INVESTIGACIÓN e INNOVACIÓN en INGENIERÍAS
ISSN2344-8652

Detección de signos de retinopatía hipertensiva usando algoritmos de aprendizaje automático en imágenes digitales de fondo de ojo

Detection of signs of hypertensive reninopathy using machine learning algorithms in digital fundus

Christian Delgado Molina **Jairo Vásquez López**
Leonardo Africany Villamil **Rubiel Vargas Cañas**

Universidad Del Cauca, Colombia

OPEN ACCESS

Recibido: 20/01/2023
Aceptado: 10/04/2023
Publicado: 15/05/2023

Correspondencia de autores:
ccdelgado@unicauca.edu.co

Resumen

Objetivo: Aplicar una arquitectura de detección como estrategia para identificar y estadificar signos de retinopatía hipertensiva en imágenes digitales de fondo de ojo. **Metodología:** El modelo propuesto se fundamenta en la metodología CRISP-ML que es el método más utilizado en proyectos de aprendizaje automático y consta de seis etapas. Estas etapas se siguen de manera exhaustiva, pero se realizan adaptaciones donde sea necesario. **Resultados:** Los resultados obtenidos en este trabajo muestran que el sistema propuesto es capaz de detectar signos de retinopatía hipertensiva en una imagen de fondo de ojo con una precisión del 96%. **Conclusiones:** las técnicas de clasificación de imágenes y aprendizaje automático se presentan como una alternativa para el apoyo de la toma de decisiones necesarias en la detección de signos de esta complicación ocular. Con esto se espera que beneficien tanto los médicos especialistas como los pacientes con retinopatía hipertensiva.

Palabras clave: Retinopatía hipertensiva, aprendizaje automático, detección, precisión, YOLO.

Abstract

Objective: The objective of this work is to apply detection architecture as a strategy to identify and stage signs of hypertensive retinopathy in digital fundus images. **Methodology:** the proposed model is based on the CRISP-ML methodology, which is the most widely used method in machine learning projects and consists of six stages. These stages are followed exhaustively, but adaptation is made where necessary. **Results:** the result obtained in this work show that the proposed system is capable of detecting signs of hypertensive retinopathy in fundus image with an accuracy of 96%. **Conclusions:** image classification and machine learning techniques are presented as an alternative to support the necessary decision-making in detection of signs of this ocular complication. This is expected to benefit both medical specialist and patients with hypertensive retinopathy.

Keywords: hypertensive retinopathy, machine learning, detection, accuracy, YOLO.

Como citar (IEEE): C. Delgado-Molina., L. Africany-Villamil., J. Vásquez-López y R. Vargas-Cañas. “Detección de signos de retinopatía hipertensiva usando algoritmos de aprendizaje automático en imágenes digitales de fondo de ojo". *Investigación e Innovación en Ingenierías*, vol. 11, n°1, 104-124, 2023. DOI: https://doi.org/10.17081/invinno.11.1.6364

Introducción

La enfermedad cardiovascular es la primera causa de muerte a nivel mundial, tan solo en Colombia, los padecimientos cardíacos causaron el 17,1 % de los fallecimientos registrados en hombres y el 18% en mujeres en el 2022 [1]. El principal factor de riego es la hipertensión arterial (HTA) que causa complicaciones que pueden llegar a ser graves cuando no se tratan a tiempo [2]. Según la Organización Mundial de la Salud (OMS) [3], en el mundo hay 1.130 millones de personas con HTA, y la mayoría de ellas, cerca de dos tercios, vive en países de bajos y medianos ingresos. Según el Ministerio de Salud y Protección Social de Colombia (Minsalud) [4], aproximadamente 4 de cada 10 adultos en Colombia sufren de HTA, y el 60% de estos no lo saben aún, además estos factores aumentan con la edad, y en general, los casos son mayores en el sexo femenino que en el masculino [5, 6, 7, 8]. La retinopatía hipertensiva (RH) es un daño en la retina ocasionado por la elevación abrupta de la presión arterial sea por causa primaria o secundaria, con disminución de la agudeza visual que puede avanzar hasta la ceguera. Diferentes estudios han demostrado que la RH se asocia a la subida de las cifras tensionales. Según la Organización Mundial de la Salud (OMS), se estima que aproximadamente 1.300 millones de personas viven con alguna forma de deficiencia visual [9].

El diagnóstico de la RH se da mediante un examen de fondo de ojo que complementa el estudio sistémico de muchos pacientes. En dicho examen, se pueden evidenciar los marcadores de la patología vascular en la circulación coronaria y cerebral tales como: las características circulatorias y el aspecto de los capilares retinianos, y los cambios micro-vasculares en la retina, en consecuencia, puede evaluarse el árbol vascular de la retina, el nervio óptico y la retina en toda su extensión siendo muy útil para predecir el riesgo de las principales enfermedades cardiovasculares [10]. La detección de la RH es una actividad que consume recursos importantes en términos de tiempo y dinero. Desafortunadamente este procedimiento se realiza actualmente de manera manual, donde los especialistas utilizan diferentes equipos o dispositivos que ayudan a visualizar la retina e identificar los signos de esta enfermedad, cruces entre venas, exudados, hemorragias y drusas que en ocasiones no son fáciles de ver en la imagen del fondo de ojo. Adicionalmente, comparar gran cantidad de datos e imágenes médicas y reconocer los patrones necesarios que puedan identificar cada enfermedad, puede inducir en los especialistas agotamiento y errores al comparar gran cantidad de datos y casos [10]. Un obstáculo adicional es que son pocos los médicos capacitados en esta rama, según el MinSalud solo el 4,1% de los médicos son especialistas en oftalmología y comúnmente se encuentran en las grandes ciudades. [11].

Además de los métodos habitualmente utilizados en medicina para realizar el diagnostico existen métodos de diagnóstico asistidos por computador que ofrecen confiabilidad, reproducibilidad y cuantificación de diferentes variables, lo cual facilita el diagnóstico clínico [11]. Por esto, se han reportado diversos estudios en la literatura relacionados con la HR que buscan identificar los diferentes signos que indican presencia de presión arterial en un paciente. Uno de ellos es un sistema automatizado propuesto por Sarmad Khitran [12] para la detección de la HR utilizando la relación arterio-vena (AV). El sistema consiste en un método para la clasificación de vasos, como arterias y venas utilizando un vector de características y un clasificador híbrido, donde se utilizan dos bases de datos de imágenes de fondo de ojo digitales disponibles públicamente, VICAVR y DRIVE. La técnica propuesta por Irshad, S. [13] consiste en clasificar los vasos en arterias y venas, utilizando una máscara de vaso binario y una localización del centro del disco óptico (OD) para extraer la Región de interés (ROI) alrededor del OD logrando una precisión de 81,3%. La investigación realizada por Triwijoyo, B. K [14]. Consiste en un sistema de detección precoz de la enfermedad para ello se procede a utilizar la imagen del fondo de ojo como una entrada de una red neuronal convolucional (CNN) para determinar si hay algún síntoma de HR o no, el conjunto de datos de imágenes utilizado es DRIVE y, se obtuvo una precisión de 98,6%. La investigación que propone Arasy, R. [15] es un sistema para la detección

de la retinopatía hipertensiva mediante el análisis de componentes principales (PCA) y la red neuronal de retro-propagación (BNN), donde las imágenes de la retina se tomaron de la base de datos STARE y mostraron una precisión del 86,36%. Aunque en estos estudios se identifica la relación Arterio-Vena (AV), no son incluidos otros signos como cruces y exudados importantes en el diagnostico dado por el especialista. Tampoco cuantifican los signos ni su localización, lo cual es útil para estadificar la complicación como leve, moderada o severa, y realizar un seguimiento y pronóstico al paciente.

En este artículo se presenta la implementación de algoritmos basados en técnicas de aprendizaje automático que permiten identificar las características y signos presentes en la RH en pacientes hipertensos; que luego implementado en una herramienta computacional puede contribuir a disminuir el tiempo de análisis de un examen de fondo de ojo en un entorno clínico.

Método y procedimientos

El desarrollo de este proyecto se basó en la metodología CRISP-ML y se siguieron las cuatro primeras etapas, con el fin de detectar signos de Retinopatía Hipertensiva. Para ellos se construyó un conjunto de datos de imágenes que permiten identificar los signos presentes en la patología, en la etapa de modelado se consideraron dos fases: entrenamiento y prueba, y finalmente en la fase prueba se evalúa el rendimiento utilizando un conjunto de datos de prueba, es decir, que no se utilizaron para el entrenamiento.

ConFiguración base de datos

Las imágenes para la identificación de los signos se obtuvieron de bases de datos públicas que son muy utilizadas en este tipo de investigaciones y trabajos relacionados con esta patología.

E-ophta. [16] este conjunto de datos se divide en dos categorías: una llamada E-Ophta MA y la otra E-Ophta EX. E-Ophta MA posee 148 imágenes con micro aneurismas y pequeñas hemorragias, y 233 imágenes sin lesiones; en cambio E-Ophta EX posee un conjunto de 47 imágenes de fondo de ojo con una segmentación de exudados y 35 imágenes sin ningún tipo de lesión y que se encuentran etiquetadas como imágenes normales.

Kaggle DR Dataset [17] esta base de datos es proporcionada por EyePACS, (n.d.) para colaborar en la investigación y desarrollo de trabajos relacionados a la detección y diagnóstico de la retinopatía diabética. En total posee 88.702 imágenes, de las cuales 35.126 son para tareas de entrenamiento y 53.576 para pruebas.

ROC (Rerinopathy Online Challenge). [18] contiene 100 imágenes digitales a color de fondo de ojo con micro aneurismas en todas las imágenes. Estas imágenes se encuentran aleatoriamente distribuidas y se dividen en 50 imágenes para entrenamiento y 50 imágenes para pruebas.

Se recolectaron un total de 200 imágenes de fondo de ojo que presentan micro aneurismas y hemorragias y 150 imágenes con exudados y drusas.

Preparación de los datos

Etiquetado

Una vez se hallan seleccionado las imágenes se utiliza una herramienta gratuita de código abierto para etiquetar imágenes gráficamente, la herramienta está escrita en Python y usa QT para su interfaz gráfica y responde al nombre de labelImg [19]. Teniendo en cuenta la importancia del conjunto de datos para poder entrenar el modelo, se hace uso de LabelImg donde las etiquetas se utilizan para ayudar a identificar los componentes de los datos que se desea que el modelo identifique en conjuntos de datos. LabelImg genera

para cada imagen un archivo marcado con formato Yolo que tiene la información en un archivo txt que presenta en su primera columna la clase a la que pertenece el objeto seleccionado, en la segunda y tercera columna se representa el punto central en x y y del rectángulo delimitador y en la cuarta y quinta columna se representa el ancho y alto del rectángulo.

Aumento de Datos [20]

Para este proyecto se utilizó aumento de datos mediante el paquete de código abierto CLODSA [21] que realiza transformaciones a la imagen. Se empleó esta biblioteca porque es flexible para adaptarse a diferentes necesidades, además mejora la precisión de los modelos para la detección. Las transformaciones que se aplicaron sobre las imágenes originales para el signo de hemorragias y micro aneurismas fueron volteos horizontales y verticales aleatorios y rotaciones de 90º, con esto se logra un aumento considerable del número de imágenes para realizar el entrenamiento. Esta misma operación se realizó sobre las imágenes originales con el signo de exudados y drusas.

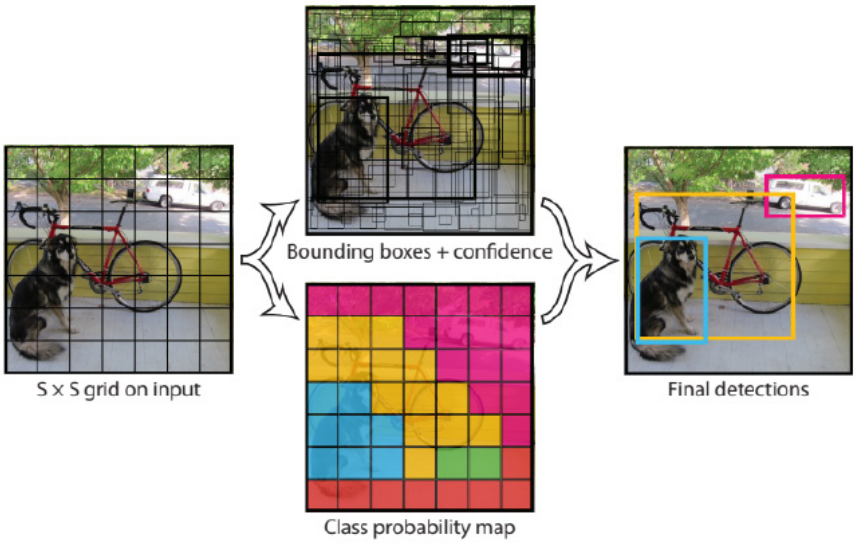
Modelado

El modelo seleccionado para esta investigación es YOLO [22], que es una arquitecturas utilizada para detección de objetos. En este tipo de enfoques se genera primero las potenciales cajas delimitadoras en una imagen y luego se clasifica cada una de estas cajas [23]. Después de la clasificación, en una etapa de post procesamiento se utiliza para refinar las cajas delimitadoras, eliminar las detecciones duplicadas, y volver a clasificar las cajas en base a otros objetos de la imagen. Estos sistemas son lentos y difíciles de optimizar porque cada componente individual debe ser entrenado por separado. YOLO, por el contrario, utiliza una sola red convolucional que predice simultáneamente múltiples cajas delimitadoras y probabilidades de clase para esas cajas (Figura 1). Se entrena en imágenes completas y optimiza directamente el rendimiento de la detección. Este modelo unificado tiene varios beneficios sobre los métodos tradicionales de detección de objetos [23]:

YOLO es extremadamente rápido. Simplemente se ejecuta la red neuronal en una nueva imagen para predecir las detecciones. Además, YOLO alcanza más del doble de la precisión media de otros sistemas en tiempo real.

YOLO trabaja globalmente sobre la imagen. A diferencia de las técnicas de ventana deslizantes y propuestas de regiones, YOLO ve toda la imagen durante el tiempo de entrenamiento y prueba, de manera que implícitamente codifica la información contextual sobre las clases, así como su apariencia.

Figura 1. Ilustración del funcionamiento de YOLO



Fuente: [23]

ConFiguración de archivo y datos del modelo Todos los ajustes de conFiguración que se analizan (Tabla 1) se basan en una GPU Tesla P100 de 16 GB disponible en Google Colab.

Tabla 1. Parámetros usados para el entrenamiento del modelo de red neuronal YOLO
Fuente: [23]

img	Define el tamaño de la imagen de entrada, para el entrenamiento. Debe ser múltiplo de 32
bath	Determina el tamaño del lote de imágenes.
max_batches	Define el número de épocas o iteraciones de entrenamiento. Como mínimo debe cumplir con (# de clases) * 2000.
Steps (pasos)	Se definen a partir del max_batch, es decir, (80% of max_batches), (90% of max_batches).
Filter (filtros)	El número de filtros que se debe aplicar depende del número de clases y se define como: filters = (# de clases + 5) * 3
data	Establece la ruta al archivo data y name donde esta las rutas y clases.
cfg:	Especifica la conFiguración del modelo seleccionado.
Random	Este permite ahorrar memoria en el entrenamiento; si es igual a 1 el variará el tamaño de los lotes durante el entrenamiento, lo que hace que demore el entrenamiento, mientras que si es cero se tendrá un lote de entrada fija y esto reduce memoria y acelera el entrenamiento.

Fuente: Adaptada de [23]

Entrenamiento

En la investigación se trabajó con dos modelos de YOLO: YOLOv3 y YOLOv4, tanto en sus versiones grandes como en las pequeñas (YOLOv3_tiny y YOLOv4_tiny). Con estos modelos se realizaron cuatro experimentos variando su conFiguración. En el archivo de conFiguración Darknet-53 (cfg), se cambió el lote de 64 a 32, se estableció max_batches en 7.000 y los pasos en 5.600 y 6.300. Se entrenó el modelo para 7.000 iteraciones con un tamaño de lote de 32. Las tasas de aprendizaje se programaron para reducirse en los pasos 5.600 y

6.300. Luego se procedió a determinar el número de filtros dependiendo de la cantidad de clases (Figura 2) que en este caso fueron dos ya que cada modelo detecta dos signos.

Figura 2. ConFiguraciones para el primer experimento con la versión pequeña de Yolo (yolov3_tiny) en resolución fija 416x416

```
[net]
# Testing
#batch=1
#subdivisions=1
# Training
batch=32
subdivisions=16
width=416
height=416
channels=3
momentum=0.9
decay=0.0005
angle=0
saturation = 1.5
exposure = 1.5
hue=.1

learning_rate=0.001
burn_in=1000
max_batches = 7000
policy=steps
steps=5600,6300
scales=.1,.1
```

```
[convolutional]
batch_normalize=1
filters=256
size=3
stride=1
pad=1
activation=leaky

[convolutional]
size=1
stride=1
pad=1
filters=21
activation=linear

[yolo]
mask = 1,2,3
anchors = 10,14,  23,27,  37,58,  81,82,  135,169,  344,319
classes=2
num=6
jitter=.3
ignore_thresh = .7
truth_thresh = 1
random=0
```

Fuente: Elaboración propia

Evaluación

La evaluación del modelo se basa en relación a su desempeño teniendo como métricas la matriz de confusión y los indicadores que se derivan de ella: Especificidad, sensibilidad, área bajo la curva ROC y precisión; y obteniendo el valor de la media de los AP calculados para todas las clases (mAp) para seleccionar el modelo de mejor rendimiento (modelo que es más estable y consistente).

Resultados y discusión

Conjunto de datos

Se recolectaron un total de 200 imágenes de fondo de ojo que presentan micro aneurismas y hemorragias; 150 imágenes con exudados y drusas de acuerdo a las bases de datos mencionadas en la sección 1.1. Las imágenes de fondo de ojo se clasificaron de acuerdo a su dimensión y formato, y se agruparon con la idea de que cada modelo pueda detectar dos signos presentes en la retinopatía hipertensiva, para el signo de hemorragias y micro aneurisma (Tabla 2) y para exudados y drusas (Tabla 3).

Tabla 2. Cantidad de imágenes de cada resolución en el conjunto de datos utilizado para hemorragias y miro-aneurismas

Dimensión	Formato	Total de imágenes
1440x960	JPEG	80
2912x2912	JPEG	30
1504x1000	JPEG	20
1500x1152	JPG	30
2544x1696	JPG	40

Fuente: Elaboración propia

Tabla 3. Cantidad de imágenes de cada resolución en el conjunto de datos utilizado para exudados y drusas

Dimensión	Formato	Total de imágenes
2544x1696	JPEG	80
776x582	JPEG	30
1440x960	JPEG	40

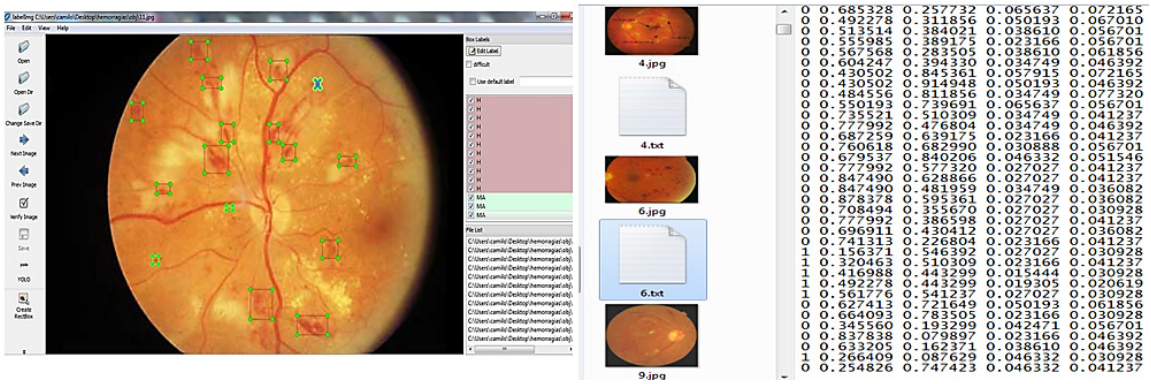
Fuente: Elaboración propia

Adecuación de los datos

Etiquetado

En esta etapa se realizó el respectivo etiquetamiento de cada imagen para obtener el rectángulo delimitador y su respectivo archivo txt en formato YOLO (Figura 3). Este mismo proceso se realizó para el modelo que detecta Exudados y drusas.

Figura 3. Etiquetas de hemorragias y micro aneurismas mediante LabelImag

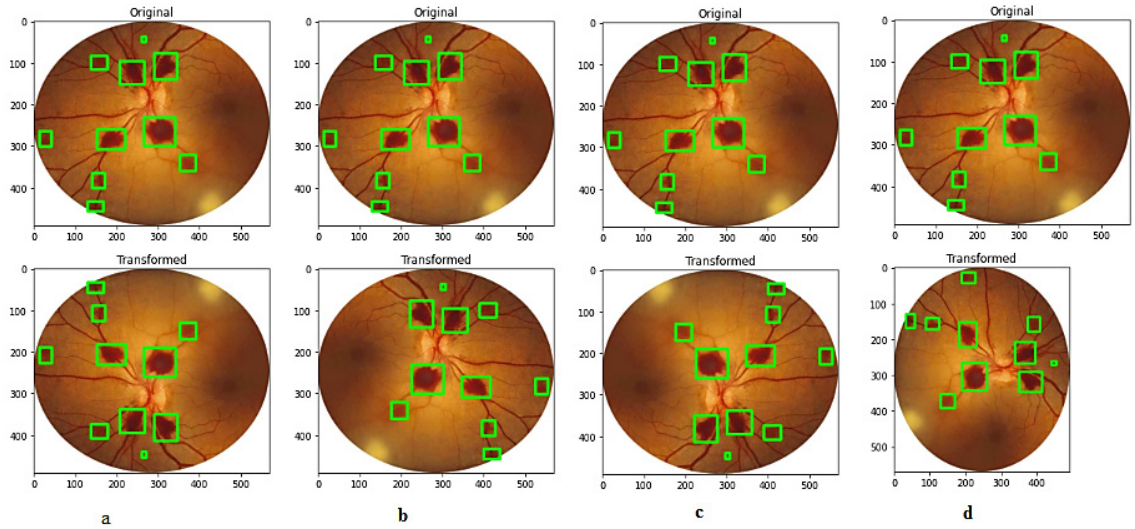


Fuente: Elaboración propia

Aumento de Datos

El método de aumento de datos se empleó para mejorar la precisión de los modelos en la detección de los signos presentes en una imagen de fondo de ojo en la retinopatía hipertensiva. Para ello, se aumentó considerablemente el número de imágenes para el entrenamiento mediante transformaciones de las imágenes originales (Figura 4), con esto se logró un aumento de datos en un total de 708 imágenes para realizar el entrenamiento. Esta misma operación se realizó sobre las 150 imágenes originales con el signo de exudados y drusas, para obtener un total de 800 imágenes para el entrenamiento (Tabla 4).

Figura 4. Aplicación de CLODSA; (a) transformación Volteo vertical; (b) transformación volteo horizontal; (c) Transformación volteo vertical y horizontal; (d) transformación Rotación 90°



Fuente: Elaboración propia

Tabla 4. Cantidad de imágenes con aumento de datos para el entrenamiento

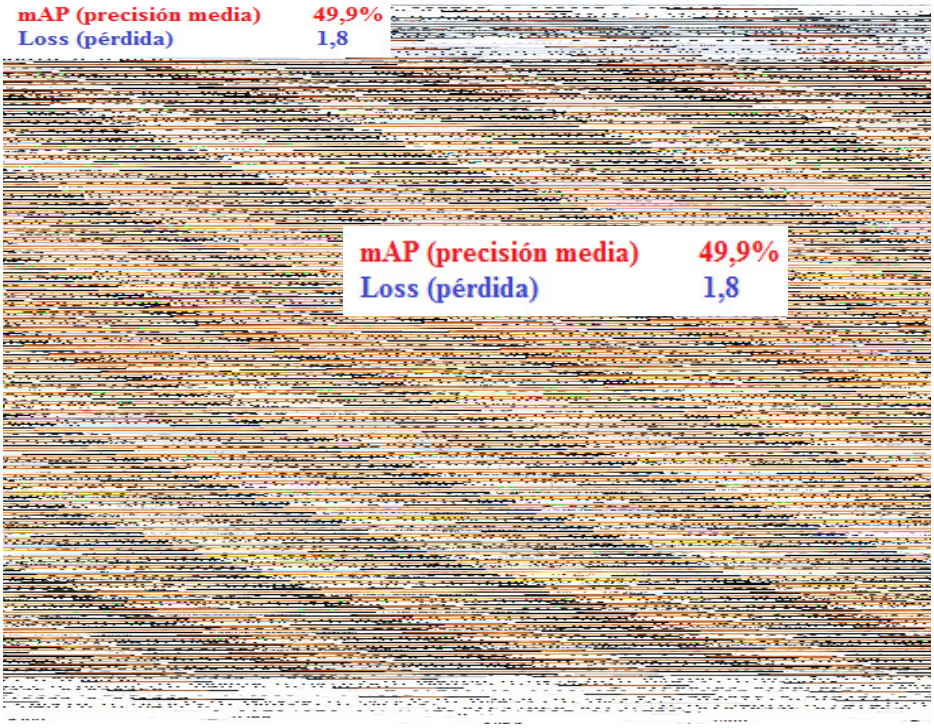
Modelo	Imágenes	Aumento de datos
Signo hemorragias y micro-aneurismas	200	708
Signo exudados y drusas	150	800

Fuente: Elaboración propia

Modelado

Para esta etapa se realizaron cuatro experimentos basados en la arquitectura YOLOv3, YOLOv3_tiny, YOLOv4 y YOLOv4_tiny. En el entrenamiento del experimento número uno se presentó un mAp de 49,9% durante 7.000 iteraciones y una pérdida de 1,8 (Figura 5).

Figura 5. Resultado del mAp durante 7.000 iteraciones para el primer experimento con la versión pequeña de Yolo (yolov3_tiny) en resolución fija 416x416



Fuente: Elaboración propia

Para el experimento dos se tiene una resolución múltiple, para este caso Darknet [23] proporciona un parámetro aleatorio, entonces en los archivos de conFiguración del modelo pequeño de yolov3, el valor predeterminado de random es cero, lo que indica que no se utilizará ninguna resolución aleatoria (o resolución múltiple) durante el entrenamiento. En este experimento se cambia ese valor para que exista la resolución aleatoria. El resto de conFiguraciones y parámetros se mantuvieron constantes (Figura 6).

Figura 6. ConFiguraciones para el segundo experimento con la versión pequeña de Yolo (yolov3_tiny) en multiresolución

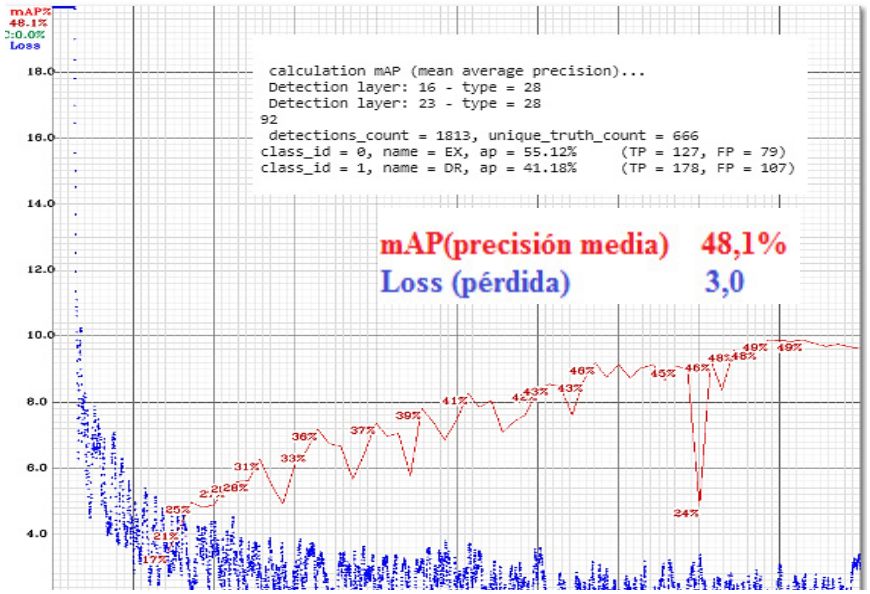
```
[net]
# Testing
#batch=1
#subdivisions=1
# Training
batch=32
subdivisions=16
width=416
height=416
channels=3
momentum=0.9
decay=0.0005
angle=0
saturation = 1.5
exposure = 1.5
hue=.1

learning_rate=0.001
burn_in=1000
max_batches = 7000
policy=steps
steps=5600,6300
scales=.1,.1
```

```
[convolutional]
batch_normalize=1
filters=256
size=3
stride=1
pad=1
activation=leaky

[convolutional]
size=1
stride=1
pad=1
filters=21
activation=linear

[yolo]
mask = 1,2,3
anchors = 10,14,  23,27,  37,58,  81,82,  135,169,  344,319
classes=2
num=6
jitter=.3
ignore_thresh = .7
truth_thresh = 1
random=1
```

Fuente: Elaboración propia

El entrenamiento del experimento dos mostró un desempeño con un mAp de 48,1% durante 7.000 iteraciones, y una pérdida de 3,0, que es mucho mayor que en el experimento uno con el entrenamiento de resolución fija. Esto es de esperar ya que los datos de entrenamiento se vuelven difíciles cuando se entrena en imágenes más pequeñas. Pero al mismo tiempo, el modelo vio muchos escenarios variados, teniendo un mAp cercano al del experimento uno (Figura 7).

Figura 7. Resultado del mAP durante 7.000 iteraciones para el segundo experimento con la versión pequeña de Yolo (yolov3_tiny) en multiresolución



Fuente: Elaboración propia

Para el experimento tres, se mantuvo la conFiguración del experimento dos y se cambió a la versión grande de YOLOV3. El ancho y la altura se conFiguraron en 608, es decir una resolución de 608 × 608, el lote y la subdivisión de 32, el max_batches de 7.000 y los pasos de 5.600 y 6.300. Estos siguen siendo los mismos que en el caso del entrenamiento de los modelos pequeños (Figura 8).

Figura 8. ConFiguraciones para el tercer experimento con la versión grande de Yolov3 en multiresolución

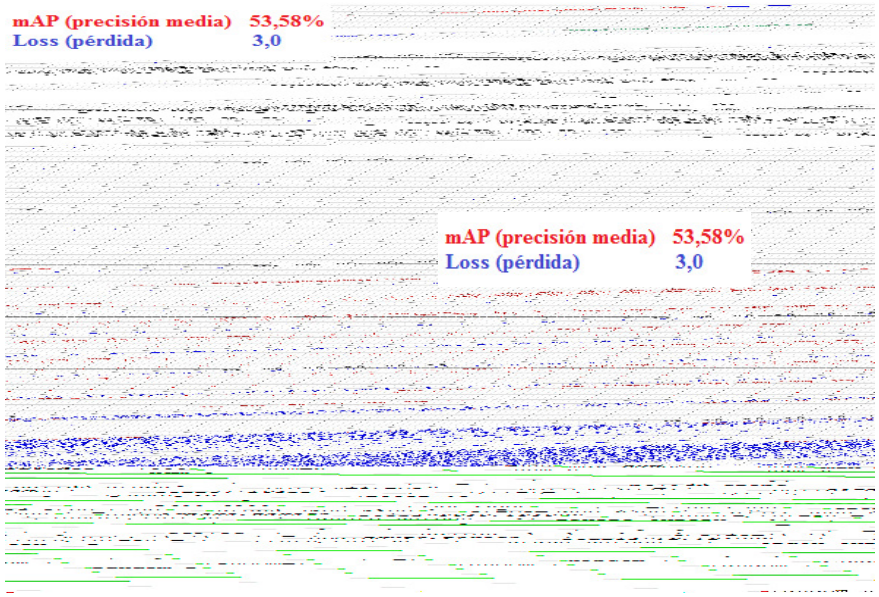
```
# Testing
#batch=1
#subdivisions=1
# Training
batch=32
subdivisions=32
width=608
height=608
channels=3
momentum=0.9
decay=0.0005
angle=0
saturation = 1.5
exposure = 1.5
hue=.1

learning_rate=0.001
burn_in=1000
max_batches = 7000
policy=steps
steps=5600,6300
scales=.1,.1

[convolutional]
batch_normalize=1
filters=16
```

```
[convolutional]
batch_normalize=1
filters=256
size=3
stride=1
pad=1
activation=leaky

[convolutional]
size=1
stride=1
pad=1
filters=21
activation=linear

[yolo]
mask = 1,2,3
anchors = 10,14,  23,27,  37,58,  81,82,  135,169,  344,319
classes=2
num=6
jitter=.3
ignore_thresh = .7
truth_thresh = 1
random=1
```

Fuente: Elaboración propia

En el entrenamiento del experimento tres se observó un mAp de 53,58 % durante 7.000 iteraciones, al final del entrenamiento, con una pérdida de 3,0, similar al del experimento dos, que sigue siendo mayor que en el experimento uno con el entrenamiento de resolución fija. Esto también era de esperar ya que los datos de entrenamiento se vuelven difíciles cuando se entrena en imágenes más pequeñas. Pero al mismo tiempo, el modelo vio muchos escenarios variados, pero teniendo un mAP mejor que los experimentos uno y dos, además de que la versión YOLOv3 grande presenta 53 capas mientras que la versión pequeña YOLOv3_tiny solo utiliza 13 de estas capas, es de esperar que el resultado del mAP sea mucho mejor (Figura 9).

Figura 9. Resultado del mAP durante 7.000 iteraciones para el tercer experimento con la versión grande de Yolov3 en multiresolución



Fuente: Elaboración propia

En el experimento cuatro se cambió a resolución fija (Figura 10) y los demás parámetros se mantuvieron constantes.

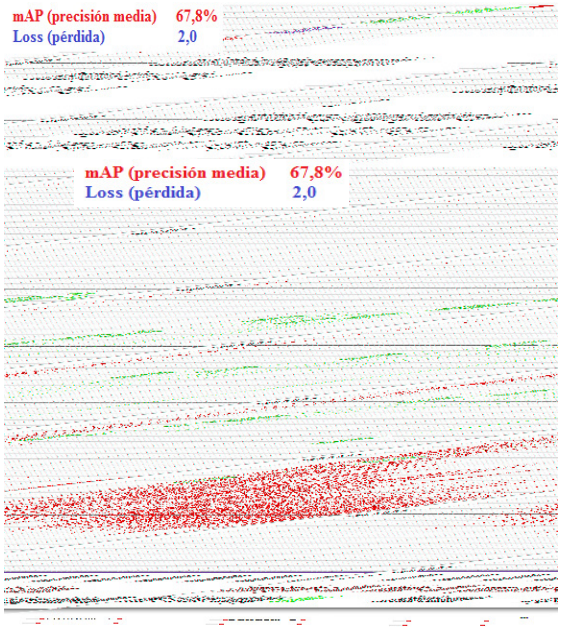
Figura 10. ConFiguraciones para el cuarto experimento con la versión grande de Yolov3 en resolución fija 608x608

```
# Testing
#batch=1
#subdivisions=1
# Training
batch=32
subdivisions=32
width=608
height=608
channels=3
momentum=0.9
decay=0.0005
angle=0
saturation = 1.5
exposure = 1.5
hue=.1

learning_rate=0.001
burn_in=1000
max_batches = 7000
policy=steps
steps=5600,6300
scales=.1,.1

[convolutional]
batch_normalize=1
filters=16

[convolutional]
batch_normalize=1
filters=256
size=3
stride=1
pad=1
activation=leaky

[convolutional]
size=1
stride=1
pad=1
filters=21
activation=linear

[yolo]
mask = 1,2,3
anchors = 10,14,  23,27,  37,58,  81,82,  135,169,  344,319
classes=2
num=6
jitter=.3
ignore_thresh = .7
truth_thresh = 1
random=0
```

Fuente: Elaboración propia

En el entrenamiento del experimento cuatro se obtuvo un mAP de 67,08 % durante 7.000 iteraciones, al final del entrenamiento, la pérdida de 2,0, que es mucho mejor que el experimento dos y tres. El mAP es mucho mejor que los experimentos uno, dos y tres, lo cual permite evidenciar que si se prolongan más iteraciones con esta conFiguración se puede obtener muy buenos resultados (Figura 11).

Figura 11. Resultado del mAP durante 7000 iteraciones para el cuarto experimento con la versión grande de Yolov3 en resolución fija 608x608.



Fuente: Elaboración propia

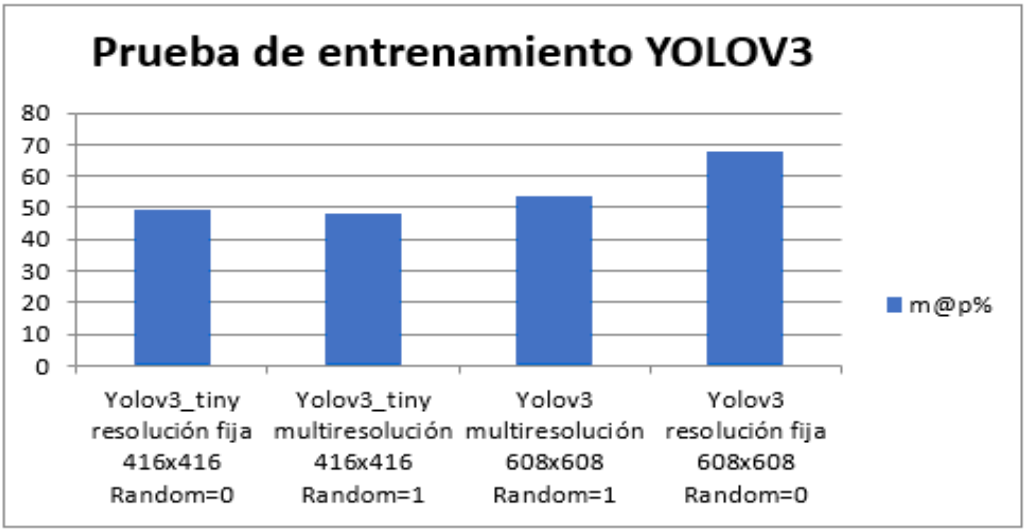
Para comparar los resultados con YOLOv3 se organizan en una Tabla (Tabla 5) y en una Figura (Figura 12).

Tabla 5. Resultados de experimentos con YOLOv3

Versión Yolo	Yolov3_tiny resolución fija 416x416 Random=0	Yolov3_tiny multiresolución 416x416 Random=1	Yolov3 multiresolución 608x608 Random=1	Yolov3 resolución fija 608x608 Random=0
mAP%	49,4	48,1	53,58	68,08

Fuente: Elaboración propia

Figura 12.esultado de los experimentos con YOLOv3



Fuente: Elaboración propia

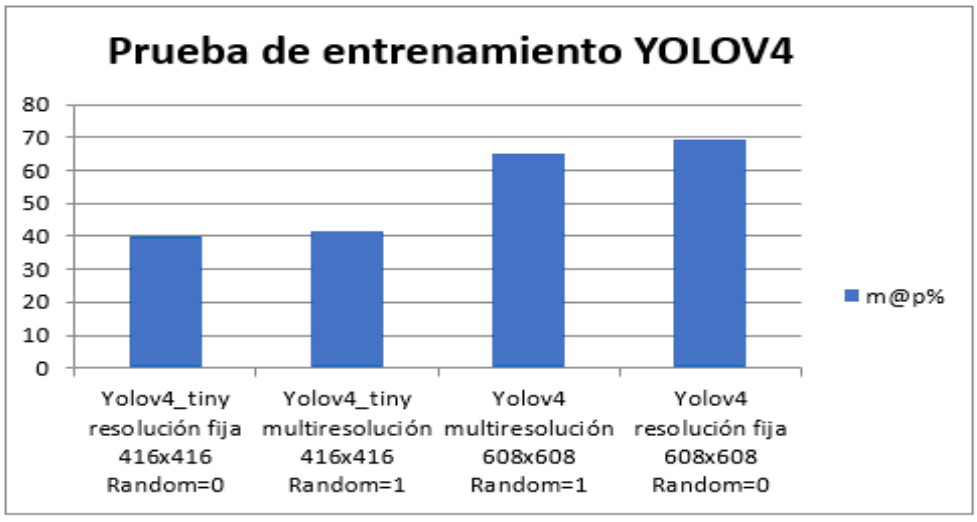
Con la arquitectura YOLOv4 que es la versión actualizada de YOLOv3 se realizaron los mismos experimentos con el fin de observar que tanto mejora el mAp con respecto a YOLOv3 (Tabla 6 y Figura 13).

Tabla 6. Resultados de experimentos con YOLOv4

Versión Yolo	Yolov4_tiny resolución fija 416x416 Random=0	Yolov4_tiny multiresolución 416x416 Random=1	Yolov4 multiresolución 608x608 Random=1	Yolov4 resolución fija 608x608 Random=0
mAp%	40,02	41,50	65,32	69,34

Fuente: Elaboración propia

Figura 13. Resultado de los experimentos con YOLOv4



Fuente: Elaboración propia

Estos experimentos permiten seleccionar el modelo de mejor rendimiento (modelo que es más estable y consistente), YOLOv4 [29], para continuar con las siguientes fases.

Entrenamiento

Una vez elegida la versión de mejor desempeño, se procede a dividir el conjunto de datos en entrenamiento, validación y prueba para los dos escenarios (Tabla 7).

Tabla 7. Distribución de entrenamiento, validación y prueba

Modelo	Entrenamiento	Validación	prueba
Signo hemorragias y micro-aneurismas	602	71	35
Signo exudados y drusas	680	80	40

Fuente: Elaboración propia

Posteriormente se conFigura YOLOv4 de acuerdo con el experimento donde se presentaron los mejores resultados. Las herramientas utilizadas para el entrenamiento y posterior visualización de los resultados se pueden ver en la Tabla 8.

Tabla 8. Herramientas utilizadas para el entrenamiento

GOOGLE COLAB	LABELIMAGE	YOLO	PYTHON
Permite ejecutar y programar en Python sin conFiguración	Para la distribución de las etiquetas de las imágenes.	Es un algoritmo de detección de objetos.	Lenguaje de programación utilizado en el proyecto

Fuente: Elaboración propia

En este punto es importante aclarar que en este proyecto se utilizó la técnica de transfer learning, que es un método de aprendizaje de maquina donde un modelo diseñado para una tarea es reusado como el punto de partida para otra tarea. En este caso particular, se hizo uso de esta técnica para el entrenamiento del modelo partiendo del modelo yolov4.conv.137 el cual es capaz de predecir 80 clases en una imagen o video. Los modelos se entrenaron durante diferentes iteraciones, obteniendo una precisión media del 80% y 88,45% (Tabla 9).

Tabla 9. Resultados del entrenamiento de los modelos en diferentes iteraciones

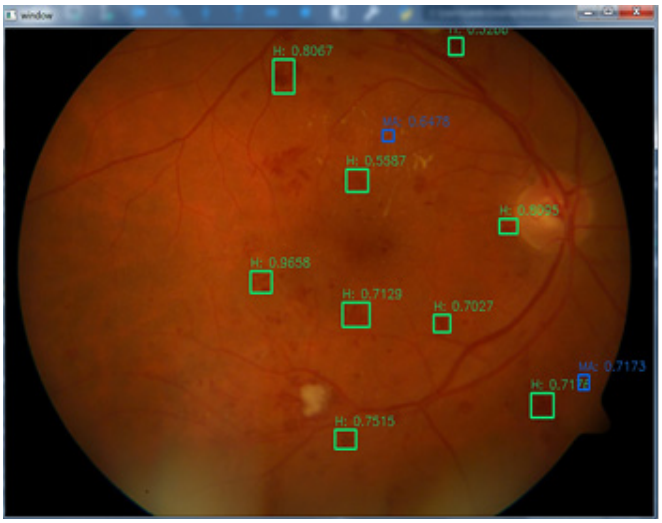
Modelo	Iteraciones	%maP
Hemorragias (H) y micro-aneurismas (MA)	14700	80%
Exudados (EX) y drusas (DR)	20000	88,49%

Fuente: Elaboración propia

Detección con el modelo entrenado

Con el modelo entrenado se realizaron pruebas en un nuevo conjunto de datos manteniendo la conFiguración y los pesos (Figura 14 y 15).

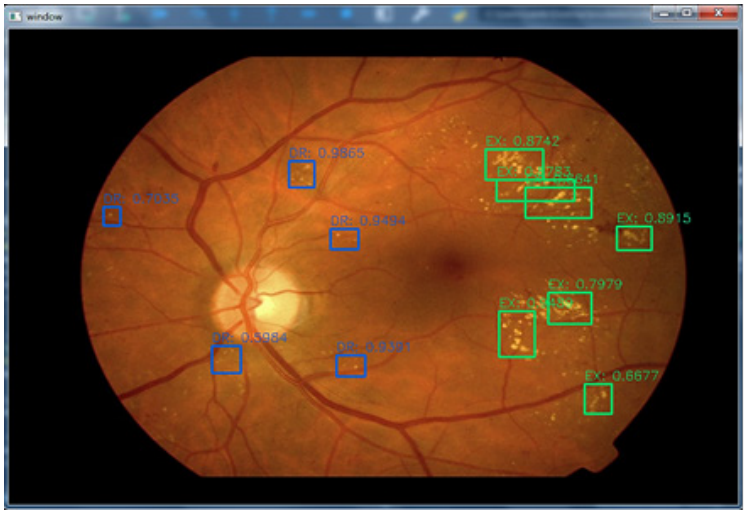
Figura 14. Detección de Hemorragias y Micro aneurismas




```
numero de objetos detectados= {11}
numero de objetos detectados= H: 0.9658
numero de objetos detectados= H: 0.8095
numero de objetos detectados= H: 0.8067
numero de objetos detectados= H: 0.7515
numero de objetos detectados= H: 0.7176
numero de objetos detectados= MA: 0.7173
numero de objetos detectados= H: 0.7129
numero de objetos detectados= H: 0.7027
numero de objetos detectados= MA: 0.6478
numero de objetos detectados= H: 0.5587
numero de objetos detectados= H: 0.5266
```

Fuente: Elaboración propia

Figura 15. Detección de Exudados y Drusas




```
numero de objetos detectados= {12}
numero de objetos detectados= DR: 0.9865
numero de objetos detectados= DR: 0.9494
numero de objetos detectados= EX: 0.9489
numero de objetos detectados= DR: 0.9391
numero de objetos detectados= EX: 0.8915
numero de objetos detectados= EX: 0.8783
numero de objetos detectados= EX: 0.8742
numero de objetos detectados= EX: 0.7979
numero de objetos detectados= DR: 0.7035
numero de objetos detectados= EX: 0.6677
numero de objetos detectados= EX: 0.6641
numero de objetos detectados= DR: 0.5984
```

Fuente: Elaboración propia

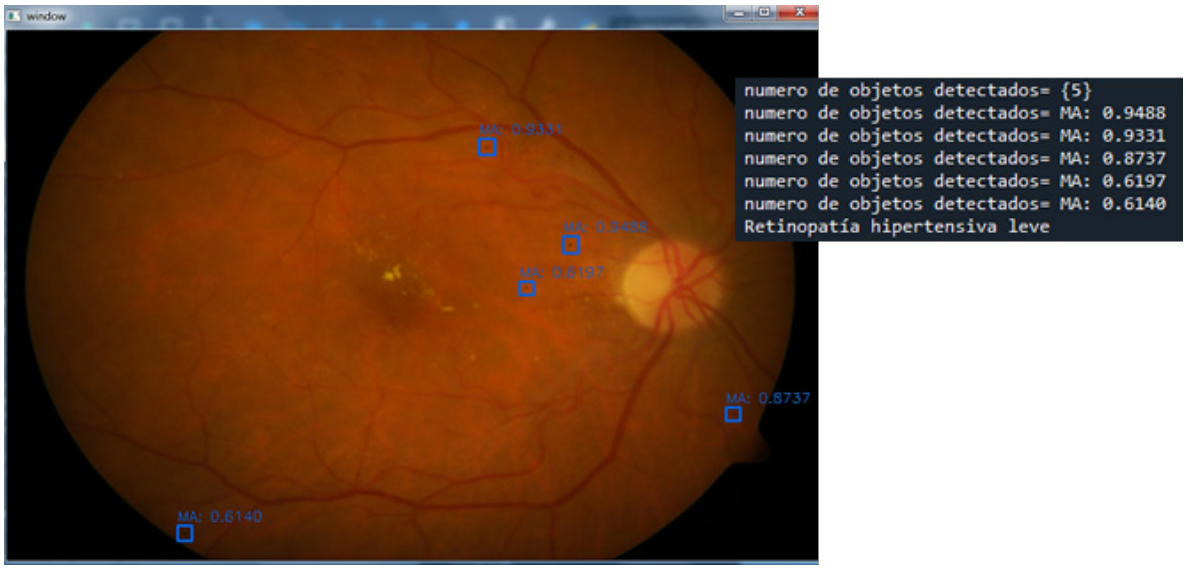
De acuerdo con la cantidad de signos presentes en la imagen de fondo de ojo se asoció una categoría de Retinopatía: leve, moderada o severa, con referencia a la gradación de Sociedad Española de Retina y Vítreo (SERV) y la Sociedad Española de Oftalmología (SEO) (Tabla 10, Figuras 16 y 17).

Tabla 10. Estatificación de retinopatía hipertensiva de acuerdo a cantidad de signos detectados

Equivalencia	Clasificación internacional	Clasificación de la SERV y de la SEO.
Solo MA o solo EX	Retinopatía Leve	Menor e igual a 5
MA , H y EX	Retinopatía Moderada	Menos de 20 y mayor a 5
MA, H, EX y DR	Retinopatía Severa	Mayor e igual de 20

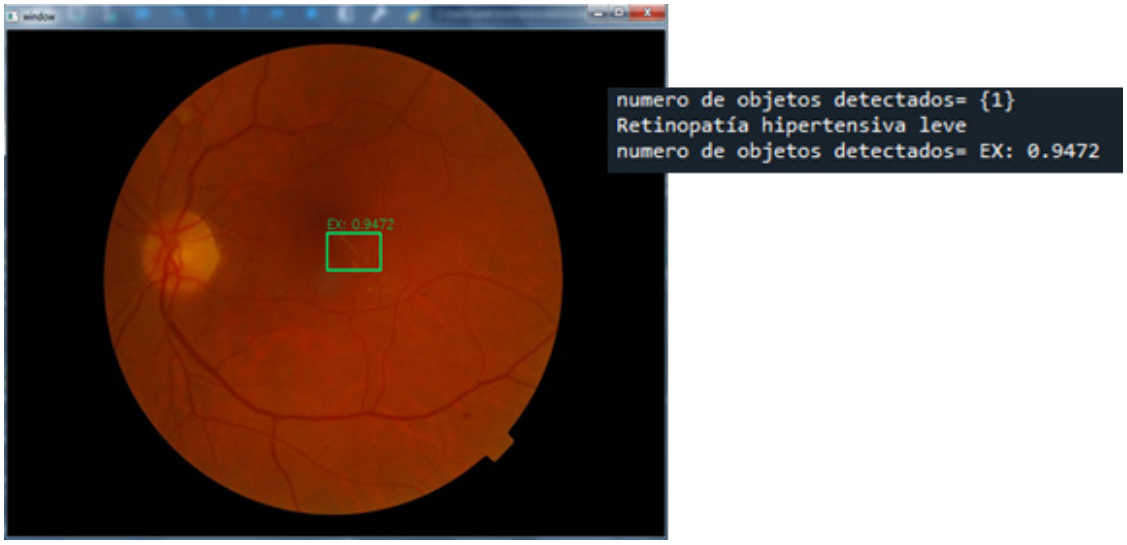
Fuente: Elaboración propia

Figura 16. Estatificación de la retinopatía hipertensiva con el algoritmo propuesto en Python



Fuente: Elaboración propia

Figura 17. Estatificación de la retinopatía hipertensiva con el algoritmo propuesto en Python

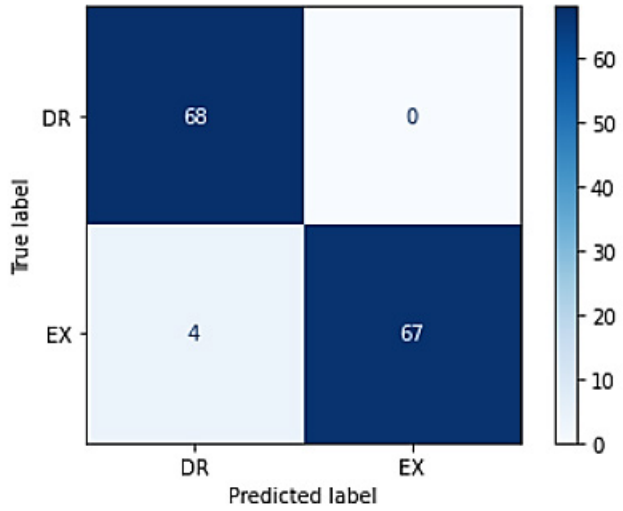


Fuente: Elaboración propia

Evaluación

Al evaluar los modelos mediante las métricas correspondientes a la detección de objetos con un set de datos no visto por el modelo, para el escenario uno, utilizando 525 imágenes, , se calculó la matriz de confusión (Figura 19), las métricas de evaluación derivadas (Tabla 11) y la curva ROC (Figura 20).

Figura 19. Matriz de confusión del modelo 1



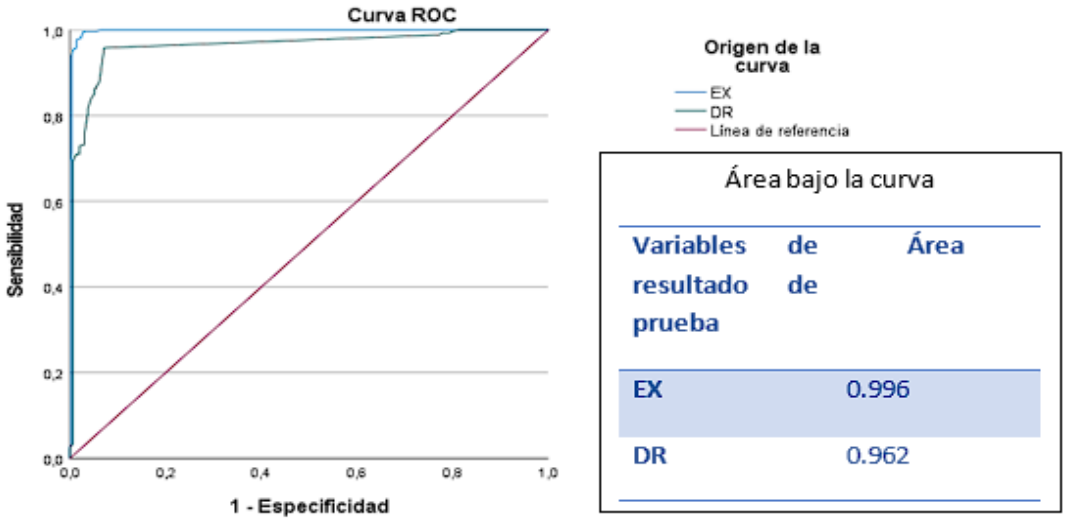
Fuente: Elaboración propia

Tabla 11. Métricas de evaluación del modelo 1

	Precisión	Recall	F1-score
DR	0,94	1,00	0,97
EX	1,00	0,94	0,97

Fuente: Elaboración propia

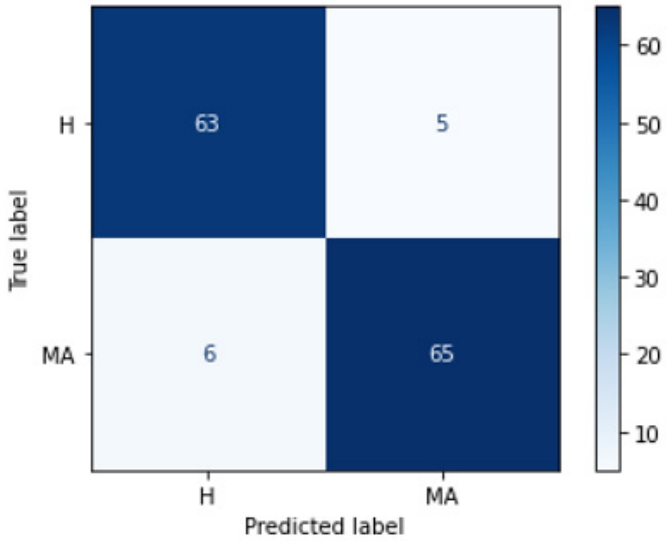
Figura 20. Curva ROC del modelo 1



Fuente: Elaboración propia

Para el escenario dos, detección de Hemorragias y micro-Aneurisma, se tiene la matriz de confusión (Figura 21), las métricas de evaluación (Tabla 12) y la curva ROC (Figura 22).

Figura 21. Matriz de confusión del modelo 2



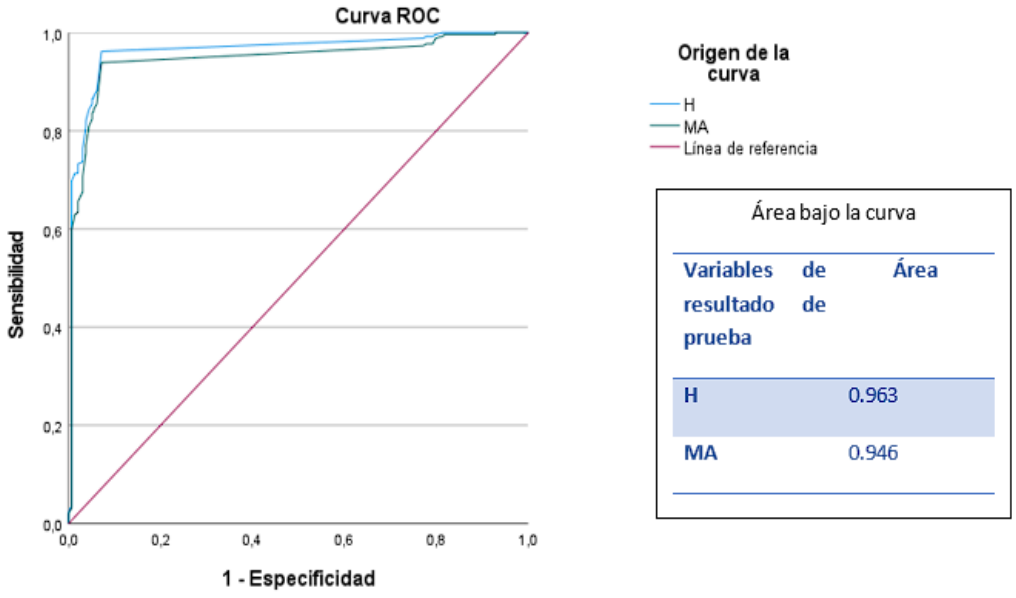
Fuente: Elaboración propia

Tabla 12. Métricas de evaluación del modelo 2

	Precisión	Recall	F1-score
H	0,91	0,93	0,92
MA	0,93	0,92	0,92

Fuente: Elaboración propia

Figura 22. Curva ROC del modelo 2



Fuente: Elaboración propia

Discusión de Resultados

Luego de validar los dos modelos de detección y haber revisado diferentes métricas y evaluado su comportamiento en lo datos de entrenamiento, validación y prueba, es claro que para la arquitectura utilizada YOLOv4 y la conFiguración de Resolución fija que se evidencio en el experimento cuatro se obtuvo mejores resultados comparado con la versión de YOLOv3. Para medir la efectividad de ambos modelos se utilizó la matriz de confusión evidenciándose que: para el modelo uno, el cual realiza detección de drusas y exudados, se tiene que los verdaderos negativos son 67 y los verdaderos positivos 68, es decir, de los 135 casos (diagonal) de un total de 139 imágenes tomadas del set de prueba, predijo correctamente las categorías de cada clase. Por otra parte en el modelo dos, el cual realiza detección de hemorragias y micro-aneurismas se obtuvo que los verdaderos negativos son 65 y los verdaderos positivos 63, es decir, de los 128 casos (diagonal) de un total de 139 tomadas del testeo, predijo correctamente las categorías de cada clase.

Evaluando los resultados de ambos modelos en la matriz de confusión se calcula las métricas de precisión, sensibilidad y F1-score. Para el modelo uno, se tiene un recall alto para drusas de un valor 1.0 y para exudados de 0.94 y una precisión de 0.94 para drusas y 1.0 para exudados, esto significa que el modelo uno detecta bien las clases y un F1-score del 97% que corrobora el buen rendimiento del modelo uno. Por otra parte, el modelo dos cuenta con resultados altos en precisión y recall, indicando que el modelo en la mayoría de las veces maneja bien la detección de las clases y un F1-score del 92% que respalda un buen rendimiento del modelo. Además, al evaluar los resultados del AUC en la curva ROC, se puede observar que el modelo uno presentó un AUC de 0,99 para EX y 0,96 para DR, mientras que el modelo dos presento un AUC de 0,96 para H y 0,94 para MA, obteniendo estos dos modelos los mejores resultados permitiendo así poder detectar de manera eficiente los signos para los cuales fueron entrenados. Esto es sinónimo de robustez de los modelos, al contar con un gran número de imágenes y un modelo previamente pre-entrenado.

Muestra la estructuración de los datos recolectados en función a los objetivos propuestos. Deben ser presentados de manera clara y precisa acorde al método planteado, incluya en este mismo apartado las Tablas y Figuras necesarias para presentar los resultados.

Conclusiones

Los resultados obtenidos en este trabajo muestran un panorama alentador en el propósito de detectar signos de retinopatía hipertensiva, además, el algoritmo de aprendizaje automático implementado muestra resultados óptimos que para el especialista pueden ser usados como soporte en la toma de decisiones frente a la patología ocular mencionada en esta investigación. El algoritmo de aprendizaje automático implementados en conjunto con el método propuesto de procesamiento de las imágenes de fondo de ojo fue capaz de detectar cuatro signos presentes en la retinopatía hipertensiva los cuales son Hemorragias, Micro-Aneurismas, Exudados y drusas con AUC del 0,99 y 0,96. Permitiendo así ser una alternativa en la asistencia del diagnóstico al especialista.

De acuerdo con los resultados de la investigación, el algoritmo de acuerdo al a cantidad de detecciones realizadas es capaz de identificar la patología ocular de manera leve, moderada y severa. La recolección del conjunto de datos es una de las etapas más importantes y se debe hacer énfasis en la correcta, recolección de las imágenes y de su respectiva anotación. Se debe hacer énfasis en el preprocesamiento de las imágenes puesto que es fundamental para obtener buenos resultados en consecuencia las fases de comprensión

de los datos y preparación de los datos llevabas a cabo en este trabajo por medio de la extracción de características fueron exitosas ya que se obtuvieron buenos resultados en la evaluación de la clasificación de la detección de signos de retinopatía hipertensiva [24].

Se implementó dos versiones del algoritmo de detección YOLO; YOLOv3 y YOLOv4 con diferentes conFiguraciones las cuales permitieron elegir la arquitectura YOLOv4 para esta investigación. Donde el enfoque presentó valores elevados en todos los índices: sensibilidad, exactitud, precisión, puntaje F1 y AUC. El trabajo realizado deja una serie de líneas abiertas para seguir investigando:

- Se puede implementar el modelo en un servidor web que permita mediante una plataforma poder interactuar con el usuario, ya que se espera que el método propuesto tenga un gran impacto en el apoyo al diagnóstico al especialista. y, por lo tanto, aumentar la posibilidad de que los pacientes tengan un diagnóstico oportuno en la patología ocular llamada Retinopatía Hipertensiva
- Incluir imágenes de pacientes de un entorno clínico que permita alimentar la base de datos para el posterior entrenamiento del algoritmo de detección.
- Desarrollar los entrenamientos en una GPU de alto rendimiento y que no estén limitados en recursos de uso.
- Realizar el entrenamiento con modelos más recientes de YOLO, como YOLOv5 o YOLOv6, para mejorar el porcentaje de la precisión media mAP en esta investigación.

Referencias bibliográficas

1. EEVV, estadisticas vitales DANE, “difuciones fetales y no fetales” 28/05/2022,online:https://www.dane.gov.co/files/investigaciones/poblacion/bt_estadisticasvitales_defunciones_Itrim_2022pr.pdf.
2. OMS,Organización mundial de la salud. “¿de que morimos?” 18/03/2019, online: online: https://www.who.int/es/news-room/fact-sheets/detail/hypertension.
3. OMS, Organización mundial de la salud : ``hipertensión`` 17 de mayo del 2021: online: https://www.who.int/es/news-room/fact-sheets/detail/hypertension.
4. MinSalud,Ministerio de Salud y protección Social de Colombia: “Conoce tus números” 14 de Junio del 2020: online: https://www.minsalud.gov.co/Paginas/Conoce-tus-numeros-para-prevenir-la-hipertension-arterial.aspx.
5. Daniela Gasca Cuello. P. Manifestaciones de la retinopatía hipertensiva y de la retinopatía diabetica en población adulta. *Scientific & Education Medical Journal / Vol. 1, N° 1, 2021*, 9.
6. Tsukikawa M, W Stacey A. A Review of Hypertensive Retinopathy and Chorioretinopathy; 12: 67-73. 2020, doi: 10.2147/OPTO.S183492.
7. Zhang Y, Zhao L, Li H, Wang Y. Risk factors for hypertensive retinopathy in a Chinese population with hypertension: The Beijing Eye study. Experimental and therapeutic medicine. Experimental and therapeutic medicine. 2020; 17 (1): 435-458, doi: 10.3892/etm.2018.6967.
8. Isabella Torres Revelo, U. A. Interpretabilidad De Un Modelo Basado En Aprendizaje Profundo Para El Diagnóstico De Retinopatía Diabética. Cali. (2020).
9. Visión atlas “numero de personas afectadas porperdida de visión”, 2020, online: https://www.iapb.org/es/learn/vision-atlas/.
10. ANORO, A. T. Retinopatía hipertensiva. *Medicina General 11*,Revisión. 2020; 524-564.

11. Herrera, M. D. Análisis Y Diseño De Un Sistema Para Identificar Signos De Retinopatía Hipertensiva A Través De Imágenes De Retina, Aplicando La Tecnología De Deep Learning. Guayaquil, 2018.
12. Sarmad Khitran, M. U. Automated System for the Detection of Hypertensive. *Image Processing Theory, Tools and Applications*, 6, 2014, DOI: 10.1109/IPTA.2014.7001984.
13. Irshad, S. Classification of Retinal Vessels into Arteries and Veins for detection of Hypertensive Retinopathy. *Cairo International Biomedical Engineering Conference*, 4, 2015.
14. Triwijoyo, B. K. The Classification of Hypertensive Retinopathy using Convolutional Neural Network. *ScienceDirect*, 8, 2017. DOI: 10.1109/CIBEC.2014.7020937.
15. [Arasy, R. Detection of hypertensive retinopathy using principal component analysis (PCA) and backpropagation neural network methods. AIP Conference Proceedings 2092, 9, 2019, https://doi.org/10.1063/1.5096735.
16. E. Decencière et al., 2013, http://dx.doi.org/10.1016/j.irbm.2013.01.010, El enlace para acceder a este dataset es: http://www.adcis.net/en/third-party/e-ophtha/,
17. Cuello Navarro, J., Barraza Peña, C. & Escorcia-Gutierrez, J. (2020). Una revisión de los métodos de deep learning aplicados a la detección automatizada de la retinopatía diabética. Revista Sextante, 23, pp. 12 - 27, 2020, El enlace para acceder a este dataset es: https://www.kaggle.com/c/diabetic-retinopathy-detection/data
18. Niemeijer et al., 2010 El enlace para acceder a esta base de datos de imágenes es http://webeye.ophth.uiowa.edu/ROC/.
19. Mauricio Menegaz. “Understanding YOLO.” Hackernoon. https://hackernoon.com/understanding-YOLO-f5a74bbc7967 . Fecha de ultima consulta: Septiembre 2022.
20. Data augmentation-assisted Deep learning of hand-drawn partially colored sketches for visual, search: https://www.researchdate.net/publication/319413978_Data_augmentation-assisted_Deep_learning_of_handfrawn_artially_colored_sktches_for_visual_search,DOI: https://doi.org/10.1371/journal.pone.0183838.
21. Asado-García, Á., Domínguez, C., García-Domínguez, M. et al. CLoDSA: a tool for augmentation in classification, localization, detection, semantic segmentation and instance segmentation tasks. BMC Bioinformatics 20, 323, 2019.
22. Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Harhali. “You Only Look Once: Unified, Real-Time Object Detection.”, 2016. https://arxiv.org/abs/1506.02640.
23. Jonathan Hui. “Real-time Object Detection with YOLO, YOLOv2, and now YOLOv3.” Fecha de última consulta: Septiembre 2022.
24. C. A. Ruíz Ramírez, D. M. Montoya Quintero, y J. A. Jimenez Builes, "Un Ambiente visual integrado de desarrollo para el aprendizaje de programación en robótica", Investigación e Innovación en Ingenierías, vol. 9, n.º 1, pp. 7–21, 2021. DOI: https://doi.org/10.17081/invinno.9.1.3957