, pero no todas están correctamente asignadas. Esto sería lo contrario a TSLN en ResNet50 donde una Exactitud más alta que el Cobertura hace mención de que la mayoría de las entradas están correctas, pero es muy específica para asignar una imagen a esta categoría.

Los resultados indican que la configuración que obtuvo mejores resultados sería la VGG16 porque en promedio con las 4 categorías dio un resultado mayor de 0.269278 en comparación con ResNet50 que tuvo un valor de 0.25118. Esto puede haber ocurrido por la baja Exactitud de 0.09062 que ResNet50 tuvo en DR en comparación con VGG16 de 0.55307.

V. Conclusiones

El propósito del proyecto era el desarrollo de un clasificador multi-etiqueta de retinopatías, utilizando un dataset ofrecido por la OMS. Asimismo, se tenían objetivos específicos como analizar las imágenes del dataset de retinopatías, la forma en que se llevó a cabo fue por medio de un análisis exploratorio de datos. En el análisis se encontró información, como cuales eran las categorías que no tenían ninguna imagen, cuál era la categoría con mayor cantidad de fotos, las dimensiones de las imágenes. El siguiente objetivo fue el preprocesamiento de imágenes, esto se completó utilizando el objeto de *ImageDataGenerator*, la segregación de las clases de acuerdo con la cantidad de imágenes que contenían y subir las imágenes a Google Drive para usarlas en Colab. El tercer objetivo, era el entrenamiento de las redes convolucionales primero solamente se usaba la red VGG16, pero después de investigar acerca de ResNet50 se decidió utilizar ambas para poder encontrar la mejor configuración para desarrollar el clasificador. El ultimo objetivo, era la evaluación de las configuraciones de la red convolucional, la forma en que se realizó es por medio de diversas métricas como la Exactitud de una clase en una red, Cobertura y el F1 Score para dar un valor mas promedio de las dos métricas anteriores donde mas cerca se encuentre a 1 mejor será el valor.

Los resultados obtenidos nos darán la respuesta a la pregunta de investigación que fue realizada, ¿Cuál es la configuración que tuvo mayor nivel de precisión al momento de clasificar las retinopatías? Conforme a los resultados obtenidos, se puede decir que la red neuronal con la mejor configuración es VGG16 con 4 clases utilizando el dataset de la OMS. Se realizaron diversos experimentos para conocer los valores que daban, como disminuir las clases que se usaban, cambiar el número de épocas, o realizar aumento de datos para las imágenes. Se concluye que esta es la mejor configuración porque obtuvo mejores resultados en las métricas que serian la Exactitud, Cobertura y el F1 Score. En este caso, la mas importante es el F1 Score que nos dio un valor por cada clase que se usó. Los valores eran muy similares, pero VGG16 obtuvo una puntuación promedio de 0.269278 donde obtuvo mejores valores en las clases DR, MH y TSLN. Mientras que ResNet50, obtuvo una puntuación de F1 promedia de 0.25118.

Para concluir, se recomienda que en trabajos futuros se encuentre la forma de utilizar distintos datasets o realizar un aumento de datos para que no exista un desequilibrio entre las retinopatías. Porque al observar los resultados en ambas redes neuronales a la retinopatía diabética (DR) es que la obtuvo los valores mas altos en Exactitud o Cobertura. Esta también es la retinopatía con mayor numero de imágenes, y es la principal causa de ceguera en las personas de la edad adulta al no hacer una diagnosis temprana, además de ser la retinopatía más común a nivel mundial [7]. En comparación con los demás resultados que todos obtuvieron valores menores a 0.5 en Exactitud y Cobertura, exceptuando la Exactitud de TSLN. Por esto se considera que se utilicen diferentes formas para aumentar la cantidad de imágenes que serán usadas.

Referencias

- [1 F. J. Alañón Fernández, M. Cardenas Lara, M. A. Alañón Fernández y A. Martos Aguilera, «ANATOMÍA Y FISIOLOGÍA DEL APARATO OCULAR,» 2011. [En línea]. Available: https://web.archive.org/web/20120617051604/http://www.sepeap.org/archivos/libros/OFTALMOLOGIA/Ar_1_8_44_APR_18.pdf. [Último acceso: 23 Agosto 2021].
- [2 D. D'Amico, «Diseases of the Retina,» *Medical Progress*, vol. 331, nº 2, pp. 95-106, 1994.
- [3 P. Watkins, «Retinopathy,» *ABC of diabetes*, vol. 326 , pp. 924-926, 2003.
- [4 P. Xiangji, J. Kai, C. Jing, L. Zhifang, W. Jian, Y. Kun, L. Yifei, X. Yufeng, S. Zhaoan, J. Jiekai, Y. Ke y Y. Juan, «Multi-label classification of retinal lesions in diabetic retinopathy for automatic analysis of fundus fluorescein angiography based on deep learning,» *Graefe's Archive for Clinical and Experimental Ophthalmology*, vol. 258, p. 779–785, 2020.
- [5 N. Rajagopala, V. Narasimhan, S. Kunnavakkam Vinjimoor y J. Aiyer, «Deep CNN framework for retinal disease diagnosis using optical coherence tomography images,» *J Ambient Intell Human Comput*, vol. 12, p. 7569–7580, 2021.
- [6 J. Yul Choi, T. Keun Yoo, J. Gi Seo, K. Jiyong, T. Taewoong Um y T. Hyungtaek Rim, «Multi-categorical deep learning neural network to classify retinal images: A pilot study employing small database,» *PLoS ONE*, vol. 12, nº 11, pp. 1-16, 2017.
- [7 A. Kollias y M. Ulbig, «Diabetic retinopathy: Early diagnosis and effective treatment,» *Deutsches Aerzteblatt Online*, vol. 107, nº 5, pp. 75-84, 2010.

- [8 W. Wang y A. Lo, «Diabetic Retinopathy: Pathophysiology and Treatments,»
] *International Journal of Molecular Sciences*, vol. 19, nº 6, pp. 1-14, 2019.
- [9 J. Torpy, T. Glass y R. Glass, «Retinopathy,» *The Journal of the American*
] *Medical Association*, vol. 298, nº 8, p. 944, 2007.
- [1 S. Pachade, P. Porwal, D. Thulkar, M. Kokare, G. Deshmukh, V.
0] Sahasrabuddhe, L. Giancardo, Q. Gwenole y F. Meriaudeau, «Retinal
Fundus Multi-Disease Image Dataset (RFMiD): A Dataset for Multi-Disease
Detection Research,» *Multidisciplinary Digital Publishing Institute*, vol. 6, nº 2,
p. 14, 2021.
- [1 D. Fong, L. Aiello, T. Gardner, G. King, G. Blankenship, J. Cavallerano, F.
1] Ferris y R. Klein, «Retinopathy in Diabetes,» *Diabetes Care*, vol. 27, pp. 84-
87, 2004.
- [1 N. Ghazi y W. Green, «Pathology and pathogenesis of retinal detachment,»
2] *Eye*, vol. 16, pp. 411-421, 2002.
- [1 R. Jager, W. Mieler y J. Miller, «Age-Related Macular Degeneration,» *The*
3] *New England Journal of Medicine*, vol. 358, pp. 2606-2617, 2008.
- [1 I. Goodfellow, Y. Bengio y A. Courville, *Deep Learning*, MIT Press, 2016.
4]
- [1 G. Shobba y S. Rangaswamy, «Machine Learning,» de *Handbook of*
5] *Statistics*, 2018, pp. 197-228.
- [1 A. Dasgupta y N. Asoke, «Classification of Machine Learning Algorithms,»
6] *International Journal of Innovative Research in Advanced Engineering*, vol. 3,
nº 3, pp. 6-11, 2016.
- [1 M. Joo, R. Venkatesan y N. Wang, «An online universal classifier for binary,
7] multi-class and multi-label classification,» de *IEEE International Conference*
on Systems, Man, and Cybernetics (SMC), Budapest, 2016.
- [1 Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu y M. Lew, «Deep learning for
8] visual understanding: A review,» *Neurocomputing*, vol. 187, pp. 27-48, 2016.

- [1 Y. Lecun, Y. Bengio y G. Hinton, «Deep Learning,» *Nature*, vol. 521, pp. 436-9] 44, 2015.
- [2 J. Feng y Z.-H. Zhou, «AutoEncoder by Forest,» de *The Thirty-Second AAAI Conference on Artificial Intelligence*, Nanjing, 2018.
- [2 K. Gregor y Y. LeCun, «Learning fast approximations of sparse coding,» 1] *ICML'10: Proceedings of the 27th International Conference on International Conference on Machine Learning*, pp. 399-406, 2010.
- [2 B. Yegnanarayana, «Basics of Artificial Neural Networks,» de *Artificial Neural Networks*, New Delhi, Prentice Hall of India, 2006, pp. 15-40.
- [2 A. Kulkarni, D. Chong y F. Batarseh, «Foundations of data imbalance and 3] solutions for a data democracy,» de *Data Democracy*, Academic Press, 2020, pp. 83-106.
- [2 M. Karakaya, «Multi Label Model Evaluation,» Kaggle, 2020. [En línea]. 4] Available: <https://www.kaggle.com/kmkarakaya/multi-label-model-evaluation#Multilabel-confusion-matrix>. [Último acceso: 15 Oct 2021].
- [2 D. Kuhlman, «Introduction -- Python 101 -- Beginning Python,» de *A Python 5] Book: Beginning Python, Advanced Python, and Python Exercises*, MIT, 2012, pp. 1-10.
- [2 P. S. Foundation, «About Python,» Python Software Foundation, 20 Abril 6] 2012. [En línea]. Available: <https://web.archive.org/web/20120420010049/http://www.python.org/about/>. [Último acceso: 22 Oct 2021].
- [2 Anaconda, «Getting started with Anaconda,» Anaconda Documentation, 7] 2021. [En línea]. Available: <https://docs.anaconda.com/anaconda/user-guide/getting-started/>. [Último acceso: 16 Oct 2021].
- [2 Conda, «Conda Documentation,» Conda, 2017. [En línea]. Available: 8] <https://conda.io/en/latest/>. [Último acceso: 16 Oct 2021].
- [2 P. Jupyter, «Jupyter,» Project Jupyter, 2021. [En línea]. Available: 9] <https://jupyter.org/>. [Último acceso: 16 Oct 2021].

- [3 Tensorflow, «Tensorflow,» Tensorflow, 2021. [En línea]. Available:
0] <https://www.tensorflow.org/>. [Último acceso: 15 Oct 2021].
- [3 N. Ketkar, «Introduction to Keras,» de *Deep Learning with Python*, Manning,
1] 2017, pp. 97-111.
- [3 P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer y W.
2] Rudiger, CRISP-DM 1.0, SPSS, 2000.
- [3 S. Mascarenhas y M. Agarwal, «A comparison between VGG16, VGG19 and
3] ResNet50 architecture frameworks for Image Classification,» 2021
*International Conference on Disruptive Technologies for Multi-Disciplinary
Research and Applications (CENTCON)*, vol. 1, pp. 96-99, 2021.
- [3 P. Nahar, S. Tanwani y N. Chaudhari, «Fingerprint Classification Using Deep
4] Neural Network Model ResNet50,» *IJRAR*, vol. 5, nº 04, pp. 1521-1537,
2018.
- [3 A. Naidu, «Step by step VGG16 implementation in Keras,» Github, 26
5] November 2019. [En línea]. Available: <https://github.com/ashushekar/VGG16>.
[Último acceso: 17 November 2022].
- [3 Educative Answers Team, «Educative Answers Team,» Educative Answers ,
6] [En línea]. Available: <https://www.educative.io/answers/definition-model-fitting>. [Último acceso: 14 11 2022].

