UNIVERSIDAD NACIONAL DEL CALLAO

ESCUELA DE POSGRADO

UNIDAD DE POSGRADO DE LA FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

"SISTEMA DE ALERTA Y VIDEOVIGILANCIA UTILIZANDO RASPBERRY PI 4B PARA LA DETECCIÓN DE ARMAS EN LA REGIÓN CALLAO, 2023"

TESIS PARA OPTAR EL GRADO ACADÉMICO DE MAESTRO EN CIENCIAS DE LA ELECTRÓNICA CON MENCIÓN EN CONTROL Y AUTOMATIZACIÓN

AUTOR: DENNIS HUAMAN YRIGOIN

ASESOR: MG. HERBERT JUNIOR GRADOS ESPINOZA

CO-ASESOR: DR. ANTENOR LEVA APAZA

LÍNEA DE INVESTIGACIÓN: INGENIERÍA Y TECNOLOGÍA

Callao, 2024

PERÚ

INFORME FINAL TESIS DE MAESTRIA - HUAMAN YRIGOIN DENNIS

7% Textos sospechosos

3% Similitudes
1% similitudes entre comillas
< 1% entre las fuentes mencionadas

5% Idiomas no reconocidos

Nombre del documento: INFORME FINAL TESIS DE MAESTRIA - HUAMAN YRIGOIN DENNIS.pdf
ID del documento: b3a3d9fecf5a35345221106e91d2fdf2a0ddfcb3
Tamaño del documento original: 5,01 MB

Depositante: FIEE PREGRADO UNIDAD DE INVESTIGACION
Fecha de depósito: 26/6/2024
Tipo de carga: interface
fecha de fin de análisis: 26/6/2024

Número de palabras: 19.904
Número de caracteres: 136.997

Ubicación de las similitudes en el documento:

Fuentes de similitudes

Fuentes principales detectadas

Nº	Descripciones	Similitudes	Ubicaciones	Datos adicionales
1	**dx.doi.org** | SISTEM PENDETEKSI PELANGGAR JARAK SOSIAL COVID-19 BERBASIS VI... http://dx.doi.org/10.35760/tr.2022.v27i2.7100 **2 fuentes similares**	< 1%		Palabras idénticas: < 1% (66 palabras)
2	**repositorio.ucv.edu.pe** https://repositorio.ucv.edu.pe/bitstream/handle/20.500.12692/121870/Blas_CJL-Jimenez_GBS-SD.pd... **3 fuentes similares**	< 1%		Palabras idénticas: < 1% (55 palabras)
3	**bibliotecadigital.udea.edu.co** https://bibliotecadigital.udea.edu.co/bitstream/10495/28886/1/MoralesOscar_2022_AproximacionAl... **1 fuente similar**	< 1%		Palabras idénticas: < 1% (41 palabras)
4	**arxiv.org** https://arxiv.org/pdf/2203.04129.pdf **2 fuentes similares**	< 1%		Palabras idénticas: < 1% (49 palabras)
5	**repositorio.unal.edu.co** http://repositorio.unal.edu.co/bitstream/handle/unal/83849/71756752.2023.pdf?sequence=2 **1 fuente similar**	< 1%		Palabras idénticas: < 1% (44 palabras)

Fuentes con similitudes fortuitas

Nº	Descripciones	Similitudes	Ubicaciones	Datos adicionales
1	**dx.doi.org** | Sistema de visión por computadora para la identificación de palma a... http://dx.doi.org/10.26507/paper.2270	< 1%		Palabras idénticas: < 1% (37 palabras)
2	**link.springer.com** | Design of an intelligent video surveillance system for crime pr... https://link.springer.com/content/pdf/10.1007/s11042-021-10809-z.pdf	< 1%		Palabras idénticas: < 1% (32 palabras)
3	**repositorio.uta.edu.ec** https://repositorio.uta.edu.ec:8443/jspui/bitstream/123456789/41254/1/t2563te.pdf	< 1%		Palabras idénticas: < 1% (39 palabras)
4	**scielo.sld.cu** | Revisión de algoritmos de detección y seguimiento de objetos con re... http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S2227-18992020000300165	< 1%		Palabras idénticas: < 1% (32 palabras)
5	**repositorio.udec.cl** http://repositorio.udec.cl/bitstream/11594/11844/1/caro_p_v_2024_MAG.pdf	< 1%		Palabras idénticas: < 1% (32 palabras)

Fuentes mencionadas (sin similitudes detectadas)

Estas fuentes han sido citadas en el documento sin encontrar similitudes.

1 http://hdl.handle.net/10654/14256
2 http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S2227
3 https://rpp.pe/peru/actualidad/coronavirus-en-el-peru-inseguridad-y
4 https://m.inei.gob.pe/media/MenuRecursivo/boletines/estadisticas_de_se
5 https://cdn.www.gob.pe/uploads/document/file/4041993/Estimaciones%2

ACTA N° 013 - 2024

ACTA DE SUSTENTACIÓN DE TESIS PARA OBTENER EL GRADO ACADÉMICO DE MAESTRO EN CIENCIAS DE LA ELECTRÓNICA CON MENCIÓN EN CONTROL Y AUTOMATIZACIÓN.

Siendo las 14:00 horas del día 13 de septiembre del año 2024, se reunieron en el Aula de Sustentación de la Unidad de Posgrado de la Facultad de Ingeniería Eléctrica y Electrónica, el Jurado de Sustentación del Trabajo de Tesis para la obtención el grado académico de **MAESTRO EN CIENCIAS DE LA ELECTRÓNICA CON MENCIÓN EN CONTROL Y AUTOMATIZACIÓN** conformado por:

Dr. ADAN ALMIRCAR TEJADA CABANILLAS	:	**PRESIDENTE**
Dr. FERNANDO MENDOZA APAZA	:	**SECRETARIO**
MSc. CARLOS HUMBERTO ALFARO RODRIGUEZ	:	**MIEMBRO**
MSc. GABRIEL AUGUSTO TIRADO MENDOZA	:	**MIEMBRO**

Con la finalidad de evaluar la sustentación del trabajo de tesis titulada **"SISTEMA DE ALERTA Y VIDEOVIGILANCIA UTILIZANDO RASPBERRY PI 4B PARA LA DETECCIÓN DE ARMAS EN LA REGIÓN CALLAO, 2023"** presentado por:

Don (ña):

HUAMAN YRIGOIN DENNIS

Acto seguido se procedió a la Sustentación del Trabajo de Tesis, con el fin de optar el grado académico de **MAESTRO EN CIENCIAS DE LA ELECTRÓNICA CON MENCIÓN EN CONTROL Y AUTOMATIZACIÓN**. Luego de la exposición, los miembros del Jurado evaluador formularon las respectivas preguntas, las mismas que fueron absueltas.

Terminada la sustentación, el Jurado Evaluador luego de deliberar, acuerda: APROBAR, con la escala de calificación cualitativa Muy Bueno, y calificación cuantitativa Dieciocho (18) el presente **TRABAJO DE TESIS**, conforme a lo dispuesto en el Art. 124° del Reglamento de Estudios de Grados y Títulos de la UNAC, aprobado por Resolución de Consejo Universitario Nº 099-2021-CU del 30 de junio de 2021. Se eleva la presente acta a la Escuela de Posgrado de la Universidad Nacional del Callao, a fin de que se declare APTO para conferir el grado académico de **MAESTRO EN CIENCIAS DE LA ELECTRÓNICA CON MENCIÓN EN CONTROL Y AUTOMATIZACIÓN.**

Se extiende el Acta, a las 14:45 hrs del mismo día.

Bellavista, 13 de septiembre del 2024

Dr. ADAN ALMIRCAR TEJADA CABANILLAS
PRESIDENTE

Dr. FERNANDO MENDOZA APAZA
SECRETARIO

MSc. CARLOS HUMBERTO ALFARO RODRIGUEZ
MIEMBRO

Msc. GABRIEL AUGUSTO TIRADO MENDOZA
MIEMBRO

INFORMACIÓN BÁSICA

FACULTAD: FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

UNIDAD DE INVESTIGACIÓN: UNIDAD DE INVESTIGACIÓN DE LA FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

TÍTULO: "SISTEMA DE ALERTA Y VIDEOVIGILANCIA UTILIZANDO RASPBERRY PI 4B PARA LA DETECCIÓN DE ARMAS EN LA REGIÓN CALLAO, 2023"

AUTOR: DENNIS HUAMAN YRIGOIN

CÓDIGO ORCID: 0000-0001-7907-9521

DNI: 72138492

ASESOR: MG. HERBERT JUNIOR GRADOS ESPINOZA

CÓDIGO ORCID: 0000-0002-5504-0734

DNI: 46168554

CO-ASESOR: DR. ANTENOR LEVA APAZA

CÓDIGO ORCID: 0000-0002-0973-0240

DNI: 25003844

LUGAR DE EJECUCIÓN: CALLAO, PERÚ

UNIDAD DE ANÁLISIS: MODELOS DE DETECCIÓN DE ARMAS

TIPO: TECNOLÓGICA - APLICADA

ENFOQUE: CUANTITATIVO

DISEÑO DE INVESTIGACIÓN: PRE-EXPERIMENTAL

TEMA OCDE: 2.02.03 -- SISTEMAS DE AUTOMATIZACIÓN, SISTEMAS DE CONTROL

HOJA DE REFERENCIA DEL JURADO Y APROBACIÓN

JURADO EXAMINADOR

1. **DR. ADAN ALMIRCAR TEJADA CABANILLAS**	:	**PRESIDENTE**
2. **DR. FERNANDO MENDOZA APAZA**	:	**SECRETARIO**
3. **MSc. CARLOS HUMBERTO ALFARO RODRIGUEZ**	:	**MIEMBRO**
4. **MSc. GABRIEL AUGUSTO TIRADO MENDOZA**	:	**MIEMBRO**

ASESOR: MG. HERBERT JUNIOR GRADOS ESPINOZA
COASESOR: DR. ANTENOR LEVA APAZA

N° LIBRO: 01

N° FOLIO: 149

N° ACTA: 013-2024

FECHA DE APROBACIÓN: 13 de septiembre de 2024

RESOLUCIÓN DIRECTORAL N° 57-2024-DUPFIEE

DEDICATORIA

A DIOS, por la sabiduría y fortaleza que me brinda día a día para afrontar las situaciones adversas de la vida.

A mi familia, que son lo más valioso que Dios me ha dado.

A mis padres Emelina y Pedro, quienes me guían sabiamente con sus consejos por el camino de la vida, ustedes son un pilar fundamental en mi formación personal y profesional. Eterno agradecimiento por su amor, paciencia y sacrificio, que han sido mi fuente de inspiración para superarme cada día más.

A mis hermanos Claudia y Franz, por haberme brindado su apoyo y motivación para seguir adelante, los quiero mucho.

AGRADECIMIENTO

A Jasmin, por su soporte y compañía en los momentos buenos y difíciles.

A mis colegas Carlos y Paul, quienes me apoyaron compartiendo sus conocimientos para el desarrollo de esta tesis.

A mis demás amigos, docentes y compañeros que aportaron con sus sugerencias y conocimientos para la culminación de esta investigación.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

ÍNDICE DE ABREVIATURAS

- Faster-RCNN: Faster Region-Convolutional Neural Network
- RCFN: Region-based Fully Convolutional Networks
- YOLO: You Only Look Once
- SSD: Single-shot detector
- INEI: Instituto Nacional de Estadística e Informática
- IMFDB: Internet Movie Firearms Database
- R-CNN: Region-based Convolutional Neural Network
- GPU: Graphic Processing Unit
- SMS: Short Message Service
- GSM: Global System for Mobile Communications
- mAP: mean Average Precision
- SMTP: Simple Mail Transfer Protocol
- POP3: Post Office Protocol 3
- IMAP: Internet Message Access Protocol
- CSI: Camera Serial Interface
- FPS: Frames per second
- API: Interfaz de programación de aplicaciones

RESUMEN

En el Perú, los delitos contra el patrimonio se han incrementado considerablemente en los últimos diez años, estos generalmente realizados con armas de fuego o punzocortantes, poniendo en riesgo la integridad de la persona afectada. Pese a los esfuerzos que realizan los gobiernos locales por mejorar la seguridad ciudadana instalando cámaras de vigilancia o mediante patrullajes policiales o de serenazgo continuos, éstos últimos también se pueden ver afectados por el desconocimiento de los tipos de armas que puedan portar los criminales que cometen delitos como robo, hurto, sicariato, etc., al momento de confrontarlos.

Ante ello, se propuso el desarrollo e implementación de un sistema de alerta y videovigilancia que pueda detectar diferentes tipos de armas las cuales fueron categorizadas en AFCA (Armas de Fuego de Corto Alcance), AFLA (Armas de Fuego de Largo Alcance) y APC (Armas punzocortantes).

En primera instancia se evaluaron los diferentes tipos de cámaras a elegir compatibles con Raspberry Pi 4B, así como el tipo de notificación a implementar. Para la detección de armas, se utilizó como base el modelo de detección YOLOv8 y se establecieron 3 modificaciones, las cuáles se denominaron (YOLOv8n-I y YOLOv8n-II, basados en el modelo YOLOv8n) y (YOLOv8s-I basado en el modelo YOLOv8s). Para cada modelo se especificaron parámetros de entrenamiento como la cantidad de épocas, cantidad de imágenes de entrenamiento y validación, tamaño del lote (batch) y tamaño de imagen de entrenamiento. Para las métricas de desempeño del sistema, se consideraron la Precision, FPS, Recall, F1-Score, Matriz de Confusión y el mAP.

Se obtuvo que, el modelo YOLOv8n-II tuvo el mejor rendimiento respecto a los modelos YOLOv8n-I y YOLOv8s-I. El modelo se escogió en base a la tasa promedio de fotogramas por segundo (FPS) de 3.5, una precisión de 83.3%, un Recall de 89.5% y un F1-Score de 88.9% para las 3 categorías. Adicionalmente se tuvo un tiempo promedio de 2 segundos para el envío de las notificaciones conteniendo las imágenes de las armas detectadas al aplicativo de Telegram.

El sistema se puso a prueba en un entorno real y a una distancia de 1.5 a 7 metros, logrando una detección de las armas correspondientes a las categorías indicadas. Este sistema puede utilizarse para ser implementado en entornos urbanos o residenciales, pudiendo notificar a las autoridades competentes sobre los tipos de armas que los delincuentes puedan portar.

ABSTRACT

In Peru, crimes against property have increased considerably in the last ten years, generally committed with firearms or sharp-edged weapons, putting the integrity of the affected person at risk. Despite the efforts made by local governments to improve citizen security by installing surveillance cameras or through continuous police patrols or *serenazgo*, the latter can also be affected by the lack of knowledge of the types of weapons that criminals who commit crimes such as robbery, theft, contract killings, etc. may carry when confronted.

Therefore, the development and implementation of an alert and video surveillance system that can detect different types of weapons was proposed, which were categorized into AFCA (short-range firearms), AFLA (long-range firearms) and APC (sharp weapons), by its acronym in Spanish.

In the first instance, the different types of cameras to choose compatible with Raspberry Pi 4B were evaluated, as well as the type of notification to implement. For the detection of weapons, the YOLOv8 detection model was used as a base and 3 modifications were established, which were named (YOLOv8n-I and YOLOv8n-II, based on the YOLOv8n model) and (YOLOv8s-I based on the YOLOv8s model). For each model, training parameters such as number of epochs, number of training and validation images, batch size, and training image size were specified. For the system performance metrics, Precision, FPS, Recall, F1-Score, Confusion Matrix and mAP were considered.

It was obtained that, the YOLOv8n-II model had the best performance with respect to the YOLOv8n-I and YOLOv8s-I models. The model was chosen based on an average frames per second (FPS) rate of 3.5, an accuracy of 83.3%, a Recall of 89.5% and an F1-Score of 88.9% for the 3 categories. Additionally, an average time of 2 seconds was used to send the notifications containing the images of the detected weapons to the Telegram application.

The system was tested in a real environment and at a distance of 1.5 to 7 meters, achieving detection of weapons corresponding to the indicated categories. This system can be used to be implemented in urban or residential environments,

being able to notify the competent authorities about the types of weapons that criminals may carry.

INTRODUCCIÓN

Los sistemas de videovigilancia son ampliamente utilizados a nivel mundial por personas y organizaciones [1], esto es gracias al desarrollo tecnológico gradual [2], y más aún por la creciente necesidad e interés de mejorar la seguridad, sea en el sector público y/o privado, pues estos sistemas permiten controlar y obtener un registro visual e histórico de cualquier actividad en una determinada área [3]. Adicionalmente, junto con estos sistemas de videovigilancia, se emplean tecnologías de recopilación de datos [4] (imágenes, vídeo, localización, etc.) para su procesamiento y tratamiento posterior en tiempo real utilizando Big Data o algoritmos de inteligencia artificial [5]. Estos algoritmos como por ejemplo Faster-RCNN, RFCN, YOLO y SSD se han ido integrando al análisis de imágenes para la detección o rastreo de objetos, y en la resolución de problemas de visión por computadora [6], que han permitido reforzar o mejorar los sistemas de videovigilancia existentes. Siendo estos últimos (YOLO y SSD) ampliamente utilizados en dispositivos de recursos computacionales y de memoria bajos [6], [7], [8].

Ante ello, la presente investigación tiene como objetivo el desarrollo de un dispositivo que emplee este tipo de algoritmos utilizando una Raspberry Pi 4B, que permita la detección de armas y el envío de alertas en para la mejora de la seguridad ciudadana. Primero, se recopilarán imágenes de diferentes bases de datos de internet para el entrenamiento de la red neuronal. Luego se evaluarán entre diferentes modelos de detección de objetos que permita tener un tiempo de respuesta menor y alta tasa de precisión para la detección. Posteriormente, se evaluará el tipo de alerta y la aplicación de mensajería a emplear (SMS, WhatsApp, Telegram) teniendo como parámetro de medición el tiempo de envío. Por último, se integrará el sistema en una Raspberry Pi 4B y se recopilarán parámetros de desempeño del algoritmo utilizado (Precision, F1-score, Recall, mAP50, etc.), el tiempo de envío de la alerta, así como la cantidad de fotogramas por segundo (FPS) que puede capturar el sistema.

Esta investigación se organizará en ocho capítulos. El primer capítulo, del planteamiento del problema, especifica la necesidad de implementar este tipo de

dispositivos o sistemas para la mejora de la seguridad ciudadana o residencial, tomando como referencia un mapeo científico mediante un simplificado análisis bibliométrico. El segundo capítulo, del marco teórico, presenta un panorama general de investigaciones nacionales e internacionales referentes al procesamiento de imágenes para la detección de armas y explica los fundamentos teóricos utilizados para esta investigación. El tercer capítulo, de las hipótesis, se establecen las suposiciones y aseveraciones a comprobar para la viabilidad del sistema, así como la operacionalización de las variables con las definiciones conceptuales de las variables planteadas del proyecto. El cuarto capítulo, de la metodología, muestra los pasos realizados para la obtención de las imágenes, los modelos modificados en base a YOLOv8 para la detección de armas, también la implementación de la notificación del evento acontecido una vez se detecten armas en fotogramas de video, y la integración de estos en el sistema propuesto. El quinto capítulo, de los resultados, muestra mediante gráficas estadísticas y figuras, las detecciones realizadas una vez el sistema se ha implementado, también muestra información sobre las métricas de desempeño del sistema desarrollado. El sexto capítulo, de la discusión de resultados, contrasta las métricas del sistema implementado con otros trabajos mencionados en los antecedentes. Por último, en el capítulo sétimo y octavo se presenta las conclusiones y recomendaciones de la investigación respectivamente.

I. PLANTEAMIENTO DEL PROBLEMA

1.1. Descripción de la realidad problemática

La inseguridad ciudadana es un problema a nivel nacional e internacional, que se ve reflejado generalmente por hechos tales como homicidios, secuestros, asaltos y terrorismo, generalmente cometidos con armas de fuego y/o punzocortantes. A pesar de que estos delitos se encuentren regulados por algunos países, ya sea con sentencias judiciales y/o penales severas, estas no siempre logran erradicar la problemática debido a que existen factores adicionales que también agravan el problema, tales como la desigualdad económica y social, el desempleo, factores culturales, o la escasa implementación de sistemas tecnológicos que permitan combatirla o en el mejor de los casos, anticiparla. Por lo que, se podrían mejorar los sistemas de seguridad existentes o implementar nuevos, que permitan mejorar la seguridad ciudadana, ofreciendo un respaldo a las personas afectadas mediante un registro visual de los criminales que utilicen armas como medio para cometer sus crímenes.

Ante ello, y como una alternativa de solución, se observa la importancia de implementar sistemas de seguridad que puedan detectar y alertar actividades delictivas que involucren el uso de armas (punzocortantes, pistolas y/o rifles) presentes en las diferentes actividades delictivas mencionadas anteriormente. Pues, estos sistemas pueden brindar una respuesta rápida a las autoridades al identificar y alertar sobre la presencia de armas de manera rápida y eficiente, mejorando la seguridad en general en los espacios públicos y/o privados. Más aún, ayudando con el cumplimiento de las regulaciones legales existentes respecto a la seguridad y el uso de armas.

Para evaluar la problemática general y tener un panorama respecto a las publicaciones científicas sobre la cual se centra esta investigación, se incorpora un breve y simplificado análisis bibliométrico, técnica ampliamente utilizada en diferentes áreas del conocimiento para proporcionar una base teórica y contextual sólida. Es por ello, que se incluye este breve análisis para definir el

origen y el avance en este campo de investigación, así como para identificar las brechas y vacíos del conocimiento.

1.1.1. Breve Análisis Bibliométrico (panorama internacional)

El análisis bibliométrico utilizó información obtenida de la base de datos de Scopus, considerando como ecuación de búsqueda la siguiente: (TITLE-ABS-KEY ("surveillance" OR "monitoring") AND TITLE-ABS-KEY ("detect*" OR "segment*" OR "identif*" OR "classif*" OR "recogn*") AND TITLE-ABS-KEY ("weapon" OR "gun" OR "handgun" OR "firearm" OR "rifle" OR "fusil" OR "shotgun" OR "pistol" OR "knife") AND TITLE-ABS-KEY ("camera" OR "video" OR "CCTV" OR "image")) AND (LIMIT-TO (DOCTYPE, "cp") OR LIMIT-TO (DOCTYPE, "ar") OR LIMIT-TO (DOCTYPE,"ch")) AND (LIMIT-TO (LANGUAGE, "English")). Estas palabras utilizadas como filtro, se avocan principalmente a encontrar investigaciones enfocadas en el reconocimiento de armas en entornos videovigilados.

Como criterio de inclusión se consideraron a artículos de revista, artículos de conferencia, capítulos de libros, y solo se examinaron investigaciones realizadas en idioma inglés. Los datos obtenidos, luego de haber sido filtrados se exportaron en formato “.csv” y se cargaron en el software Bibliometrix [9]. El diseño del análisis bibliométrico se muestra en la Fig. 1.

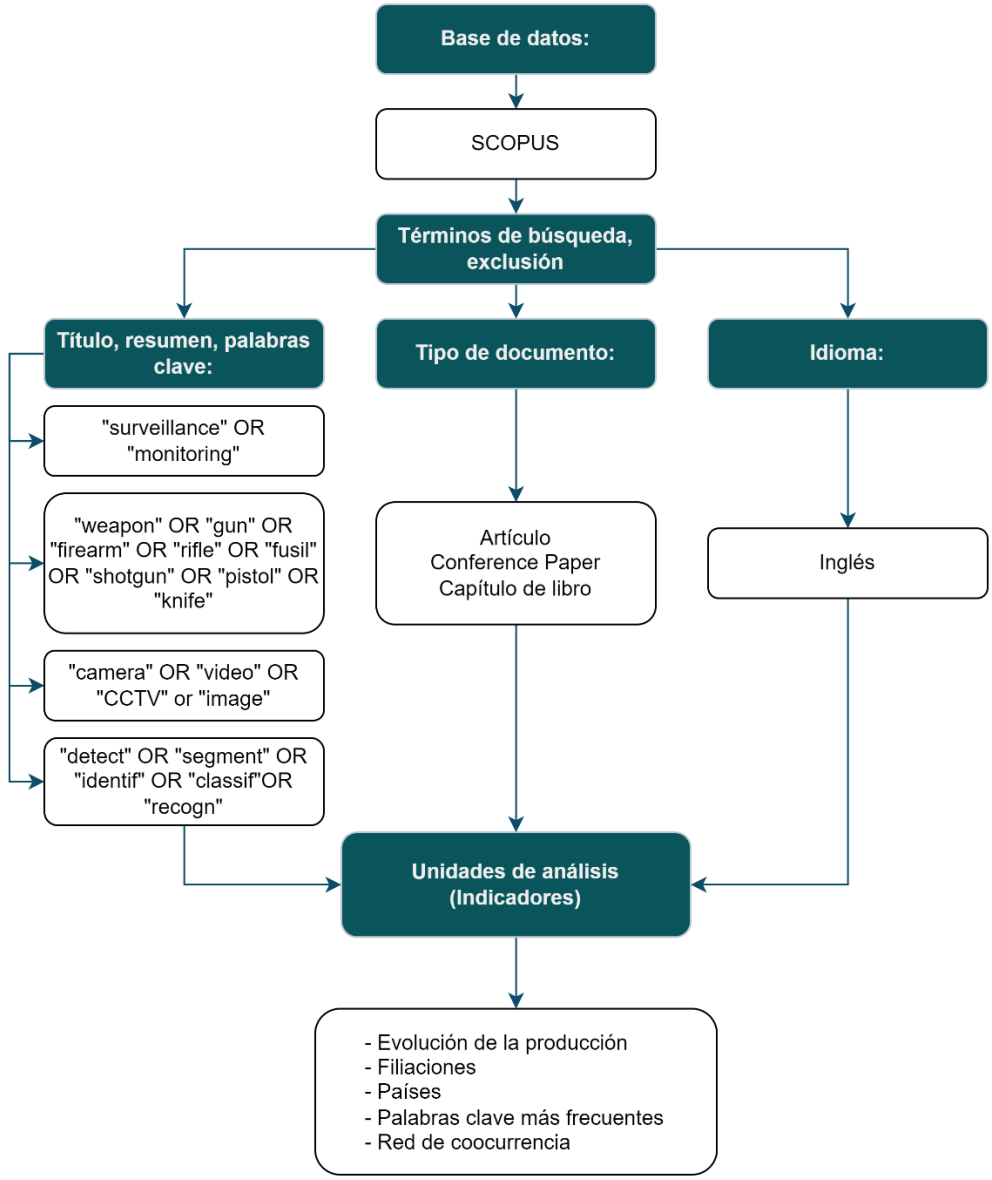


Figura 1. Diseño de la revisión bibliométrica.

Fuente: Elaboración propia.

1.1.1.1. Evolución de la producción científica global

El incremento de la cantidad de publicaciones científicas referente a los sistemas de vigilancia para la detección de armas refleja la importancia e influencia que han tenido a lo largo de estos años. De acuerdo con la información recopilada como se muestra en la Fig. 2, se observó que de los 577 documentos obtenidos de Scopus, el 57,9% (334 documentos) son documentos de conferencia, el 38,1% (220 documentos) son artículos de revista y el 4,0% (23 documentos) son capítulos de libros. La tasa anual de crecimiento desde 1973 al 2023 fue del

8,71%. A partir del 2006, se tuvo un crecimiento acelerado en las publicaciones sobre sistemas de vigilancia para la detección de armas, teniendo una tendencia exponencial a partir del año 2015, esto se debe al incipiente desarrollo de algoritmos de detección de objetos tales como YOLO [10], Single Shot MultiBox Detector (SSD) [11], Faster R-CNN [12], Fast R-CNN [13] y R-CNN minus R [14].

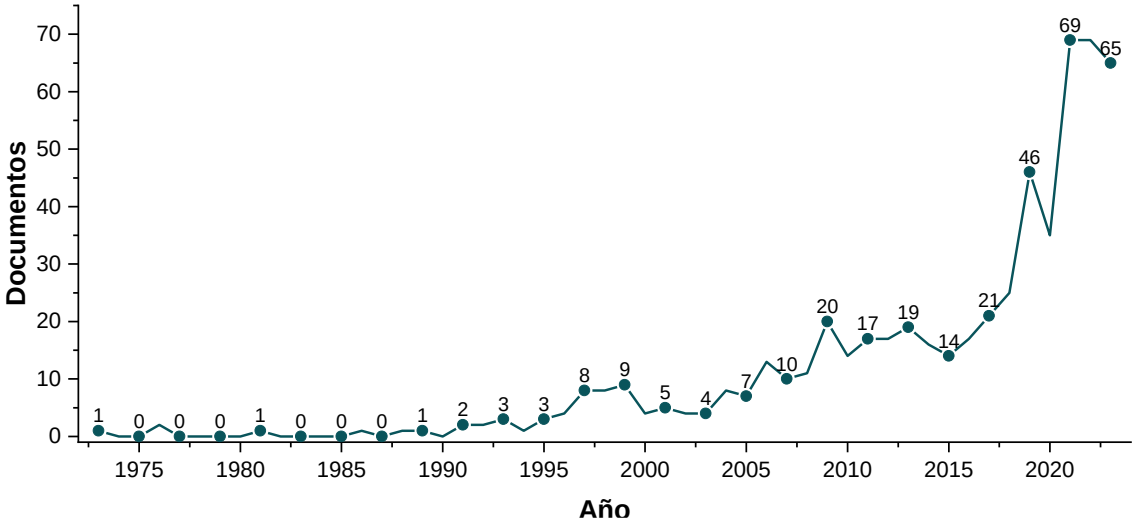


Figura 2. Cantidad de publicaciones por año.

Fuente: Elaboración propia.

1.1.1.2. Filiaciones más relevantes

En la Fig. 3 se muestra la relación de las afiliaciones más productivas e influyentes en el campo de sistemas de detección de armas, siendo la Battelle Memorial Institute (Estados Unidos) la que lidera con 25 documentos. Seguido por la Universidad de Granada (España) con 23 documentos. Entidades principalmente extranjeras (de Estados Unidos, Europa y Asia), no encontrándose alguna universidad con una alta cantidad de publicaciones en Sudamérica. Esto indica, posiblemente, que investigaciones referentes a sistemas de detección de armas no han sido implementadas o son escasas en esta área geográfica.

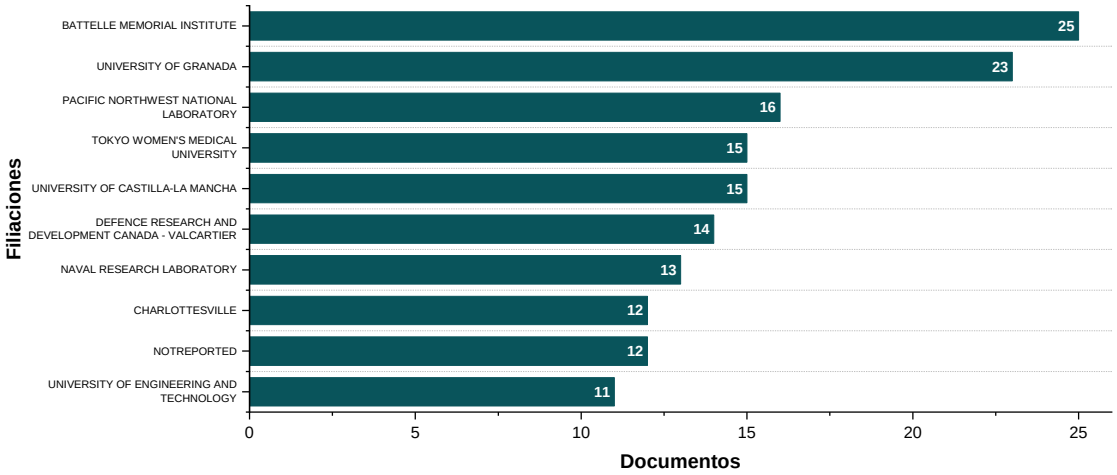


Figura 3. Filiaciones más relevantes.

Fuente: Elaboración Propia.

1.1.1.3. Países con mayor producción científica

Esto último se contrasta con el mapa mostrado en la Fig. 4, donde se aprecia que investigaciones en ese campo han sido desarrolladas en gran medida en países como Estados Unidos (564 documentos), India (410 documentos) y China (221 documentos). En Latinoamérica, se obtuvieron documentos para Perú 8 documentos, 15 documentos para Brasil y 1 documento para Colombia. Estos números indican cuántos investigadores participan en calidad de coautores, sugiriendo la importancia de investigar en esta área del conocimiento.

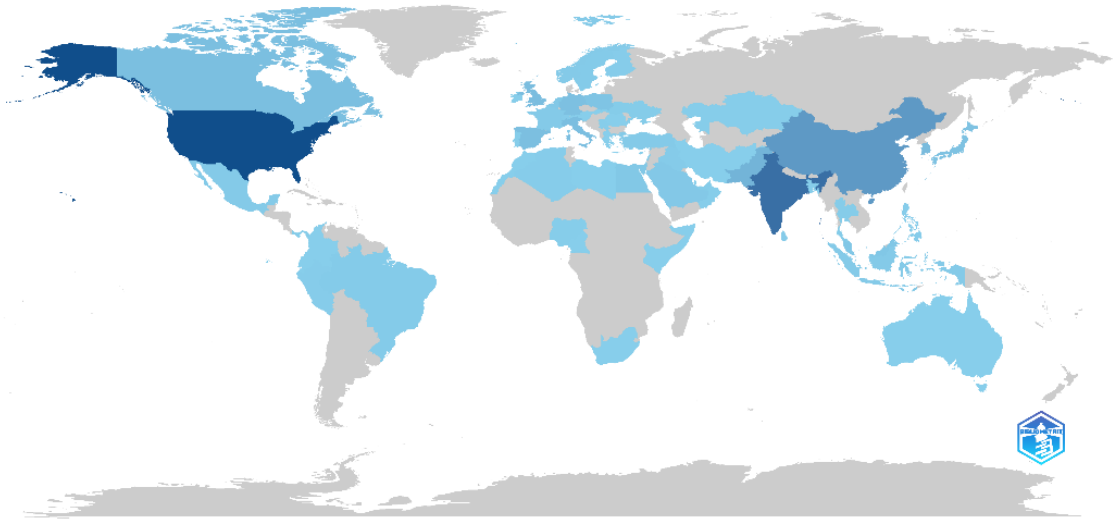


Figura 4. Producción científica por países.

Fuente: Elaboración propia utilizando la librería Bibliometrix.

1.1.1.4. Palabras clave autogeneradas más frecuentes

Esta sección nos brinda un panorama respecto al análisis de contenido de los documentos, como la identificación de los temas emergentes o de tendencia en de investigación [15]. Se puede observar de acuerdo con la Fig. 5, que las palabras clave extraídas más comunes dentro de este campo de investigación son “security systems” con una frecuencia f = 186, “deep learning” con f = 113, “object detection” con f = 91, “monitoring” con f = 82 y “cameras” con f = 73. Esto indica que existe una sinergia de investigaciones con la utilización de estas palabras clave en la realización de manuscritos científicos. De esto se puede desprender también que, cada vez se busca implementar sistemas de seguridad con tecnologías de aprendizaje profundo o inteligencia artificial para la detección de objetos, no necesariamente para la detección de armas, sino de forma general.

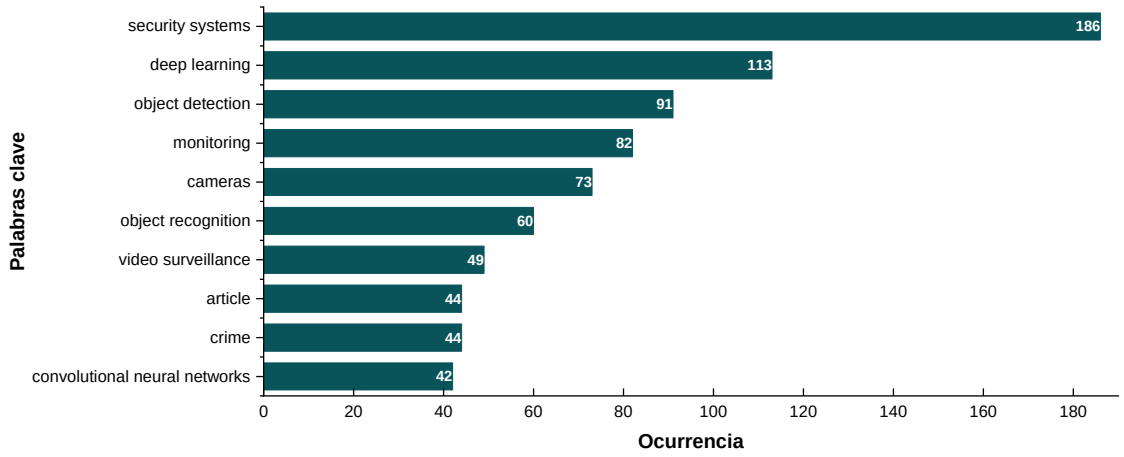


Figura 5. Palabras clave por Ocurrencia (Keyword Plus).

Fuente: Elaboración propia.

De igual manera, del análisis previo, se puede extrapolar para la Fig. 6, donde se analizan las palabras clave colocadas por los autores de los manuscritos científicos, que, tanto los términos “deep learning” y “object detection” son los más utilizados e identifican una tendencia de investigación, validando el análisis respecto a la cantidad de publicaciones existentes, donde se emplean técnicas de aprendizaje profundo para la resolución de problemas que involucren detección de objetos en imágenes.

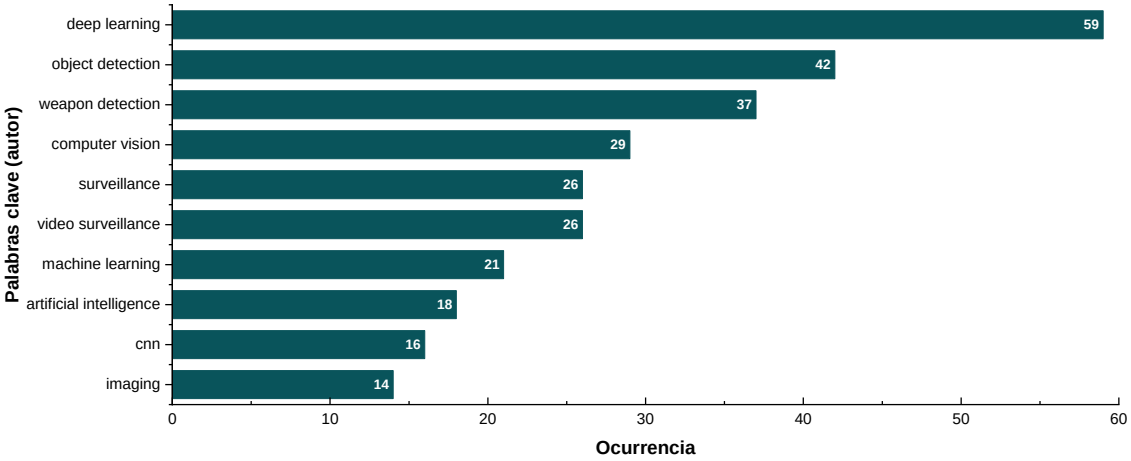


Figura 6. Palabras claves por ocurrencia (Autor).

Fuente: Elaboración propia.

1.1.1.5. Red de coocurrencia de palabras clave (autor)

En la Fig. 7 se presenta el análisis de la estructura conceptual de un campo de investigación, este mapa en particular muestra las palabras clave usadas por los autores de los manuscritos consultados, siendo la más predominante “deep learning” con 59 ocurrencias, “object detection” con 42 ocurrencias, seguido por “weapon detection” con 37. La estrecha correlación que existe entre Deep learning y weapon detecion se puede comprender a través de una exploración extensa de trabajos académicos relevantes, como los de Bhatti et. al [16] y Jain et. al [17]. Esto puede indicar, que estas palabras clave también pueden determinar un área de especialización predominante o en desarrollo de acuerdo con los datos obtenidos, de igual manera, puede indicar que estos autores podrían estar realizando o colaborando con proyectos similares. Sin embargo, esto no puede describir la calidad o impacto de las publicaciones realizadas por los autores, y tampoco garantizaría si el trabajo es innovador o influyente.

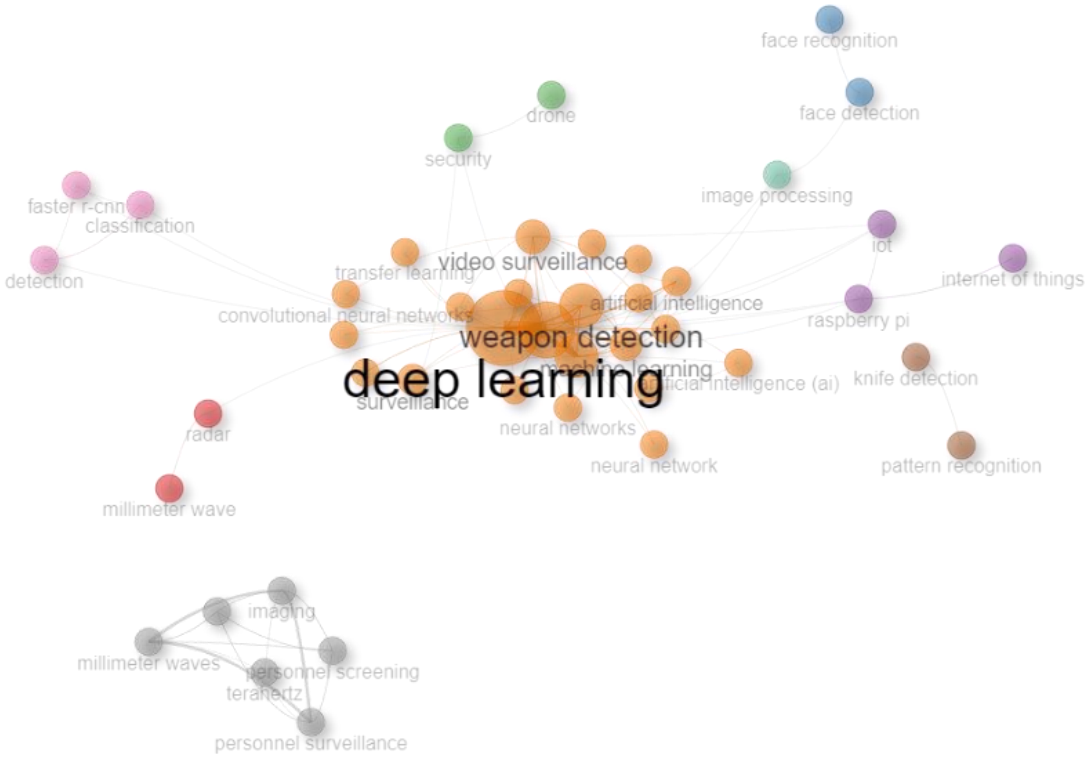


Figura 7. Red de Coocurrencias de palabras clave (Autores)

Fuente: Elaboración propia utilizando la librería Bibliometrix.

Ante lo expuesto anteriormente, se refleja que el área del Deep learning o aprendizaje profundo está en constante crecimiento de acuerdo con las investigaciones recopiladas de Scopus para este breve y simplificado análisis bibliométrico, por lo que se valida también que existe una escasez de investigaciones enfocadas a la aplicación de IA para la detección de armas. Por lo que es necesario profundizar y explorar diferentes soluciones que ayuden a concretar proyectos que se enfoquen a esa problemática. Al ser el área de inteligencia artificial un campo muy vasto y con mucho trabajo colaborativo, cada día se desarrollan diferentes algoritmos que ayudan a perfeccionar los existentes o creando nuevos con el fin de optimizarlos, de este modo se pueden aplicar estos nuevos algoritmos con la finalidad de obtener mejores resultados. Y como si fuera poco, también se podrían ir germinando nuevas líneas de investigación que ayuden a consolidar el campo del Deep learning y su aplicación en áreas relacionadas a ciudades inteligentes, seguridad ciudadana y residencial.

1.1.2. Breve Análisis (panorama nacional)

Luego de haber revisado el panorama internacional a través de las publicaciones científicas, a nivel nacional podemos apreciar que en los últimos años se ha visto un incremento gradual de los delitos contra el patrimonio, estos delitos no solo acarrean en agresión física, sino que, en algún caso extremo, también peligra la integridad de la persona afectada, y en el peor de los casos, conlleva a la muerte de la persona por ofrecer resistencia o por tratar de proteger sus bienes.

Pese al incremento de la seguridad ciudadana con las cámaras de vigilancia y los patrullajes policiales continuos en los distritos para la vigilancia de las calles y vecindarios, en general, estos no se logran abastecer para acaparar grandes distritos o lugares no muy concurridos, debido en cierta medida al desconocimiento de éstos por parte del personal competente. Por lo tanto, es necesario reducir los tiempos de respuesta desde que la persona agraviada comunica a las autoridades correspondientes sobre el delito, hasta que las autoridades policiales o de serenazgo se apersonan al lugar de los hechos.

Por ello, es de suma importancia que se tengan tiempos de respuesta casi inmediatos por parte del personal de serenazgo o de la policía, con el fin de evitar posibles pérdidas de vidas humanas durante la ejecución de un acto delictivo y en el mejor de los casos, atrapar a los criminales.

En el Perú, los sistemas de videovigilancia son utilizados por los gobiernos locales y regionales para combatir los índices de delincuencia que, después de la pandemia del COVID-19, se han incrementado sucesivamente luego de que las restricciones sanitarias y de confinamiento fueran gradualmente levantadas [18]. Esto último se contrasta según datos obtenidos por el INEI, el cual indica que en el semestre de julio – diciembre del año 2022 considera que, a nivel nacional urbano, alrededor del 18,5% de la población urbana mayor de 15 años, el 20,5% de personas en ciudades de más de veinte mil habitantes y el 13,4% de personas de centros poblados urbanos entre dos mil y menos de veinte mil habitantes se han visto afectados por algún hecho delictivo, en la Fig. 8 se aprecia el porcentaje de los afectados según el tipo de arma [19].

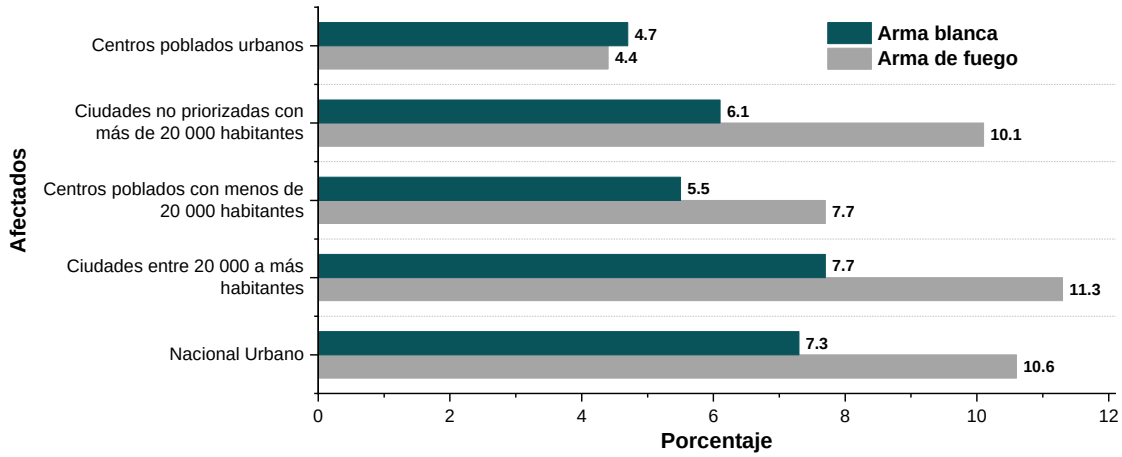


Figura 8. Porcentaje de población afectada por hechos delictivos según el tipo de arma

Fuente: Instituto Nacional de Estadística e Informática [19]

Estos valores son alarmantes, pues se aprecia que un porcentaje considerable se ha visto expuesto ante este tipo de asaltos con armas letales, los cuales podrían haberse desencadenado en hechos funestos.

La preocupación se incrementa al ver las cifras de las estimaciones de las tasas de homicidios por cada cien mil habitantes brindadas por el MININTER [20], que muestra las proyecciones de los últimos 14 años desde el 2010 – 2023, un fragmento de esa información (10 años) se puede apreciar en la Fig. 9.

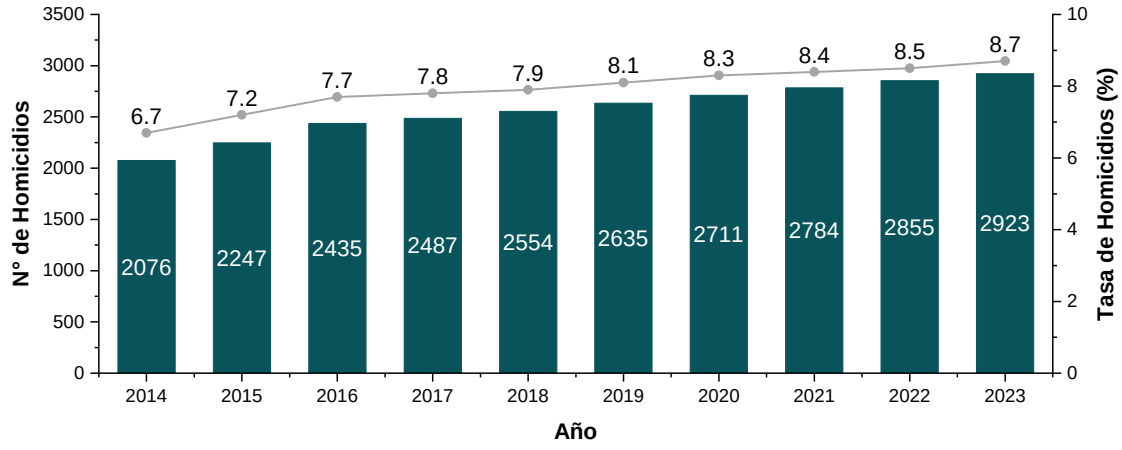


Figura 9. Proyección de número y tasa de homicidios desde 2014 al 2023

Fuente: Ministerio del Interior [20]

A pesar de que las municipalidades cuenten con cámaras de vigilancia o incluso tengan integradas tecnologías de detección de intrusos, detección de rostros o de transmisión de vídeo en tiempo real, estos no cuentan con la capacidad de detectar el peligro que tendrían estas personas al momento de ser acechadas por individuos armados, ya sea con armas blancas o de fuego.

Por lo que es necesario implementar sistemas que integren este tipo de detecciones para incrementar las funcionalidades de éstas, pudiendo ayudar a reducir los riesgos latentes de sufrir asaltos con armas y de modo preventivo para que las autoridades sepan con qué tipo de amenaza se enfrentan.

1.2. Formulación del problema

1.2.1. Problema general

¿El desarrollo de un sistema de alerta y videovigilancia utilizando Raspberry Pi 4B puede realizar la detección de armas en la Región Callao, 2023?

1.2.2. Problemas específicos

PE1: ¿Cuál es el modelo de detección de objetos a utilizar en un sistema de alerta y videovigilancia utilizando Raspberry Pi 4B para la detección de armas en la Región Callao, 2023?

PE2: ¿Cuál es el tipo de notificación a utilizar en un sistema de alerta y videovigilancia utilizando Raspberry Pi 4B para la detección de armas en la Región Callao, 2023?

1.3. Objetivos

1.3.1. Objetivo general

Desarrollar un sistema de alerta y videovigilancia utilizando Raspberry Pi 4B para la detección de armas en la Región Callao, 2023.

1.3.2. Objetivos específicos

OE1: Evaluar el modelo de detección de objetos a utilizar en el sistema de alerta y videovigilancia utilizando Raspberry Pi 4B para la detección de armas en la Región Callao, 2023.

OE2: Evaluar el tipo de notificación a utilizar en el sistema de alerta y videovigilancia utilizando Raspberry Pi 4B para la detección de armas en la Región Callao, 2023

1.4. Justificación

1.4.1. Justificación tecnológica

El presente dispositivo por desarrollar busca implementar funcionalidades adicionales que los sistemas de videovigilancia convencionales no pueden realizar, como la detección de armas en tiempo real y el envío de alertas, e integrados en sistemas de bajos recursos computacionales como la Rasbperry Pi 4B.

1.4.2. Justificación práctica

El desarrollo de un sistema de alerta y videovigilancia utilizando Raspberry Pi 4B para la detección de armas en la región Callao, 2023 permite notificar a las autoridades competentes cuando se ejecuten delitos con armas, pudiendo contribuir a la reducción de los índices de criminalidad e incrementando la seguridad ciudadana.

1.4.3. Justificación económica

El sistema desarrollado permitirá a las municipalidades y/o entidades públicas o privadas, minimizar costos por el personal que realiza el monitoreo en cámaras de videovigilancia debido a la automatización del proceso de identificación y envío de alertas por delitos con armas.

1.5. Delimitantes

1.5.1. Delimitante teórica

La presente investigación se limita al estudio de algoritmos de detección de objetos aplicados en sistemas de bajo poder computacional como una Raspberry Pi 4B. Asimismo, se evaluarán tecnologías disponibles para la emisión de alertas por medio de conexión a una red celular, internet y aplicaciones móviles con el fin de obtener un menor tiempo de respuesta.

1.5.2. Delimitante temporal

La presente investigación tendrá una duración de 12 meses, viéndose principalmente afectada por la dificultad que se tuvo para encontrar réplicas de armas de fuego de corto y largo alcance debido a las regulaciones existentes.

1.5.3. Delimitante espacial

La investigación se realizó en la Región Callao, el uso del dispositivo creado está limitado a utilizarse en entornos urbanos, rurales y residenciales. Además, esta investigación se limita a detectar armas en ambientes diurnos y/o bien iluminados.

II. MARCO TEÓRICO

2.1. Antecedentes: Internacionales y nacionales

2.1.1. Internacionales

En investigaciones realizadas a nivel internacional, se encontró en el artículo de Ingle et al. [21], los autores desarrollaron un método de detección de diferentes tipos de armas como pistolas y cuchillos en tiempo real utilizando una red neuronal convolucional. La base de datos de imágenes que utilizaron para entrenar la red neuronal fue de ImageNet e IMFDB, Open Image Dataset V4 y Olmos [22] donde el 75% de las imágenes obtenidas se usaron para el entrenamiento, 15% para la validación y el 10% para las pruebas. El desarrollo del algoritmo se hizo con las librerías de Tensorflow y Keras, el entrenamiento de la red neuronal se realizó con una tarjeta gráfica Nvidia RTX 2060, y para la prueba del algoritmo se utilizó una cámara web Logitech C920 Pro HD con una Raspberry Pi 4B. El método del algoritmo propuesto en la investigación para la detección de pistolas y cuchillos obtuvo una tasa de detección del 97.50% en el conjunto de datos de ImageNet e IMFDB, 90.5% con Open Image Dataset V4, 93% con el conjunto de datos de Olmos y 90.7% de precisión en cámaras multivista.

En la tesis de Oñate [23], se tuvo como objetivo el diseño y construcción de nodos inteligentes para la detección de armas dentro de una red de videovigilancia empleando técnicas de visión artificial, para ello utilizaron una base de datos de 4 500 imágenes con armas y 8 000 imágenes sin armas para el entrenamiento de la red neuronal. El desarrollo del algoritmo se implementó con el lenguaje de programación Python utilizando la librería de OpenCV, la alerta del sistema se implementó con un módulo GSM SIM 800L en conjunto con una Raspberry Pi V3 y una cámara de Raspberry Pi B1.3. Las pruebas realizadas en la investigación determinaron que el tiempo promedio de envío de la alerta GSM fue de 6.5 segundos y para el envío y almacenamiento en la base de datos de Gmail tuvo un tiempo promedio de 10 segundos.

En la tesis de Suárez [24], se desarrolló un sistema para la detección de actividades criminales en sistemas de videovigilancia para su uso por la Policía Nacional de Colombia. Este sistema fue implementado utilizando técnicas de Deep Learning y de Redes Neuronales Convolucionales (R-CNN) para la detección de armas de fuego de corto alcance y hurtos utilizando la arquitectura de la red neuronal AlexNet. El entrenamiento de la red neuronal se realizó en una computadora con una GPU GTX 1070, y realizándolo en 100 épocas. El desarrollo de la investigación demostró que la utilización de R-CNN es viable para su implementación en sistemas de detección de actividades criminales. El autor concluyó que el sistema está limitado por la calidad de la imagen suministrada a la red neuronal, como baja iluminación y condiciones climáticas u otros factores que pueden afectar la detección las actividades criminales.

Aguilar [25], realizó una investigación que abordó la identificación de armas utilizando una tarjeta Raspberry Pi 3B+. El objetivo principal de la tesis fue identificar personas utilizando armas en tiempo real. El diseño de la investigación abordó en primera instancia la adquisición de imágenes de armas con una minicámara a una resolución de 4608x2592 píxeles para el entrenamiento, luego aplicó un preprocesamiento redimensionando las imágenes a 640x480 píxeles. El entrenamiento lo realizó con la librería de OpenCV utilizando imágenes positivas y negativas de las armas (pistolas y cuchillos) y de personas mediante un clasificador HAAR y LBP. Los principales resultados indicaron que la utilización del clasificador tipo HAAR obtuvo mejores resultados con una precisión del 78% para las armas (pistolas y cuchillos) y el clasificador LBP con 60 y 72% respectivamente.

En el trabajo de Criollo-Leal y Avendaño-Guzmán [26], tuvieron como objetivo el desarrollo de un método para la detección de armas utilizando técnicas de aprendizaje profundo. Para la recopilación de los datos, obtuvieron videos de YouTube donde se visualizaban armas de mano para el modelamiento del algoritmo. Utilizaron la arquitectura Faster-RCNN como modelo de detección, teniendo una métrica de desempeño mayor al 60% de mAP, considerado aceptable por los autores para emplearse para la detección de armas.

2.1.2. Nacionales

A nivel nacional, en la tesis de Huaman [27], se desarrolló un sistema para la detección de armas utilizando la red neuronal convolucional YOLO, logrando un rendimiento de 45 fotogramas por segundo con una tasa de detección del 97.30% aproximándose a una detección en tiempo real.

En el artículo Leturia et al. [28], los autores desarrollaron un algoritmo para la detección de objetos (cuchillos, bates de beisbol, pistolas) y patrones de movimiento utilizando el modelo de YOLOv3 en combinación con técnicas de algoritmos genéticos. La base de datos empleada estuvo conformada por 100 imágenes y modificadas con técnicas de “Data Augmentation” para la diversificación de las imágenes. La precisión del modelo desarrollado logró un 81% de detección y al utilizarse en conjunto con el algoritmo genético se logró un 96.5% de precisión para la detección de comportamientos anormales como mejor resultado.

En la tesis de Pacco [29], se desarrolló un sistema de videovigilancia y alerta utilizando una Raspberry Pi 4 para el reconocimiento de armas (pistolas, cuchillos), con el clasificador de imágenes Haar Cascade. El dataset de imágenes que utilización fue de 500 y 1000 imágenes respectivamente (positivas y negativas) para el entrenamiento de la red neuronal empleando una cámara Hikvision Turbo HD. Concluyeron que el desarrollo de la investigación obtuvo una tasa de detección de armas del 90.30% y 88.31% de precisión y exactitud respectivamente. La utilización de alertas por WhatsApp en la investigación no mostró tiempos cortos de respuesta (6 segundos de retardo) frente a las actividades delictivas debido a que se utilizan librerías para navegar por la plataforma mencionada antes de enviar las notificaciones. Las imágenes obtenidas por cámaras digitales tuvieron que ser redimensionadas debido a la resolución mayor que se obtienen de ellas para hacer la inferencia con el algoritmo de detección desarrollado.

En el trabajo de Aliaga y Pariona [30], tuvieron como objetivo principal el desarrollar un algoritmo para la identificación de gestos corporales en asaltos a mano armada en un local comercial. Utilizaron el lenguaje de programación

Python 3.7 y una tarjeta Raspberry Pi. Para la validación del modelo utilizaron métricas como la Curva ROC, matriz de confusión, precisión y el coeficiente de Matthews. Como resultado de la investigación, lograron un 89.01% de precisión y un 84.40% de rendimiento.

En la tesis de Blas y Jimenez [31], tuvieron como objetivo la implementación de un algoritmo para la detección de actos delictivos en Comas. El diseño de la investigación fue pre-experimental con enfoque cuantitativo. Utilizaron YOLO como algoritmo para la detección de objetos sobre una muestra de 16 videos para el análisis. Dentro de los parámetros para la evaluación del algoritmo utilizaron métricas de precisión, sensibilidad, F1 score y exactitud. En los resultados obtenidos tuvieron un 89% de precisión, 69% de sensibilidad, 72% de F1 score y 63% de exactitud.

2.2. Bases teóricas

2.2.1. Sistemas de videovigilancia

Desde el punto de vista tradicional, la videovigilancia, implica la observación de la conducta humana, actividades y/o alteraciones del entorno por parte de operadores cualificados, con la intención de salvaguardar un área determinada [32], la descripción gráfica de la definición se muestra en la Fig. 10.

Figura 10. Sistemas de videovigilancia tradicional.

Fuente: Agencia Peruana de Noticias [33].

Por otra parte, un sistema de videovigilancia inteligente radica en su capacidad para la realización de análisis automáticos de la escena, sin la necesidad de personal cualificado, y, además, abarcan diferentes tareas cruciales como la detección, seguimiento e identificación de personas u objetos [34] que pueden presentar un riesgo potencial para las personas circundantes, tal como muestra la Fig. 11.

Figura 11. Sistemas de videovigilancia inteligente.

Fuente: Extraído de [32].

En ambos puntos de vista, tanto tradicional como inteligente, estos sistemas se utilizan con dispositivos de almacenamiento para guardar las imágenes recopiladas, o se integran a través del uso de internet para la transmisión de la información de los hechos a ubicaciones remotas.

2.2.2. Sistemas de alerta

Este tipo de sistemas funcionan como un modo de comunicación, transmitiendo información crucial sobre amenazas potenciales. Poseen la capacidad de transmitir y recibir esta información, lo que permite a las personas u organizaciones abordar rápidamente cualquier peligro o emergencia inminente, garantizando respuestas rápidas a peligros potenciales por parte de las autoridades [35].

2.2.2.1. Tecnologías de difusión de alertas

El desarrollo progresivo de la ciencia y tecnología ha permitido la creación de dispositivos o componentes que puedan ir integrándose en sistemas de bajo

consumo, como celulares, computadoras de bolsillo, dispositivos IoT y relojes inteligentes.

Éstos a su vez tienen incluidos diferentes tecnologías de comunicación, como el envío de información por SMS, correo electrónico, notificaciones de mensajería instantánea, llamadas de voz, entre otros.

- **SMS**

Por su traducción al español "*Servicio de mensajes cortos*" por la limitación de caracteres que tienen [36], es una forma efectiva de enviar rápidamente notificaciones a otros dispositivos móviles, aprovechando las capacidades de los proveedores de servicios de mensajería para trasmitir texto automáticamente, además de enviar notificaciones en zonas donde no haya conexión a internet [37]. Para el uso de este tipo de alerta, se necesitan de dispositivos que incorporen un módulo GSM que utilizan una SIM o chip de algún operador de telefonía [38], [39], entre los dispositivos que integran este módulo tenemos por ejemplo el SIM800L, SIM900, SIM7600, etc.

- **Correo electrónico**

Es una plataforma digital que facilita la creación, transmisión y recepción de diferentes tipos de contenido digital entre usuarios utilizando el protocolo SMTP y POP3/IMAP, para el envío y recepción respectivamente. Se requiere de una conexión a internet y un dispositivo con capacidad de navegación web o compatible con aplicaciones de correo electrónico [40].

- **Notificaciones de mensajería instantánea**

Las notificaciones de mensajería instantánea son alertas que se envían dispositivos electrónicos, incluidos teléfonos móviles o computadoras, para informar a los usuarios sobre diferentes sucesos. Estas notificaciones se generan cada vez que se producen nuevos mensajes, llamadas o interacciones dentro de la plataforma de mensajería que, a su vez, pueden integrar también el envío archivos, audios, imágenes y videos [41]. Este proceso se realiza mediante la utilización de servicios y protocolos para garantizar la entrega rápida y fluida de la alerta al destinatario previsto. Dentro de las ventajas que tienen este tipo

de tecnologías, es que permite a los usuarios recepcionar y enviar información o notificaciones en tiempo real. Más aún, estas notificaciones pueden automatizarse sin la necesidad de que exista un usuario que envíe la notificación, ya que se puede crear un Bot para que realice este proceso, y aplicaciones móviles como Telegram integran esta función [42].

2.2.4. Visión por computadora

La visión por computadora, también conocida como percepción visual, abarca una colección de tecnologías y herramientas que permiten a las computadoras capturar imágenes del mundo real, procesarlas y extraer información valiosa mediante un análisis para la resolución de tareas específicas [43].

2.2.4.1. Detección de objetos

Se refiere a la identificación y el posicionamiento preciso de objetos dentro del contenido visual. Este proceso tiene como objetivo identificar en dónde se encuentran los diversos objetos presentes dentro de una escena o fotograma, proporciona información invaluable sobre la disposición espacial exacta de cada objeto mediante la utilización de coordenadas delimitadoras, también conocidas como cuadros delimitadores [44], tal como se muestra en la Fig. 12.

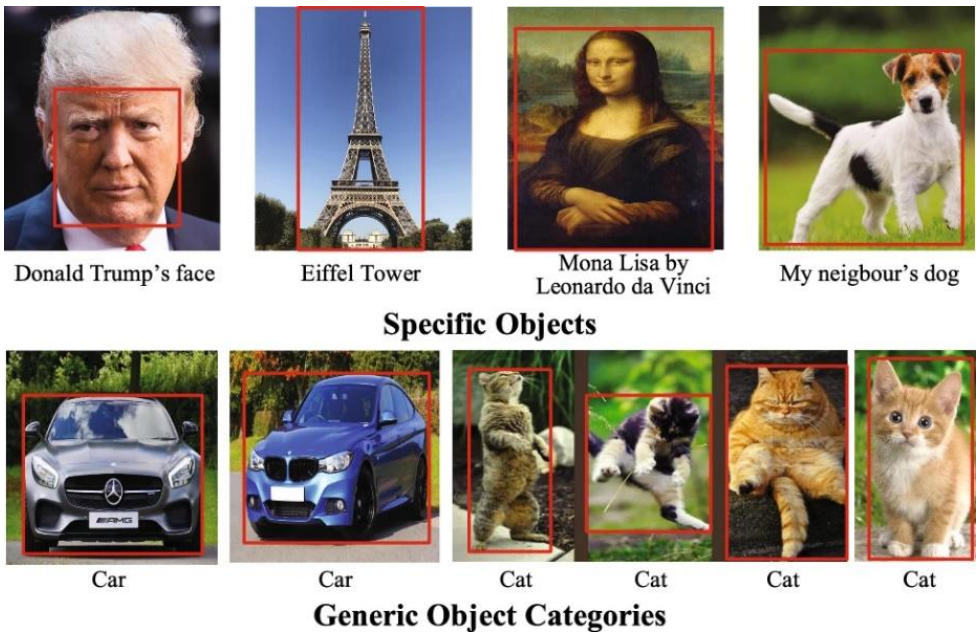


Figura 12. Ejemplos de detección de objetos.

Fuente: Extraído de [45].

- **Detección de armas**

La detección de armas implica la abstracción de armas de fuego u otros armamentos dentro de medios visuales, que sirven como un componente de la detección de objetos. Esta tecnología encuentra una amplia aplicación en los ámbitos de la seguridad, incluidos, entre otros, aeropuertos, estaciones de tren, instituciones educativas y diversos espacios públicos. Al emplear técnicas sofisticadas de visión por computadora, esta disciplina abarca tanto metodologías convencionales como de aprendizaje profundo [46].

2.3. Marco conceptual

2.3.1. Redes Neuronales Convolucionales (CNN)

Son un tipo de modelo de aprendizaje diseñado para analizar imágenes o videos, este tipo de redes tienen la capacidad de extraer características como bordes, texturas y patrones complejos que le permiten realizar predicciones. El núcleo de funcionamiento de las CNN se basa específicamente en la operación de convolución, siendo un filtro o también denominado kernel a una matriz usualmente de $3x3$, $5x5$ o de otras dimensiones de acuerdo con los requerimientos, que se aplica y se desliza sobre toda la región de la imagen. Estos filtros, refinan continuamente sus valores durante todo el proceso permitiéndoles comprender los datos analizados. Además, al utilizar CNN, se puede reducir el tamaño de los datos y al mismo tiempo extraer las características fundamentales, reduciendo así la complejidad de la red [47]. El proceso de aplicación del kernel sobre un dato de entrada se puede apreciar en la Fig. 13.

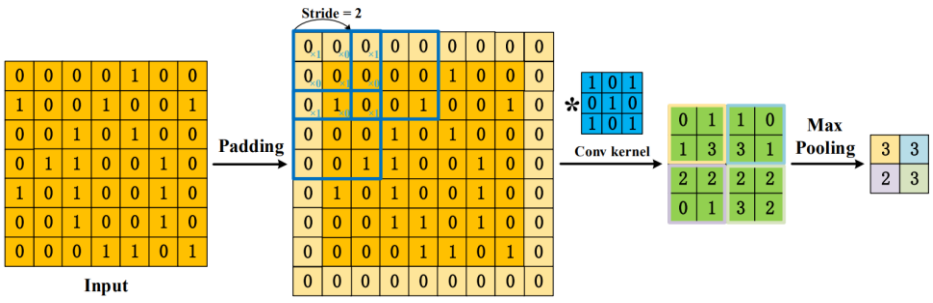


Figura 13. Proceso de una CNN bidimensional.

Fuente: Extraído de [48].

Asimismo, de forma simplificada se muestra en la Fig. 14 una CNN de cinco capas.

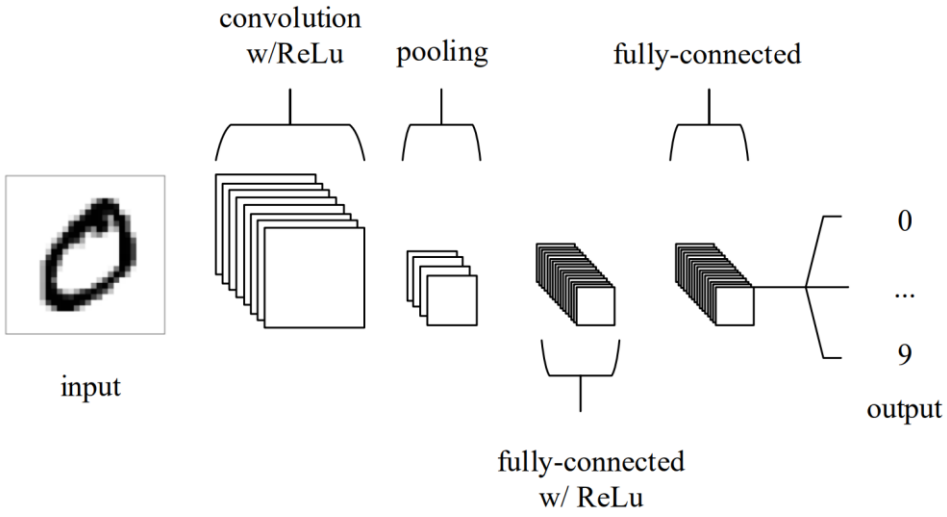


Figura 14. Arquitectura de una CNN comprendida por 5 capas.

Fuente: Extraído de [49].

2.3.2. Modelos de detección de objetos

2.3.2.1. R-CNN

Este modelo fue propuesto en 2013 por Girshick *et al.*, en la Fig. 15 se muestra el funcionamiento del modelo en primera instancia utiliza regiones características independientes que especifican las posibles detecciones sobre una imagen ingresada, luego una CNN extrae información sobre las características de cada región y finalmente utilizando SVM lineales, define la detección de las categorías de objetos sobre la imagen [50], [51].

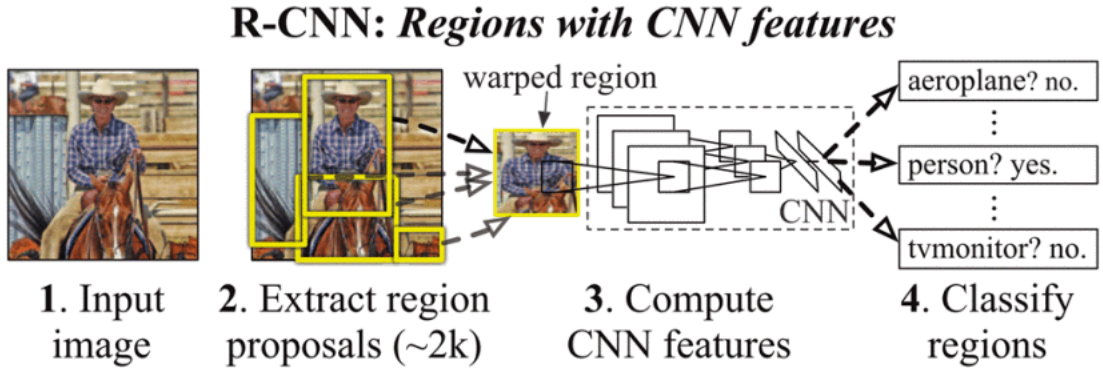


Figura 15. Proceso propuesto por el modelo R-CNN.

Fuente: Extraído de [51].

2.3.2.2. Fast R-CNN

Este modelo también fue propuesto por Girshick en el 2015, el modelo propone una mejora sobre la anterior arquitectura diseñada de R-CNN, implementando una agrupación de ROI's, (regiones de interés), que reduce la carga computacional ya que no genera regiones características independientes con posibles detecciones sobre la imagen [13], la arquitectura propuesta se muestra en la Fig. 16.

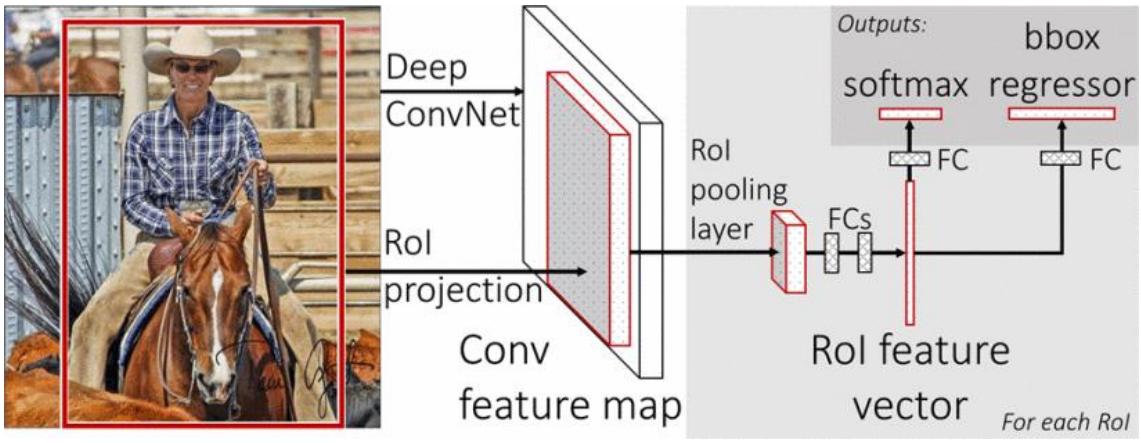


Figura 16. Arquitectura Fast R-CNN.

Fuente: Extraído de [13].

2.3.2.3. Faster R-CNN

Modelo propuesto por Ren *et al.* en 2015, este eleva las capacidades del modelo Fast R-CNN al incorporar una red de propuesta de regiones (RPN) junto a una red CNN, permitiendo que la RPN aproveche las características convolucionales de la red de detección, estas RPN se generan de manera eficiente, eliminando la necesidad de escanear cada posición potencial dentro de las imágenes, lo que permite tener un costo computacional mínimo [12]. El principio de funcionamiento de la red Faster R-CNN se muestra en la Fig. 17.

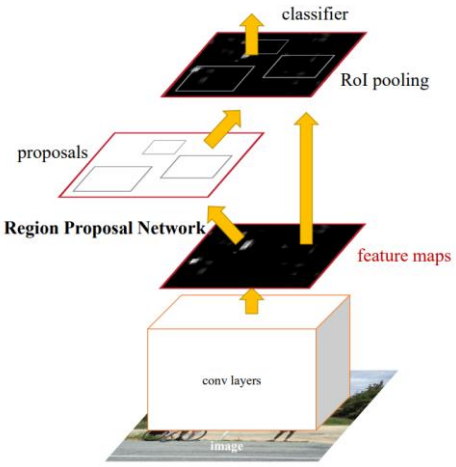


Figura 17. Red Faster R-CNN.

Fuente: Extraído de [12].

2.3.2.4. YOLO

Este modelo emplea una red neuronal que procesa de forma integral toda la imagen, dividiéndola en regiones para luego predecir cuadros delimitadores y mapas de probabilidades para cada región, utilizando umbrales de confianza y supresión no máxima (NMS). Cada cuadro delimitador predicho posee valores como el centro del objeto detectado (x, y), la altura y ancho (h, w) del objeto detectado respecto a toda la imagen, y el valor de confianza, que muestra la predicción de un objeto sobre todos los cuadros de la imagen que representan la categoría predefinida para ese mismo objeto. La Fig. 18 muestra el funcionamiento del modelo YOLO.

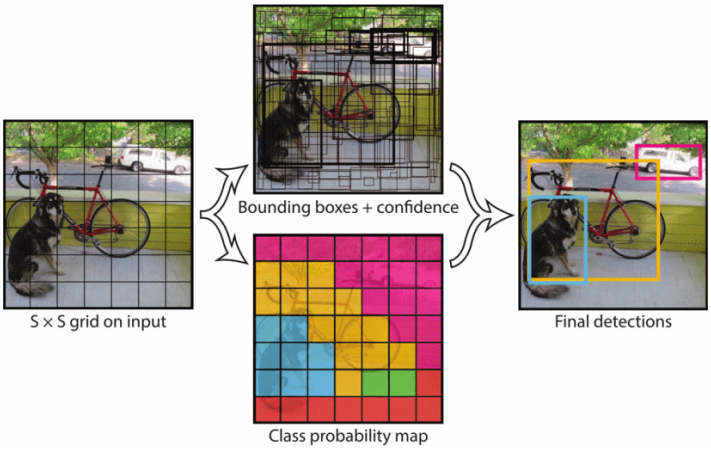


Figura 18. Funcionamiento del modelo YOLO.

Fuente: Extraído de [10].

2.4. Definición de términos básicos

Aumento de datos: Es una práctica ampliamente utilizada dentro del ámbito de la inteligencia artificial, específicamente dentro de las redes neuronales convolucionales (CNN), para la obtención de un mayor conjunto de imágenes o información cuando se tienen datos limitados [52].

Matriz de confusión: Permite evaluar la eficacia de un modelo de clasificación tanto en el ámbito estadístico como en el del aprendizaje automático. Esta matriz presenta la precisión de las predicciones, distinguiendo entre las clasificaciones precisas y erróneas realizadas por el modelo, mientras alinea las clases verdaderas que se encuentran dentro del conjunto de datos.

Raspberry Pi: La Raspberry Pi es una microcomputadora que funciona con sistema operativo Linux y es compatible con lenguajes de programación como Scratch, Python, etc. Además, tiene pines que permiten comunicarse con diferentes tipos de sensores utilizando protocolos I2C, SPI y UART [53].

Bot: Herramienta o programa utilizado para la automatización de tareas específicas y que buscan imitar el comportamiento humano que puede tener diferentes aplicaciones, estos también pueden trabajar de forma mucho más rápida [54].

Computación de borde: Se refiere a la ubicación y despliegue de la potencia informática en un dispositivo electrónico físico que se encuentra cerca de la fuente de los datos, lo cual permite obtener mayor agilidad y confiabilidad para los usuarios finales, esto debido a la reducción de la latencia que se genera al no enviarse los datos a procesar a servidores externos, sino que estos son procesados en el propio dispositivo [55].

Transferencia de aprendizaje: Es una técnica que utiliza una red neuronal que ha sido previamente entrenada con un gran conjunto de datos, para que luego, los parámetros existentes de dicha red neuronal puedan ser utilizados para entrenar un nuevo modelo con usos personalizados [56].

Tasa de aprendizaje: Parámetro fundamental de las redes neuronales profundas, este valor es dinámico, y su elección es crucial para determinar si un

modelo alcanzará una precisión elevada y una convergencia mucho más rápida [57].

Cuantización: Técnica utilizada para disminuir el costo computacional y de memoria requerido al ejecutar inferencias en un modelo de redes neuronales, mediante una reducción del número de bits en los pesos del modelo, pudiendo impactar también en la precisión del modelo [58].

Non-maximum supression (NMS): Es una técnica que se aplica principalmente en modelos de detección de objetos, su objetivo es la de seleccionar el cuadro delimitador que mejor se ajusta al objeto entre un grupo de cuadro superpuestos [59].

Instancias positivas y negativas: Se refiere a la característica o atributo que nos interesa detectar o clasificar como “positiva” o “negativa” respectivamente.

Predicciones positivas y negativas: Se refiere a una predicción que el modelo identifica como perteneciente a la clase deseada como “positiva” o “negativa” respectivamente.

III. HIPÓTESIS Y VARIABLES

3.1. Hipótesis

Hipótesis general:

El desarrollo de un sistema de alerta y videovigilancia utilizando Raspberry Pi 4B realiza la detección de armas y envío de alertas en la Región Callao, 2023.

Hipótesis específicas:

HE1: El modelo de detección de objetos seleccionado e implementado en una Raspberry Pi 4B realiza la detección de armas en la Región Callao, 2023.

HE2: El tipo de alerta seleccionado e implementado en una Raspberry Pi 4B realiza el envío de alertas y/o notificaciones de la detección de armas en la Región Callao, 2023.

3.1.1. Operacionalización de variable

Tabla 1. Operacionalización de la variable independiente y dependiente.

VARIABLE	DEFINICIÓN CONCEPTUAL	DEFINICIÓN OPERACIONAL	DIMENSIÓN	INDICADOR
Variable Independiente Sistema de Alerta y Videovigilancia	Un sistema de alerta y videovigilancia es un dispositivo electrónico que permite el registro de la actividad en escenas públicas o privadas mediante una cámara [60], y tiene integrado un hardware de comunicación que notifica cuando se detecta alguna actividad o evento específico.	Será medido a través del registro visual de la calidad de las imágenes y videos captados por el sistema, la interfaz de comunicación y el tiempo de envío de los datos.	Tipo de Cámara	Resolución Interfaz Autofoco
			Tipo de Alerta	Tiempo de envío
Variable Dependiente Detección de armas	Es proceso computacional que permite la detección de armas específicas en imágenes o videos [61], [62].	Es la variable que indica el modelo de detección de objetos a utilizar para la detección de armas, evaluado con métricas de desempeño del algoritmo.	Armas punzocortantes	mAP Precision Recall FPS F1-score
			Armas de fuego de corto alcance	
			Armas de fuego de largo alcance	

Fuente: Elaboración propia.

IV. METODOLOGÍA DEL PROYECTO

4.1. Diseño metodológico

De acuerdo con [63], [64], esta investigación es de tipo tecnológica-aplicada, pues la esencia de este trabajo es la de resolver un problema en particular de nuestra sociedad a través de la aplicación de resultados de los conocimientos de investigaciones básicas; utilizando modelos de detección de objetos basados en aprendizaje por computadora para su implementación en microcomputadores con el fin de detectar armas, y realizar la identificación del tipo de comunicación para el envío de la notificaciones.

Esta investigación posee un diseño preexperimental tal como muestra la Fig. 19, pues solo existe un grupo de prueba al cual se realizará una sola medición (posprueba), además que se carece de control sobre las variables desconocidas [63], [64], [65].

Figura 19. Diseño de la Investigación.

Fuente: Adaptado de Sampieri R. y Mendoza C. [65].

Donde:

G: Grupo de Estudio (Imágenes y videos de escenas con armas)

X: Variable Independiente (Sistema de alerta y videovigilancia utilizando Raspberry Pi 4B)

M: Medición (Métricas de evaluación del sistema)

4.2. Método de investigación

Cuantitativo, esto es, porque se describen las variables existentes de una determinada muestra o población. La finalidad de este método es la de evaluar y analizar posteriormente los datos, para cumplir los requerimientos de la presente investigación a través de métricas de evaluación.

4.2.1. Diseño general del sistema propuesto

El diagrama de funcionamiento propuesto del sistema para la detección y alerta de armas se muestra en la Fig. 20. El sistema analiza continuamente los entornos videovigilados donde se encuentre instalado, los cuales pueden ser avenidas, calles, transporte público, oficinas, negocios, colegios, hogares, etc., mediante un módulo de cámara instalado en el Raspberry Pi 4B. Luego, esta imagen es analizada empleando un modelo de detección de objetos, que especifica también el tipo de arma. El modelo valida si existe un arma dentro del entorno videovigilado, y, de encontrarse algún tipo de arma en específico, el sistema envía una notificación al aplicativo de Telegram, con el fin de que pueda notificar a las autoridades competentes para que puedan apersonarse a la escena del delito o crimen.

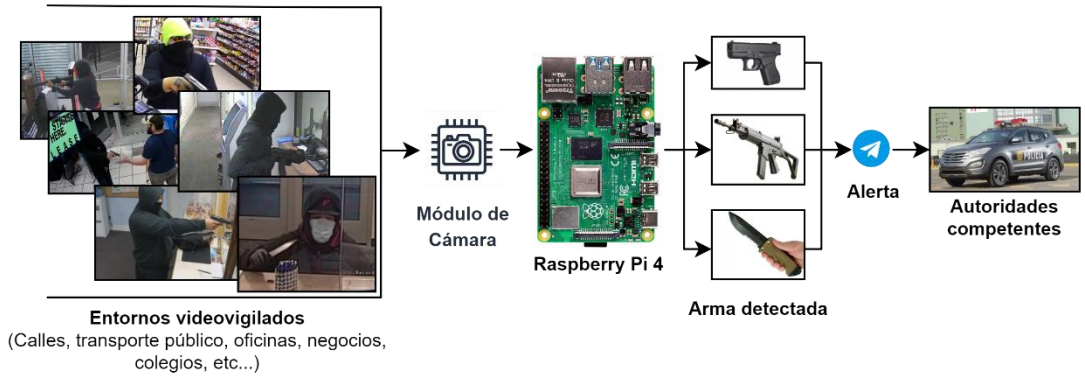


Figura 20. Diagrama de funcionamiento del sistema propuesto.

Fuente: Elaboración propia.

4.2.2. Etapas del sistema

4.2.2.1. Selección de los tipos de armas a detectar

De acuerdo con lo visto en el marco teórico, se puede inferir que los delitos pueden efectuarse tanto con armas de fuego como armas punzocortantes. Por lo que se consideraron como tipo de armas de fuego a clasificar pistolas y revólveres dentro de la categoría de armas de corto alcance (AFCA); rifles, ametralladoras y armas de francotirador en la categoría de armas de largo alcance (AFLA); y cuchillos, navajas en la categoría de armas punzocortantes

(APC). Esto con el fin de tener variedad para la detección de armas en los entornos videovigilados, así como el de alertar sobre con qué tipo de amenaza se enfrentarían las autoridades al momento de ser notificadas.

4.2.2.2. Selección del módulo de cámara

Los módulos de cámara elegidos que pueden ser utilizados con Rasberry Pi se detallan en la Tabla 2, los criterios para la selección de la cámara incluyeron resolución de la imagen, cantidad de megapíxeles, sensibilidad a la luz, compatibilidad con el puerto CSI y el precio. Se descartaron los modelos NOIR (Sin filtro de luz Infrarrojos) debido a que se priorizó la detección en ambientes diurnos e iluminados, también se descartaron las cámaras que utilizan interfaz USB para su conexión debido a que se requiere que exista una menor latencia y una alta tasa de transferencia para el envío de las imágenes. Sin embargo, esto sí lo proporcionan las cámaras con interfaz CSI ya que permite una conexión directa desde el módulo de la cámara hacia el procesador de la Raspberry. Otra desventaja, es el tamaño de las cámaras USB, ya que aumenta ligeramente la complejidad al momento de integrar funcionalmente el sistema en entornos de videovigilancia, por lo que la utilización de módulos de cámaras con interfaz CSI hacen más compacto el sistema en comparación con las cámaras USB.

Tabla 2. Comparación de módulos de cámaras para RPI 4.

Criterio	HQ CAMERA	RPI CAM V3	ARDUCAM V3
Sensor	IMX477R	IMX708	IMX519
Resolución	4056 x 3040	4608 x 2592	4656 × 3496
Megapíxeles	12.3 MP	12 MP	16MP
CSI	SI	SI	SI
Velocidad de fotogramas de video	1080p@30 fps, 720p@60 fps, 640x480@60/90 fps	1080p@50 fps, 720p@100 fps, 480p@120 fps	4K@30 fps, 1080p@30 fps, 720p@60 fps
Dimensiones (mm)	38x38x18.58	25x24x11.5	25x23.86x10.67
Características adicionales	Lentes adicionales, IR-CUT	PDAF, CDAF, High SNR	PDAF, CDAF
Precio (soles)	S/350.00	S/140.00	S/180.00

Fuente: Elaboración propia.

La cámara elegida fue el módulo IMX519 (ver Fig. 21), debido a que posee un equilibrio de características para la aplicación de esta investigación, incluyendo el costo y la calidad y también el autofoco, permitiendo que la cámara ajuste automáticamente la distancia focal cuando exista movimiento de personas y objetos, logrando obtener una imagen nítida, esencial para aplicaciones de visión por computadora y detección de objetos.

Figura 21. Sensor de cámara IMX519.

Fuente: Extraído de [66].

4.2.2.3. Selección del tipo de notificación

Actualmente se disponen de diferentes herramientas y tecnologías para poder notificar sobre los posibles eventos identificados por el sistema, en la Tabla 3 se puede visualizar una comparativa realizada de diferentes tipos de notificación.

Tabla 3. Comparativa de diferentes tipos de notificación

Característica	Telegram	Whatsapp	Correo electrónico	SMS	Llamadas de Voz
Protocolo de Comunicación	API propia	API propia	SMTP, IMAP, POP3	GSM	Red de telefonía
Tipo de Mensaje	Texto, Multimedia	Texto, Multimedia	Texto, Multimedia	Texto	Voz, texto
Comunicación Asíncrona	Sí	Sí	Sí	Sí	No
Creación de bots	Alta	Limitada	No	No	No
Interfaz de Usuario	Fácil e Intuitiva	Fácil e Intuitiva	Variable según el cliente de correo	Limitada	Variable según dispositivo
Almacenamiento de mensajes	Ilimitado	Ilimitado	Limitado	No	No

Fuente: Elaboración propia

Ante lo mostrado en la Tabla 3, se puede inferir que las características de las notificaciones por aplicaciones móviles en la nube predominan frente a las convencionales como correo electrónico, mensajes de texto (SMS) o llamadas de voz.

Para los requerimientos de la investigación, se priorizó el envío de archivos multimedia (texto e imágenes), por lo que las notificaciones convencionales quedaron descartadas. El correo electrónico tiene la desventaja de depender del almacenamiento de acuerdo con el proveedor de servicios de correo electrónico a utilizar (Gmail, Outlook, Yahoo!, iCloud, etc..).

Por otra parte, las notificaciones por mensajes de texto y llamadas de voz no se almacenan generalmente de forma permanente en servidores externos, sino dentro del mismo dispositivo, por lo que dificulta su accesibilidad a través de otras plataformas pudiendo ponerse en riesgo la información almacenada en caso el dispositivo se extravíe. Otro rasgo importante que considerar es que, para las llamadas de voz y mensajes de texto a pesar de poder enviarse rápidamente, estos pueden verse afectados por situaciones donde exista congestión en la red móvil local.

En contraste, las aplicaciones móviles mencionadas en la tabla comparativa tienen la ventaja de almacenar la información en la nube de manera ilimitada, y también se podrían acceder a ello desde cualquier dispositivo y de manera simultánea, aunque también un punto en concreto a considerar es que su desempeño y rapidez se ven influenciados en gran medida por la velocidad de la conexión a internet que posean, así como la ubicación geográfica (país) donde se encuentre el dispositivo, que puede limitar el acceso a estas.

Ante las consideraciones anteriores presentadas, la notificación utilizada fue el aplicativo de Telegram, se consideró utilizar esta aplicación frente a las demás existentes como Whatsapp u otras plataformas como correo electrónico, SMS o llamadas de voz, debido a que Telegram permite la creación de Bots personalizados para las necesidades específicas del proyecto. Estos Bots también pueden interactuar con los usuarios para realizar determinadas acciones a través de comandos y mensajes, el cual permite realizarse también

con diferentes personas y al mismo tiempo. De igual importancia, permite almacenar los mensajes de forma permanente, pudiendo acceder a ellos sin la necesidad de realizar copias de seguridad de los archivos que se envíen dentro aplicativo, con el fin de que exista un historial de detecciones que sirvan como evidencia para futuras acciones legales.

Creación y configuración del Bot para las notificaciones en Telegram

Para crear y configurar el Bot, ingresamos al aplicativo de Telegram y buscamos la herramienta de BotFather. La interfaz de creación de Bots de Telegram se muestra en la Fig. 22.

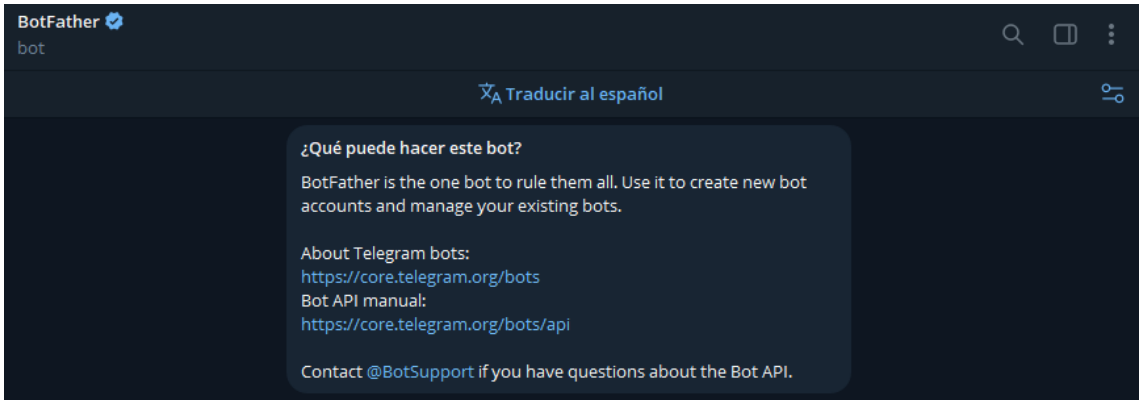


Figura 22. Interfaz del BotFather de Telegram.

Fuente: Elaboración propia.

Luego ingresamos los comandos para nombrar al bot a crear, en este caso **RPI4BAlertUNACbot,** y con ello, se nos brindará el nombre del bot y un Token que nos permitirá acceder luego desde la Raspberry, mediante la librería telepot, hacia el Bot para realizar el envío de las notificaciones. El mensaje que se nos retornará de BotFather luego de crear el bot se visualiza en la Fig. 23. La habilitación de la conexión a internet fue mediante el puerto gigabit ethernet que posee la Raspberry Pi 4B, permitiendo reducir costos de implementación con otros sensores o módulos que permitan su conexión a internet como por ejemplo a través de módulos con ranura SIM.

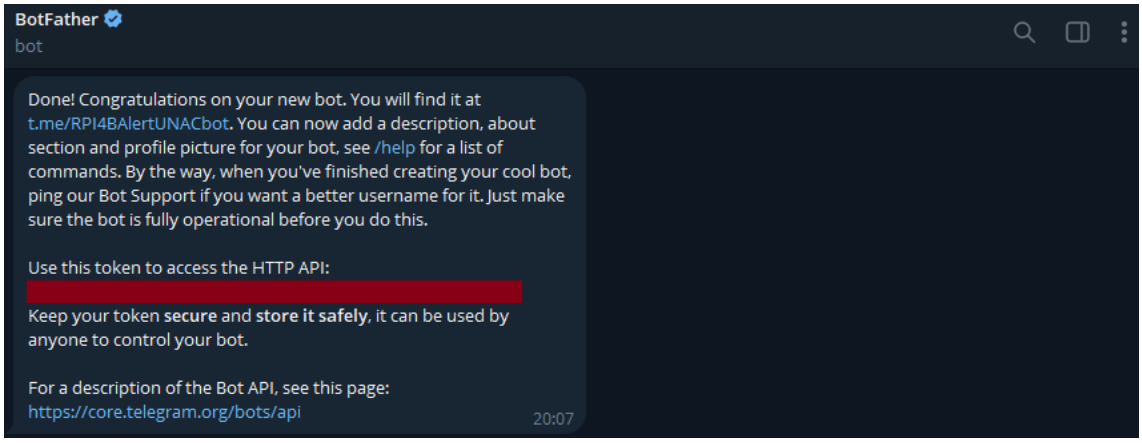


Figura 23. Mensaje de confirmación de creación del Bot.

Fuente: Elaboración propia.

4.2.2.4. Selección del modelo de detección

Dentro del abanico de modelos de detección YOLO existentes, se utilizó la versión YOLOv8, lanzada en enero de 2023 por Ultralytics, misma empresa que realizó también YOLOv5. En la Fig. 24, YOLOv8 se muestra como un modelo eficiente, en contraste con versiones anteriores de YOLO, logrando una mejora destacable en cuestión de precisión y velocidad detección [67].

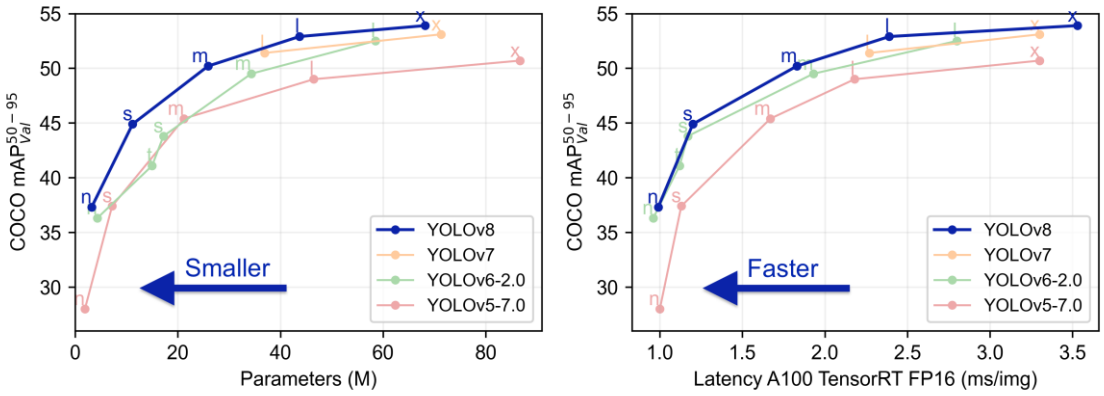


Figura 24. Comparación del modelo YOLOv8 con versiones anteriores.

Fuente: Extraído de [68].

Esta versión, tiene una precisión más alta, al utilizar modelos más pequeños y rápidos como YOLOv8n, permitiendo ejecutarse en dispositivos con limitada capacidad computacional [69], [70]. YOLOv8 además dispone de cinco modelos

(YOLOv8x, YOLOv8l, YOLOv8m, YOLOv8s y YOLOv8n), que pueden utilizarse para tareas de clasificación, detección y segmentación, siendo YOLOv8x el más preciso, alcanzando un mAP50-95 de 53.9% sobre imágenes con un tamaño de 640 píxeles y utilizando una tarjeta gráfica NVIDIA A100 con TensorRT [68], tal como se muestra en la Tabla 4.

Tabla 4. Precisión de los modelos de YOLOv8 con el set de datos de COCO.

Model	size (pixels)	mAP^{val} 50-95	Speed CPU ONNX (ms)	Speed A100 TensorRT (ms)	params (M)	FLOPs (B)
YOLOv8n	640	37.3	80.4	0.99	3.2	8.7
YOLOv8s	640	44.9	128.4	1.2	11.2	28.6
YOLOv8m	640	50.2	234.7	1.83	25.9	78.9
YOLOv8l	640	52.9	375.2	2.39	43.7	165.2
YOLOv8x	640	53.9	479.1	3.53	68.2	257.8

Fuente: Extraído de [68].

Ante lo visto anteriormente con las comparativas hechas de desempeño que posee frente a otros modelos de YOLO, se probaron diferentes configuraciones de YOLOv8, tomando a YOLOv8n y YOLOV8s como modelos de detección base a los cuales se les aplicó la técnica de transferencia de aprendizaje. Se eligieron estos modelos con la finalidad de que puedan ser ejecutados en la Raspberry Pi 4B a un bajo costo computacional.

4.2.2.5. Establecimiento del entorno de entrenamiento en Google Colab

Se configuró la plataforma para realizar el entrenamiento del modelo de detección de objetos YOLOv8 en Google Colab, esta nos permite ejecutar código en lenguaje Python y también por la facilidad de utilizar gratuitamente y de forma limitada una Tarjeta Gráfica NVIDIA T4.

Para tener el tiempo suficiente que demanda realizar el entrenamiento del modelo y para las pruebas, se adquirieron 100 unidades de procesamiento ofrecidos en el servicio de Google Colab.

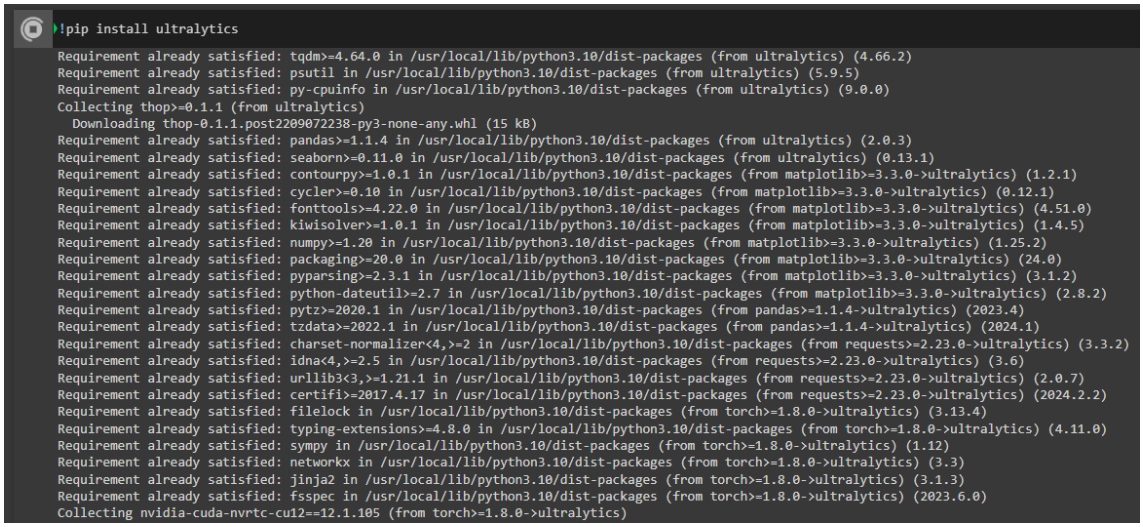
Primero, se subieron a una carpeta de Google Drive las imágenes con los datos de entrenamiento y validación en un formato zip, y se descomprimió el contenido

siguiendo la siguiente estructura de ubicación de carpetas y archivos, como se muestra en la Fig. 25.

Figura 25. Estructura jerárquica de ubicación de los archivos de trabajo.

Fuente: Elaboración propia.

Una vez realizado, en el entorno virtual de Google Colab, se instalaron las librerías y dependencias de Ultralytics, el cual nos permite utilizar el modelo de YOLOv8 y sus modelos respectivos. En la Fig. 26 se muestra el proceso de instalación de la librería.



Instalación de la librería ultralytics

Fuente: Elaboración propia

Luego, se editó el archivo ‘config.yaml’, donde se especificó la ubicación de las imágenes y etiquetas para el entrenamiento del modelo YOLOv8n y YOLOv8s,

así como la cantidad de clases a detectar, el contenido del archivo 'config.yaml' se puede observar en la Fig. 26, con las clases a detectar.

```
train: /content/drive/MyDrive/COLAB/data/images/train
val: /content/drive/MyDrive/COLAB/data/images/val

# Classes
names:
  0: AFCA
  1: AFLA
  2: APC
```

Figura 26. Contenido del archivo 'config.yaml'

Fuente: Elaboración propia

Al haber realizado esta configuración previa, se procedió a establecer el entorno para el entrenamiento. La tarjeta gráfica utilizada dentro del entorno de Google Colab fue la NVIDIA Tesla V100 con 16 GB de memoria VRAM, dentro de los parámetros de entrenamiento se consideraron los siguientes:

Tabla 5. Comparación de parámetros de los modelos entrenados.

Parámetro/Modelo	YOLOv8n-I	YOLOv8n-II	YOLOv8s
Cantidad de épocas	200	100	100
Optimizador	AdamW	AdamW	AdamW
Tasa de aprendizaje	0.001	0.001429	0.001
Tamaño de lote	32	320	64
Imgsz	640	320	640
Cantidad de imágenes utilizadas	1500	4800	4800

Fuente: Elaboración propia

Una vez configurado el entorno virtual, se eligió la cantidad de imágenes que contendría cada modelo, la obtención de las imágenes y las técnicas utilizadas para obtener una mayor cantidad de información se detallan en el apartado 4.5.

La Fig. 27 muestra la interfaz de Google Colab realizando el entrenamiento del modelo YOLOv8 con uno de los parámetros especificados en la Tabla 5.

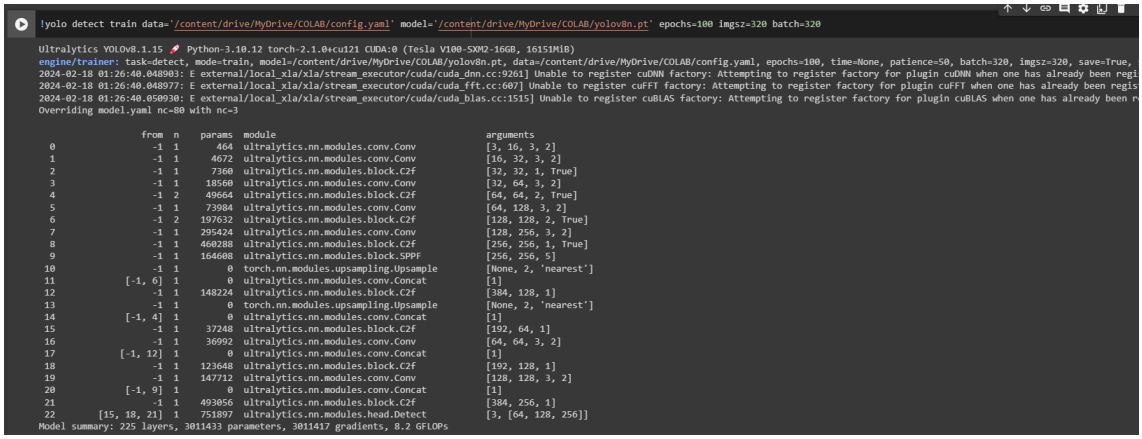


Figura 27. Interfaz de Google Colab durante el entrenamiento.

Fuente: Extraído de Google Colab.

Técnicas de optimización

Una vez realizado el entrenamiento, se obtuvieron los modelos entrenados con los pesos ajustados durante la transferencia de aprendizaje con las tres categorías (APC, AFCA Y AFLA).

Los modelos conseguidos funcionan con operaciones de 32bits de punto flotante, por lo que para poder ejecutar el modelo con rapidez y se reduzca la capacidad computacional necesaria para su despliegue en una Raspberry Pi 4B, se aplicó la técnica de cuantización, esta técnica redujo las operaciones a 8 bits. Los modelos obtenidos denominados como “best.pt” automáticamente por YOLOv8 para cada entrenamiento, se convirtieron a modelos de TensorFlow Lite.

4.2.2.6. Establecimiento del entorno en la Raspberry Pi 4B

La Raspberry Pi 4B utilizada tiene una memoria RAM de 8GB, integra dispositivos de red inalámbrica (WLAN) y un puerto LAN para la transmisión de información a Internet. El modelo del sistema operativo elegido para la Raspberry Pi 4B fue el Raspberry Pi OS Bullseye Legacy 64-bit. La instalación de este S.O. fue a través de la herramienta Raspberry Pi Imager en una tarjeta microSD de 128GB. El uso de la herramienta se muestra en la Fig. 28.

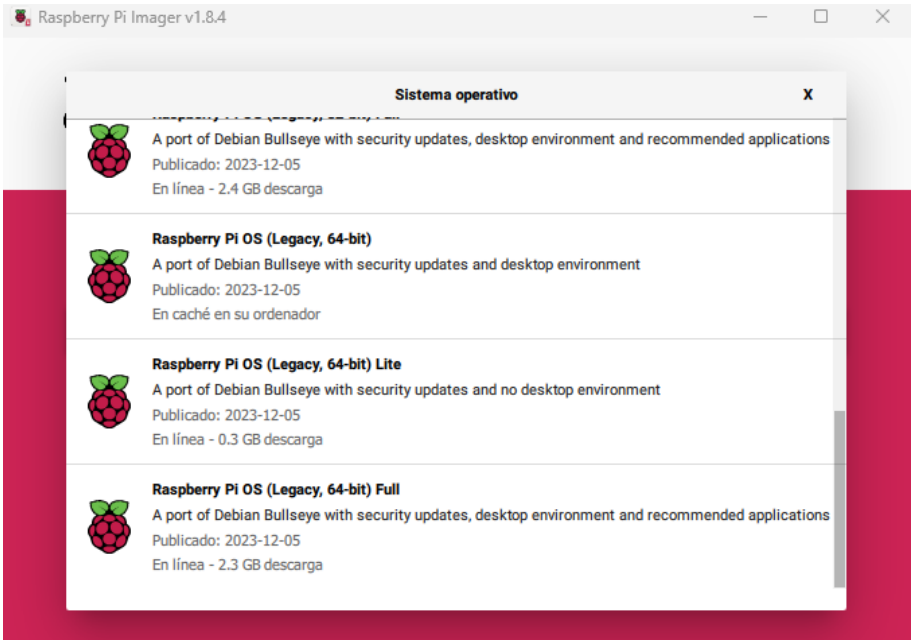


Figura 28. Interfaz del Raspberry Pi Imager

Fuente: Elaboración propia.

Luego de haber instalado el sistema operativo, se procedió a instalar las siguientes librerías de opencv, ultralytics, pytorch, tensorflow, tensorboard y picamera2, esta última para configurar el módulo de cámara IMX519 elegido. La Raspberry Pi 4B fue manipulada desde una computadora utilizando RealVNC, la Fig. 29 muestra la interfaz una vez instalada las librerías necesarias.

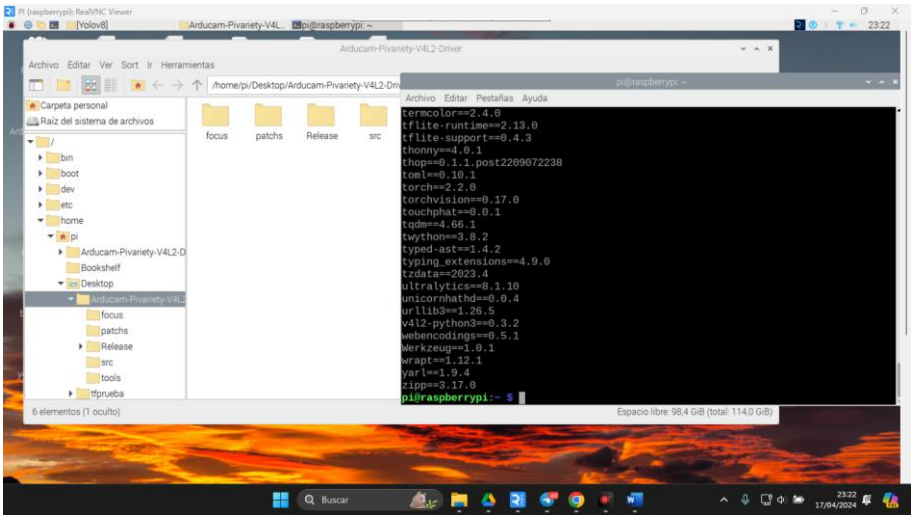


Figura 29. Interfaz del sistema operativo con las librerías instaladas.

Fuente: Elaboración propia.

4.3. Población y muestra

La población estuvo conformada por imágenes y fotogramas de videos que contienen armas de fuego (pistolas, revólveres, rifles) y armas punzocortantes (cuchillos y navajas), la muestra estuvo conformada por 4620 imágenes recopiladas de diferentes bases de datos, y luego aumentadas a 4800 utilizando técnicas de aumento de datos para el entrenamiento del modelo de detección YOLOv8, esta técnica se detalla en el apartado 4.5.

4.4. Lugar de estudio y periodo desarrollado

La investigación se realizó en la región Callao y el periodo comprendido para la ejecución y desarrollo de la investigación fue de 12 meses.

4.5. Técnicas e instrumentos para la recolección de la información

Adquisición del conjunto de imágenes de armas para el entrenamiento

Se recopilaron imágenes de diferentes fuentes, como fragmentos de videos de YouTube, datasets de la Internet Movie Firearms Database (IMFDB) [71], Datasets de la IEEE Dataport [72], [73] y de acceso abierto [74], se seleccionaron como imágenes para el entrenamiento a aquellas donde se mostraron armas de fuego de corto y largo alcance, así como armas punzocortantes.

Las imágenes extraídas de las bases de datos mencionadas fueron preseleccionadas con el criterio de que posean una dimensión mayor a 500 píxeles en ancho o en alto con el fin de obtener una buena calidad de éstas para el entrenamiento del modelo de detección de objetos. El código mostrado en la Fig. 30 fue utilizado para realizar este filtrado del banco de imágenes.

```python
import cv2
import os
from pathlib import Path, PurePath

i = 0

for file in os.listdir():
    print(file)

    if file.endswith('.py'):
        break
    im = cv2.imread(file)
    h = im.shape[0]
    w = im.shape[1]

    if h > 499 or w > 499:
        print("OK")
    else:
        os.remove(file)
```

Figura 30. Código implementado para filtrar las imágenes con una cantidad de píxeles mayor a 500 en ancho o alto.

Fuente: Elaboración propia

Las imágenes se etiquetaron en tres categorías: Armas de Fuego de Corto Alcance: AFCA, Armas de Fuego de Largo Alcance: AFLA, y Armas Punzocortantes: APC.

Primera categoría: Armas de fuego de Corto Alcance (AFCA)

Las armas de fuego de corto alcance comprendieron a pistolas y revólveres. La muestra de la primera categoría estuvo conformada por 1600 imágenes, una muestra de ellas se puede observar en la Fig. 31.

Figura 31. Recopilación de las imágenes de la categoría AFCA.

Fuente: Elaboración propia.

Segunda categoría: Armas de Fuego de Largo Alcance (AFLA)

Las armas de fuego de largo alcance comprendieron rifles de asalto modernos, escopetas y armas de francotirador. La muestra de la segunda categoría estuvo compuesta por 1420 imágenes, un fragmento de ellas se puede observar en la Fig. 32.

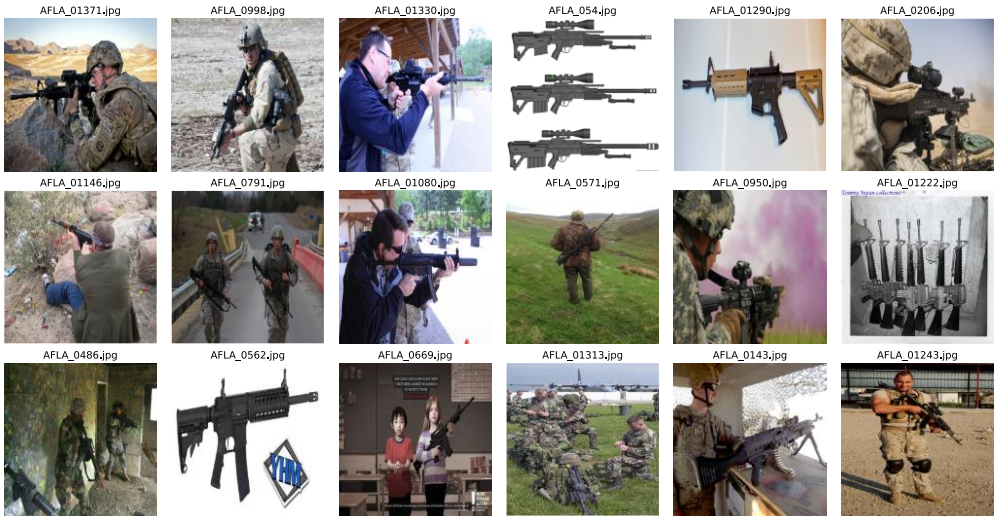


Figura 32. Recopilación de armas de la categoría AFLA.

Fuente: Elaboración propia.

Tercera categoría: Armas Punzocortantes (APC)

Las armas punzocortantes comprendieron a cuchillos y navajas. El muestreo de la tercera categoría tuvo 1600 imágenes, y un fragmento de ellas se puede observar en la Fig. 33.

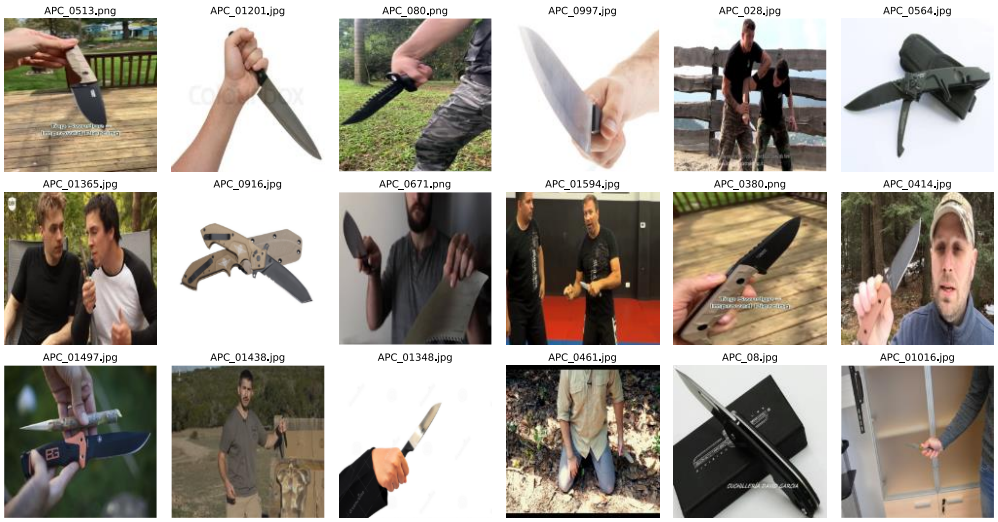


Figura 33. Recopilación de imágenes de la categoría APC.

Fuente: Elaboración propia.

De las categorías mencionadas anteriormente, se obtuvieron con facilidad imágenes de la clase AFCA y APC, teniendo dificultad de encontrar imágenes de calidad para la categoría AFLA, por lo que se procedió a emplear la técnica denominada “aumento de datos”.

Data Augmentation o aumento de datos

Para tener más variedad del set de datos elaborado de la categoría AFLA, se procedió a utilizar la técnica de data augmentation o “aumento de datos”, esto es, aplicar transformaciones sobre el set de datos existentes para incrementar la cantidad de información disponible [75].

Para ello se utilizó la librería de Keras, el código mostrado en la Fig. 34 ejemplifica el uso de la técnica de data augmentation para las imágenes del set de datos de la categoría AFLA.

```
from keras.preprocessing.image import ImageDataGenerator
from skimage import io

datagen = ImageDataGenerator(
    rotation_range = 45,
    width_shift_range = 0.1, height_shift_range = 0.1,
    shear_range = 0.2, zoom_range = 0.2,
    horizontal_flip = True, vertical_flip = True,
    fill_mode = 'constant', cval = 125
)

x = io.imread('test/AFLA_0238.jpg')
x = x.reshape((1, ) + x.shape)
i = 0

for batch in datagen.flow(x, batch_size = 16, save_to_dir = 'test/augmented',
                          save_prefix = 'aug', save_format = 'jpg'):
    i += 1
    if i > 20:
        break
```

Figura 34. Código utilizado para la técnica de data augmentation.

Fuente: Elaboración propia.

Con ello se aplicaron rotaciones sobre las imágenes existentes (desde 0° a 45°), se aplicaron variaciones corrimiento de la imagen vertical y horizontal en un 10%, perspectiva horizontal y vertical en 10%, aumento y alejamiento de la imagen en 10%, giro horizontal y vertical de la imagen, los cuales fueron añadidas a la categoría AFLA y permitieron tener las 1600 imágenes necesarias para equipar el set de datos de cada categoría, la aplicación de la técnica mencionada se aprecia en la Fig. 35.

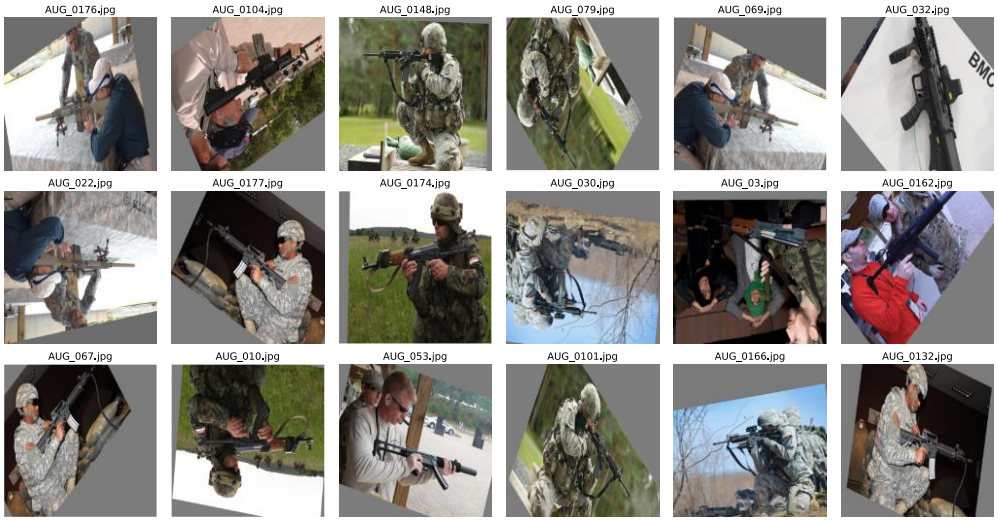


Figura 35. Aplicación de Data Augmentation para la Categoría AFLA.

Fuente: Elaboración propia utilizando ImageDataGenerator de Keras.

Selección de imágenes para entrenamiento y validación del modelo

Se consideró realizar tres pruebas sobre modelos base (YOLOv8n y YOLOv8s) para determinar cuál de ellos tuvo mejor rendimiento al momento de ser implementada en la Raspberry Pi 4B. Los modelos de prueba se denominaron YOLOv8n-I, YOLOv8n-II y YOLOv8s-I. Para el modelo YOLOv8n-I se consideraron 1500 imágenes (divididas en 1200 de entrenamiento y 300 de validación) tomando la proporción de 80/20. Para YOLOv8n-II y YOLOv8s-I, se realizó la partición del set de datos considerando un 85% para el entrenamiento del modelo y un 15% para validación, esto debido a la grande cantidad de imágenes. Teniendo respectivamente: 4080 imágenes para entrenamiento (1360 de la categoría AFCA, 1360 de la categoría AFLA y 1360 de la categoría APC) y 720 imágenes para validación (240 de la categoría AFCA, 240 de la categoría AFLA y 240 de la categoría APC).

En la Fig. 36 se muestra la cantidad de imágenes utilizadas para el entrenamiento y validación de cada variante de los modelos escogidos de YOLOv8n-II y YOLOv8s-I.

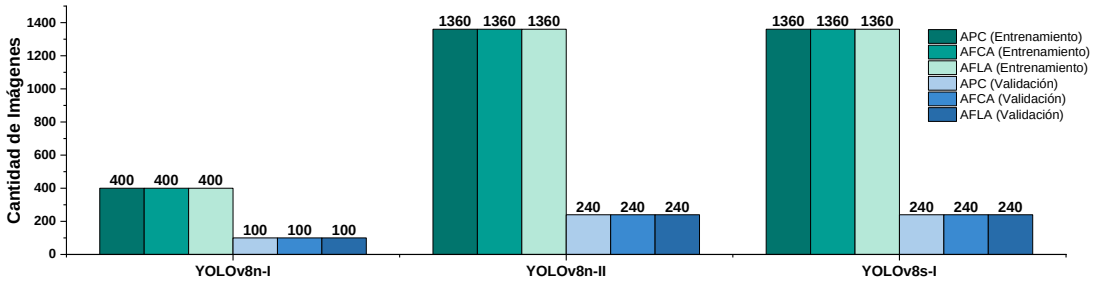


Figura 36. Imágenes de entrenamiento por cada modelo YOLOv8.

Fuente: Elaboración propia

Etiquetado de las imágenes

Para el etiquetado de las imágenes, se utilizó la herramienta de LabelImg [76], debido a su facilidad de uso ya que posee una interfaz gráfica. Las etiquetas utilizadas fueron AFCA, AFLA y APC. El uso de la herramienta para el etiquetado de los datos se muestra en la Fig. 37.

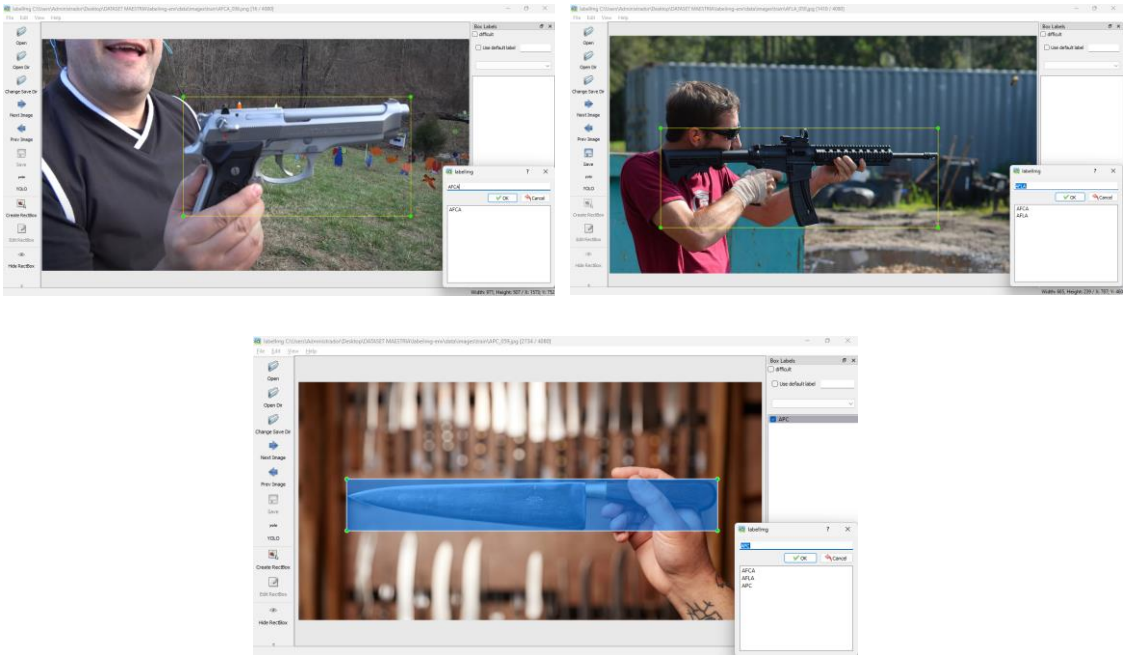


Figura 37. Proceso de etiquetado de las imágenes utilizando LabelImg.

Fuente: Elaboración propia utilizando LabelImg.

4.6. Análisis y procesamiento de datos

Las siguientes métricas se utilizaron para evaluar el rendimiento general del modelo, los valores que se obtendrán de las métricas directamente se recopilarán después de la ejecución del entrenamiento del modelo de YOLOv8.

4.6.1. Métricas de desempeño

Precisión (P)

Esta métrica es un valor porcentual que se calcula utilizando el VP (verdadero positivo) y FP (falso positivo), la ecuación se calcula como la cantidad de verdaderos positivos entre el número total de predicciones positivas [77]:

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

Recall o sensibilidad (R)

Esta métrica es un valor porcentual que se calcula utilizando el VP (verdadero positivo) y el FN (falso negativo), la ecuación se calcula como la cantidad de verdaderos positivos entre el número total de instancias positivas [77]:

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

F1-Score

Esta métrica se calcula como la media armónica que considera tanto la Precision y el Recall [77], se calcula de la siguiente manera:

$$F1 = 2 \times \frac{P \times R}{P + R}$$

Precisión promedio media (mAP)

Es una métrica que permite evaluar la efectividad de un modelo de detección de objetos. Esta no solo indica si un modelo de detección de objeto puede detectar múltiples clases de objetos correctamente, sino también con precisión. Este valor se obtiene promediando las AP de todas las clases de objetos presentes en el conjunto de datos [77]. La forma de calcular este valor se da mediante la siguiente ecuación:

$$mAP = \frac{1}{N} \sum_{i=0}^{N} AP_i$$

Donde N es el número total de clases y el AP_i es la precisión promedio (AP) de la i-ésima clase.

Matriz de confusión

Es una herramienta que se utiliza para evaluar el rendimiento de un modelo de clasificación, está diseñada para evaluar la capacidad del modelo para clasificar correctamente objetos de diferentes clases. Muestra los valores de VP, FP, VN Y FN resultantes de las predicciones del modelo en relación con las clases reales dentro del conjunto de datos.

4.7. Aspectos Éticos en Investigación

La realización de la presente investigación se fundamenta en el “Código de ética de investigación” de la Universidad Nacional del Callao, basándose en: la probidad, profesionalismo, transparencia, objetividad, igualdad, compromiso,

honestidad, confidencialidad, independencia, diligencia y dedicación con la finalidad de asegurar el conocimiento, la comprensión y mejora de la condición humana y el desarrollo progresivo de la sociedad.

V. RESULTADOS

5.1. Resultados descriptivos

5.1.1. Cantidad de imágenes recopiladas

En la recopilación de la información se obtuvieron 4800 imágenes de diferentes fuentes, teniendo las clases APC, AFCA Y AFLA, cada una de ellas con 1600 imágenes, la distribución realizada en entrenamiento y validación se puede apreciar en la Fig. 38.

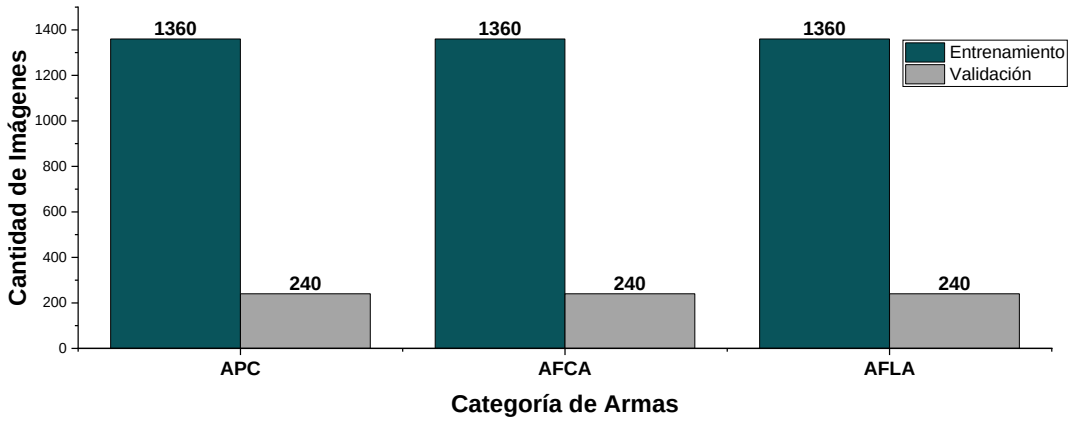


Figura 38. Distribución de las imágenes en entrenamiento y validación.

Fuente: Elaboración propia.

El balanceo de los datos se hizo con el fin de equiparar la información de las clases o categorías y evitar el desarrollo de sesgos al momento de realizar el entrenamiento, pues si existiesen clases con menos información (imágenes), el modelo detectado podría tener un rendimiento menor para determinadas clases.

5.1.2. Tiempos de entrenamiento por modelo

La cantidad de horas de entrenamiento por cada modelo de YOLOv8, se puede observar en la siguiente Fig. 39.

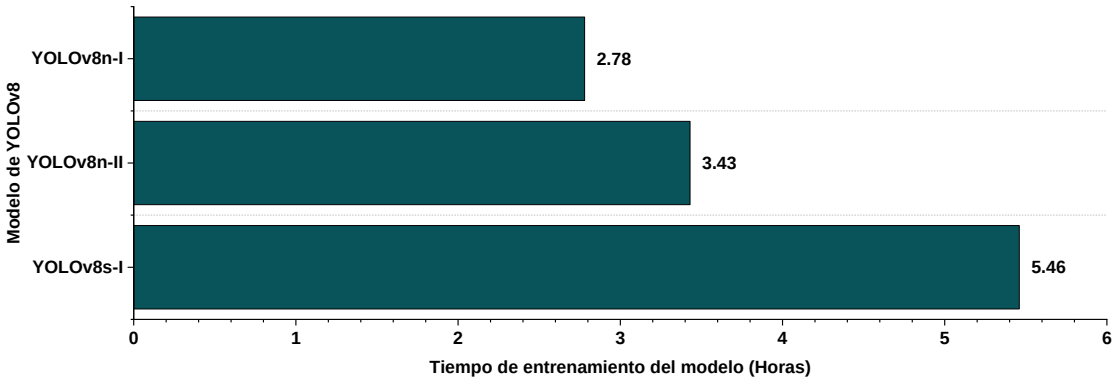


Figura 39. Tiempos de entrenamiento por cada modelo YOLOv8.

Fuente: Elaboración propia

Se visualiza que para el modelo base de YOLOv8s-I, el algoritmo se entrenó en 5.46 horas debido a que tiene una mayor cantidad de pesos en su arquitectura. Para el modelo YOLOv8n-I, se tuvo un tiempo de 2.78 horas de entrenamiento, esto a pesar de que fue entrenado con una menor cantidad de imágenes (1200 de entrenamiento y 300 de validación), debido también al tamaño del batch (lote) de 32 como parámetro. Y para el modelo base de YOLOv8n-II se tuvo un tiempo de 3.43 horas para 100 épocas y 4800 imágenes (4080 de entrenamiento y 720 de validación). La cantidad de tiempo utilizada para el entrenamiento del modelo YOLOv8n-II fue relativamente cercano al del modelo YOLOv8n-I, sugiriendo que la configuración del tamaño del lote, así como el tamaño de la imagen redujeron el tiempo de ejecución para cada época de entrenamiento de este modelo.

5.1.3. Tamaño de los archivos luego de la cuantización de los modelos

A continuación, en la Fig. 40 se muestra el tamaño de los archivos en MB, luego de la aplicación de la técnica de cuantización para que éstos puedan ser ejecutados en la Raspberry Pi 4B. La cuantización se realizó exportando utilizando la librería de Ultralitycs al formato TFLite.

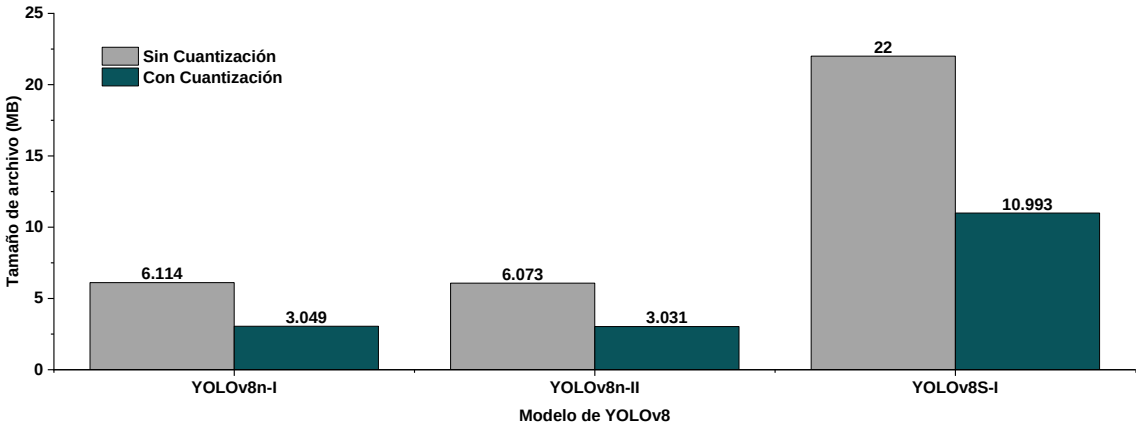


Figura 40. Tamaño de los modelos luego de aplicar cuantización.

Fuente: Elaboración propia.

Los tamaños de los modelos de YOLOv8 utilizados se redujeron en casi un 50% en todos los casos luego de aplicar cuantización, debido a que esta técnica no almacena los pesos en números de 32 bits, sino en un formato de 8 bits enteros.

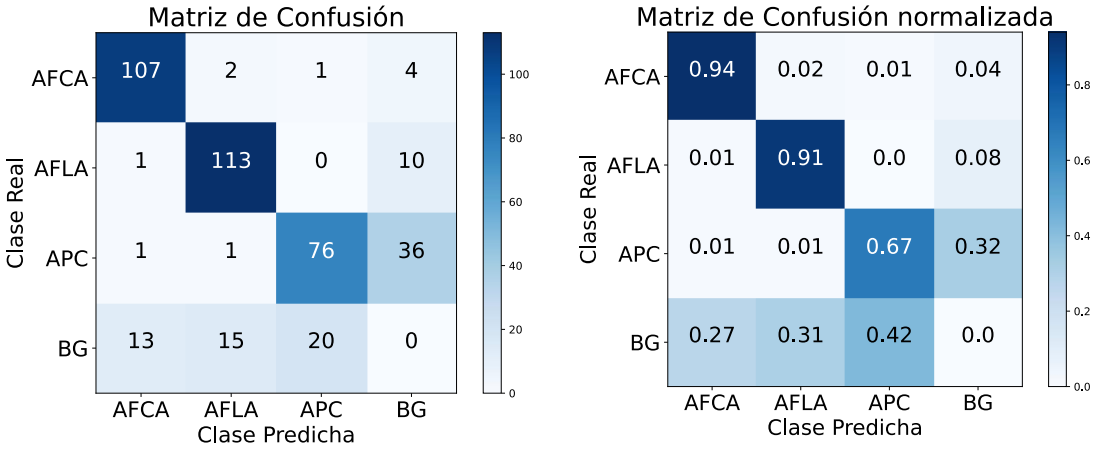
5.1.2. Métricas del entrenamiento del modelo de detección de objetos

Métricas de desempeño por modelo

Luego de haber hecho el entrenamiento utilizando Google Colab y aplicando transferencia de aprendizaje en YOLOv8. Se presentan a continuación los resultados obtenidos considerando las métricas de evaluación, mismas que se recopilaron usando la plataforma de Tensorboard para visualizar el entrenamiento de cada época de los modelos.

Métricas para el modelo YOLOv8n-I

La matriz de confusión del modelo se muestra en la Fig. 41, donde se visualizan los valores de VP, VN, FP Y FN para cada clase. La clase AFCA tuvo un mayor Recall con 94%, mientras que la menor fue la de la clase APC con 67%. Para este modelo con el conjunto de datos de 1500, existe una detección errónea por parte de las clases confundiéndolas con el fondo de la imagen (32% para APC). Esto puede indicar que el modelo no realiza correctamente la inferencia para imágenes de la categoría APC al confundirlas probablemente con objetos similares del entorno dentro de la imagen.



(a) (b)

Figura 41. Matriz de confusión del modelo YOLOv8n-l (a) y su versión normalizada (b).

Fuente: Elaboración propia.

En la Fig. 42 se aprecia las tres categorías para el modelo entrenado YOLOv8n-l, de la cual se observa que la categoría donde tuvo menores valores para realizar una correcta predicción fue para la clase APC (arma punzocortante), teniendo valores para la Precision, Recall y F1-Score de 78.4%, 66.7% y 72% respectivamente. Para la clase AFCA (arma de fuego de corto alcance), se tuvo mejores valores de métricas tanto para la Precisión, Recall y F1-Score, obteniendo 87.7%, 93.9% y 90.7% respectivamente.

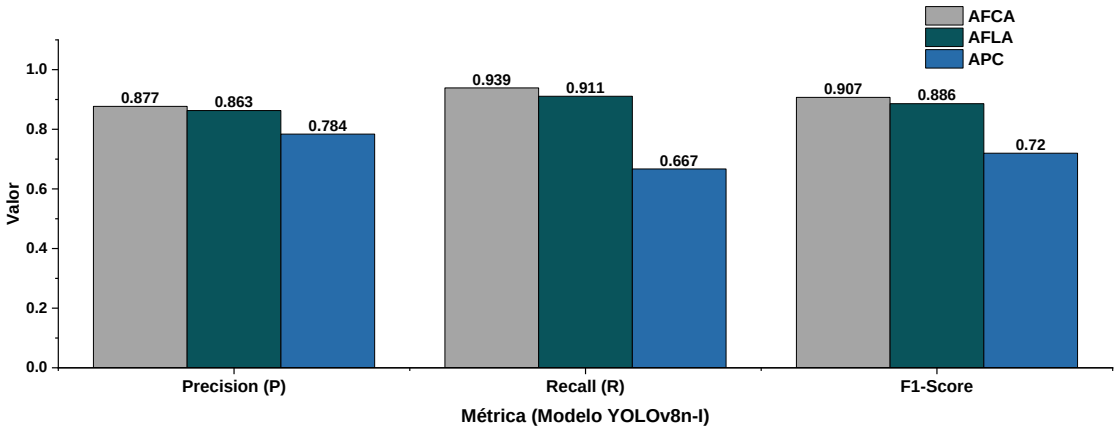
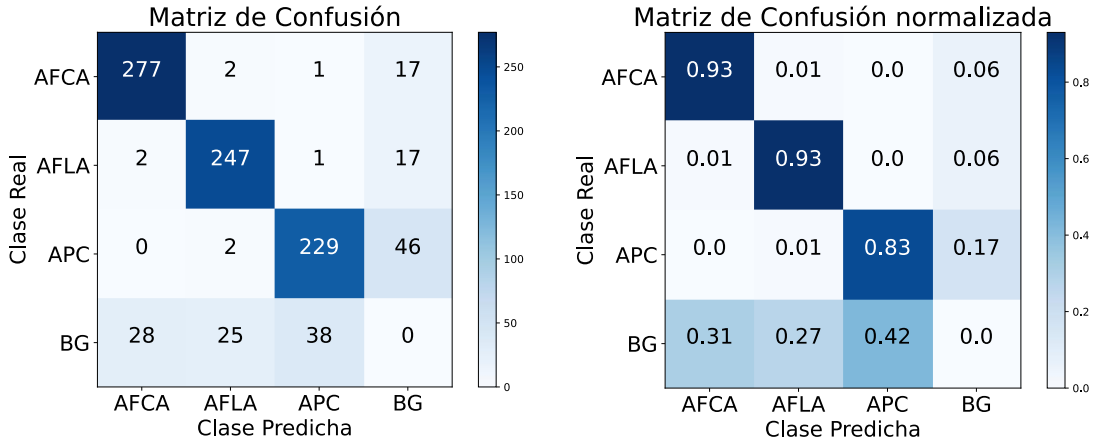


Figura 42. Métricas de desempeño del modelo YOLOv8n-l

Fuente: Elaboración propia.

Métricas del modelo YOLOv8n-II

La matriz de confusión del modelo se muestra en la Fig. 43, donde se visualizan los valores de VP, VN, FP Y FN para cada clase. Las clases AFCA y AFLA tuvieron una mayor tasa de detección con 93%, mientras que la menor fue de la clase APC con 83%. Para este modelo con el conjunto de datos de 4800, existe una detección errónea confundiéndolas con el fondo de la imagen (17% para APC), menor en comparación con el modelo YOLOv8n-I.



(a) (b)

Figura 43. Matriz de confusión del modelo YOLOv8n-II (a), y su versión normalizada (b).

Fuente: Elaboración propia.

En la Fig. 44 se aprecia las tres categorías para el modelo entrenado YOLOv8n-II, de la cual se observa que la categoría donde tuvo menores valores para realizar una correcta predicción fue para la clase APC (arma punzocortante), teniendo valores de Precision, Recall y F1-Score, obteniendo 85.1%, 82.7% y 83.9% respectivamente. Para la clase AFCA (arma de fuego de corto alcance), se tuvo mayores valores de métricas para la Precisión, Recall y F1-Score, obteniendo 90.2%, 93.3% y 91.7% respectivamente. Se observa que la diferencia porcentual entre las clases no es mayor a comparación con el modelo de YOLOv8n-I.

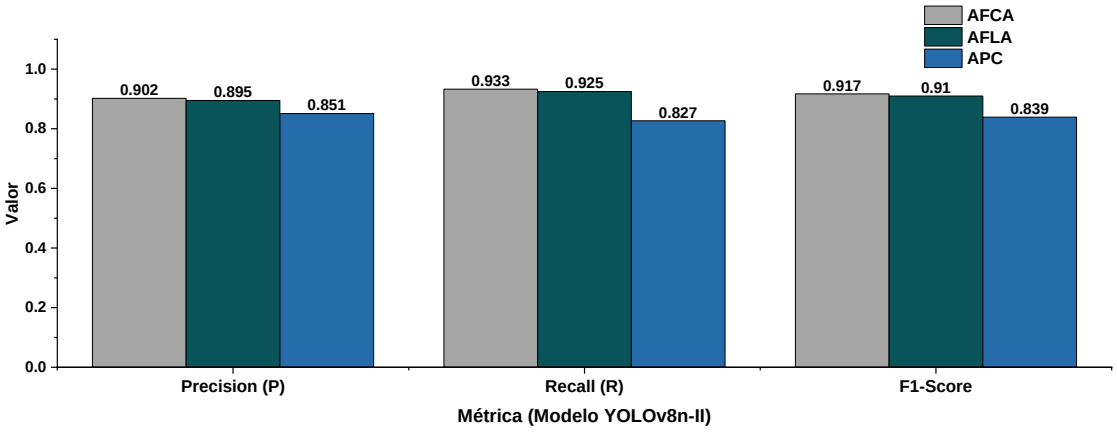
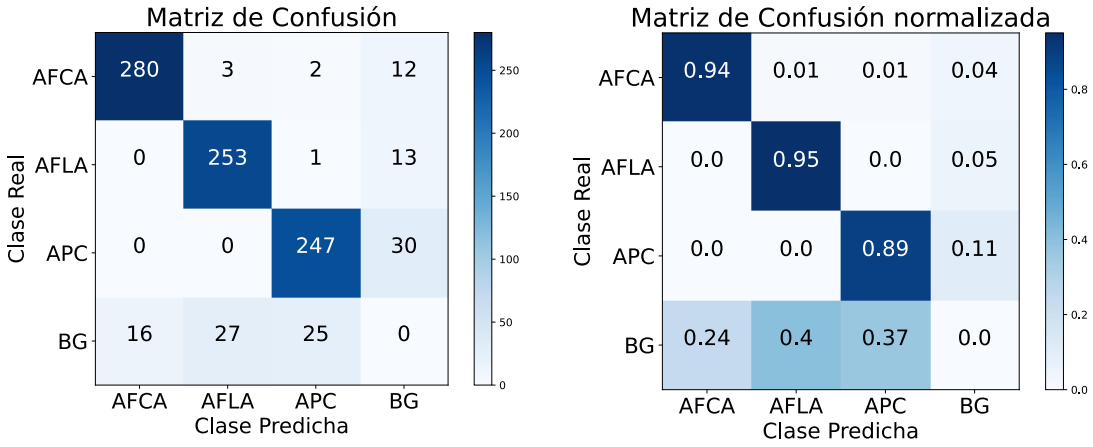


Figura 44. Métricas de desempeño del modelo YOLOv8n-II.

Fuente: Elaboración propia.

Métricas del modelo YOLOv8s-I

La matriz de confusión del modelo se muestra en la Fig. 45, donde se visualizan los valores de VP, VN, FP Y FN para cada clase. La clase AFCA tuvo una mayor tasa de detección con 94%, mientras que la menor fue de la clase APC con 89%. Para este modelo existe una detección errónea mucho menor por parte de las clases, clasificándolas con el fondo de la imagen (11% para APC).



(a) (b)

Figura 45. Matriz de confusión del modelo YOLOv8s-I (a), y su versión normalizada (b).

Fuente: Elaboración propia.

En la Fig. 46 se aprecia las tres categorías para el modelo entrenado YOLOv8s-I, de la cual se observa que la categoría que tuvo menor valor para realizar una

correcta predicción fue para la clase APC (arma punzocortante), teniendo valores de Recall y el F1-Score de 89.2% y 89.5% respectivamente. Para la clase AFCA (arma de fuego de corto alcance), se tuvo valores de métricas para la Precisión, Recall y F1-Score de 94.6%, 94.3% y 94.4%, mejor que el modelo YOLOv8n-II.

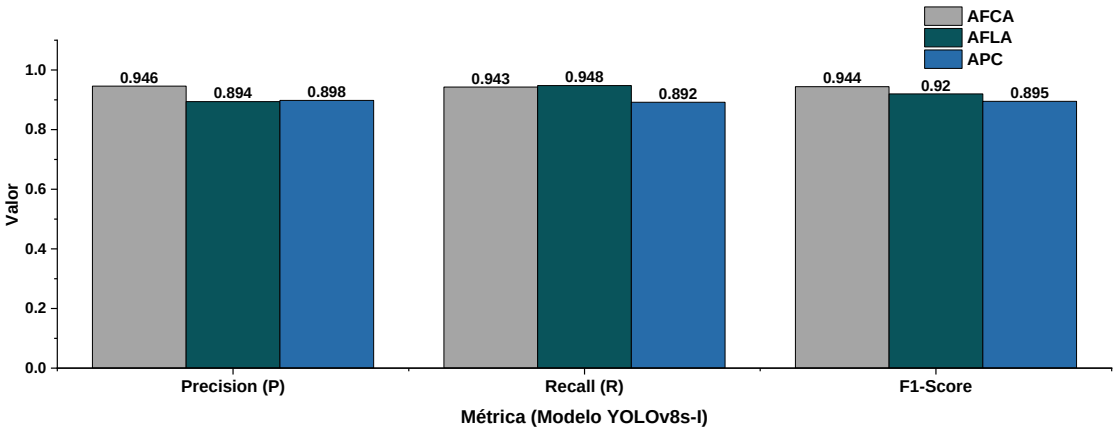


Figura 46. Métricas de desempeño del modelo YOLOv8s-I

Fuente: Elaboración propia.

Comparación de métricas de desempeño entre modelos

Para la clase de AFCA, se muestra en la Fig. 47, las métricas de desempeño de los diferentes algoritmos, teniendo un mayor valor de F1-Score para el modelo YOLOv8s-I de 94.4%, seguido por el modelo YOLOv8n-II con 91.7% y finalmente con menor valor para el modelo YOLOv8n-I con 90.7%.

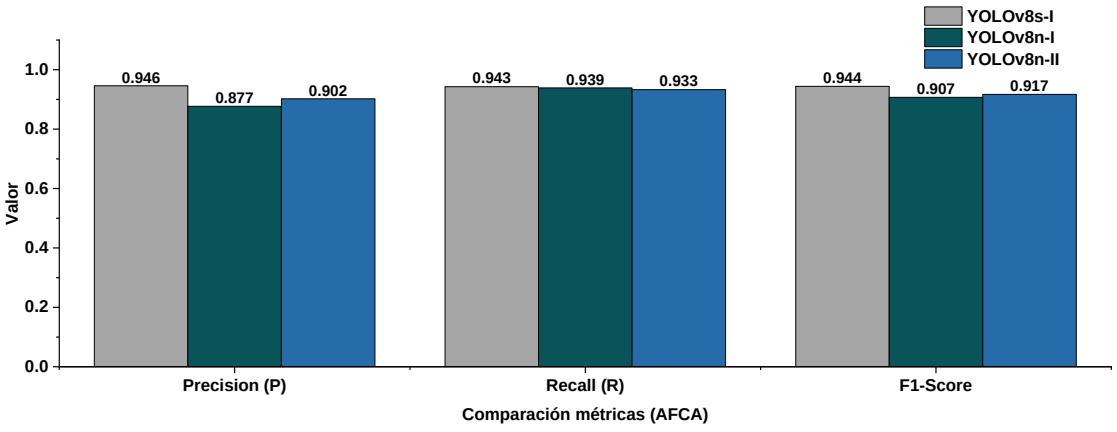


Figura 47. Comparación de métricas de los modelos para la clase AFCA.

Fuente: Elaboración propia.

Para la clase de AFLA, se muestra en la Fig. 48 las métricas de desempeño de los diferentes modelos, teniendo un mayor valor de F1-Score para el modelo YOLOv8s-I de 92%, seguido por el modelo YOLOv8n-II con 91% y finalmente con menor valor para el modelo YOLOv8n-I con 88.6%.

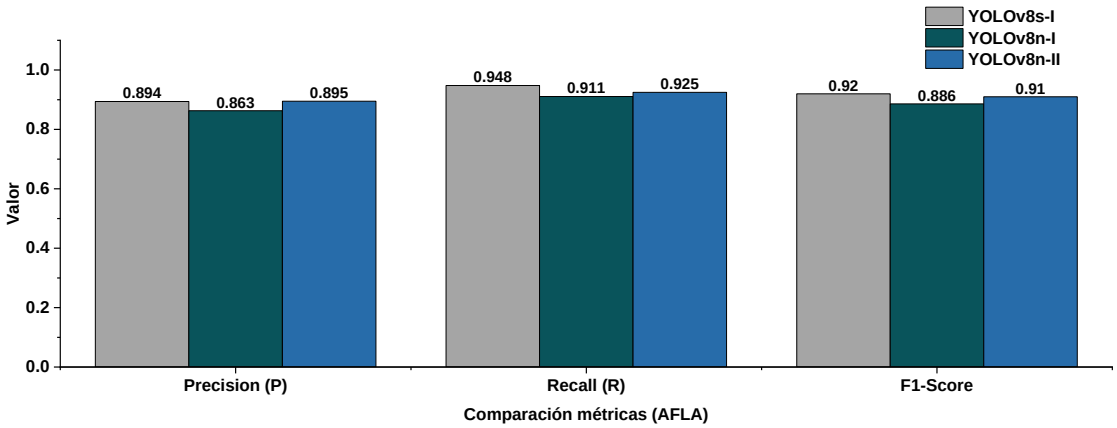


Figura 48. Comparación de métricas de los modelos para la clase AFLA.

Fuente: Elaboración propia.

Para la clase de APC, se muestra en la Fig. 49, las métricas de desempeño de los diferentes algoritmos, teniendo un mayor valor de F1-Score para el modelo YOLOv8s-I de 89.5%, seguido por el modelo YOLOv8n-II con 83.9% y finalmente con menor valor en las métricas para el modelo YOLOv8n-I con 72%.

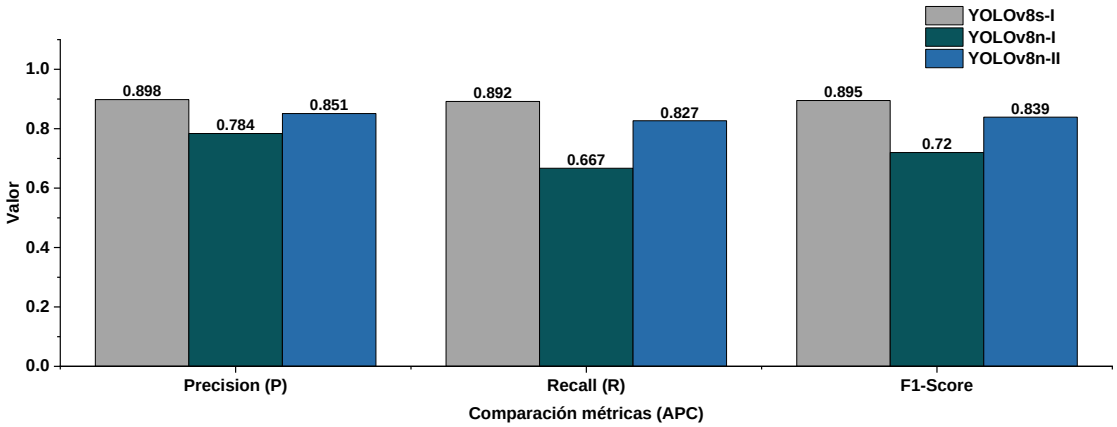


Figura 49. Comparación de métricas de los modelos para la clase APC.

Fuente: Elaboración propia.

Comparación de métricas de desempeño durante el entrenamiento (histórico) de los modelos de detección

mAP50

En la Fig. 50, podemos apreciar la métrica de los modelos entrenados, destacando que, para la métrica de mAP50, el cual considera correcto si existe al menos un 50% de superposición con el objeto real y al mismo tiempo no existe superposición con el fondo en ese mismo porcentaje, los modelos YOLOv8s-I y YOLOv8n-II, fueron los únicos con una precisión superior al 90%, logrando valores finales del 94.5% y 91.7% respectivamente, y para el modelo YOLOv8n-I un 84.6%, siendo este muy inferior a los anteriores.

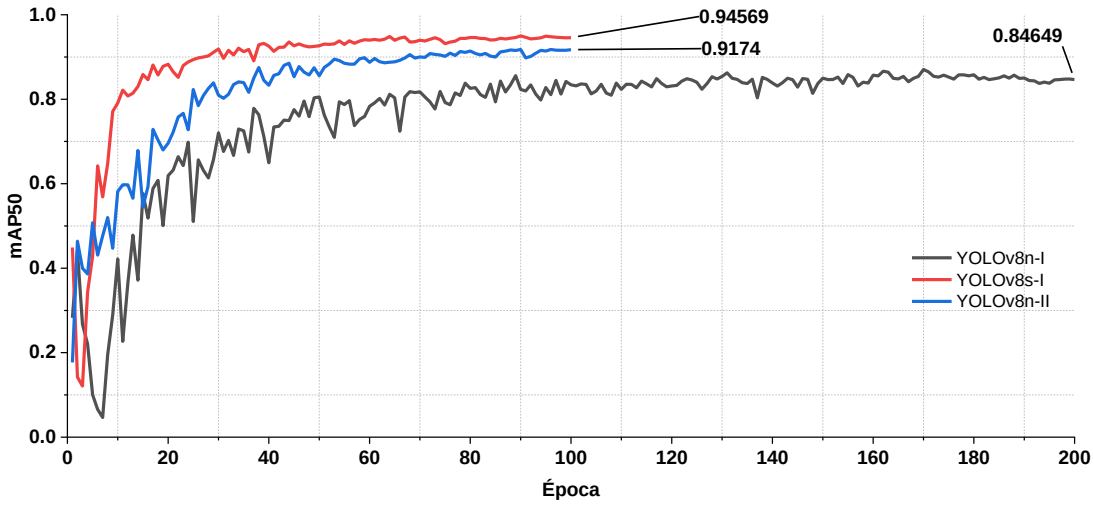


Figura 50. Métrica mAP50

Fuente: Elaboración propia

mAP50-95

En la Fig. 51, podemos apreciar la métrica de los modelos entrenados, observando que para la métrica de mAP50-90, el cual considera correcto si existe al menos un 50-90% de superposición con el objeto real y al mismo tiempo no existe superposición con el fondo en ese mismo porcentaje, los modelos YOLOv8s-I y YOLOv8n-II, lograron valores finales del 77.7% y 71.2% respectivamente, y para el modelo YOLOv8n-I un 65.9%, siendo este muy inferior a los dos anteriores.

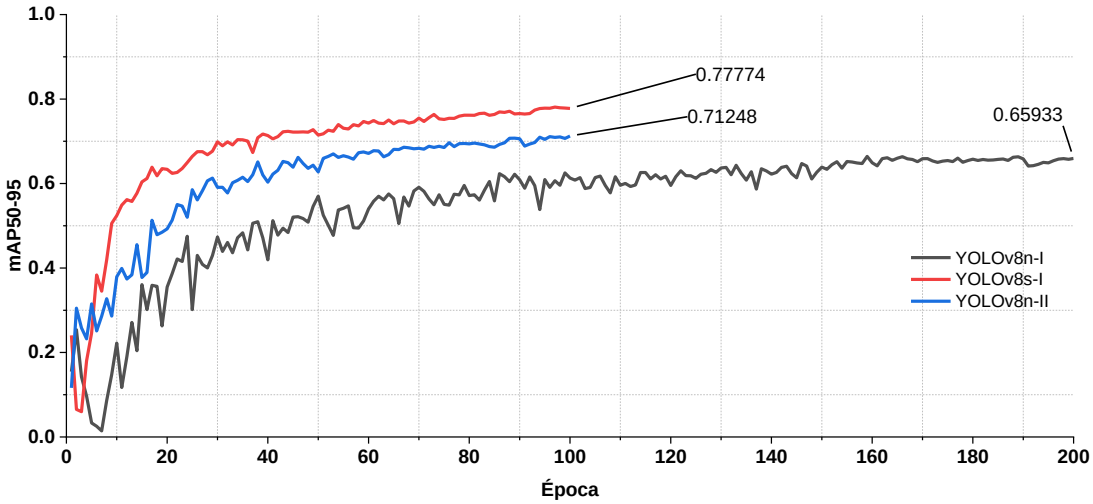


Figura 51. Métrica mAP50-95

Fuente: Elaboración propia

Precision

Para la métrica de precisión, tal como muestra la Fig. 52, se aprecia que los valores para los modelos YOLOv8s-I y YOLOv8n-II tuvieron un 92.6% y 91.9% respectivamente, sin embargo, para el modelo YOLOv8n-I logró un valor final de 86.3%, con valores pico sobre el 90% pero teniendo valores oscilantes durante las primeras 10 épocas de entrenamiento. Este último modelo tuvo un rendimiento inferior. Estos valores en general para esta métrica indican que el modelo predijo correctamente las clases de los objetos en las imágenes.

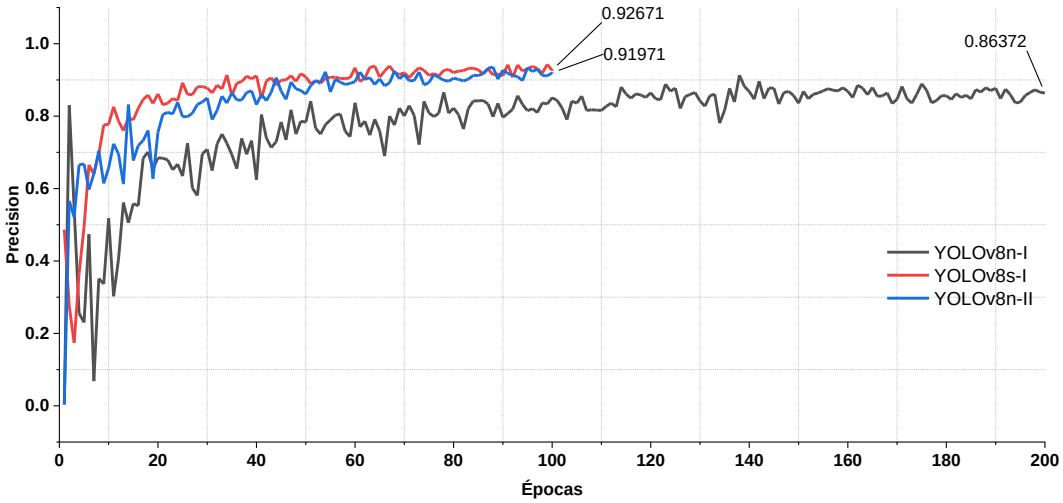


Figura 52. Métrica Precision

Fuente: Elaboración propia

Recall

Para la métrica de Recall, tal como muestra la Fig. 53, se aprecia que los valores para los modelos YOLOv8s-I y YOLOv8n-II tuvieron un 90.5% y 85.1% respectivamente, sin embargo, para el modelo YOLOv8n-I logró un valor final de 79.4%. El valor de esta métrica indica que los modelos predijeron correctamente los casos positivos en las imágenes de validación.

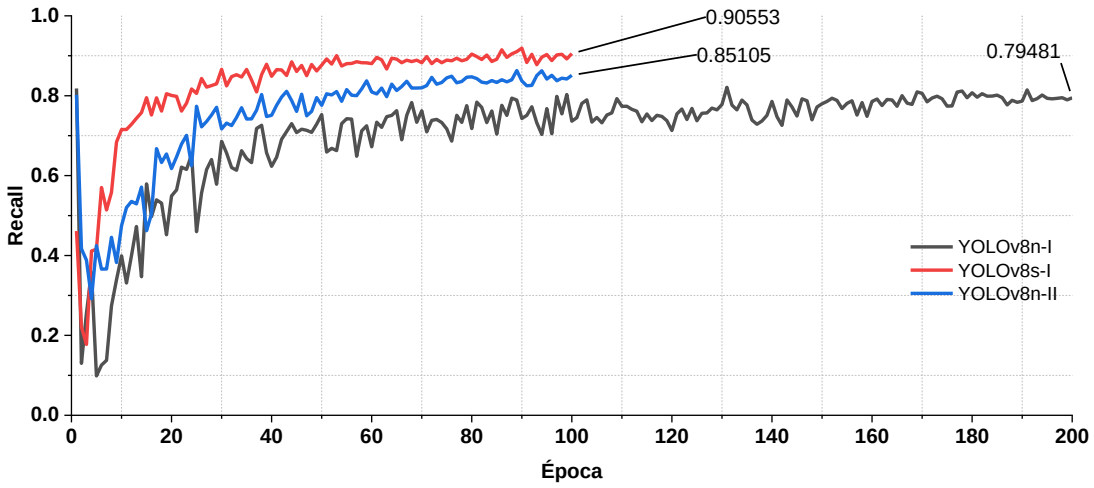
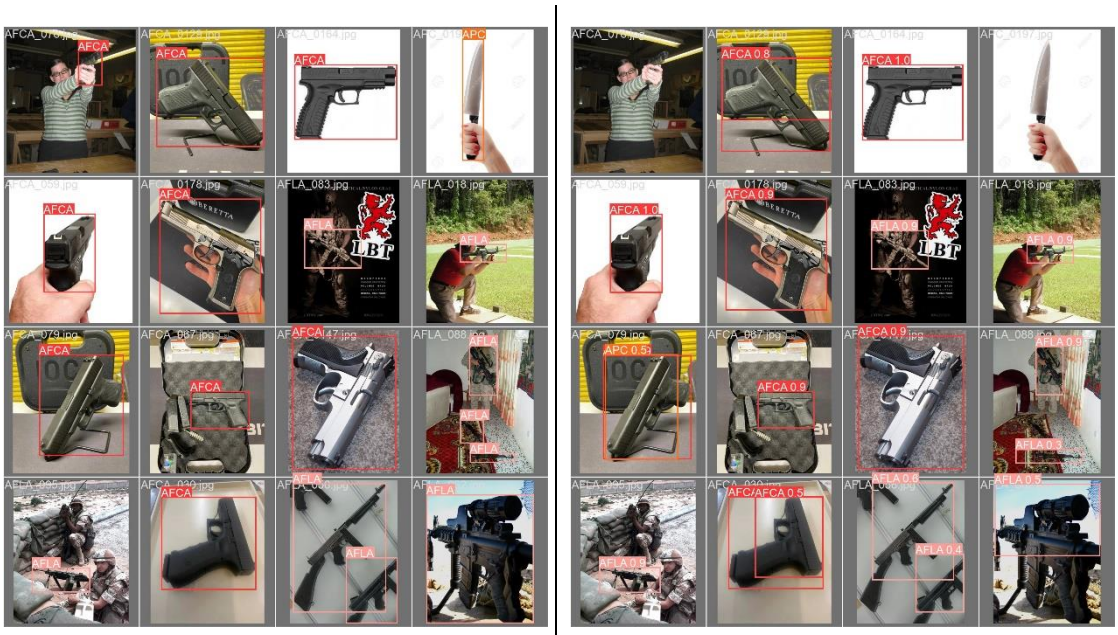


Figura 53. Métrica Recall

Fuente: Elaboración propia

Resultados de detección del modelo con el set de datos de validación

Para el modelo YOLOv8n-I, en la Fig. 54 (a) se muestra las imágenes etiquetadas de validación para el modelo con su respectiva clase. Asimismo, en la Fig. 54 (b), se muestra las predicciones hechas por el modelo con las imágenes de validación, se observa que no pudo identificar correctamente algunas de las clases, y las cajas de detección no se ajustan al objeto identificado y/o son clasificados erróneamente.



(a) (b)

Figura 54. Conjunto de imágenes de validación del modelo YOLOv8n-I (a), resultados de la detección (b)

Fuente: Elaboración propia

Para el modelo YOLOv8s-I, en la Fig. 55 (a) se muestra las imágenes etiquetadas de validación para el modelo con su respectiva clase. Asimismo, en la Fig. 55 (b), se muestran las predicciones hechas por el modelo con las imágenes de validación, se observa que pudo identificar correctamente las clases en ese conjunto de muestra, y las cajas de detección se ajustaron mejor al objeto identificado.

Esto puede deberse debido a que el modelo posee una mayor cantidad de pesos y las operaciones que realiza consumen una mayor cantidad de memoria computacional logrando operaciones más precisas a un alto costo.

(a) (b)

Figura 55. Conjunto de imágenes de validación del modelo YOLOv8s-I (a), resultados de la detección (b)

Fuente: Elaboración propia

Para el modelo YOLOv8n-II, en la Fig. 56 (a) se muestra las imágenes etiquetadas de validación para el modelo con su respectiva clase. Asimismo, en la Fig. 56 (b), se muestran las predicciones hechas por el modelo con las imágenes de validación, se observa que también se pudo identificar correctamente las clases en esa imagen de muestra, y las cajas de detección se ajustan mejor al objeto identificado.

(a) (b)

Figura 56. Conjunto de imágenes de validación del modelo YOLOv8n-II (a), resultados de la detección (b)

Fuente: Elaboración propia

5.2. Resultados inferenciales

Para esta investigación, la inferencia se define como la ejecución del modelo entrenado, elegido e implementado en el sistema en acción para procesar datos nuevos o no etiquetados con el fin de realizar predicciones, a continuación, se muestra la inferencia realizada del dispositivo en escenarios reales, en la cual se evaluó la precisión, velocidad de detección y tiempo de envío de notificaciones frente a situaciones en donde se visualizan objetos de las categorías con las que fue entrenado dicho modelo.

En la Fig. 57 se muestra la primera prueba con la categoría APC, hecha por el sistema de alerta y monitoreo implementado con la Rasbperry Pi 4B, la distancia aproximada de la persona a la cámara fue de 1.5 a 3 metros, las imágenes recopiladas fueron realizadas directamente por la cámara IMX519 conectada a la interfaz CSI del Raspberry Pi 4B.

Figura 57. Predicción del sistema para la categoría APC a una distancia de 1.5 a 3 metros.

Fuente: Elaboración propia

En las Fig. 58 se muestra la prueba hecha por el sistema para la clase AFCA, la distancia aproximada de la cámara a la persona fue de 1.5 a 3 metros.

Figura 58. Predicción del sistema para la categoría AFCA a una distancia aproximada de 1.5 a 3 metros.

Fuente: Elaboración propia.

En las Fig. 59 se muestra la prueba hecha por el sistema para la clase AFLA, la distancia aproximada de la cámara a la persona fue de 1.5 a 3 metros.

Figura 59. Predicción del sistema para la categoría AFLA a una distancia aproximada de 1.5 a 3 metros.

Fuente: Elaboración propia

En la Fig. 60 se observa prueba ejecutada para la categoría APC con una distancia de 6 a 7 metros aproximadamente.

Figura 60. Predicción del sistema para la categoría APC a una distancia de 6 a 7 metros.

Fuente: Elaboración propia.

En la Fig. 61 se observa prueba ejecutada para la categoría AFCA con una distancia de 6 a 7 metros aproximadamente.

Figura 61. Predicción del sistema para la categoría AFCA a una distancia de 6 a 7 metros.

Fuente: Elaboración propia.

En la Fig. 62 se observa prueba ejecutada para la categoría AFLA con una distancia de 6 a 7 metros aproximadamente.

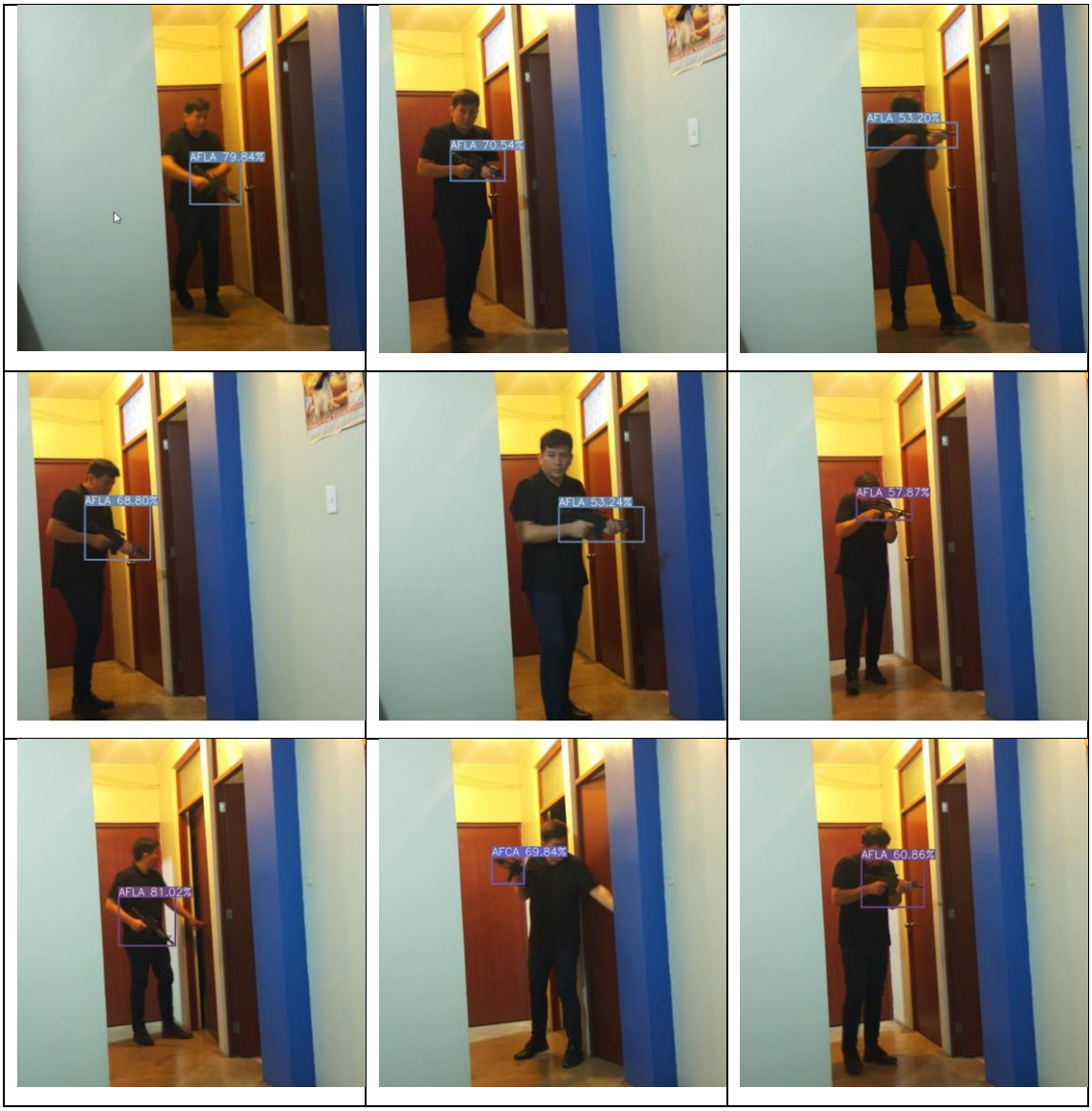


Figura 62. Predicción del sistema para la categoría AFLA a una distancia de 6 a 7 metros.

Fuente: Elaboración propia.

Resultados de envío de las notificaciones con las imágenes de la detección mediante el Bot de Telegram

En la Fig. 63 se observa las detecciones hechas por el sistema y enviados a Telegram, el tiempo de envío de las alertas fueron en intervalos de 2 segundos en promedio, estas alertas se enviaron mediante internet usando el módulo WiFi integrado de la Raspberry Pi 4B.

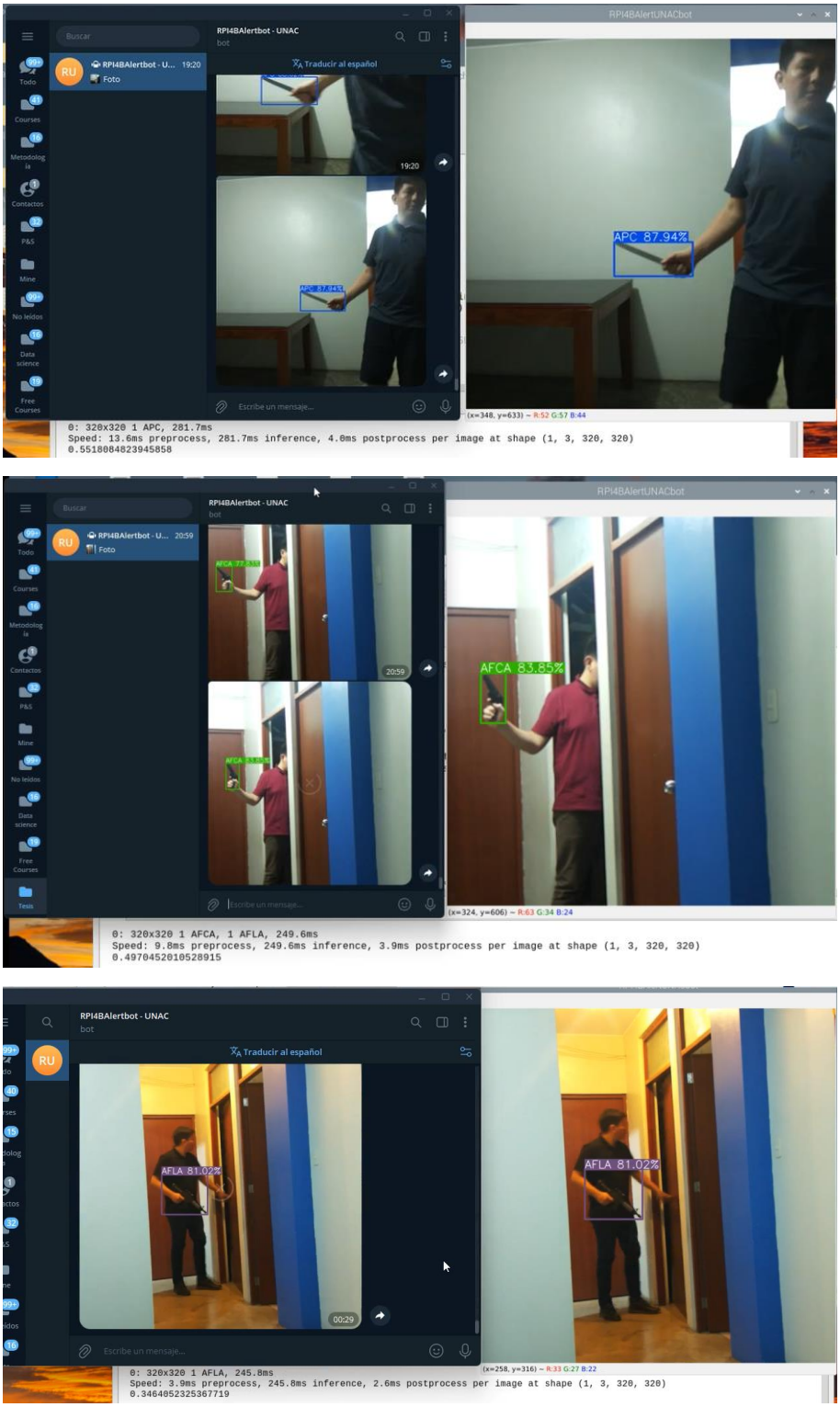


Figura 63. Detección de las armas y envío de la notificación vía Telegram para las tres categorías.

Fuente: Elaboración propia.

VI. DISCUSIÓN DE RESULTADOS

6.1. Contrastación y demostración de la hipótesis con los resultados

Hipótesis general:

El desarrollo de un sistema de alerta y videovigilancia utilizando Raspberry Pi 4B realiza la detección de armas y envío de alertas en la Región Callao, 2023.

Contrastación de la hipótesis general

Se logró el desarrollo de un sistema de alerta y videovigilancia empleando una microcomputadora Raspberry Pi 4B, el sistema se implementó con una cámara con interfaz CSI que permitió enviar las imágenes directamente para el procesado con el modelo entrenado de YOLOv8n-II. Este modelo luego de analizar las imágenes capturadas por la cámara realiza una predicción, y en caso de encontrarse algún arma de las categorías predefinidas (APC, AFCA y AFLA) envía una notificación por el aplicativo móvil Telegram.

Por lo que, se demuestra la hipótesis principal es válida, ya que, de acuerdo con los resultados obtenidos, este sistema logra realizar la detección de armas utilizando modelos de detección de objeto existentes como YOLOv8, pudiéndose ejecutar en dispositivos de bajo consumo. Asimismo, se logró el envío de notificaciones con la detección realizada con las categorías entrenadas.

Hipótesis específicas:

HE1: El modelo de detección de objetos seleccionado e implementado en una Raspberry Pi 4B realiza la detección de armas en la Región Callao, 2023.

Contrastación de la hipótesis específica 1

De los modelos de YOLOv8 existentes, se eligió a la variación YOLOv8n-II para su implementación en la Raspberry Pi 4B, este modelo se ejecutó y realizó las predicciones sin sacrificar la cantidad de fotogramas, obteniendo 3.5 FPS en promedio, y sin sacrificar las métricas propuestas de rendimiento.

Por lo que se demuestra que la hipótesis específica 1 es válida.

HE2: El tipo de alerta seleccionado e implementado en una Raspberry Pi 4B realiza el envío de alertas y/o notificaciones en la detección de armas en la Región Callao, 2023.

Contrastación de la hipótesis específica 2

Se eligió e implementó el tipo de alerta que realice el envío de la información con un tiempo promedio de 2 segundos y permitió automatizar el proceso de envío de las notificaciones, esto se logró utilizando la librería de Python “telepot”, el cual permitió enviar imágenes a un Bot preconfigurado en la aplicación móvil Telegram, desde la Raspberry Pi 4B.

Por lo que se demuestra que la hipótesis específica 2 es válida.

6.2. Contrastación de los resultados con otros estudios similares

Del trabajo citado de Ingle et al., desarrollaron un modelo de detección de armas utilizando una cámara USB y una GPU Nvidia RTX 2060, teniendo una tasa de detección de 97.5%, 90.5% y 90.7% de precisión con diferentes sets de datos. Para la presente investigación, se tuvo una precisión de 88.3%, cercana a la precisión obtenida por ellos.

Respecto al trabajo de Oñate, obtuvieron un tiempo de envío de la alerta de 6.5 segundos una vez se detectan armas, y 10 segundo en promedio para el envío de estas. Este trabajo de investigación logró un tiempo promedio de 2 segundos para el envío de las alertas, asimismo, esta alerta enviada mediante mensaje al aplicativo móvil permite automáticamente también su almacenamiento.

En la Investigación de Aguilar, el autor obtuvo una precisión del 78% para armas (pistolas y cuchillos) en un modelo implementado en una Raspberry Pi 3B+. Sin embargo, la presente investigación logró una precisión del 88.3%, mostrando un mejor desempeño.

En la tesis de Pacco, el sistema de videovigilancia implementado para el reconocimiento de armas (pistolas y cuchillos) implementado con el clasificador de imágenes Haar Cascade mostró una tasa de detección de armas del 90.3% y 88.31% de precisión y exactitud, adicionalmente, utilizaron Whatsapp como

medio para enviar las notificaciones, logrando un tiempo promedio de 6 segundos. Por otra parte, esta investigación desarrollada, logró obtener menores tiempos para el envío de las notificaciones, con un tiempo promedio de 2 segundos luego de hecha la detección del arma.

De acuerdo con el trabajo de Aliaga y Pariona, se tuvo propuso desarrollar un algoritmo para analizar los gestos corporales en asaltos a mano armada en un local comercial, utilizando Raspberry Pi, dentro de las métricas obtenidas por su investigación, lograron un 89.01% de precisión y un 84.40% de exactitud. La presente investigación, logró obtener un 88.3% de precisión y 89.5 de exactitud, mostrando un rendimiento similar en la precisión y superior en la exactitud.

6.3. Responsabilidad ética de acuerdo con los reglamentos vigentes

El presente trabajo de investigación se fundamenta en el “Código de ética de investigación” de la Universidad Nacional del Callao, basándose en: la probidad, profesionalismo, transparencia, objetividad, igualdad, compromiso, honestidad, confidencialidad, independencia, diligencia y dedicación con la finalidad de asegurar el conocimiento, la comprensión y mejora de la condición humana, así como el desarrollo progresivo de la sociedad.

VII. CONCLUSIONES

La detección de armas en entornos videovigilados es de vital importancia ya que la integridad y seguridad de las personas dependen de ello. Esta investigación mostró la implementación de sistema de detección de armas categorizadas en tres clases (armas de corto alcance, armas de largo alcance y armas punzocortantes) adoptando el algoritmo de detección de objetos YOLOv8. En este estudio se compararon los resultados de los modelos YOLOv8s y YOLOv8n sus variantes experimentales (YOLOv8n-I y YOLOv8n-II), para diferentes cantidades de épocas, tamaño de set de datos y tamaño de imagen ingresadas al algoritmo para entrenamiento. Luego, estos modelos fueron implementados en una Rasbperry Pi 4B con el fin de evaluar el rendimiento en tiempo real. Los parámetros considerados para evaluar el rendimiento fueron la Precisión, Recall, F1-Score, mAP50 y el mAP50-95. Se demostró que el modelo que mejor se ajusta para utilizar con la Rasbperry Pi 4B fue la variante propuesta de YOLOv8n-II, siendo superior en tiempos de respuesta en la detección en comparación con la variante YOLOv8s-I, el cual consume recursos excesivos debido a la cantidad de parámetros y conexiones que esta posee, incrementando también la cantidad computacional requerida para su ejecución.

El algoritmo de YOLOv8n-II entrenado tuvo un 88.3% de precisión, estando en un rango de detección óptimo para la identificación de armas, pudiendo detectar adecuadamente objetos superpuestos o incluso en condiciones de luz y perspectiva distintas, logrando un valor promedio de 3.5 FPS en la detección de armas y 2 segundos en promedio para el envío de la notificación una vez hecha la detección.

Este sistema podría ser utilizado en ambientes de circuito cerrado y para el monitoreo en espacios urbanos, donde existen altos índices delictivos, que, en coordinación con las autoridades competentes, podría reducir los asaltos o robos que hagan uso de estos tipos de armas, mitigando la posibilidad de pérdidas de vidas humanas.

Telegram se posiciona como una aplicación polivalente y con tiempos de procesamiento cortos para el envío de mensajes con contenido multimedia

utilizando las librería telepot para manipular el envío de mensajes, el cual puede funcionar como un Bot que emite alertas y también como una base de datos, ya que almacena las imágenes de las detecciones en tiempo real que han sido enviadas por el sistema de detección de objetos, siendo esto de gran utilidad para aplicaciones IoT, de monitoreo y seguimiento, que pueden ser aplicados a diferentes proyectos.

Se pueden usar plataformas en la nube como Google Colab, para desarrollar modelos de inteligencia artificial que requieran procesamiento computacional elevado, ya que permiten usar entornos virtuales con tarjetas gráficas de uso gratuito y de paga para entrenar diferentes tipos de algoritmos existentes o por desarrollar.

La implementación del modelo YOLOv8 es práctica y se puede implementar utilizando pocas líneas de código, ya sea utilizando la línea de comandos (CLI) o con el lenguaje de programación Python. En comparación de desarrollar un algoritmo desde cero, esto reduce el tiempo necesario para implementar una idea de proyecto de investigación de clasificación, detección o segmentación de objetos.

Esta investigación brinda a los investigadores una base para poder utilizar el modelo de detección de objetos YOLOv8 en dispositivos de bajo consumo como la Rasbperry Pi 4B, así de como implementar de notificaciones o alertas de actividades predefinidas por el investigador que involucren la detección de objetos en cuadros de imágenes en videos.

Finalmente, esta investigación no solo se restringe a la detección de armas, pudiéndose utilizar como referencia para implementar otro tipo de detección de objetos que se apliquen en tiempo real.

VIII. RECOMENDACIONES

- Utilizar cámaras infrarrojas para poder realizar detecciones de armas en ambientes poco iluminados o nocturnos, ya que estas cámaras permiten detectar objetos en situaciones de baja luminosidad y visibilidad.
- Dentro de la gama de modelos YOLOV8 existentes, se sugiere experimentar modificando los hiper parámetros de entrenamiento (tamaño de lote, cantidad de épocas y tamaño de imagen de entrada), así como la cantidad de imágenes utilizadas para el entrenamiento, con el fin de encontrar mejores métricas de desempeño.
- Experimentar utilizando otros dispositivos de bajo consumo como Jetson nano, Orange Pi, etc, y evaluar otros modelos existentes basados en YOLO.
- Comparar los otros modelos existentes para detectar objetos y evaluar si su implementación en la Raspberry Pi u otro microcomputador ofrece un desempeño superior.

IX. REFERENCIAS BIBLIOGRÁFICAS

[1] O. Rasheed, A. Ishaq, M. Asad, y T. S. S. Hashmi, “Multiplatform Surveillance System for Weapon Detection using YOLOv5”, en *2022 17th International Conference on Emerging Technologies (ICET)*, IEEE, nov. 2022, pp. 37–42. doi: 10.1109/ICET56601.2022.10004690.

[2] C. Ineneji y M. Kusaf, “Hybrid weapon detection algorithm, using material test and fuzzy logic system”, *Computers and Electrical Engineering*, vol. 78, pp. 437–448, sep. 2019, doi: 10.1016/J.COMPELECENG.2019.08.005.

[3] J. G. Ríos, “Descripción del estado del arte de la seguridad comercial, industrial y ciudadana de los sistemas de cámaras de seguridad en la ciudad de Cartagena – Bolívar”, Trabajo de Grado, Universidad Militar Nueva Granada, Colombia, 2016. Consultado: el 25 de marzo de 2023. [En línea]. Disponible en: http://hdl.handle.net/10654/14256

[4] V. Lio, “CIUDADES, CÁMARAS DE SEGURIDAD Y VIDEO-VIGILANCIA: ESTADO DEL ARTE Y PERSPECTIVAS DE INVESTIGACIÓN”, *Astrolabio*, núm. 15, pp. 273–302, dic. 2015, doi: 10.55441/1668.7515.N15.9903.

[5] C. S. Sung y J. Y. Park, “Design of an intelligent video surveillance system for crime prevention: applying deep learning technology”, *Multimed Tools Appl*, vol. 80, núm. 26–27, pp. 34297–34309, nov. 2021, doi: 10.1007/S11042-021-10809-Z.

[6] D. Ameijeiras Sánchez, H. R. González Diez, y Y. Hernández Heredia, “Revisión de algoritmos de detección y seguimiento de objetos con redes profundas para videovigilancia inteligente”, *Revista Cubana de Ciencias Informáticas*, vol. 14, núm. 3, 2020, Consultado: el 27 de septiembre de 2023. [En línea]. Disponible en: http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S2227-18992020000300165

[7] S. Borde, C. Patil, C. Sonawane, y M. Singh, “Object Recognition based on Deep Learning Algorithms using Embedded IoT with Interactive Interface”, *Proceedings of the 7th International Conference on Intelligent Computing and Control Systems, ICICCS 2023*, pp. 1581–1586, 2023, doi: 10.1109/ICICCS56967.2023.10142821.

[8] A. N. Amudhan y A. P. Sudheer, “Lightweight and computationally faster Hypermetropic Convolutional Neural Network for small size object detection”, *Image Vis Comput*, vol. 119, p. 104396, mar. 2022, doi: 10.1016/J.IMAVIS.2022.104396.

[9] M. Aria y C. Cuccurullo, “bibliometrix: An R-tool for comprehensive science mapping analysis”, *J Informetr*, vol. 11, núm. 4, pp. 959–975, nov. 2017, doi: 10.1016/J.JOI.2017.08.007.

[10] J. Redmon, S. Divvala, R. Girshick, y A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection”, *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-December, pp. 779–788, jun. 2015, doi: 10.1109/CVPR.2016.91.

[11] W. Liu *et al.*, “SSD: Single Shot MultiBox Detector”, *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9905 LNCS, pp. 21–37, dic. 2015, doi: 10.1007/978-3-319-46448-0_2.

[12] S. Ren, K. He, R. Girshick, y J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”, *IEEE Trans Pattern Anal Mach Intell*, vol. 39, núm. 6, pp. 1137–1149, jun. 2015, doi: 10.1109/TPAMI.2016.2577031.

[13] R. Girshick, “Fast R-CNN”, en *2015 IEEE International Conference on Computer Vision (ICCV)*, IEEE, dic. 2015, pp. 1440–1448. doi: 10.1109/ICCV.2015.169.

[14] K. Lenc y A. Vedaldi, “R-CNN minus R”, pp. 5.1-5.12, jun. 2015, doi: 10.5244/c.29.5.

[15] E. Gil-González, L. A. Pérez-Maqueda, P. E. Sánchez-Jiménez, y A. Perejón, “Flash Sintering Research Perspective: A Bibliometric Analysis”, *Materials*, vol. 15, núm. 2, p. 416, ene. 2022, doi: 10.3390/ma15020416.

[16] M. T. Bhatti, M. G. Khan, M. Aslam, y M. J. Fiaz, “Weapon Detection in Real-Time CCTV Videos Using Deep Learning”, *IEEE Access*, vol. 9, pp. 34366–34382, 2021, doi: 10.1109/ACCESS.2021.3059170.

[17] H. Jain, A. Vikram, Mohana, A. Kashyap, y A. Jain, “Weapon Detection using Artificial Intelligence and Deep Learning for Security Applications”, en *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)*, IEEE, jul. 2020, pp. 193–198. doi: 10.1109/ICESC48915.2020.9155832.

[18] Redacción RPP, “Coronavirus en el Perú: Inseguridad y delincuencia ciudadana durante la pandemia en el país”. Consultado: el 13 de julio de 2023. [En línea]. Disponible en: https://rpp.pe/peru/actualidad/coronavirus-en-el-peru-inseguridad-y-delincuencia-ciudadana-durante-la-pandemia-en-el-pais-noticia-1310011

[19] D. Carhuavilca *et al.*, “Estadísticas de Seguridad Ciudadana - Informe Técnico N°01”, Instituto Nacional de Estadística e Informática, 2023. Consultado: el 13 de julio de 2023. [En línea]. Disponible en: https://m.inei.gob.pe/media/MenuRecursivo/boletines/estadisticas_de_seguridad_ciudadana_jul_dic_2022.pdf

[20] “Estimaciones de indicadores del Plan Nacional de Seguridad Ciudadana 2019-2023”, Ministerio del Interior, Lima, Perú, 2018. Consultado: el 14 de julio de 2023. [En línea]. Disponible en: https://cdn.www.gob.pe/uploads/document/file/4041993/Estimaciones%2

0de%20indicadores%20del%20PNSC%202019-2023.pdf?v=1673626712

[21] P. Y. Ingle y Y.-G. Kim, “Real-Time Abnormal Object Detection for Video Surveillance in Smart Cities”, *Sensors*, vol. 22, núm. 10, p. 3862, may 2022, doi: 10.3390/s22103862.

[22] R. Olmos, S. Tabik, y F. Herrera, “Automatic handgun detection alarm in videos using deep learning”, *Neurocomputing*, vol. 275, pp. 66–72, ene. 2018, doi: 10.1016/J.NEUCOM.2017.05.012.

[23] F. P. Oñate Miranda, “Diseño y construcción de nodos inteligentes para detección de armas dentro de una red de video-vigilancia utilizando visión artificial”, Tesis de pregrado, Escuela Superior Politécnica de Chimborazo, 2020. Consultado: el 24 de septiembre de 2023. [En línea]. Disponible en: http://dspace.espoch.edu.ec/handle/123456789/13783

[24] J. E. Suárez Páez, “Arquitectura de detección de actividades criminales basada en análisis de vídeo en tiempo real”, Universitat Politècnica de València, Valencia (Spain), 2020. doi: 10.4995/Thesis/10251/153162.

[25] E. J. Aguilar Cabrera, “Identificación en tiempo real de personas en posesión de pequeñas armas dentro de un ambiente vídeo-vigilado”, Tesis de Licenciatura, Universidad de las Fuerzas Armadas, 2018. Consultado: el 3 de diciembre de 2023. [En línea]. Disponible en: http://repositorio.espe.edu.ec/jspui/handle/21000/15377

[26] B. A. Criollo-Leal y N. Díaz-Rondón, “Método de detección automática de armas de mano en video usando aprendizaje profundo”, Tesis de Licenciatura, Universidad Católica de Colombia, 2019. Consultado: el 4 de diciembre de 2023. [En línea]. Disponible en: https://hdl.handle.net/10983/24010

[27] J. M. Huaman Cruz, “Detección de armas de fuego utilizando redes convolucionales profundas”, Tesis de pregrado, Universidad Nacional de San Agustín de Arequipa, 2019. Consultado: el 26 de septiembre de 2023. [En línea]. Disponible en: http://repositorio.unsa.edu.pe/handle/UNSA/10578

[28] W. Leturia-Rodriguez y L. Urrelo-Huiman, “Object Detection and Movement Patterns Using Neural Networks and Genetic Algorithms for the Identification of Armed Robbery”, *Latin-American Journal of Computing*, vol. 8, núm. 2, pp. 46–57, jul. 2021, doi: 10.5281/ZENODO.5770889.

[29] J. A. Pacco Enriquez, “Desarrollo e implementación de un sistema CCTV antirrobo inteligente capaz de detectar armas de manera eficiente y a bajo costo al interior de la joyería Chavelis de Arequipa usando un Raspberry PI 4”, Tesis de pregrado, Universidad Nacional de San Agustín de Arequipa, 2022. Consultado: el 26 de septiembre de 2023. [En línea]. Disponible en: http://hdl.handle.net/20.500.12773/14858

[30] A. M. Aliaga Olivares y E. J. Pariona Lozano, “Desarrollo y validación de un algoritmo basado en visión artificial para el reconocimiento de un posible asalto con arma de fuego en un local comercial de Lima”, Tesis de Licenciatura, Universidad Tecnológica del Perú, 2022. Consultado: el 3 de diciembre de 2023. [En línea]. Disponible en: http://repositorio.utp.edu.pe/handle/20.500.12867/6190

[31] J. L. Blas Campos y Jimenez Galindo Bruno Steffano, “Algoritmo inteligente de vigilancia para la detección de actos delictivos en el distrito de Comas”, Tesis de Licenciatura, Universidad César Vallejo, 2023. Consultado: el 3 de diciembre de 2023. [En línea]. Disponible en: https://repositorio.ucv.edu.pe/handle/20.500.12692/121870

[32] M.-O. D. Alejandra, “Re-identificación de personas a través de sus características soft-biométricas en un entorno multi-cámara de video-vigilancia”, *Ingeniería, Investigación y Tecnología*, vol. 17, núm. 2, pp. 257–271, abr. 2016, doi: 10.1016/j.riit.2016.06.010.

[33] “Cajamarca inaugura moderno sistema de videovigilancia para reforzar seguridad”, Agencia Peruana de Noticias Andina. Consultado: el 30 de septiembre de 2023. [En línea]. Disponible en: https://andina.pe/agencia/noticia-cajamarca-inaugura-moderno-sistema-videovigilancia-para-reforzar-seguridad-832674.aspx

[34] Á. García-Martín y J. M. Martínez, “Post-processing approaches for improving people detection performance”, *Computer Vision and Image Understanding*, vol. 133, pp. 76–89, abr. 2015, doi: 10.1016/j.cviu.2014.09.010.

[35] H. J. Eslava Blanco, N. Serrano P., y F. A. Castro, “Sistema de alerta de riesgos en hogares mediante SMS”, *Visión electrónica*, vol. 6, núm. 2, pp. 15–30, jul. 2012, doi: 10.14483/22484728.3883.

[36] G. Ubale y S. Gaikwad, “SMS Spam Detection Using TFIDF and Voting Classifier”, *2022 International Mobile and Embedded Technology Conference, MECON 2022*, pp. 363–366, 2022, doi: 10.1109/MECON53876.2022.9752078.

[37] “Sistema Industrial de Notificaciones: ¿Cómo mantenerme informado en todo momento?”, SINCI. Consultado: el 30 de septiembre de 2023. [En línea]. Disponible en: https://sinci.com/sistema-industrial-de-notificaciones-como-mantenerme-informado-en-todo-momento/

[38] G. Sushanth y S. Sujatha, “IOT Based Smart Agriculture System”, *2018 International Conference on Wireless Communications, Signal Processing and Networking, WiSPNET 2018*, nov. 2018, doi: 10.1109/WISPNET.2018.8538702.

[39] R. S. Ransing y M. Rajput, “Smart home for elderly care, based on wireless sensor network”, *2015 International Conference on Nascent Technologies in the Engineering Field, ICNTE 2015 - Proceedings*, feb. 2015, doi: 10.1109/ICNTE.2015.7029932.

[40] U. Goel, K. Shah, S. Singh, y M. A. Qadeer, “EMS: The talking mail service”, en *2011 IEEE 3rd International Conference on Communication Software and Networks*, IEEE, may 2011, pp. 622–626. doi: 10.1109/ICCSN.2011.6014346.

[41] K. Shramko, “Las 15 mejores aplicaciones de mensajería instantánea gratuitas para empresas”. Consultado: el 3 de enero de 2024. [En línea]. Disponible en: https://trueconf.com/es-es/blog/noticias/aplicaciones-de-mensajeria-instantanea-para-empresas.html

[42] I. Rahil, W. Bouarifi, O. Mustapha, y R. Ghizlane, “ADVANCING REAL-TIME VIDEO VIOLENCE DETECTION: A DEEP LEARNING APPROACH WITH INTEGRATED TELEGRAM ALERTING”, *J Theor Appl Inf Technol*, vol. 101, núm. 21, pp. 7021–7032, 2023, [En línea]. Disponible en: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85179458810&partnerID=40&md5=77b076eb1a9b3b462f1235a7d08e941e

[43] A. Voulodimos, N. Doulamis, A. Doulamis, y E. Protopapadakis, “Deep Learning for Computer Vision: A Brief Review”, *Comput Intell Neurosci*, vol. 2018, pp. 1–13, 2018, doi: 10.1155/2018/7068349.

[44] Z.-Q. Zhao, P. Zheng, S.-T. Xu, y X. Wu, “Object Detection With Deep Learning: A Review”, *IEEE Trans Neural Netw Learn Syst*, vol. 30, núm. 11, pp. 3212–3232, nov. 2019, doi: 10.1109/TNNLS.2018.2876865.

[45] L. Liu *et al.*, “Deep Learning for Generic Object Detection: A Survey”, *Int J Comput Vis*, vol. 128, núm. 2, pp. 261–318, feb. 2020, doi: 10.1007/s11263-019-01247-4.

[46] S. Narejo, B. Pandey, D. Esenarro vargas, C. Rodriguez, y M. R. Anjum, “Weapon Detection Using YOLO V3 for Smart Surveillance System”, *Math Probl Eng*, vol. 2021, pp. 1–9, may 2021, doi: 10.1155/2021/9975700.

[47] A. Vaishya, *Mastering OpenCV with Python*. Delhi: Orange Education PVT Ltd, 2023.

[48] Z. Li, F. Liu, W. Yang, S. Peng, y J. Zhou, “A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects”, *IEEE Trans Neural Netw Learn Syst*, vol. 33, núm. 12, pp. 6999–7019, abr. 2020, doi: 10.1109/TNNLS.2021.3084827.

[49] K. O’Shea y R. Nash, “An Introduction to Convolutional Neural Networks”, *Int J Res Appl Sci Eng Technol*, vol. 10, núm. 12, pp. 943–947, nov. 2015, doi: 10.22214/ijraset.2022.47789.

[50] J. Krohn, G. Beyleveld, y Aglaé. Bassens, *Deep Learning Illustrated A Visual, Interactive Guide to Artificial Intelligence*. Hoboken: Pearson Education, Limited, 2019.

[51] R. Girshick, J. Donahue, T. Darrell, y J. Malik, “Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation”, en *2014 IEEE*

Conference on Computer Vision and Pattern Recognition, IEEE, jun. 2014, pp. 580–587. doi: 10.1109/CVPR.2014.81.

[52] S. Pereira, A. Pinto, V. Alves, y C. A. Silva, “Brain Tumor Segmentation Using Convolutional Neural Networks in MRI Images”, *IEEE Trans Med Imaging*, vol. 35, núm. 5, pp. 1240–1251, may 2016, doi: 10.1109/TMI.2016.2538465.

[53] “¿Que es Raspberry Pi? - Raspberry Pi”, MCIElectronics. Consultado: el 3 de enero de 2024. [En línea]. Disponible en: https://raspberrypi.cl/que-es-raspberry/

[54] L. Calvo, “¿Qué es un bot? Tipos de bots y sus principales funciones - Blog”. Consultado: el 3 de enero de 2024. [En línea]. Disponible en: https://es.godaddy.com/blog/que-es-un-bot/

[55] Red Hat, “El concepto del edge computing”, Red Hat. Consultado: el 4 de febrero de 2024. [En línea]. Disponible en: https://www.redhat.com/es/topics/edge-computing

[56] A. Koirala, K. B. Walsh, Z. Wang, y C. McCarthy, “Deep learning – Method overview and review of use for fruit detection and yield estimation”, *Comput Electron Agric*, vol. 162, pp. 219–234, jul. 2019, doi: 10.1016/j.compag.2019.04.017.

[57] Y. Wu *et al.*, “Demystifying Learning Rate Policies for High Accuracy Training of Deep Neural Networks”, ago. 2019.

[58] Hugging Face, “Quantization”. Consultado: el 17 de enero de 2024. [En línea]. Disponible en: https://huggingface.co/docs/optimum/concept_guides/quantization

[59] V. Subramanyam, “Non Max Suppression (NMS)”. Consultado: el 9 de marzo de 2024. [En línea]. Disponible en: https://medium.com/analytics-vidhya/non-max-suppression-nms-6623e6572536

[60] O. Elharrouss, N. Almaadeed, y S. Al-Maadeed, “A review of video surveillance systems”, *J Vis Commun Image Represent*, vol. 77, p. 103116, may 2021, doi: 10.1016/j.jvcir.2021.103116.

[61] S. Khalid, A. Waqar, H. U. Ain Tahir, O. C. Edo, y I. T. Tenebe, “Weapon detection system for surveillance and security”, *2023 International Conference on IT Innovation and Knowledge Discovery, ITIKD 2023*, 2023, doi: 10.1109/ITIKD56332.2023.10099733.

[62] A. Rehman y L. G. Fahad, “Real-Time Detection of Knives and Firearms using Deep Learning”, *2022 24th International Multitopic Conference, INMIC 2022*, 2022, doi: 10.1109/INMIC56986.2022.9972915.

[63] J. A. Fernández Vásquez, C. N. Purihuaman Leonardo, O. López Regalado, y M. J. Sánchez Chero, *Metodología de la investigación científica y tecnológica*. Colloquium, 2021. Consultado: el 7 de diciembre

de 2023. [En línea]. Disponible en: https://colloquiumbiblioteca.com/index.php/web/article/view/95

[64] C. Espinoza Montes, *Metodología de Investigación Tecnológica Pensando en Sistemas*. Universidad Nacional del Centro del Perú, 2014. Consultado: el 7 de diciembre de 2023. [En línea]. Disponible en: http://repositorio.uncp.edu.pe/handle/20.500.12894/1148

[65] R. Hernández Sampieri y C. P. Mendoza Torres, *Metodología de la investigación: las rutas cuantitativa, cualitativa y mixta*. México, D.F.: McGraw-Hill, 2018.

[66] ArduCam, "Arducam IMX519", ArduCam. Consultado: el 11 de noviembre de 2023. [En línea]. Disponible en: https://www.arducam.com/product/imx519-autofocus-camera-module-for-raspberry-pi-arducam-b0371/

[67] F. M. Talaat y H. ZainEldin, "An improved fire detection approach based on YOLO-v8 for smart cities", *Neural Comput Appl*, vol. 35, núm. 28, pp. 20939–20954, oct. 2023, doi: 10.1007/s00521-023-08809-1.

[68] "Ultralytics YOLOv8", Ultralytics. Consultado: el 22 de noviembre de 2023. [En línea]. Disponible en: https://github.com/ultralytics/ultralytics

[69] A. Vina, "From YOLO to YOLOv8: Tracing the Evolution of Object Detection Algorithms | by Abirami Vina | Nerd For Tech | Medium", Medium. Consultado: el 3 de enero de 2024. [En línea]. Disponible en: https://medium.com/nerd-for-tech/from-yolo-to-yolov8-tracing-the-evolution-of-object-detection-algorithms-eaed9a982ebd

[70] J. R. Terven y D. M. Cordova-Esparza, "A Comprehensive Review of YOLO: From YOLOv1 and Beyond", abr. 2023, doi: 10.3390/make5040083.

[71] "IMFDB: Internet Movie Firearms Database". Consultado: el 8 de diciembre de 2023. [En línea]. Disponible en: https://imfdb.org/wiki/Main_Page

[72] M. Zahrawi, "Weapon Detection YOLOv7", IEEE DataPort, 2022. doi: 10.21227/3akm-cb29.

[73] Q. Nguyen, "Gun Dataset Labelled", IEEE Dataport, 2020. doi: 10.21227/db68-5854.

[74] F. Pérez Hernandez y A. Castillo Lamas, "Weapon detection datasets", Universidad de Granada, 2020. Consultado: el 8 de diciembre de 2023. [En línea]. Disponible en: https://github.com/ari-dasci/OD-WeaponDetection

[75] J. L. Salazar González, C. Zaccaro, J. A. Álvarez-García, L. M. Soria Morillo, y F. Sancho Caparrini, "Real-time gun detection in CCTV: An open problem", *Neural Networks*, vol. 132, pp. 297–308, dic. 2020, doi: 10.1016/j.neunet.2020.09.013.

[76] Tzutalin, “LabelImg”, 2015. Consultado: el 23 de diciembre de 2023. [En línea]. Disponible en: https://github.com/tzutalin/labelImg

[77] J. Kaur y W. Singh, “Tools, techniques, datasets and application areas for object detection in an image: a review”, *Multimedia Tools and Applications 2022 81:27*, vol. 81, núm. 27, pp. 38297–38351, abr. 2022, doi: 10.1007/S11042-022-13153-Y.

X. ANEXOS

Matriz de Consistencia

Tabla 6. Matriz de Consistencia.

<table>
<tr><th>Problema</th><th>Objetivos</th><th>Hipótesis</th><th>Variables</th><th>Dimensión</th><th>Indicador</th><th>Método</th></tr>
<tr><td rowspan="3">PG: ¿El desarrollo de un sistema de alerta y videovigilancia utilizando Raspberry Pi 4B puede realizar la detección de armas en la región Callao, 2023?</td><td rowspan="3">OG: Desarrollar un sistema de alerta y videovigilancia utilizando Raspberry Pi 4B para la detección de armas en la región Callao, 2023.</td><td rowspan="3">HG: El desarrollo de un sistema de alerta y videovigilancia utilizando Raspberry Pi 4B realiza la detección de armas y envío de alertas en la Región Callao, 2023.</td><td rowspan="4">X: Sistema de Alerta y Videovigilancia</td><td rowspan="3">Tipo de Cámara</td><td>Resolución</td><td rowspan="6">Tipo y diseño de investigación:
Tecnológica-aplicada, pre-experimental

Método de Investigación:
Cuantitativo

Población y muestra:
Conjunto de imágenes de armas. Se seleccionaron 4800 imágenes de armas (cuchillos, pistolas y fusiles)

Lugar de estudio:
Callao, Perú

Técnicas para el análisis e interpretación de datos:
Se utilizaron métricas de desempeño y gráficos estadísticos implementados con Python y el Software Origin Pro.</td></tr>
<tr><td>Interfaz de comunicación</td></tr>
<tr><td>Autofoco</td></tr>
<tr><td rowspan="2">PE1: ¿Cuál es el modelo de detección de objetos a utilizar en un sistema de alerta y videovigilancia utilizando Raspberry Pi 4B para la detección de armas en la región Callao, 2023?</td><td rowspan="2">OE1: Evaluar el modelo de detección de objetos a utilizar en el sistema de alerta y videovigilancia utilizando Raspberry Pi 4B para la detección de armas en la región Callao, 2023.</td><td rowspan="2">HE1: El modelo de detección de objetos seleccionado e implementado en una Raspberry Pi 4B realiza la detección de armas en la Región Callao, 2023.</td><td rowspan="2">Alerta</td><td rowspan="2">Tiempo de envío</td></tr>
<tr><td rowspan="4">Y: Detección de armas</td></tr>
<tr><td rowspan="3">PE2: ¿Cuál es el tipo de alerta a utilizar en un sistema de alerta y videovigilancia utilizando Raspberry Pi 4B para la detección de armas en la región Callao, 2023?</td><td rowspan="3">OE2: Evaluar el tipo de alerta a utilizar en el sistema de alerta y videovigilancia utilizando Raspberry Pi 4B para la detección de armas en la región Callao, 2023</td><td rowspan="3">HE2: El tipo de alerta seleccionado e implementado en una Raspberry Pi 4B realiza el envío de alertas y/o notificaciones de la detección de armas en la Región Callao, 2023.</td><td>Armas punzocortantes</td><td rowspan="3">mAP
Precision
Recall
FPS
F1-Score</td></tr>
<tr><td>Armas de fuego de corto alcance</td></tr>
<tr><td>Armas de fuego de largo alcance</td></tr>
</table>

Fuente: Elaboración propia.

Código para la configuración de la cámara IMX519

```
from picamera2 import Picamera2
picam2 = Picamera2()
dispW = 1080
dispH = 1080
picam2.preview_configuration.main.size = (dispW, dispH)
picam2.preview_configuration.main.format= "RGB888"
picam2.preview_configuration.align()
picam2.configure("preview")
picam2.set_controls({"AfMode": 2, "LensPosition": 0})
picam2.start()
```

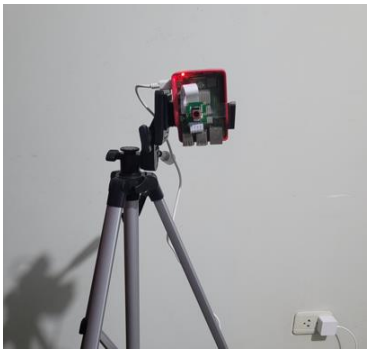
Código para la configuración de enlace de la Raspberry con Telegram

```
import telepot

def handle(msg):
    global telegramText
    global chat_id
    global receiveTelegramMessage
    chat_id = msg['chat']['id']
    telegramText = msg['text']
    print("Message received from " + str(chat_id))
    if telegramText == "/start":
        bot.sendMessage(chat_id, "Bienvenido al RPI4BAlertUNACbot")
    else:
        receiveTelegramMessage = True


bot = telepot.Bot('XXXXXXXXXXX:XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX')
bot.message_loop(handle)
receiveTelegramMessage = False
sendTelegramMessage = False
```

Imágenes del sistema implementado



Nota final: Para consultas sobre el código general implementado y sobre la base de datos utilizada para el entrenamiento del algoritmo, pueden remitir un correo electrónico a: dennis.hyrigoin@hotmail.com