PROYECTO DE TITULACIÓN

Departamento de Ciencias de la Computación

Carrera de Ingeniería en Tecnologías de la Información

Propuesta de mejoras de alertas de seguridad de dispositivos de IoT mediante Inteligencia Artificial

Duque Quevedo, Odalys Rashel

Ing. Salazar Armijos, Diego Ricardo, Ph.D.

Santo Domingo, 01 de marzo de 2024

Reporte de Verificación de Contenido

Copyleaks

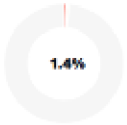
Plagiarism and AI Content Detection Report

DUQUE ODALYS IEEE final.docx

Scan details

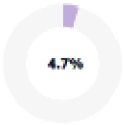
Scan time:	Total Pages:	Total Words:
March 1th, 2024 at 15:35 UTC	44	10817

Plagiarism Detection



Types of plagiarism		Words
Identical	0.8%	85
Minor Changes	0.3%	31
Paraphrased	0.4%	40
Omitted Words	0%	0

AI Content Detection



Text coverage		Words
AI text	4.7%	512
Human text	95.3%	10305

Learn more

Firma:

DIEGO RICARDO SALAZAR ARMIJOS

.....

Ing. Salazar Armijos Diego Ricardo, PhD

Director

Departamento de Ciencias de la Computación

Carrera de Ingeniería en Tecnologías de la Información

Certificación

Certifico que el trabajo de integración curricular: "**Propuesta de mejoras de alertas de seguridad de dispositivos de IoT mediante Inteligencia Artificial**" fue realizado por la señorita **Duque Quevedo, Odalys Rashel,** el mismo que cumple con los requisitos legales, teóricos, científicos, técnicos y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, además fue revisado y analizada en su totalidad por la herramienta de prevención y/o verificación de similitud de contenidos; razón por la cual me permito acreditar y autorizar para que se lo sustente públicamente.

Santo Domingo de los Tsáchilas, 01 de marzo del 2024

Firma:

DIEGO RICARDO SALAZAR ARMIJOS

.....

Ing. Salazar Armijos Diego Ricardo, PhD

C.C: 1710481027

Departamento de Ciencias de la Computación

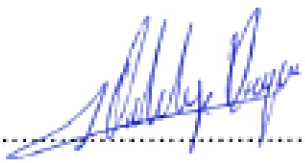
Carrera de Ingeniería en Tecnologías de la Información

Responsabilidad de Autoría

Yo, **Duque Quevedo, Odalys Rashel**, con cédula de ciudadanía n° 2300552565, declaro que el contenido, ideas y criterios del trabajo de integración curricular: **"Propuesta de mejoras de alertas de seguridad de dispositivos de IoT mediante Inteligencia Artificial"** es de mi autoría y responsabilidad, cumpliendo con los requisitos legales, teóricos, científicos, técnicos, y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, respetando los derechos intelectuales de terceros y referenciando las citas bibliográficas.

Santo Domingo de los Tsáchilas, 01 de marzo de 2024

Firma



.....

Duque Quevedo, Odalys Rashel

C.C.:2300552565

ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

Departamento de Ciencias de la Computación

Carrera de Ingeniería en Tecnologías de la Información

Autorización de Publicación

Yo **Duque Quevedo, Odalys Rashel**, con cédula de ciudadanía n° 2300552565, autorizo a la Universidad de las Fuerzas Armadas ESPE publicar el trabajo de integración curricular: "**Propuesta de mejoras de alertas de seguridad de dispositivos de IoT mediante Inteligencia Artificial**" en el Repositorio Institucional, cuyo contenido, ideas y criterios son de mi responsabilidad.

Santo Domingo de los Tsáchilas, 01 de marzo de 2024

Firma

.....

Duque Quevedo, Odalys Rashel

C.C.:2300552565

DEDICATORIA

Dedico esta tesis a mis padres Franklin German Duque Marcillo y Raquelita del Pilar Quevedo Ayala, ya que han sido el soporte y motor durante este tiempo de preparación y esfuerzos no individuales, incluyendo a mi hermano Franklin Brayan Duque Quevedo y mi hermana Nahomy Raquel Duque Quevedo, quienes han luchado junto a mí en todo este trayecto, dándome siempre palabras de aliento, apoyándome de forma directa o indirecta en todo este proceso, por tanto, entrego este triunfo a todos ellos.

Extiendo esta dedicatoria hacia Wellington Gonzáles, Diego Gonzáles y Jonathan Quevedo por el infinito amor demostrado a mi familia, al ayudarnos en nuestra rehabilitación. Brindarles un humilde Dios les pague por todo su sacrificio, ya que eso no se puede pagar con algo material. Dios siempre los bendiga a ustedes y a sus familias.

Duque Quevedo, Odalys Rashel

AGRADECIMIENTO

Agradezco a Dios por sobre todas las cosas, por la vida, por la salud mía y de mi familia, pues pese a todas las dificultades enfrentadas él nos dio la guía, el camino y siempre estaré en deuda por ello.

Agradezco a la prestigiosa Universidad de las Fuerzas Armadas “ESPE”, a mis docentes educadores que supieron compartir sus valiosos conocimientos para poder alcanzar esta meta.

Agradezco a mi tutor de tesis el Ing. Diego Ricardo Salazar Armijos, Ph.D., al dar su guía, paciencia y saberes, esperando que su vida sea llena de alegría y bendiciones siempre.

Duque Quevedo, Odalys Rashel

Índice

Índice de figuras

Índice de tablas

RESUMEN

El aumento de robos en los domicilios y la creciente inseguridad en Santo Domingo y el país, contempla preciso fortalecer los sistemas de seguridad para evitar el acceso no deseado a los hogares, que puede tener consecuencias fatales. El Internet de las cosas (IoT) junto con la Inteligencia Artificial (AI) ofrecen servicios mejorados para atacar esta problemática al incluir ramas de Machine Learning y Deep Learning. Por esta razón, se propone un esquema mejorado de seguridad al detectar personas con una mayor precisión con la ayuda del modelo YOLOv8x-pose en una cámara IoT de una casa prototipo, evitando saturar al usuario de notificaciones falsas y justificando la importancia necesaria a este aviso. La propuesta incluye un bot y un chat privado creados en Telegram para las notificaciones, enviadas desde Python y Firebase con Realtime Database y su Storage donde se carga previamente una imagen de la detección, así como videos en formatos MP4 y AVI. Entrega así también un sitio web con Angular para observar en tiempo real las detecciones por imagen y video, gestionando la autenticación para el acceso.

Palabras clave — Videovigilancia, detección de intrusos, aprendizaje automático, visión artificial, IoT

ABSTRACT

The increase in home robberies and the growing insecurity in Santo Domingo and the country requires strengthening security systems to prevent unwanted access to homes, which can have fatal consequences. The Internet of Things (IoT) together with Artificial Intelligence (AI) offer improved services to attack this problem by including branches of Machine Learning and Deep Learning. For this reason, an improved security scheme is proposed by detecting people with greater precision with the help of the YOLOv8x-pose model in an IoT camera of a prototype house, avoiding saturating the user with false notifications and justifying the necessary importance of this notice. The proposal includes a bot and a private chat created in Telegram for notifications, sent from Python and Firebase with Realtime Database and its Storage where an image of the detection is previously loaded, as well as videos in MP4 and AVI formats. It also delivers a website with Angular to observe image and video detections in real time, managing authentication for access.

Keywords: Video Surveillance, Intrusion Detection, Machine Learning, Computer Vision, IoT

I. INTRODUCCIÓN

El presente proyecto propone un sistema mejorado de seguridad con la implementación del modelo YOLOv8x-pose de Ultralytics sobre una cámara IoT, con la finalidad de mejorar las métricas de precisión en la detección de personas, lo que se consiguió mediante la programación del citado algoritmo y el envío de alertas en Python, para lo cual se empleó un sistema de almacenamiento de imágenes y videos en Firebase en formatos MP4 y AVI, los cuales se procesaron mediante el mencionado algoritmo para su posterior reenvío de alertas con un Bot de Telegram a un grupo privado que notifica al usuario sobre la intrusión detectada, para el efecto se incluyó una aplicación web en Angular que mantiene el registro de la detección mejorada de los intrusos con la cámara.

La seguridad de los hogares está amenazada por el aumento de la delincuencia, generando la necesidad de fortalecer los sistemas de control de acceso y vigilancia para prevenir potenciales amenazas. A medida que la tecnología avanza, la automatización de los hogares y el IoT se están convirtiendo en tendencia en diversos países.

Los sistemas mencionados no solo ofrecen servicios más avanzados, sino que también se benefician del auge de la AI, así como de sus disciplinas derivadas, como el Machine Learning y Deep Learning. Este proyecto propone como objetivo principal establecer un esquema de protección mediante la implementación de cámaras IoT, aprovechando la potencia de la Inteligencia Artificial. Esto permitirá mejorar las alertas preventivas, brindando a los usuarios la certeza de que cuentan con un sistema eficiente para detectar y prevenir accesos no autorizados a sus viviendas.

La integración de la Inteligencia Artificial en el ámbito de la seguridad del hogar representa un avance significativo en la capacidad de anticiparse a situaciones de riesgo. Con la capacidad de aprender y adaptarse, estos sistemas son capaces de proporcionar alertas más precisas y personalizadas, creando un entorno residencial más seguro y confiable para los usuarios.

La detección humana, se ha convertido en una tarea esencial en aplicaciones como seguridad, control de flujo de personas y vigilancia, donde se busca a partir de la visión artificial obtener de forma precisa la ubicación y posición exacta de la persona detectada.

El avance de las redes convolucionales marcó un hito en la detección de objetos, en especial con el uso de algoritmos de una sola etapa, como los modelos YOLO, que han comprobado ser veloces y exactos al predecir la imagen completa, así es el caso de la detección de personas, ejemplo de aquello es la versión mejorada YOLOv5 usada en la propuesta [1] para la detección de peatones considerando condiciones de tiempo y nivel de luz mediante un algoritmo denominado multiespectral, en el cual se construyen dos subredes que integran imágenes RGB y térmicas sobre bases de datos públicas de peatones que permiten el entrenamiento de las redes.

Actualmente YOLO lanzó la octava versión de su modelo ganándose su lugar como Inteligencia Artificial de última generación con visión de vanguardia [2], cabe destacar un poco de su historia nombrando a los desarrolladores Joseph Redmon y Ali Farhadi quienes lanzaron en 2015 esta tecnología [3], generando una tendencia por su rapidez y precisión, es por ello que ahora su última versión sorprende con una paleta de opciones respecto a detectar, segmentar, clasificar, seguir y estimar pose, cuya última opción se entrenó específicamente en personas.

Este trabajo explorará y analizará estas mejoras en la detección humana por estimación de pose, destacando la eficacia de los algoritmos YOLO mejorados y sus contribuciones a la precisión y velocidad en aplicaciones de seguridad y vigilancia [4]. Con la combinación de estas técnicas avanzadas, se busca proporcionar una visión más completa y detallada de la presencia humana en entornos específicos, mejorando así la capacidad de respuesta y la eficiencia de los sistemas de detección en tiempo real.

A. Antecedentes

Con los avances en tecnologías de aprendizaje profundo, los sistemas de videovigilancia han adoptado tecnologías de análisis de video basadas en AI para mejorar la seguridad. Lo que implica altos costos de procesamiento informático que las cámaras básicas no los pueden soportar y el fabricante recurre a ofrecer un servicio en la nube para que se haga uso de estos servicios de AI y puedan detectar adecuadamente a los intrusos humanos.

Wond & Kim [5] proponen en "Toward an Online Continual Learning Architecture for Intrusion Detection of Video Surveillance", proponen una arquitectura de videovigilancia híbrida de extremo a extremo para la detección confiable de objetos. La propuesta incluye inteligencia tanto en el front-end como en el back-end, debido a que los actuales sistemas se fundamentan en la variación de matices y temperatura del color para detectar movimientos.

Para la detección de personas actualmente se utilizan técnicas de aprendizaje automático con algoritmos que han evidenciado su eficiencia, por ejemplo: (YOLOF, YOLOX y TooD) en el conjunto de datos refrentes a la detección de postura humana en imágenes térmicas (IPHPDT) utilizando detectores de referencia (IPH-YOLOF, IPH-YOLOX e IPH-TOOD)[6]. Muestran que este detector de referencia logra una detección precisa de la pose humana en imágenes térmicas, con una precisión promedio del 70,4%. La introducción de IPHPDT busca fomentar más investigaciones sobre la detección de la posición humana en imágenes térmicas infrarrojas y llamar la atención sobre esta desafiante tarea.

La verificación y análisis de la actividad humana (HVAA), se utiliza principalmente para observar y monitorear patrones de movimiento humano mediante imágenes y videos RGB. Interpretar las interacciones humanas a través de imágenes RGB es una tarea compleja de aprendizaje automático basada en parámetros como la velocidad de detección, la posición y la dirección de los componentes del cuerpo humano. El enfoque propuesto por [7] en "Human Verification over Activity Analysis via Deep Data Mining" presenta un análisis de la actividad humana para el reconocimiento de eventos, destacando la extracción de características basada en inteligencia contextual, utiliza imágenes de interacciones humanas como datos de entrada, aplicando pasos de eliminación de ruido y análisis de persona a persona para mejorar la precisión.

Liu L, Jiao Y & Meng F [8]proponen en "Key Algorithm for Human Motion Recognition in Virtual Reality Video Sequences Based on Hidden Markov Model" analizar en detalle el reconocimiento de movimientos humanos en secuencias de vídeo de realidad virtual (VR) utilizando modelos ocultos de Markov. El proceso abarca desde la recopilación y el preprocesamiento de videos de realidad virtual, la detección de primer plano, la extracción de parámetros de características humanas hasta el reconocimiento de movimiento utilizando el modelo oculto de Markov.

B. Definición de la problemática

La inseguridad en Santo Domingo de los Tsáchilas, Ecuador, requiere de acciones que contribuyan a mitigar los robos y otros delitos en las viviendas. Una solución para contrarrestar esta problemática es el uso de cámaras de videovigilancia, sin embargo, el aplicar estos dispositivos no garantiza una detección de intrusos efectiva, pues requiere de monitorización continua, sintiendo entonces la necesidad de integrar una cámara inteligente que permita detectar movimiento y personas. Pese a existir esta alternativa, se evidencia una brecha en la precisión para lograr este fin.

Se figura entonces otro escenario a tomar en cuenta al aplicar cámaras inteligentes para mitigar la inseguridad, dado que, al no tener la precisión esperada para la detección de personas, se genera una cantidad considerable de notificaciones, que al ser tan frecuente, el usuario obvia el aviso por suponer una falsa alarma, dejando nuevamente abierta la puerta a esta problemática, sin obtener una respuesta esperada a la necesidad de neutralizar la inseguridad.

Según la Fiscalía General del Estado (FGE) [9], existen en el año 2021 un total de 62,584 denuncias por robo en Ecuador de las cuales 8,198 eran por robo a domicilios, representando el 12.9% de las denuncias en el país, así también en ese año Santo Domingo se situó como la séptima provincia más peligrosa, con 1,989 denuncias donde el 14.33% representaba a denuncias por robo a domicilios. Se analiza entonces los datos brindados por la FGE en el año 2022 donde evidencia respecto al año anterior un incremento del 27.06% de denuncias por robo y un 2.29% por robos a domicilio en el país, ubicando ese año a Santo Domingo en la sexta provincia más peligrosa, con un incremento respecto al 2021 del 66.97% por robos en general y en un 15.09% por robos a domicilios.

Es por tanto que se manifiesta la importancia de entregar una estrategia integral para solventar esta necesidad involucrando a las comunidades, autoridades locales e inyectando medidas de prevención, como es el mejorar la iluminación pública para evitar dar facilidad a cometer estos actos delictivos y salir sin consecuencias, el agregar un sistema de vigilancia para la comunidad fortaleciéndose con organismos de seguridad, e implementar tecnologías avanzadas como son las cámaras de videovigilancia inteligentes pero con mecanismos de alarma eficaces.

C. Justificación e importancia

Mejorar la detección de intrusos en cámaras de seguridad es de suma importancia debido a que las alertas falsas pueden hacer que los usuarios se descuiden de las notificaciones por recibir constantemente este tipo de alertas, lo que podría provocar robos u otros delitos a la propiedad. La tecnología adecuada puede ayudar a identificar y responder con precisión situaciones de riesgo, proporcionando mayor seguridad y protección a propiedades y personas.

Optimizar los métodos de alarma al mejorar la precisión para la detección de personas en las cámaras de videovigilancia es de total relevancia para evitar que el usuario ignore las notificaciones al considerarlas previamente como una alerta falsa, debido a la cantidad de avisos fallidos en su experiencia, entregando por tanto un sistema obsoleto para mitigar el riesgo de sufrir robos u otro tipo de delito en las viviendas. La integración de una tecnología avanzada que facilite la detección correcta de personas puede brindar el mecanismo para responder con precisión a la necesidad de notificar la intrusión en los domicilios evitando el robo en las viviendas y disminuyendo la inseguridad que cada vez aumenta por falta de métodos eficientes para enfrentarla.

Un algoritmo de visión artificial preciso contribuye a una respuesta rápida y eficiente en situaciones de seguridad, esto es crucial para prevenir incidentes delictivos, al tener un mayor número de aciertos (verdaderos positivos) sobre el total de predicciones.

La detección de personas es importante para diversas aplicaciones, como la vigilancia en áreas urbanas, la monitorización de infraestructuras críticas y la gestión del tráfico. Un algoritmo más efectivo mejora la utilidad y confiabilidad de estas aplicaciones, los avances en la detección de personas permiten una mejor integración con otros sistemas de seguridad, como el

reconocimiento facial o la identificación de comportamientos sospechosos, fortaleciendo así el ecosistema de seguridad en su conjunto, el uso y mejora de los algoritmos de detección de personas no solo fortalece la seguridad, sino que también tiene un impacto positivo en la eficiencia operativa, la integración con otros sistemas y el cumplimiento normativo. Estos avances contribuyen significativamente a la efectividad y utilidad de los sistemas de videovigilancia en diversos contextos.

D. Alcance

El presente proyecto tiene como alcance implementar un algoritmo de Deep Learning para mejorar la detección de intrusos empleando para ello una cámara inteligente con sensores de movimiento, en una casa prototipo en la Cooperativa Galo Plaza en Santo Domingo, Ecuador, el prototipo implementado pretende mejorar la capacidad del sistema para identificar intrusos - personas, permitiendo una detección más efectiva y reduciendo posibles falsas alarmas.

E. Objetivos

1) Objetivo General

Aplicar un algoritmo o modelo de Machine Learning condicionado en el porcentaje de predicción en la detección de personas para dispositivos IoT de una vivienda prototipo.

2) Objetivos específicos

- Establecer un esquema de seguridad mediante el uso de cámaras que incluyan sensores de movimiento para la detección de personas.
- Implementar un modelo de Machine Learning con el objetivo de mejorar los resultados obtenidos en la fase inicial para mejorar la precisión de predicción de personas.

F. Marco teórico

En 2023, Ecuador experimentó un aumento en la inseguridad. La tasa de homicidios aumentó en casi 16 por cada 100,000 habitantes en octubre. En el primer semestre de 2023, se registraron 3,599 homicidios intencionales, lo que equivale a un promedio diario de 19.72 casos. Ecuador se encuentra entre los 10 países con mayor criminalidad del mundo. El país podría superar su propio récord de muertes violentas, alcanzando una tasa de 40 homicidios por 100,000 habitantes.

La creciente problemática de inseguridad en los domicilios de Santo Domingo, Ecuador, implica desafíos importantes y produce temor en la comunidad. Este incremento puede atribuirse a diversos factores, como desigualdades socioeconómicas y falta de recursos para aplicar medidas de seguridad efectivas.

1) Internet de las Cosas

El IoT se define según [10] como el acoplamiento a través de internet de datos, procesos, cosas y personas, facilitando la interconexión de dispositivos y entornos. Se constituye a partir de sensores, actuadores, sistemas de red y procesadores de datos, abarcando ciudades, industrias y redes inteligentes, se puede observar su aplicación en focos, puertas, electrodomésticos, cámaras, domótica para la toma de decisiones inteligente [11].

2) Inteligencia Artificial

Según [12] la AI se define como la capacidad de los computadores de tomar decisiones imitando la inteligencia humana para sus acciones, utilizan por tanto algoritmos para aprender patrones en base a volúmenes significativos de datos, adaptándose y mejorando con la práctica, se visualiza su campo en el reconocimiento de imágenes, detección, clasificación, distribución y predicción, además de englobar el aprendizaje automático, profundo, procesamiento del lenguaje natural e incluyéndose en la robótica y visión por computadora.

3) Machine Learning

El aprendizaje automático es un campo de la inteligencia artificial que se centra en el desarrollo de algoritmos y modelos que permiten a las máquinas aprender patrones a partir de datos. Estos modelos pueden realizar tareas específicas sin estar programados explícitamente y mejorar su desempeño con la experiencia. Se divide en aprendizaje supervisado, no supervisado y por refuerzo. El aprendizaje automático tiene aplicaciones en una variedad de áreas, desde el reconocimiento de voz hasta el diagnóstico médico y las recomendaciones personalizadas [13]

a) El aprendizaje supervisado:

Según Russell & Norving [14], es un enfoque en el campo de la inteligencia artificial en el cual los algoritmos son entrenados utilizando un conjunto de datos que incluye ejemplos emparejados de entrada y salida. Durante el proceso de entrenamiento, el modelo aprende a mapear la entrada a la salida esperada, permitiéndole hacer predicciones o tomar decisiones cuando se enfrenta a nuevas entradas. Este método se caracteriza por la supervisión continua del modelo a través de la comparación entre sus predicciones y las salidas reales del conjunto de datos de entrenamiento.

b) El aprendizaje no supervisado:

Es una rama de la inteligencia artificial en la cual los algoritmos son entrenados sin tener acceso a datos etiquetados. A diferencia del aprendizaje supervisado, no se proporcionan salidas específicas durante el entrenamiento. En cambio, el modelo busca descubrir patrones, estructuras o relaciones intrínsecas en los datos de entrada sin guía externa. En este enfoque [14], el sistema debe identificar por sí mismo las características significativas y la organización subyacente en los datos.

4) Deep Learning

El Deep Learning (DL) se define como una rama de la AI y el ML que emplea redes neuronales artificiales (ANN) para procesar y aprender patrones complejos a partir de un gran volumen de datos, utiliza por tanto arquitecturas de redes neuronales multicapa para el procesamiento de estas cantidades de datos construyendo un nuevo modelo basado en datos, este enfoque presenta su tecnología ubicándola en la Industria 4.0 siendo efectivo en el análisis de sentimientos, procesamiento de lenguaje natural, inteligencia empresarial y reconocimiento visual [15].

5) La visión artificial

Se explica de una rama de la AI que usa algoritmos y un conjunto de tecnologías para el procesamiento de imágenes, cuyo objetivo es simular la capacidad visual de las personas. Busca que los dispositivos puedan comprender, analizar y tomar decisiones en base a datos visuales. Se puede distinguir en aplicaciones de diversos ámbitos para la detección, segmentación o clasificación de objetos [16].

a) Los algoritmos de visión artificial

Se conceptualizan por [17] como la capacidad de procesar datos visuales y adoptar la información relevante encontrada en imágenes o vídeos, son por tanto fundamentales en el campo de la visión por computador, permitiendo realizar tareas en relación con el reconocimiento de esquemas, detección y seguimiento de movimiento.

6) Los algoritmos de segmentación de imágenes

Están diseñados para analizar y dividir una imagen en regiones, se centran en identificar y delimitar áreas que comparten características visuales similares, englobando diversos algoritmos, incluyendo técnicas de umbralización, crecimiento de regiones y métodos basados en contornos [16].

a) Las técnicas de umbralización

Se define como la búsqueda de dividir una imagen al identificar umbrales de intensidad de píxeles, estableciendo límites para diferenciar regiones en base a la intensidad de la escala de grises o el color, se considera una herramienta esencial en el campo de la visión artificial [16].

b) El crecimiento de regiones

Se define como la capacidad de dividir una imagen identificando regiones contiguas con características visuales similares, donde su enfoque está en la agrupación de píxeles adyacentes que cumplen con criterios de similitud específicos, constituyendo así segmentos coherentes en la imagen, esta técnica es utilizada en aplicaciones de visión por computador, como son en el campo de la medicina para la segmentación de objetos o para identificar áreas homogéneas en imágenes diversas [16].

c) Las técnicas basadas en contornos

Son procesos diseñados para analizar y dividir una imagen identificando las fronteras de los objetos presentes y delinean las detenciones y cambios en la intensidad de los píxeles para definir límites precisos entre regiones, abarcan métodos como el contorno activo (o "snakes") y el contorno elástico [5].

d) Las técnicas de clasificación

Russell & Norving [14] las definen como enfoques computacionales que buscan asignar categorías o etiquetas a datos, basándose en patrones aprendidos a partir de conjuntos de entrenamiento. Estas metodologías se centran en desarrollar modelos capaces de reconocer y asignar nuevas instancias a categorías predefinidas. Las técnicas de clasificación son esenciales en aplicaciones diversas, desde reconocimiento de imágenes hasta procesamiento del lenguaje natural.

7) Algoritmos para la clasificación de imágenes con Python

Los distintos tipos de algoritmos para la clasificación de imágenes con Python abarcan diversas técnicas computacionales diseñadas para asignar categorías a datos visuales. Estos métodos, implementados en el lenguaje de programación Python, incluyen enfoques como redes neuronales convolucionales, máquinas de soporte vectorial y bosques aleatorios. Estos algoritmos se utilizan ampliamente en aplicaciones de reconocimiento de patrones visuales y análisis de imágenes [18]

a) Las redes neuronales convolucionales (CNN)

Se presencia en el análisis de datos visuales y se evidencia su implementación en el ámbito de la inteligencia artificial, están especialmente diseñados para reconocer patrones en imágenes a partir de la utilización de capas convolucionales que extraen características específicas, aplicadas en clasificación, segmentación y detección, destaca su capacidad para la captura de características locales [15].

b) Las máquinas de soporte vectorial (SVM)

Son ideales para la clasificación y regresión. Estas metodologías, también conocidas como Support Vector Machines, se destacan por su capacidad para encontrar límites de decisión óptimos entre diferentes categorías, utilizando vectores de soporte. Las SVM son ampliamente aplicadas en problemas de aprendizaje supervisado y tienen diversas aplicaciones, desde reconocimiento de patrones hasta análisis de datos complejos [19].

c) Los bosques aleatorios (RT)

Los bosques aleatorios también conocidos como Random Forests son ampliamente empleados en problemas complejos y variados, como el reconocimiento de patrones y análisis de datos [20], se caracterizan por construir múltiples árboles de decisión durante el entrenamiento y combinar sus resultados para mejorar la precisión y generalización del modelo.

8) Los métodos en el campo de visión por computadora

Se utilizan para analizar e interpretar datos visuales, englobando por tanto una variedad de técnicas, iniciando en algoritmos dese segmentación de imágenes hasta clasificación y reconocimiento de patrones, estos procedimientos buscan dotar a las máquinas de la capacidad para entender y extraer información valiosa a partir de imágenes o vídeos y encuentran su aplicación en diversas áreas, como es el del reconocimiento facial o el procesamiento de imágenes [16].

a) Algoritmos de detección de personas

Se tratan de mecanismos esenciales en sistemas de seguridad, suelen utilizar técnicas de visión artificial y Deep Learning, se presenta a continuación algunas técnicas comunes utilizadas por estos algoritmos:

- **Haar Cascades:** Este método utiliza clasificadores en cascada entrenados para detectar características específicas en imágenes, con estas propiedades el clasificador puede discernir entre regiones de la imagen que contienen el objeto y aquellas que no [21].
- **HOG (Histogram of Oriented Gradients):** Analiza la distribución de gradientes de intensidad en una imagen para identificar objetos, incluidas las personas.
- **CNN:** Son fundamentales en el DL y se entrenan para reconocer patrones complejos en imágenes, permitiendo una detección más precisa de personas.
- **YOLO (You Only Look Once):** Este es un tipo de arquitectura de red neuronal diseñada para detectar objetos en tiempo real en imágenes. Puede identificar y localizar personas en una sola pasada a través de la imagen.
- **SSD (Single Shot Multibox Detector):** Similar a YOLO, SSD es otra arquitectura que puede detectar múltiples objetos, incluidas personas.

Tabla 1

COMPARACIÓN DE LOS ALGORITMOS DE DETECCIÓN DE PERSONAS.

Modelo	Ventaja	Desventaja
HOG	Explican la estructura a partir de la extracción detallada de gradientes locales dentro de una ventana [22].	No se considera óptimo para la identificación en tiempo real pues su precisión disminuye y ocupa mucho rendimiento computacional [23]
HAAR CASCADES	Poco uso de recursos computacionales, fáciles de implementar.	Requieren un entrenamiento personalizado, y son sensibles a cambios de luz, posición, forma o tamaño lo que afecta su precisión.
CNN	Son flexibles y adaptables pues manejan una arquitectura de redes neuronales en Deep Learning.	Requiere grandes recursos computacionales su velocidad dependerá del tamaño de complejidad y la red.
SSD	Detección en tiempo real, versatilidad en cuanto a posición e iluminación y precisión alta.	Dificultad en la detección de objetos pequeños o deformados y necesidad de grandes recursos computacionales para su aplicación, tiempo de procesamiento dependiente a su enfoque de extracción de características en múltiples escalas (lento).
YOLO	Detección en tiempo real, capaz de segmentar, clasificar y estimar pose, alta precisión y velocidad de respuesta, nueva versión entrenada para la detección de personas según puntos clave.	Requiere grandes recursos computacionales por lo que puede ser lento, sin embargo, ofrece versiones adaptables a cada procesador para contrarrestar esta desventaja

9) Python

Es un lenguaje de programación de alto nivel y de propósito general ampliamente utilizado en la comunidad de desarrollo de software. Destaca por su sintaxis clara y legible, facilitando la escritura de código y la colaboración entre programadores. Python es versátil y soporta diversos paradigmas de programación, siendo aplicado en una variedad de campos como desarrollo web, análisis de datos, inteligencia artificial y automatización de tareas [24].

a) OpenCV

Es una biblioteca de visión por computadora de código abierto que proporciona herramientas y funciones para el procesamiento de imágenes y visión artificial. Desarrollada en C++ pero con extensiones para varios lenguajes, OpenCV se utiliza en una amplia variedad de aplicaciones, desde reconocimiento facial hasta segmentación de objetos. Es una herramienta esencial en la comunidad de visión por computadora debido a su versatilidad y capacidad para trabajar con diversas plataformas [25].

10) Cámaras IoT

Son dispositivos independientes de video vigilancia capaces de detectar movimiento, almacenar en la nube, tener visión nocturna, monitoreo remoto con imágenes real time, además de contar con notificaciones más precisas al detectar un movimiento, un sonido o incluso una persona dependiendo el tipo de cámara y el plan que se adquiera respecto al almacenamiento que en la mayoría de los casos puede ser por tarjeta Micro SD o en la nube.

a) YCC 360 PLUS OUTDOOR

Es una cámara de videovigilancia inteligente que tiene visión panorámica de 360º con gran cobertura de áreas, grabación de video en alta definición, detección de movimiento y sonido, visión nocturna, audio bidireccional, alertas de luz y sonido y el ser compatible con aplicaciones móviles para un fácil control a partir de un dispositivo inteligente, permite configurar un área caliente y determinar el horario activo de la cámara además de opciones de grabación por notificaciones o video continuo.

b) TAPO C200

Es una videocámara TP-Link inteligente de tipo Pan/Tilt Wifi referente a cobertura en movimiento horizontal de 360º y vertical de 114º, tiene visión nocturna de 850nm con distancia de hasta 9m, video de alta definición, notificaciones y detección de movimiento, sonido e incluso personas, seguimiento inteligente, audio bidireccional, comunicación tipo llamada, alarmas de luz y sonido, almacenamiento por tarjeta Micro SD y por la nube configurando sea por notificaciones o en video continuo.

II. METODOLOGÍA

El proyecto se fundamentó en una investigación aplicada, cuyo objetivo es establecer una solución en virtud de las necesidades de problema para un individuo o comunidad, centrándose en hallar y consolidar conocimiento para su aplicación [13], busca abordar estos desafíos mediante la implementación de algoritmos o un modelo de detección de personas que incorpore técnicas avanzadas de aprendizaje automático y visión por computadora. La clave es aplicar un modelo que distinga con precisión la presencia humana, reduciendo al mínimo los casos de falsos positivos.

En cuanto a la propuesta para el aplicativo se consideró pertinente el uso la metodología eXtreme Programming (XP), en virtud de que es una metodología ágil reconocida por su enfoque colaborativo y flexible [26] y [27], se alinea perfectamente con los desafíos dinámicos y las demandas cambiantes de proyectos en constante evolución como el que nos ocupa.

El esquema presentado en la Fig. 1 indica el proceso de detección de personas aplicando el modelo YOLOv8x-pose en Python tomando de entrada la cámara de prueba, en donde al detectar una persona con el modelo se envía automáticamente a Firebase Realtime Database una imagen de esa detección, además de videos en formato AVI y MP4, se adjuntan dichos archivos en el Storage de Firebase, para luego enviar desde Firebase a un bot en Telegram un mensaje, notificación o alerta de haber detectado un intruso adjuntando la imagen en la detección y el video en tiempo real de la detección. Finalmente, en la parte del FrontEnd, se maneja Angular para observar los datos recibidos en Firebase con una interfaz amigable al usuario.

De acuerdo con la delimitación del problema y considerando las necesidades establecidas para la detección de intrusos y establecimiento de alerta, se considera la siguiente propuesta metodológica:

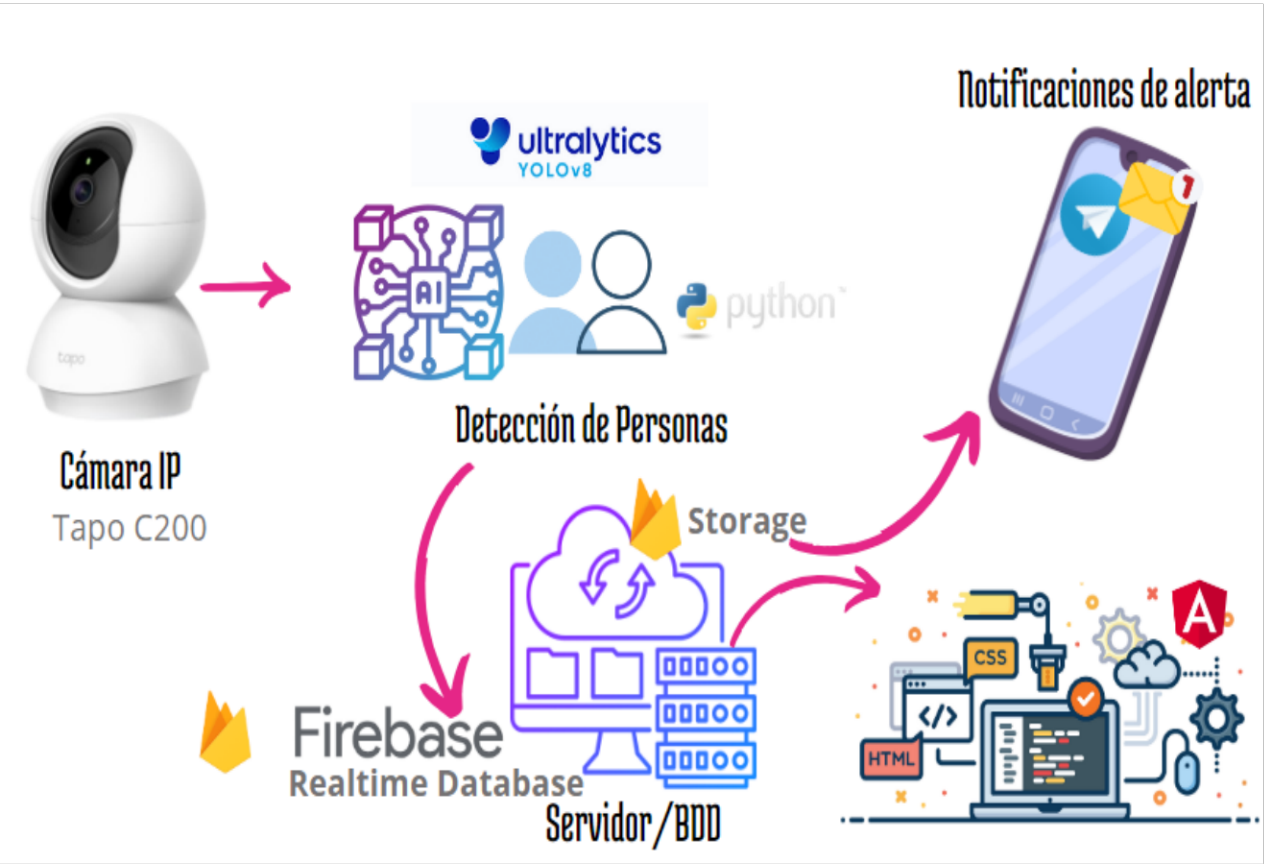


Fig. 1. Esquema para la detección de intrusos.

La propuesta utiliza el modelo pre entrenado YOLOv8x-pose de Ultralytics, que es específico para la detección de personas, cuyo algoritmo ofrece una predicción y detección profunda de estos, mediante etiquetas y puntos clave del objeto en cuestión.

La combinación de Firebase para la visualización de datos en tiempo real con el envío de las notificaciones en Telegram y la creación del aplicativo web en Angular junto con la detección de objetos mediante YOLO proporciona una plataforma poderosa para la creación de aplicaciones web interactivas y análisis de datos en tiempo real. Esta integración permite una variedad de aplicaciones, desde sistemas de vigilancia y seguridad hasta análisis de tráfico y monitoreo, con el potencial de expandirse aún más con el desarrollo continuo en estos campos.

Con el propósito de estimar la eficiencia de la propuesta para detectar intrusos, se llevaron a cabo las actividades:

A. Fase1: Evaluación del estado base de la detección de intrusos en la cámara de prueba:

Se realizó una evaluación del estado base de la detección de intrusos en la cámara de prueba Tapo C200, la cual cuenta con un sensor de movimiento integrado, además alertas de detección de personas, cabe destacar que este modelo de cámara maneja tres puntos clave para las notificaciones en relación a movimiento, llanto de bebé y personas, sin embargo base a tener toda esta tecnología integrada, la efectividad para la detección de intrusos no es la esperada y satura de notificaciones al usuario con la necesidad de este servicio, siendo por tanto obligatorio el examinar los resultados analizando las alertas en verdaderos positivos, falsos positivos, verdaderos negativos y falsos negativos para evidenciar el panorama inicial a solucionar.

En base a lo anterior se construyó una matriz de confusión inicial que reflejó con claridad la efectividad y limitaciones del sistema actual. En este caso se muestra la matriz inicial en la Fig. 40 de la sección de Resultados para las pruebas hechas en un día típico efectuado el 22 de diciembre de 2023.

B. Fase 2: Implementación de un modelo o algoritmo para mejorar la detección de intrusos:

En esta etapa, se implementó el modelo de detección YOLOv8x-pose, el cual se trata de un modelo que usa puntos clave o referencias para representar distintas partes de un objeto donde su ubicación se representa en conjuntos 2D o 3D para estimación de pose, además este modelo viene entrenado solo con un tipo de detección dirigido netamente a personas, por lo cual es ideal en nuestro proyecto, destacando que la única dependencia a instalarse es Ultralytics YOLO [28], se realizaron pruebas para verificar su eficiencia. En la sección de Resultados Fig. 45, se muestra una segunda matriz de confusión para comparar los resultados obtenidos después de implementar el algoritmo y evaluar su impacto en la precisión de la detección.

C. Fase 3: Implementación de Firebase para almacenamiento de detecciones en tiempo real

Se creó una base de datos NoSQL en la nube Firebase Realtime Database para que permita el almacenamiento y la sincronización de datos en tiempo real incluyendo el Storage de Firebase,

con una integración sencilla con aplicaciones web o móviles a través de SDKs para varias plataformas. Se visualiza la creación del proyecto en la Fig. 2 y las configuraciones detalladas desde la Fig 3 a la Fig. 5. Los datos se sincronizan automáticamente entre los clientes conectados en tiempo real, lo que facilita la visualización de datos actualizados instantáneamente.

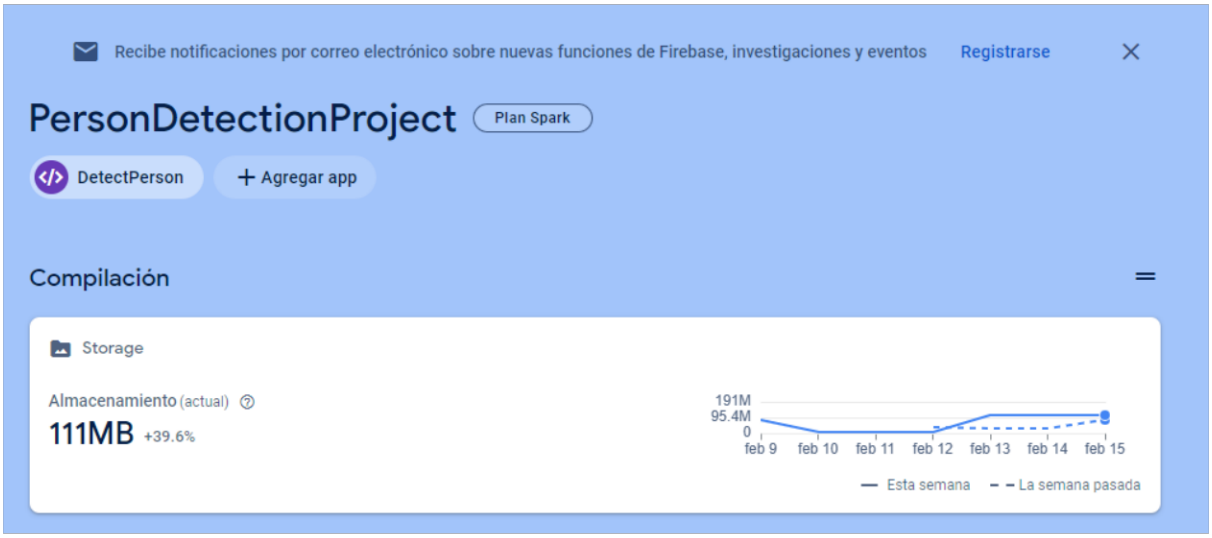


Fig. 2. Creación del proyecto PersonDetectionProject y de la aplicación web DetectPerson en Firebase.

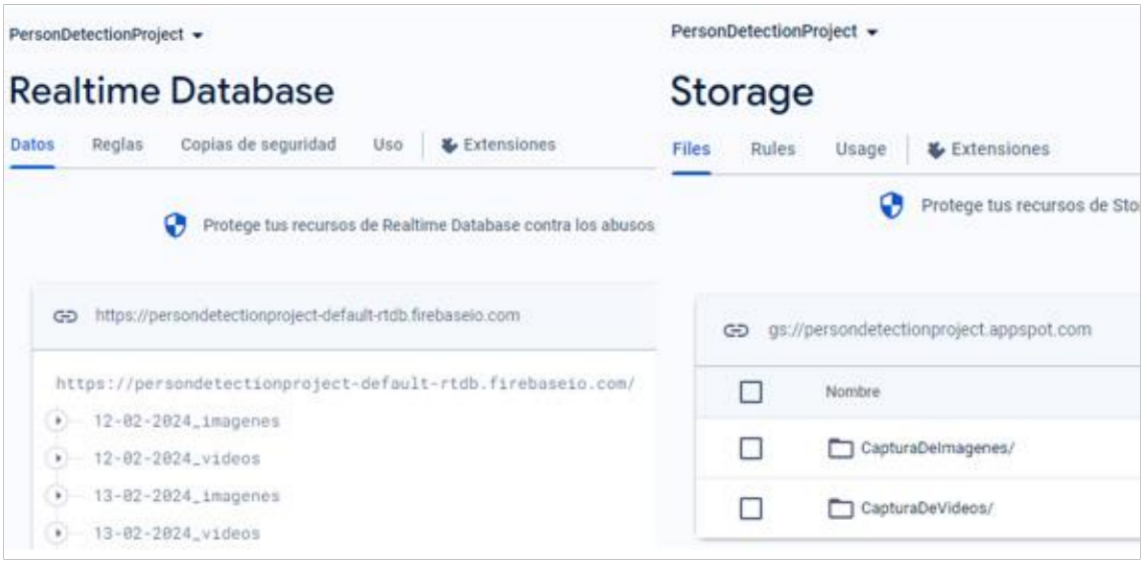


Fig. 3. Creación del Realtime Database y Storage en Firebase.

Fig. 4. Inclusión de la autenticación para el manejo del proyecto.

Fig. 5. Cambio de las reglas en la base de datos en tiempo real para verificar la autenticación al acceder a los datos.

Se procede a establecer la lógica para la detección con YOLO en Python en la Fig. 9, y se incluyen las SDK necesarias con la configuración de Firebase de nuestra app y nuestro proyecto para poder hacer uso de nuestra base de datos en tiempo real y de nuestro Storage en Python, incluyendo inclusive en estos pasos la autenticación para poder manipular y enviar nuestras detecciones con el algoritmo a Firebase, por lo cual descargamos de Firebase las credenciales e instalamos la biblioteca firebase_admin para realizar tareas administrativas y la biblioteca pyrebase para las operaciones CRUD, se puede observar la configuración correspondiente en las Fig. 6, 7, 8 y 9.

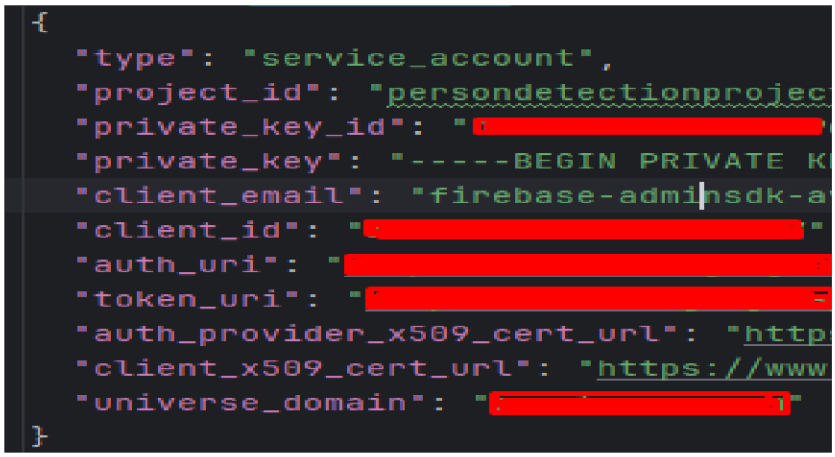


Fig. 6. Credenciales del Proyecto de Firebase en Python.

```
import firebase_admin
from firebase_admin import credentials, db
import pyrebase
```

Fig. 7. Importaciones necesarias para integrar Firebase en Python.

```python
# Configuración de Firebase
cred = credentials.Certificate('c.json')

firebase_admin.initialize_app(cred, options: {
    'databaseURL': config_env['DATABASE_URL'],
    'databaseAuthVariableOverride': {
        'uid': config_env['CUSTOM_TOKEN']
    }
})

firebaseConfig = {
    "apiKey": config_env['API_KEY'],
    "authDomain": config_env['AUTH_DOMAIN'],
    "databaseURL": config_env['DATABASE_URL'],
    "projectId": config_env['PROJECT_ID'],
    "storageBucket": config_env['STORAGE_BUCKET'],
    "messagingSenderId": config_env['MESSAGING_SENDER_ID'],
    "appId": config_env['APP_ID'],
    "measurementId": config_env['MEASUREMENT_ID'],
    "serviceAccount": "c.json"
}
#Incializacion en la app de Firebase
firebase = pyrebase.initialize_app(firebaseConfig)

#Referencia a la BDD y al Storage de Firebase
db_ref=db.reference('/')
storage = firebase.storage()
```

Fig. 8. Integración de Realtime Database con autenticación y el Storage de Firebase en Python.

```
model=YOLO("yolov8x-pose.pt")
#model.export(format="openvino")
ov_model=YOLO("yolov8x-pose_openvino_model/")
classNames=["person"]
# Empezamos
while True:
    success, img = cv2.VideoCapture(rtsp_url).read()
    if not success:...
    if grabacion:
        out.write(img)
        out2.write(img)
        fecha = datetime.now().strftime("%d-%m-%Y")

    results = ov_model(img, show=True, conf=0.6)
    annotated_frame = results[0].plot()

    for r in results:
        boxes = r.boxes
        for box in boxes:
            if not grabacion:...

            if  (grabacion and ((datetime.now()-tiempo_detect).total_seconds()>120)):...

    if not results[0] and grabacion and ((datetime.now() - tiempo_detect).total_seconds() > 10):...

    # Leemos el teclado
    if cv2.waitKey(1) & 0xFF == ord('q'):...

cv2.VideoCapture(rtsp_url).release()
cv2.destroyAllWindows()
```

Fig. 9. Implementación del modelo YOLO.

Se destaca también el formato RTSP para la visualización en tiempo real de la cámara por lo cual al tener dos tipos de cámara de prueba en modelo Tapo C200 y YCC365 Plus Outdoor, se considera, el formato RTSP correspondiente para ambos modelos resaltando de manera general la siguiente sintaxis rtsp://usuario:contraseña@dirección_ip:puerto/canal.

Se muestra el proceso para la creación de cuenta en la cámara de prueba TapoC200 en la Fig. 10, habilitando el uso de RTSP.

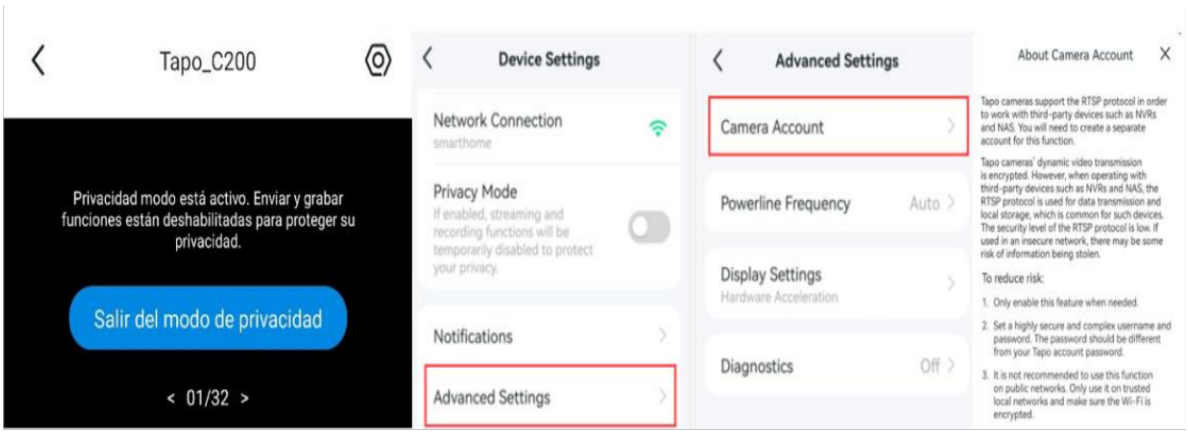


Fig. 10. Creación de cuenta en tapo para el uso de rtsp.

Debido a la necesidad de acceso remoto para la visualización de la cámara y uso del algoritmo en tiempo real, se realizaron como se visualiza en la Fig. 11 las pruebas correspondientes para el reenvío de puertos en nuestro router, sin embargo, pese a la configuración este reenvío se dirigía netamente a la IP estática del router y no a la IP pública como se pretendía.

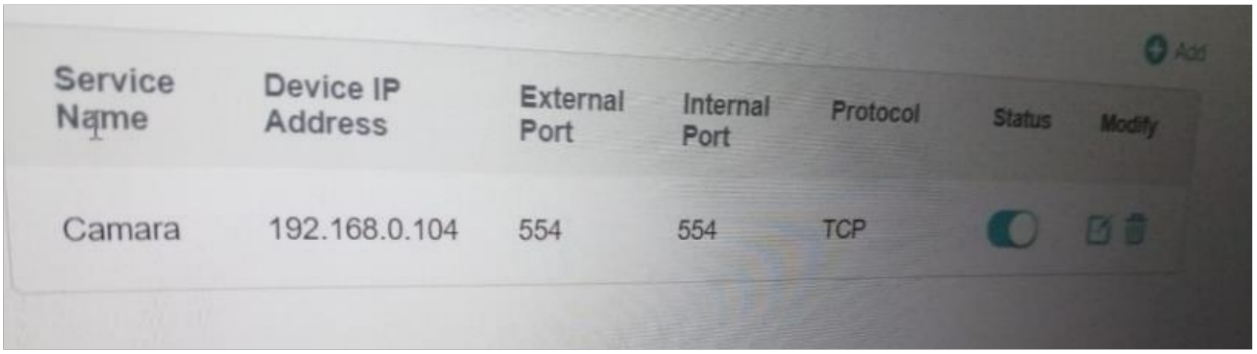


Fig. 11. Reenvío de puertos en el router con plan hogar.

Nota: En esta imagen se muestra la configuración del reenvío de puertos teniendo previamente mi IP en dinámica y estableciendo en zona de MZ a la IP estática de mi cámara, pero el reenvío solo se realizó para la IP del router mas no la IP pública asignada por mi proveedor.

Se visualiza a continuación en la Fig. 12 y 13 la cámara en tiempo real por el reproductor VLC a partir del RTSP con la IP local de la cámara.

Fig. 12.Visualización de la cámara por RTSP en VLC.

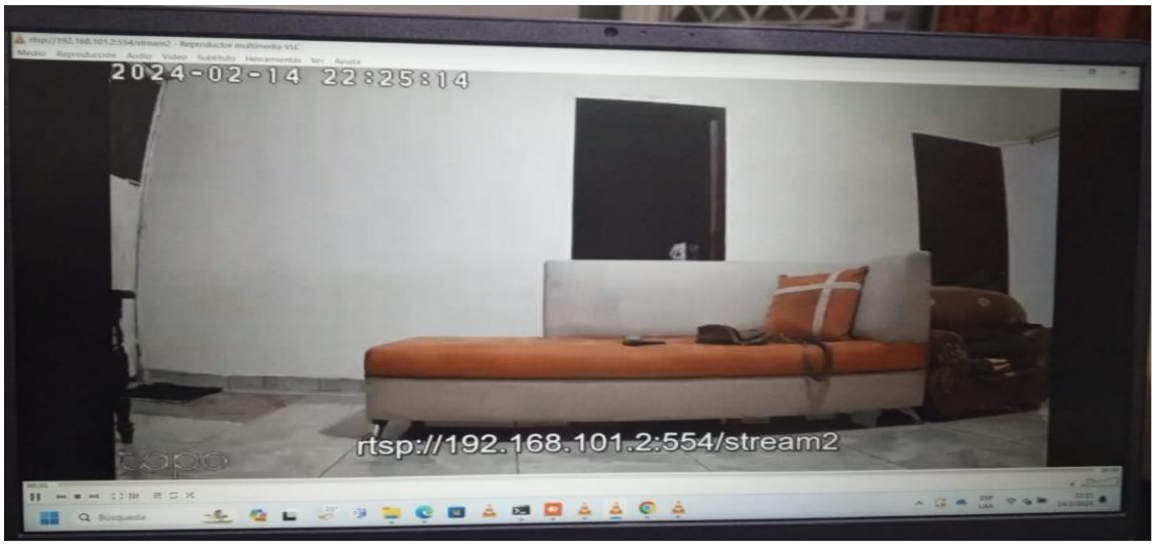


Fig. 13. Visualización de la cámara por la IP del router aplicando el reenvío de puertos.

Se creó en la Fig. 14 un dominio con No-Ip donde actualiza la IP pública del router y redirige a partir del dominio al puerto, considerando el cambio constante de la IP, dado que al colocarla directamente no se visualizaba como se presencia en la Fig. 15, sin embargo, la respuesta de igual manera resulto insatisfactoria debido a que no se habilitaban los puertos.

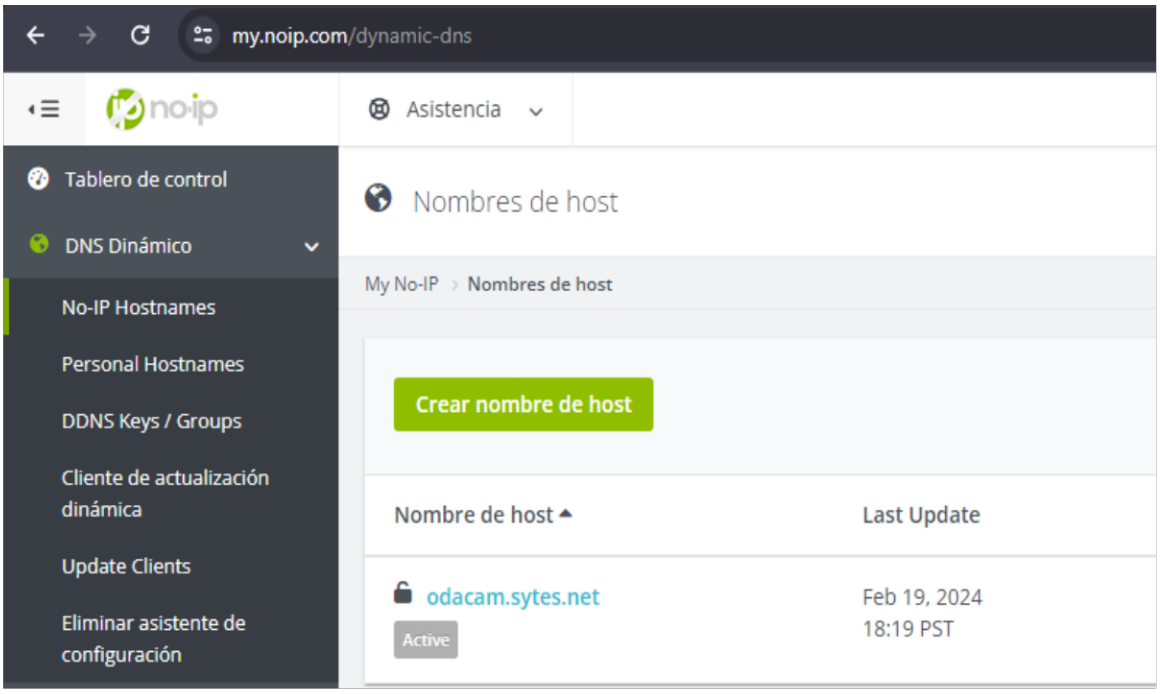


Fig. 14. Creación de dominio con No-Ip.

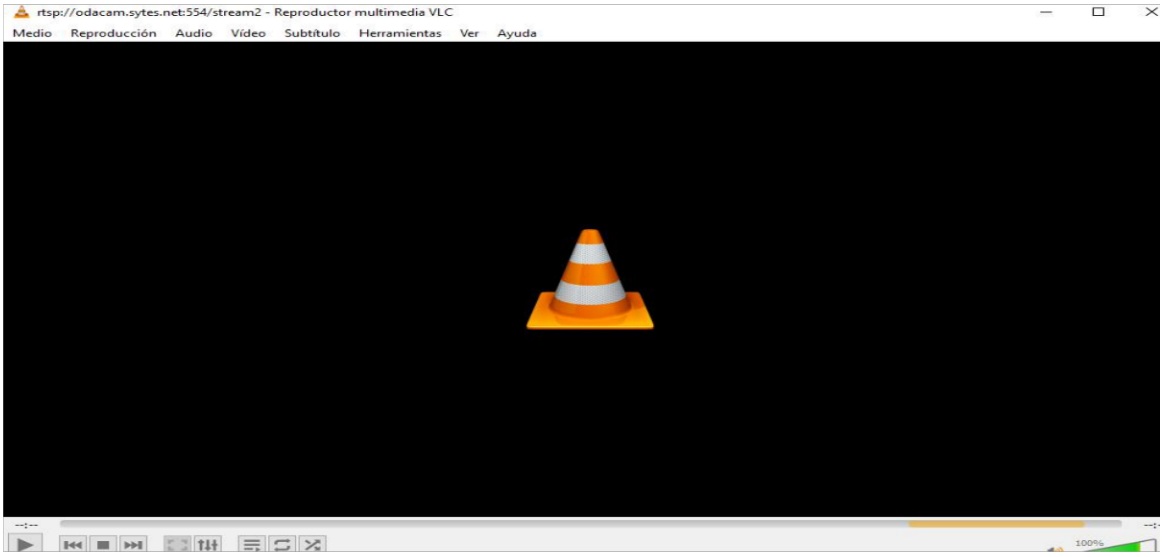


Fig. 15. Uso del dominio para visualizar la cámara con respuesta insatisfactoria.

Se observa en las Fig. 16 y 17 el intento de habilitación de puerto por mi proveedor de internet, para mi IP pública, pero sin resultados esperados:

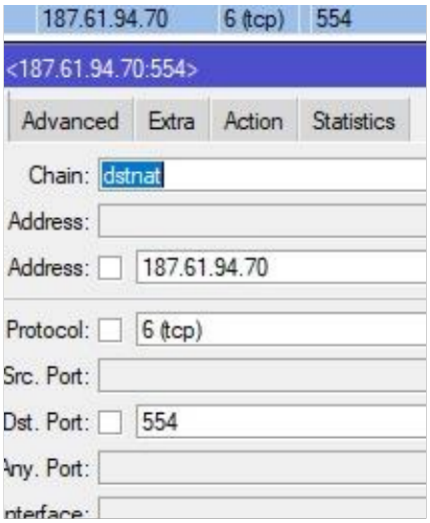


Fig. 16. Primer intento para habilitar puerto por la IP remota de router con un plan hogar.

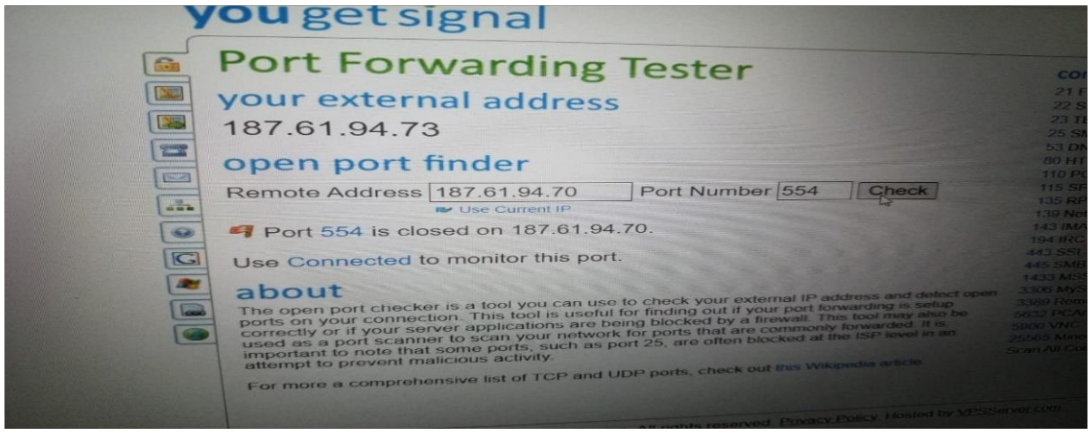


Fig. 17. Comprobación de habilitación de puerto insatisfactoria.

Se requirió un cambio de plan hogar a corporativo para poder abrir los puertos de manera respectiva en las cámaras. Se procedió por tanto a la instalación de la MikroTik por parte de nuestro proveedor y a comprobar la habilitación de puertos con nuestra nueva IP remota validando finalmente su acceso remoto, se muestra a continuación en las Fig. 18 y 19 la configuración para una de las cámaras de prueba realizando el reenvío de puertos y su visualización en la Fig. 20.

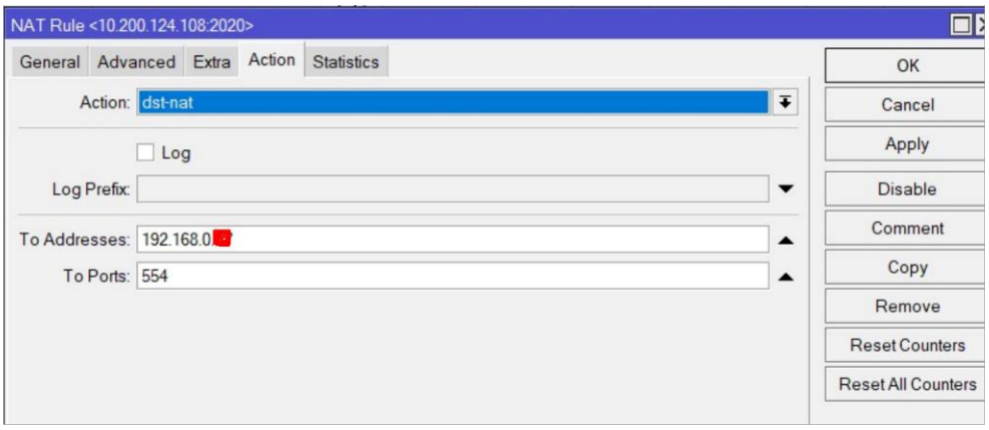


Fig. 18. Reenvío de puertos desde la IP de la cámara y su puerto interno.

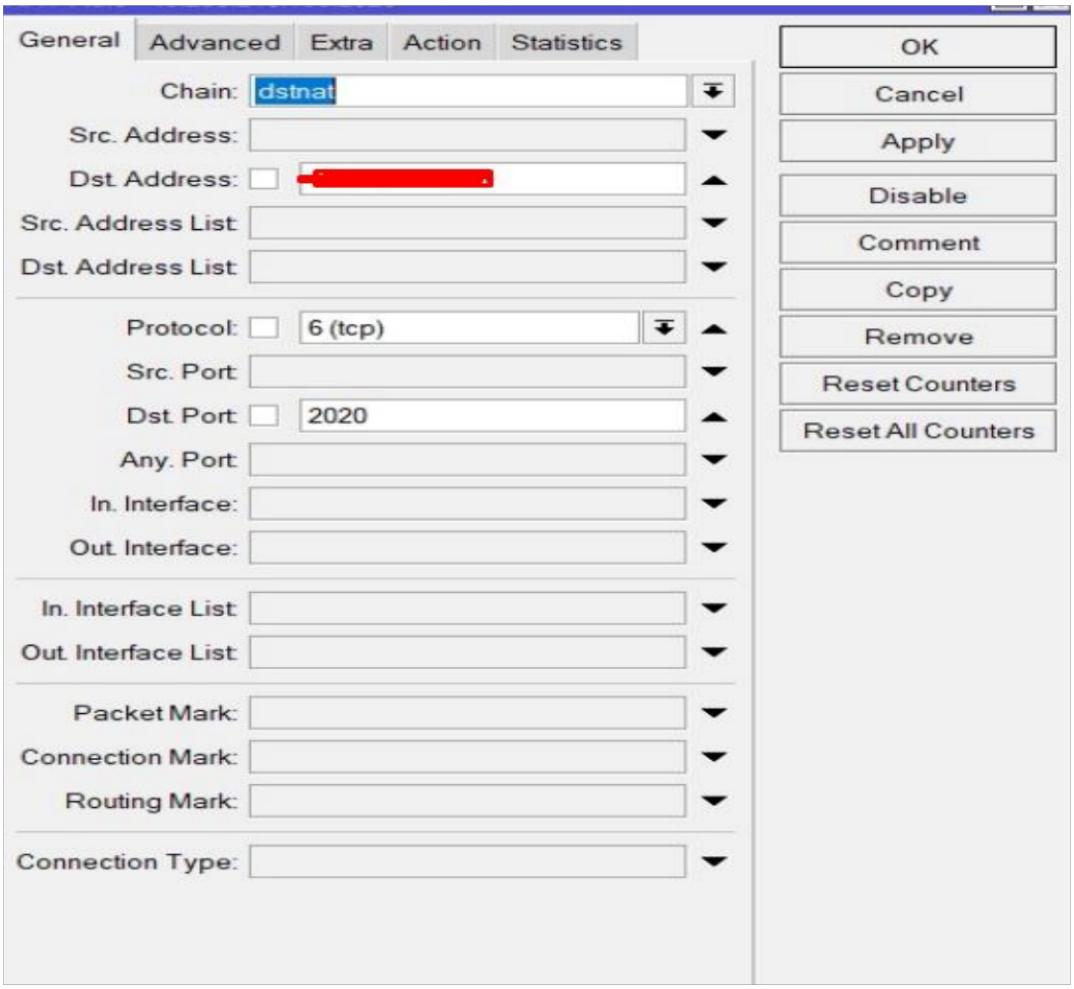


Fig. 19. Reenvío de puertos a la IP remota con su puerto externo respectivamente.

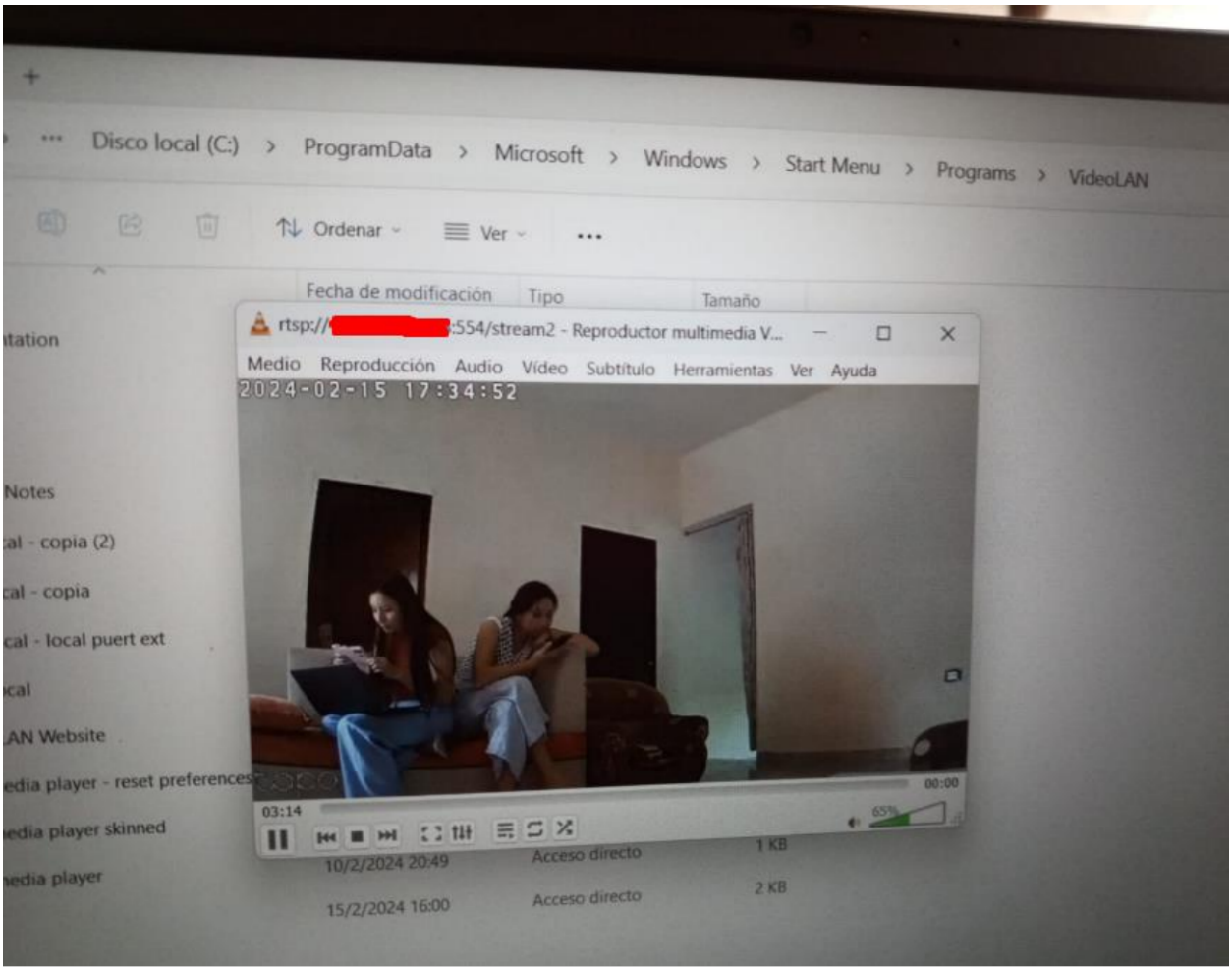


Fig. 20.Visualización de la cámara por el puerto externo y la IP remota.

Se implementó un mecanismo de normalización para el envío de datos a la base de datos en tiempo real con el objetivo de evitar el almacenamiento de múltiples detecciones en un corto período de tiempo. En lugar de enviar cada detección individualmente de forma continua, se estableció un tiempo de espera entre detecciones para evitar la sobrecarga de datos y reducir la duplicación de imágenes y notificaciones posteriormente.

Después de cada detección de intrusos, se configuró el envío de una imagen a Firebase con la primera detección como se observa en la Fig. 21. Se determinó un período de espera durante el cual el sistema no procesaría ni almacenaría nuevas detecciones en imagen. Esto aseguró que solo se enviaran a la base de datos y se notificaran las detecciones relevantes, reduciendo el ruido y la redundancia de datos.

```
image_path = f'CapturaDeImagenes/{timestamp}.jpg'
imagen_cap = f'C:/Users/Usuario/Documents/PeopleDetectionProject_Backend/CapturaDeImagenes/{timestamp}.jpg'
URL_TELEGRAM = f'http://api.telegram.org/bot{TOKEN_TELEGRAM_BOT}'
cv2.imwrite(image_path, annotated_frame)
storage.child(image_path).put(imagen_cap)
fecha = datetime.now().strftime("%d-%m-%Y")
imagen_url=storage.child(image_path).get_url(imagen_cap)
db_ref.child(f'{fecha}_imagenes').push({'timestamp': timestamp,
                                        'imagen_url': imagen_url})
```

Fig. 21. Envío de imagen a Firebase al detectar una persona a partir del modelo YOLOv8x-pose.

Durante el tiempo de espera entre detecciones, el sistema recopiló y almacenó capturas de detección en un búfer temporal. Al final del período de espera, se utilizó este búfer para crear un video que capturo toda la secuencia de detección durante ese lapso.

Después de analizar el modo de detección de la tapo C200, se observó que el video no se corta inmediatamente después de que ya no detecta a una persona. Continúa grabando durante un tiempo prolongado, incluso cuando la persona ya no está presente. Para abordar este problema, se decidió modificar el código para que las imágenes se envíen inmediatamente al detectar a un intruso con el tiempo de espera explicado, mientras que los videos se inicien al detectar al intruso, envíen la imagen de detección y se detengan si ya no se detecta a la persona intrusa o si ha transcurrido un tiempo de 2 minutos desde el inicio de la detección.

Si la persona sigue presente en la detección después de los 2 minutos, el video se enviará. Y volverá al bucle para detectar y enviar la primera detección en una imagen e iniciar la grabación, sin embargo, si la persona se retira antes de los 2 minutos, el algoritmo dejará de grabar después de esperar 5 segundos adicionales a la falta de detección. Esto asegura que los videos sean precisos y relevantes, y no incluyan tiempo innecesario sin la presencia de la persona detectada.

Una vez finalizado el video, se envía a Firebase garantizando el registro oportuno de todas las detecciones de intrusos para su posterior análisis y seguimiento.

D. Fase 4: Integración con Plataformas de Comunicación para notificaciones:

Con el objetivo de ofrecer un mecanismo efectivo de alerta en la detección de intrusos, se establecieron notificaciones automáticas hacia los dispositivos móviles. Estas notificaciones se

enviarán a través de Telegram. Esta medida permitirá una comunicación instantánea sobre los eventos detectados, lo que facilitará una respuesta rápida ante posibles intrusiones.

Se realizó por tanto la creación de un Bot en Telegram denominado BotCamaraIoT el cual recibirá las notificaciones enviadas a partir de Python y obteniendo tanto imagen como video de Firebase para enviar la notificación.

Se procedió entonces a solicitar la creación de un nuevo bot a partir de Botfather colocando el comando /newbot e ingresando en este caso el nombre que se requiere para nuestro bot OdalysBot.

Se observa en la Fig. 22 el Bot creado para este propósito, denominado BotCamaraIot y cuyo nombre de usuario fue OdalysBot.

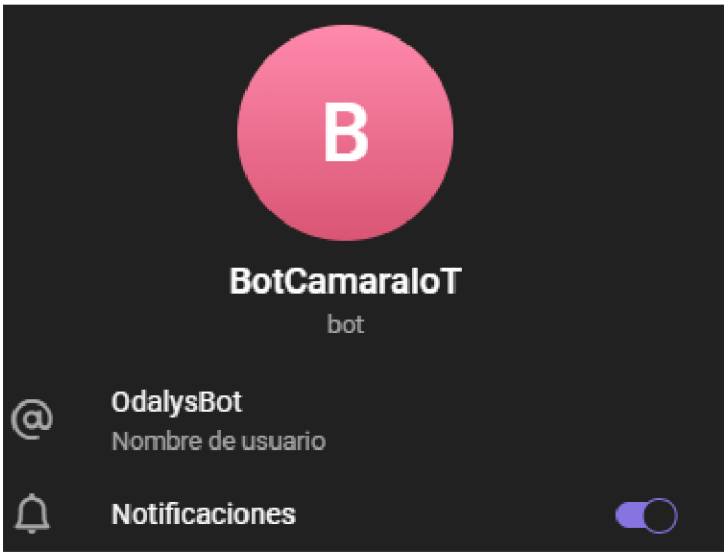


Fig. 22. Creación del BotCamaraIoT en Telegram.

Luego de haber creado el Bot en Telegram se procede en Python a la instalación de los paquetes de Telebot e importación de telebot, se solicita al Bot Father el token de acceso a nuestro Bot para manejarlo en Python. Así también se obtiene del Bot Rose el Id de nuestro chat para poder enviar mensajes a nuestro Bot, se visualiza lo descrito en la Fig. 24.

Se importan a su vez los paquetes de requests para inicializar el bot y descargar la imagen y el video de Firebase desde la URL como se visualiza en la Fig. 23, validando que si el status_code

es 200 envié por send_photo o send_video la notificación tal como se observa en la Fig. 25 y 26, se observan las detecciones recibidas en nuestro Bot en la Fig. 27.

```
import os
import requests
import telebot
```

Fig. 23. Importación de librerías.

```
TOKEN_TELEGRAM_BOT = config_env['TOKEN_TELEGRAM_BOT']
ChatId2=config_env['CHAT_ID_BOT'] #Chat del Bot
ChatId=config_env['CHAT_ID_GROUP'] #Chat del Grupo Privado
bot = telebot.TeleBot(TOKEN_TELEGRAM_BOT)
#bot.set_webhook()
```

Fig. 24. Uso del token de acceso para la autenticación en Realtime Database.

```
99          # Descargar la imagen desde la URL
100         imagen_response = requests.get(imagen_url)
101         # Verificar si la descarga fue exitosa
102         if imagen_response.status_code == 200:
103             # Enviar el video como un documento al chat de Telegram
104             bot.send_photo(ChatId, imagen_response.content,
105                            caption='Se detecto una persona revisa la imagen!')
106             os.remove(imagen_cap)
107         else:
108             print("Error al descargar la imagen")
```

Fig. 25. Descarga y verificación de la imagen para envío al chat del bot o del grupo en Telegram. Nota: A partir del método send_photo del bot se envía la imagen verificando previamente su descarga de Firebase con status_code y a su vez, una vez enviada la imagen, se remueve localmente para evitar el almacenamiento en nuestro computador.

```
# Descargar el video desde la URL
video_response = requests.get(video_url)
# Verificar si la descarga fue exitosa
if video_response.status_code == 200:
    # Enviar el video  al chat de Telegram
    bot.send_video(ChatId, video_response.content, caption='Se detecto una persona!')
    os.remove(video_cap)
    os.remove(video_cap2)
else:
    print("Error al descargar el video")
```

Fig. 26. Descarga y verificación del video para envío al chat del bot o del grupo en Telegram.

Nota: A partir del método send_video del bot se envía el video subido en formato mp4 en Firebase verificando previamente su descarga desde la url correspondiente y validando con status_code. Finalmente, al ser enviado el video, se remueve los archivos en formato avi y mp4 del almacenamiento local

Fig. 27. Detecciones realizadas con YOLOv8x-pose y envío de notificaciones a Telegram.

E. Fase 5: Visualización de Datos a partir de Angular

Se procede con la instalación del Angular versión 16 y a la integración de @angular/fire para el uso de Firabase en nuestro frontend. Se agregó por tanto las herramientas utilizadas cómo fueron Cloud Storage, Authentication y Realtime Database, se observa en la Fig. 28 lo especificado.

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { environment } from '../environments/environment';
import { provideAuth, getAuth } from '@angular/fire/auth';
import { LoginComponent } from './components/login/login.component';
import { SignUpComponent } from './components/sign-up/sign-up.component';
import { DashboardComponent } from './components/dashboard/dashboard.component';
import { AngularFireModule } from '@angular/fire/compat';
import { provideDatabase,getDatabase } from '@angular/fire/database';
import { provideStorage,getStorage } from '@angular/fire/storage';

@NgModule({
  declarations: [
    AppComponent,
    LoginComponent,
    SignUpComponent,
    DashboardComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    AngularFireModule.initializeApp(environment.firebase),
    provideAuth(() => getAuth()),
    provideDatabase(() => getDatabase()),
    provideStorage(() => getStorage())
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

Fig. 28. Creación del entorno en angular

Se procede en la Fig. 29 a la creación del servicio de autenticación con Firebase para lo cual se importa AngularFireAuth con su decorador de Injectable para permitir inyectar este servicio en otros componentes, se agrega NgZone para detectar los cambios y el servicio de Router para la navegación en distintas vistas y el uso de observables para guardar el estado de inicio de sesión y salida, se toma por parámetros el correo y contraseña de manera respectiva tanto para el Log-In como para la creación de la cuenta, finalmente se crea el último método de Log-Out para salir de la sesión iniciada.

```
import { Injectable, NgZone } from '@angular/core';
import { AngularFireAuth } from '@angular/fire/compat/auth';
import { Router } from '@angular/router';

@Injectable({
  providedIn: 'root'
})
export class AuthService {

  userData: any;

  constructor(
    private firebaseAuthenticationService: AngularFireAuth,
    private router: Router,
    private ngZone: NgZone
  ) {
    this.firebaseAuthenticationService.authState.subscribe((user) => {
      if (user) {
        this.userData = user;
        localStorage.setItem('user', JSON.stringify(this.userData));
      } else {
        localStorage.setItem('user', 'null');
      }
    })

  }

  logInWithEmailAndPassword(email: string, password: string) {···
  }

  signUpWithEmailAndPassword(email: string, password: string) {···
  }

  observeUserState() {···
  }

  //regresa true cuando el usario esta loggeado
  get isLoggedIn(): boolean {···
  }
```

Fig. 29. Creación del servicio de autenticación

Se procedió a realizar la creación de componentes en angular respectivos al inicio de sesión con la autenticación de Firebase y la creación de una cuenta, además del apartado para la visualización de los datos respectivos en tiempo real con la base de datos de Firebase, se observa a continuación en las Fig. 30 y 34 el Login con el correo autenticado previamente.

```typescript
import { Component } from '@angular/core';
import { AuthService } from 'src/app/shared/services/auth.service';

@Component({
  selector: 'app-login',
  templateUrl: './login.component.html',
  styleUrls: ['./login.component.css']
})
export class LoginComponent {

  constructor(private authService: AuthService) {

  }

  logIn(email: string, password: string) {
    this.authService.logInWithEmailAndPassword(email, password);
  }
}
```

Fig. 30. Lógica de inicio de sesión utilizando el servicio de autenticación

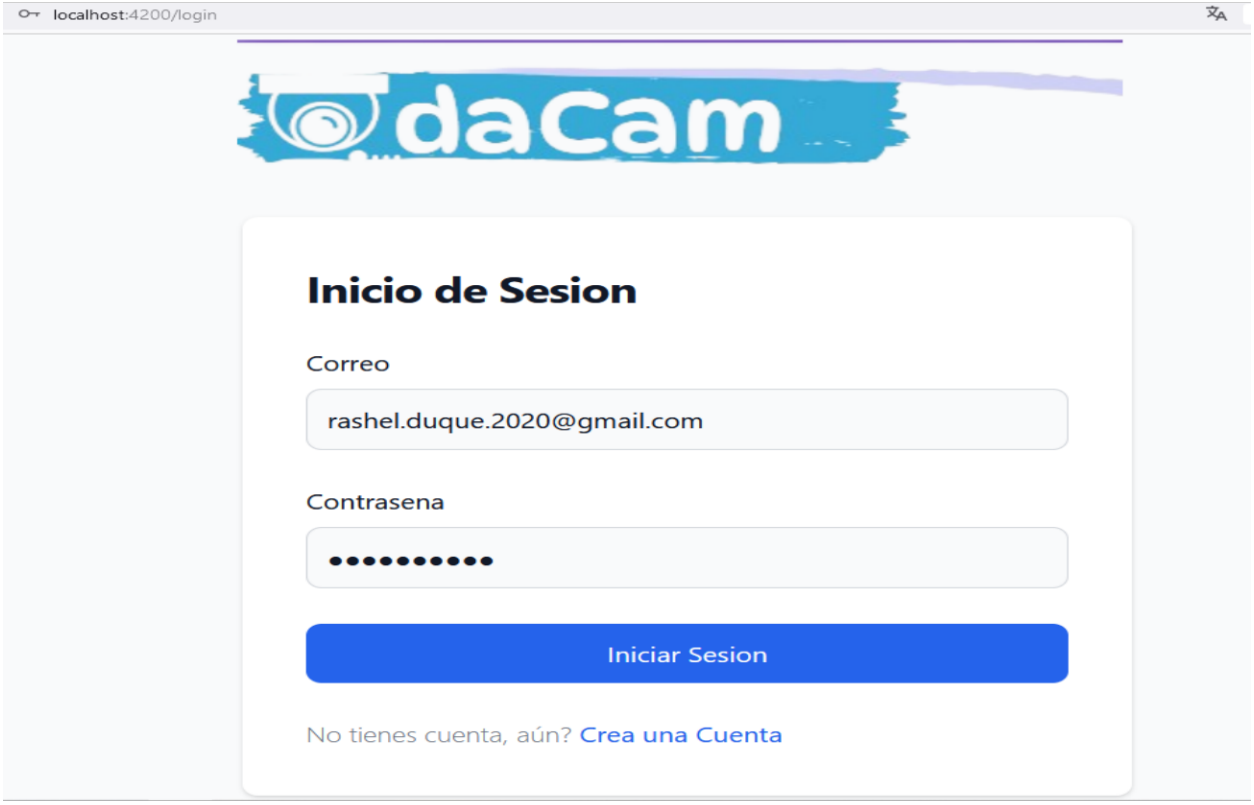


Fig. 31. Login

```typescript
import { Component } from '@angular/core';
import { AuthService } from 'src/app/shared/services/auth.service';

@Component({
  selector: 'app-sign-up',
  templateUrl: './sign-up.component.html',
  styleUrls: ['./sign-up.component.css']
})
export class SignUpComponent {
  constructor(private authService: AuthService) {

  }

  signUp(email: string, password: string) {
    this.authService.signUpWithEmailAndPassword(email, password);
  }
}
```

Fig. 32. lógica para la creación de Cuenta con autenticación

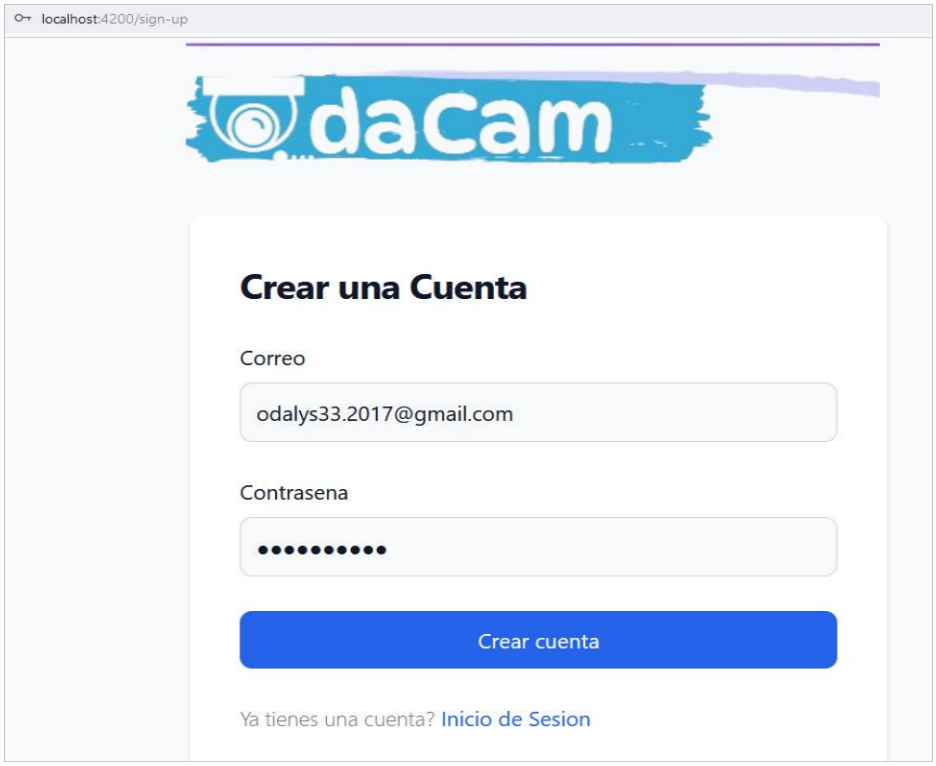


Fig. 33. Creación de la Cuenta

Fig. 34. Creación de Usuario.

Se procede en la creación de la lógica para la obtención de las imágenes y videos de nuestra base de datos en tiempo real, para lo cual se procede en la importación de getDatabase, ref, onValue necesarias para mostrar los datos guardados en Firebase, se importan las interfaces Imagen y Video para tipar los datos debido a la estructura almacenada en Firebase, se observa la inicialización de algunos arrays y variables necesarios para la lógica de nuestro aplicativo, se agregó el constructor a partir de la autenticación y se crea un método para cambiar la tabla que se va a visualizar en nuestro componente debido a que se implementaron dos enfoques distintos respecto a las imágenes y los vídeos.

Se procede a observar si el usuario está autenticado y a condicionar la visualización de los datos en caso de no estarlo, se crean las constantes respectivas a la obtención de la base de datos y a crear la referencia con la raíz de nuestra base de datos, finalmente con onValue obtenemos el valor que guardamos en Firebase considerando los objetos de toda la data y a partir de esta filtrar las colecciones que terminan respecto a imágenes y vídeos, filtrando así dos tipos de datos e iterando sobre cada objeto de los mismos para obtener los valores de cada uno y poder mostrarlos respectivamente, se visualiza lo detallado en las Fig. 35, 36, 37, 38 y 39.

```
ular-peopledetection > src > app > components > dashboard > TS dashboard.component.ts > ...
import { Component, OnInit } from '@angular/core';
import { AuthService } from 'src/app/shared/services/auth.service';
import { getDatabase, ref, onValue } from '@angular/fire/database';
import {Imagen} from 'src/app/shared/interFaces/imagen';
import { Video} from 'src/app/shared/interFaces/video';

@Component({
  selector: 'app-dashboard',
  templateUrl: './dashboard.component.html',
  styleUrls: ['./dashboard.component.css']
})
export class DashboardComponent implements OnInit{

  allData: any[] = [];
  datos: { key: string, value: Imagen }[] = [];
  datosV: { key: string, value: Video }[] = [];
  datosValues: any[] = [];
  datosValuesV: any[] = [];
  datosAplanados: any[] = [];
  datosAplanadosV: any[] = [];

  tablaActual = 'tabla1'; // tabla1 es la tabla que se mostrará por defecto

  cambiarTabla(tabla: string) {
    this.tablaActual = tabla;
  }

  constructor(private authService: AuthService) {

  }

  logOut() {
    this.authService.logOut();
  }

  ngOnInit() {
    if (!this.authService.isLoggedIn) {
      // El usuario no está autenticado, redirigirlo a la página de inicio de sesión
      console.log('El usuario no esta autenticado')
    }
    if (this.authService.isLoggedIn) {

      // El usuario no está autenticado, redirigirlo a la página de inicio de sesión
      const db = getDatabase();
      const dbref = ref(db, '/');

      onValue(dbref, (snapshot) => {
        this.allData = Object.entries(snapshot.val()).map(([key, value]) => ({key, value}));
        console.log('Datos:', this.allData); // Aquí imprimimos los datos en la consola

        // Filtrar los documentos que terminan en "_imagenes"
        this.datos = this.allData.filter((dato) => dato.key.endsWith('_imagenes'));
        console.log('Datos filtrados Imagenes:', this.datos);

         // Filtrar los documentos que terminan en "_videos"
         this.datosV = this.allData.filter((dato) => dato.key.endsWith('_videos'));
         console.log('Datos filtrados Videos:', this.datosV);

        // Itera sobre cada objeto en this.datos y extrae los valores
        this.datos.forEach(dato => {
          // Obtiene los valores de cada objeto y los agrega a datosValues
        this.datosValues.push(Object.values(dato.value));
        console.log('Datos filtrados por Valor de cada objeto:', this.datosValues);
        });

         // Itera sobre cada objeto en this.datosV y extrae los valores
        this.datosV.forEach(dato => {
          // Obtiene los valores de cada objeto y los agrega a datosValues
        this.datosValuesV.push(Object.values(dato.value));
        console.log('Datos filtrados por Valor de cada objeto Video:', this.datosValuesV);
        });

        // Aplanar el array de arrays y obtener un único array con todos los elementos
        this.datosAplanados = this.datosValues.flat();
        console.log('Datos aplanados:', this.datosAplanados);

        // Aplanar el array de arrays y obtener un único array con todos los elementos
        this.datosAplanadosV = this.datosValuesV.flat();
        console.log('Datos aplanados Video:', this.datosAplanadosV);
        const pagesize = this.datos.length
        console.log('Tamano:', pagesize);

      }, (errorObject) => {
        console.log(errorObject);
      });
```

Fig. 35. Obtención de las imágenes y videos de nuestra base de datos en tiempo real.

Se implementa una navegación con estilo Tailwind CSS, donde se agrega la opción de clic en la navegación para redirigirnos al método cambiar tabla y a la visualización de imágenes o vídeos de manera respectiva.

Fig. 36. Creación de navegación para la visualización de las tablas y el botón para cerrar sesión.

Se agrega *ngIf para dar la vista a la tabla 1 o 2 de forma correspondiente a imágenes y videos, se muestran los datos obtenidos del array datos aplanados tanto para el filtro de imágenes como videos, utilizando la directiva estructural *ngFor para iterar sobre cada objeto del array y renderizar una fila de la tabla para cada uno de ellos. Finalmente se llama o los valores de los keys de forma correspondiente en las dos tablas para mostrar sus valores.

```
<div  class="relative overflow-x-auto shadow-md sm:rounded-lg" *ngIf="tablaActual === 'tabla1'">
  <table class="w-full text-sm text-left rtl:text-right text-gray-500 dark:text-gray-400">
    <thead class="text-xs text-gray-700 uppercase bg-gray-50 dark:bg-gray-700 dark:text-gray-400">
      <tr>
        <th class="w-1/4 px-4 py-2">timestamp</th>
        <th class="w-1/4 px-4 py-2">Imagen</th>
      </tr>
    </thead>
    <tbody>
      <ng-container>
        <tr class="bg-white border-b dark:bg-gray-800 dark:border-gray-700 hover:bg-gray-50 dark:hover:bg-gray-600" *ngFor="let objeto of datosAplanados" >
          <!-- Muestra el timestamp -->
          <td>{{ objeto.timestamp }}</td>
          <!-- Muestra la imagen -->
          <td><img class="w-300 h-20 mr-2 tracking-tight" [src]="objeto.imagen_url" alt="Imagen"></td>
        </tr>
      </ng-container>
    </tbody>
  </table>
</div>
<div class="relative overflow-x-auto shadow-md sm:rounded-lg" *ngIf="tablaActual === 'tabla2'">
  <table class="w-full text-sm text-left rtl:text-right text-gray-500 dark:text-gray-400">
    <thead class="text-xs text-gray-700 uppercase bg-gray-50 dark:bg-gray-700 dark:text-gray-400">
      <tr>
        <th class="w-1/4 px-4 py-2">timestamp</th>
        <th class="w-1/4 px-4 py-2">Video de Descarga</th>
      </tr>
    </thead>
    <tbody>
      <ng-container>
        <tr class="bg-white border-b dark:bg-gray-800 dark:border-gray-700 hover:bg-gray-50 dark:hover:bg-gray-600" *ngFor="let objeto of datosAplanadosV" >
          <!-- Muestra el timestamp -->
          <td>{{ objeto.timestamp }}</td>
          <!-- Muestra el video-->
          <td><a [href]="objeto.video_url2">Descargar Video</a>
        </tr>
      </ng-container>
    </tbody>
  </table>
</div>
</body>
</html>
```

Fig. 37. Integración de *ngFor para la visualización de los datos filtrados para imágenes y videos.

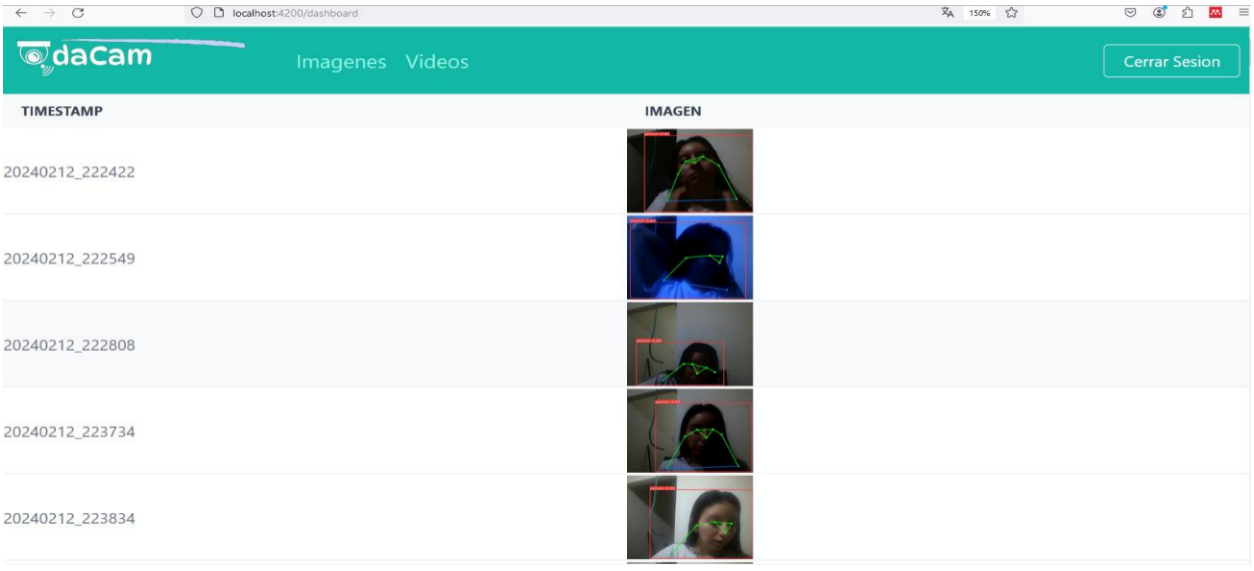


Fig. 38. Visualización de las imágenes en tiempo real en Angular.

Fig. 39.Visualización de los videos en Angular en tiempo real.

III. RESULTADOS

En esta sección se presenta el rendimiento en detalle al evaluar las métricas de precisión, sensibilidad y exactitud en la cámara de prueba y el modelo YOLOv8x-pose, generando para ello las matrices de confusión para su análisis.

A. Evaluación y resultados de la cámara de prueba para la detección de intrusos

Al analizar la matriz inicial de rendimiento del modelo que aplica la Cámara Tapo C200 para la detección de personas en la Fig. 40, se evidenció que de las 160 alertas generadas para la detección de intrusos, 82 de ellas resultaron ser falsos positivos, mientras que 78 fueron verdaderos positivos. Se observó 54 casos donde la cámara no logró detectar personas cuando estaban presentes, es decir, se etiquetaron erróneamente, resultando en falsos negativos. Por último, se registraron 150 casos de verdaderos negativos, donde la cámara acertó al identificar eventos sin intrusos.

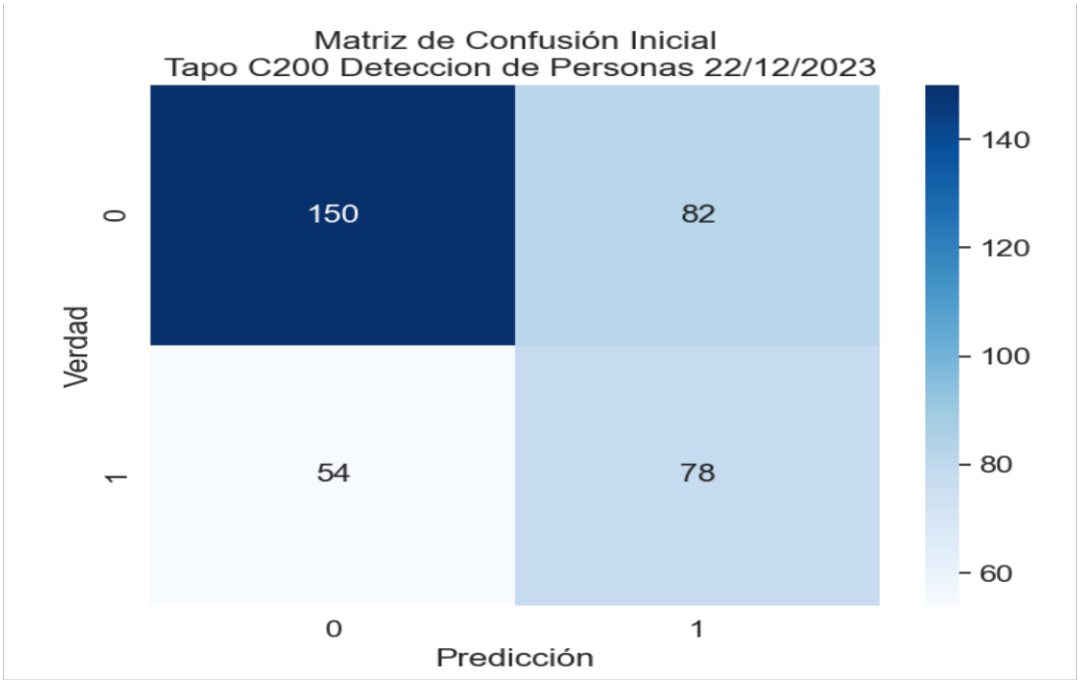


Fig. 40. Matriz de confusión inicial realizada con Cámara Tapo C200.

Nota: Matriz Inicial obtenida el 22 de diciembre de 2023 con una Cámara wifi Tapo C200: Verdaderos Positivos (VP) = 78, Verdaderos Negativos (VN) = 150, Falsos positivos (FP) = 82, Falsos Negativos (FN) = 54.

En las Fig. 41 y 42 se presentan las métricas de clasificación para analizar la precisión, sensibilidad y exactitud de la Cámara Tapo C200, se tomó por tanto los datos de la matriz de confusión inicial. Se evaluó la precisión para la detección de personas en la cámara, para lo cual se tomaron los casos de verdaderos positivos (78) y de falsos positivos (82), considerando el número de casos en que realmente detecta personas (78) sobre el número de casos total que la cámara predijo que eran personas incluyendo los falsos positivos 160, se obtuvo que la precisión base de la cámara es de 0.49.

A partir de los verdaderos positivos (78) y falsos negativos (54) se evaluó la sensibilidad en la detección tomando los casos positivos sobre la sumatoria de los verdaderos positivos 132, se obtuvo una sensibilidad de 0.59, finalmente se calculó la exactitud de la cámara tomando para ello los datos de verdaderos positivos (78), verdaderos negativos (150), falsos positivos (82) y falsos negativos (54), donde se divide la sumatoria de los verdaderos negativos (82) sobre la sumatoria (364).

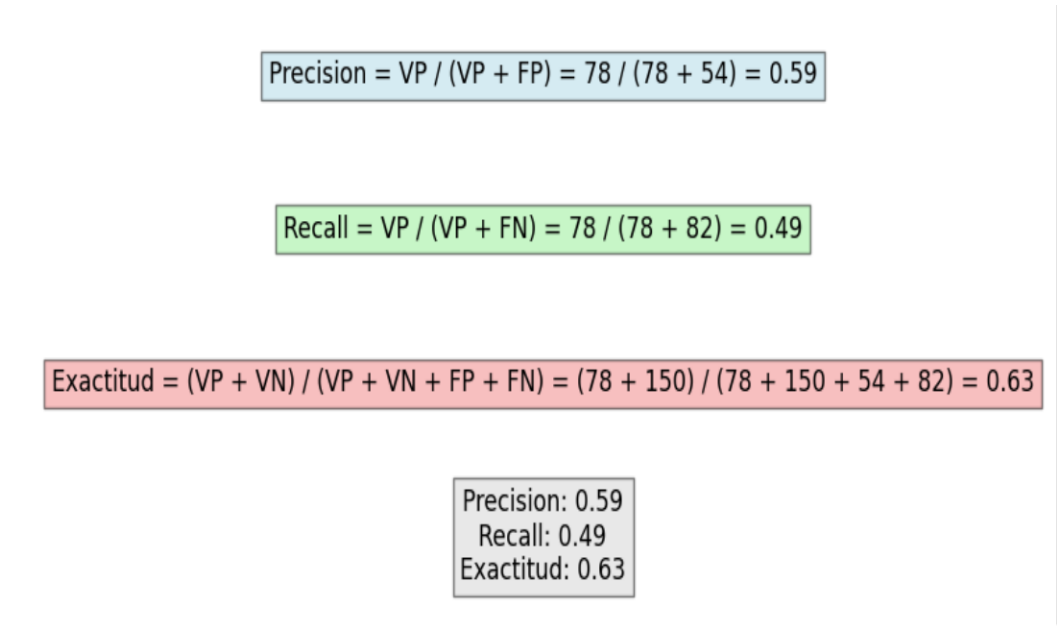


Fig. 41. Métricas de clasificación para la detección de personas en Tapo C200.

Nota: La presente figura muestra la precisión, sensibilidad y exactitud de la Cámara Tapo C200 donde nos indica un 49% de precisión para la detección de intrusos, un 59% de sensibilidad para identificar los casos positivos y finalmente la exactitud para clasificar de forma adecuada las detecciones con un 63%.

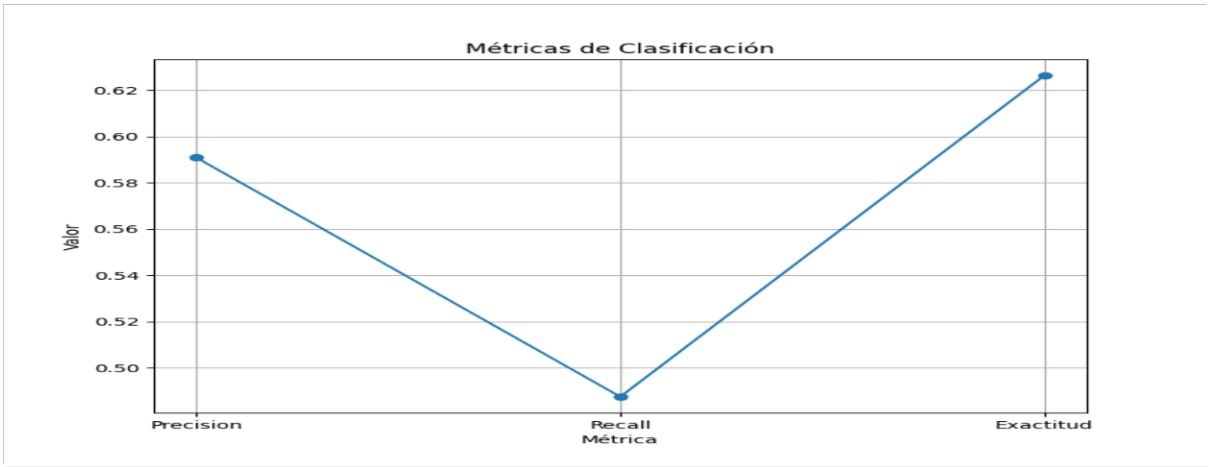


Fig. 42. Las métricas de precisión, sensibilidad y exactitud de la matriz inicial

A continuación, se muestra la Curva de Precisión-Recall en la Fig. 43 para observar el equilibrio de la precisión con la sensibilidad de la cámara en el umbral, en donde se distingue que la precisión es relativamente baja, lo que significa una cantidad importante a considerar de casos

de falsos positivos, sin embargo, la sensibilidad es alta ya que logra captar un número significativo de casos de verdaderos positivos. Puesto que nuestro objetivo es reducir el número de casos de falsos positivos. A partir de este análisis se logró encontrar la necesidad de aumentar el umbral de decisión, es decir el uso de un modelo más estricto para la detección de personas.

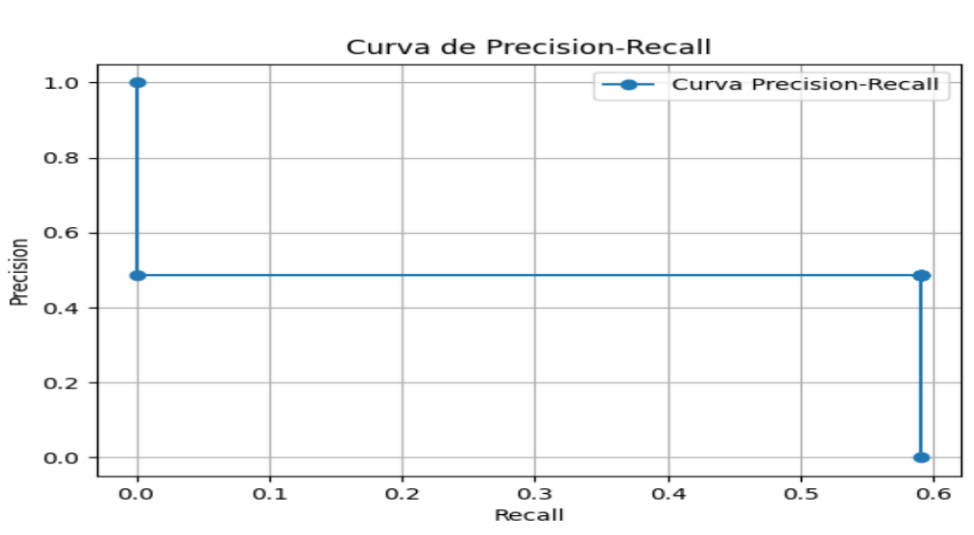


Fig. 43. Curva de Precision-Recall Inicial.

Se indican a continuación en la Fig. 44 las imágenes capturadas de la detección con la Tapo C200 con una estructura de matriz de confusión inicial para las detecciones respectivamente:

Fig. 44. Detecciones de la TAPO C200.

B. Evaluación y resultados del modelo propuesto para la detección de intrusos

Se evidencian las predicciones con ayuda del modelo YOLOv8x-pose generando la matriz de confusión. Al aplicar el modelo en las detecciones tanto positivas como negativas de la Tapo C200, se logra observar en la Fig. 45 que de los 218 casos detectados como negativos por el modelo 216 son casos de verdaderos negativos mientras que solo 2 son casos de falsos negativos, así también, se analizó y comprobó que de 146 casos positivos detectados por el modelo YOLOv8x-pose, detectó 138 casos que son de verdaderos positivos y solamente 8 casos de falsos positivos, indicando por tanto una tasa alta de verdaderos positivos además de que la tasa de falsos positivos es baja deduciendo que el modelo no suele identificar de manera errónea otras cosas como personas.

Así también se observó que el resultado de los falsos negativos es relativamente bajo dando la seguridad de que nuestro modelo muy rara vez va a fallar en la detección de personas. Unificando la idea se visualiza que la tasa de los verdaderos negativos es alta, sugiriendo la efectividad del modelo en distinción de objetos y de personas.

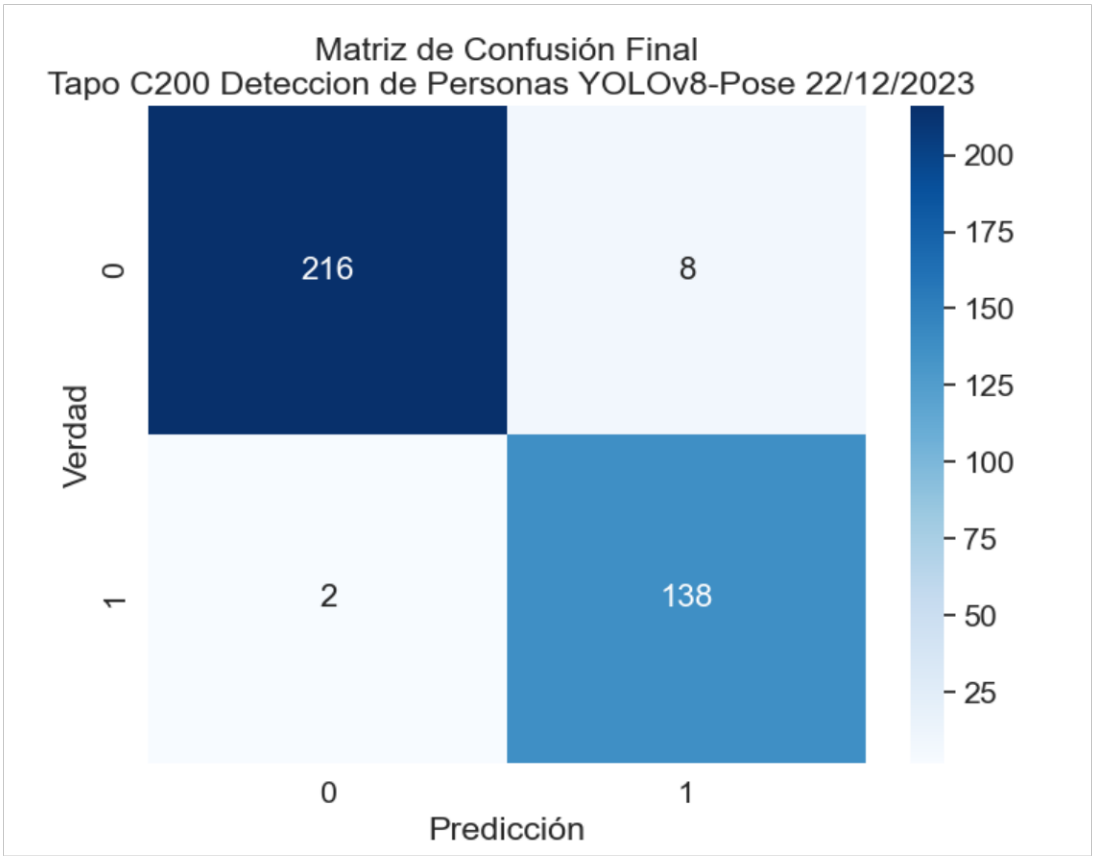


Fig. 45. Matriz de Confusión luego de aplicar el modelo YOLOv8x-pose

Se visualiza en la Fig. 46 la evidencia de algunas de las predicciones realizadas con el modelo YOLOv8x-pose

Fig. 46. Prueba del modelo YOLOv8x-pose para la detección de personas.

En la Fig. 47 se detallan los resultados de precisión, recall y exactitud con la aplicación del modelo YOLOv8x-pose, dado que se conoce 138 casos de verdaderos positivos y 8 de falsos positivos se sustituye en la formula sus valores para la obtención de la precisión significando que del total de muestras positivas el 95% son positivos reales y los restantes falsos, así también se calcula el recall al dividir los VP sobre la sumatoria de los 2 casos detectados como falsos positivos con los VP, resultando que el 99% de todas las muestras positivas en el grupo de datos fueron correctamente identificadas por el modelo.

Se agrega a estos cálculos la obtención de la exactitud luego de aplicar nuestro algoritmo para lo cual se requiere conocer el valor de los verdaderos negativos en este caso 216, realizando finalmente una sumatoria de los verdaderos positivos y verdaderos negativos sobre la sumatoria total entre verdaderos positivos, verdaderos negativos, falsos positivos y falsos negativos obteniendo una exactitud al aplicar nuestro algoritmo de un 97%.

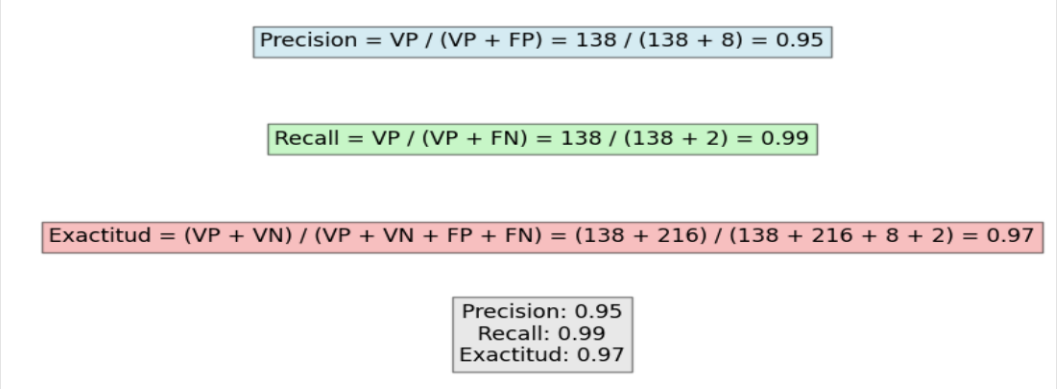


Fig. 47. Métricas de Clasificación al aplicar el modelo YOLOv8x Pose.

Nota: La presente figura muestra la precisión, sensibilidad y exactitud aplicando el modelo YOLOv8x-pose donde nos indica un 95% de precisión para la detección de intrusos, un 99% de sensibilidad para identificar los casos positivos y finalmente la exactitud para clasificar de forma adecuada las detecciones con un 97%.

Se visualizan las gráficas correspondientes en la Fig. 48 de precisión, recall y exactitud que se obtiene a partir del análisis realizado con nuestra matriz de confusión y los cálculos para cada uno de estos apartados, observando las métricas de clasificación y visualizando de una forma más dinámica su rendimiento.

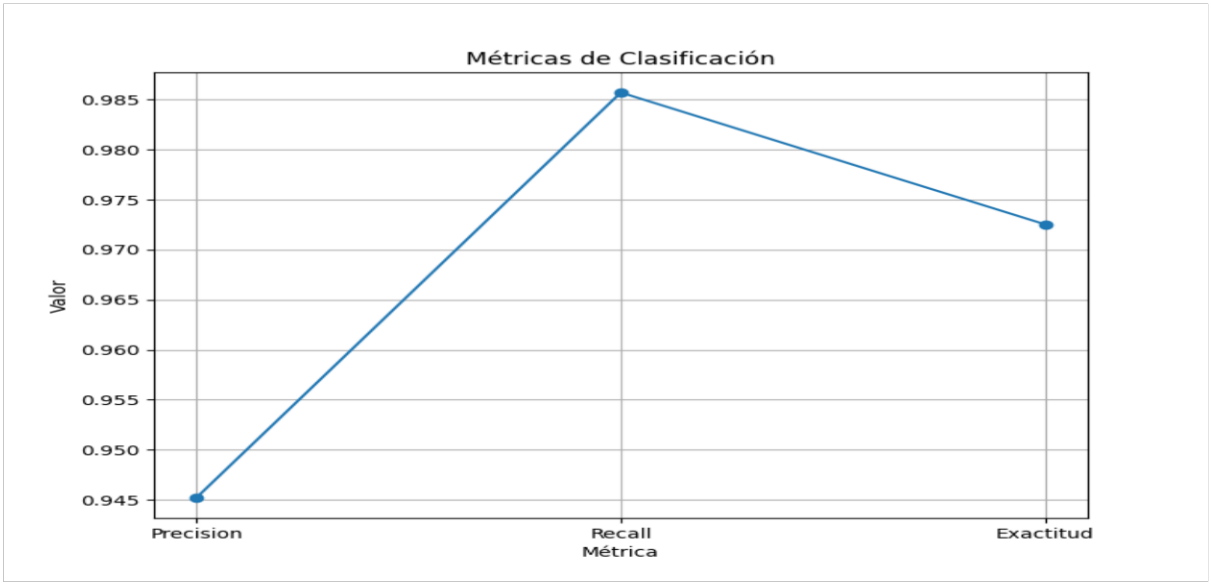


Fig. 48. Las métricas de precisión, sensibilidad y exactitud respecto al modelo YOLOv8x-pose

En la Fig. 49 se observa la curva de Precision-Recall en donde, se visualiza que se obtiene una alta precisión y un alto Recall dando a entender que existe un mejor rendimiento con la aplicación del modelo para la detección de personas.

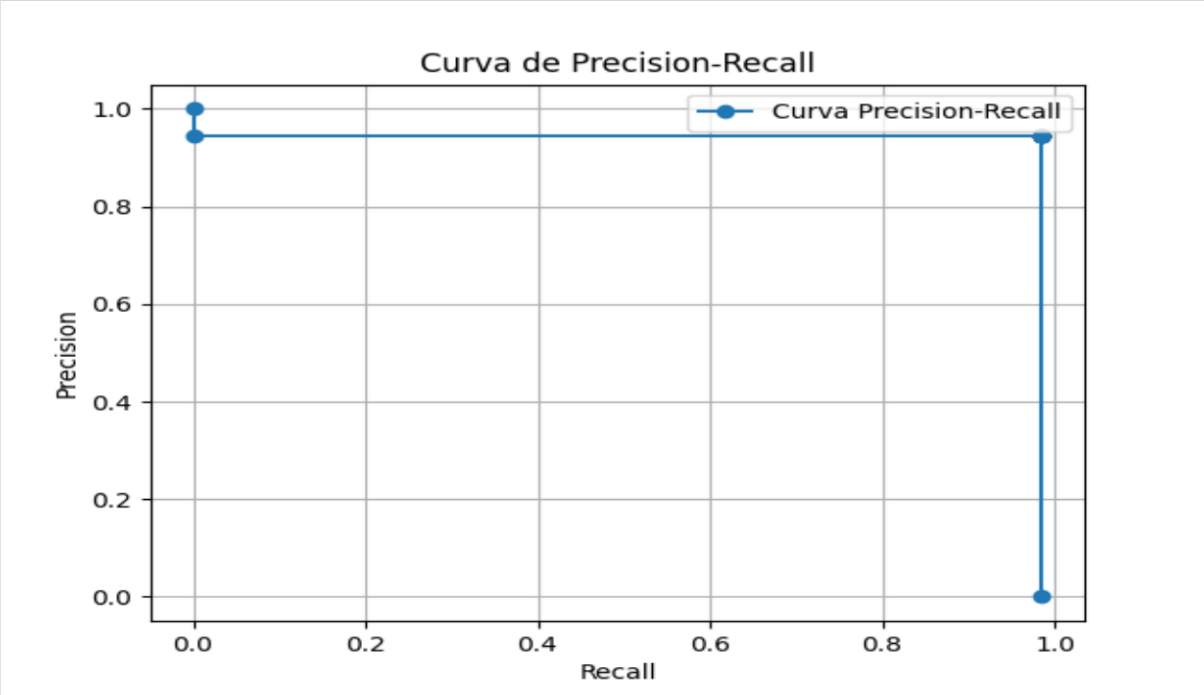


Fig. 49. Curva de Precisión-Recall

C. Evaluación y resultados

A partir del análisis realizado con las matrices de confusión generadas se logra comprobar que existe una mejora significativa de la matriz final con respecto a la inicial tal como se puede distinguir en la Fig. 50, en donde se obtuvo primeramente 150 casos de verdaderos negativos mientras que en la matriz final se consigue 216 casos detectados como tales, así también se obtiene primero 78 casos de verdaderos positivos y luego se logra 138, se continúa con los casos de falsos negativos en donde se tiene 54 casos iniciales y se disminuye este rango a simplemente 2 casos. Finalmente se obtiene el propósito en la reducción de los casos de falsos positivos al tener inicialmente 82 casos y lograr disminuir esta brecha a solo 8 casos.

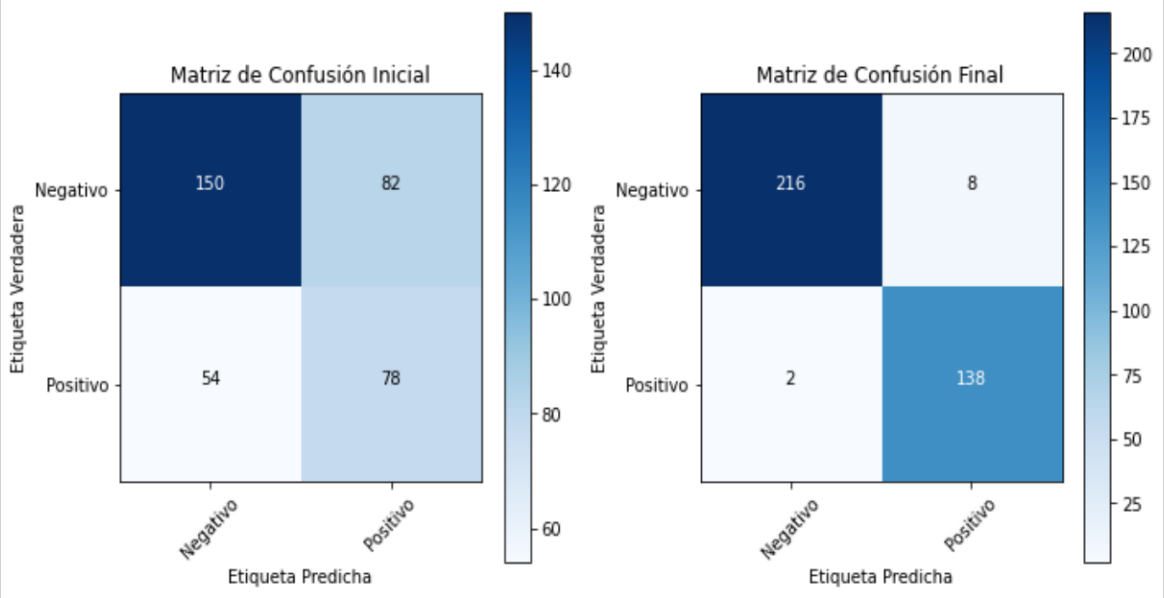


Fig. 50. Matrices de Confusión

En relación con estos resultados se lograron calcular las métricas de clasificación para cada una de estas matrices iniciales y finales, visualizando gráficamente las diferencias en precisión, recall y exactitud como indica la Fig. 51.

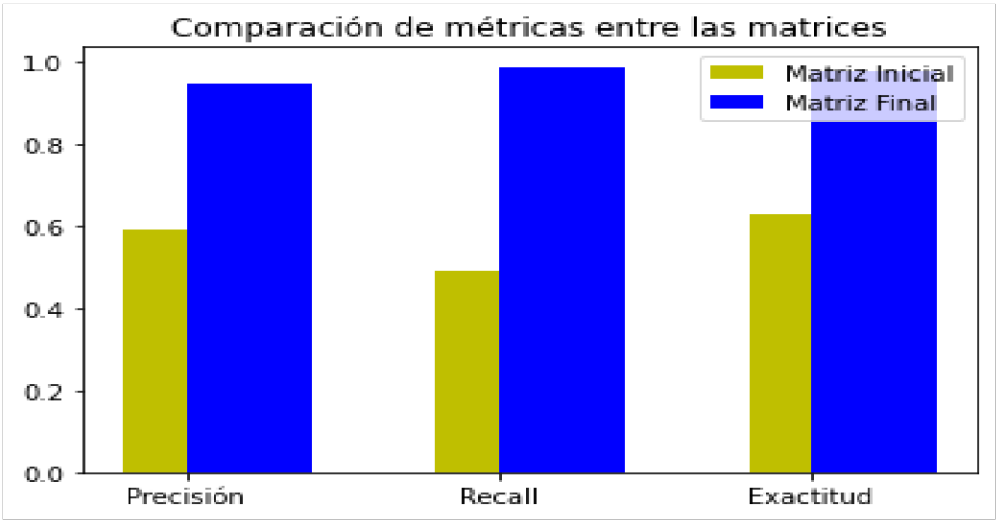


Fig. 51. Comparación de métricas entre las matrices

Se observan estas mejoras en la curva de precisión con intervalos de confianza en la Fig. 52, donde sus métricas resaltan en un rango inicial aproximado a 0.6 hacia un rango de 0.9 a 1.

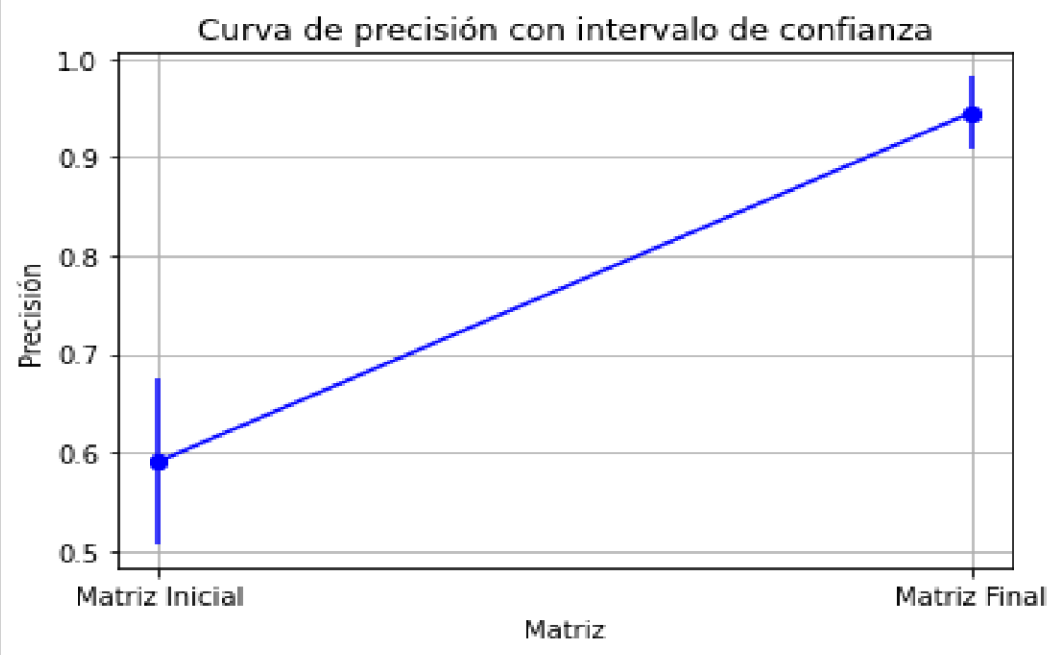


Fig. 52. Curva de precisión con intervalo de confianza de las matrices generadas

Se muestra en la Tabla 2 la comparación de las métricas de clasificación respecto a precisión, sensibilidad y exactitud obtenidas de las 364 imágenes de prueba, visualizando en porcentajes los cambios o mejoras de la Matriz Final respecto a la Matriz Inicial

Tabla 2.

COMPARACIÓN DE MÉTRICAS ENTRE LAS MATRICES

	Imágenes	**Etiquetas**	**Precisión**	**Sensibilidad**	**Exactitud**
Matriz Inicial	364	Persona	59%	49%	63%
Matriz Final		Persona	95%	99%	97%

IV. CONCLUSIONES

A partir de los resultados obtenidos con la ayuda del modelo YOLOv8x-pose de Ultralytics para la detección de personas se mejoró de una precisión inicial de 59% a una precisión final de 95%, avalando por tanto la funcionalidad del modelo para la reducción de falsos positivos en las cámaras de prueba, destacando la efectividad de las notificaciones al discriminar a los objetos y animales de la detección. Integrando herramientas como Realtime Database y Storage de Firebase, se permite guardar en el servidor todas estas detecciones en tiempo real, además de integrar la autenticación en la BDD.

Con la recepción de notificaciones con una precisión más alta en la detección de intrusos y el apoyo de un Bot en Telegram que incluye imágenes y videos en tiempo real permite tener una mayor confianza respecto a las alertas propias de la cámara. Minimizando por lo tanto riesgos a la propiedad y al usuario evitando daños significativos pues el tener un respaldo de los eventos capturados hace posible la investigación y actuación en temas legales, además, al existir estas evidencias en un grupo privado sugiere la posibilidad de compartir con otros usuarios de dicha vivienda estos registros visuales, contando con la atención a estas alertas al mejorar de forma significativa la capacidad para monitorear nuestra instalación.

La propuesta presentada permite además de una detección mejorada de intrusos, reducir la captura de eventos innecesarios en los buffers de videos, al incluir una variable de tiempo con límite máximo de 2 minutos de grabación al detectar una persona, iniciando la notificación con una imagen de la detección. Eliminando la necesidad de revisar periodos extensos de grabación en búsqueda de contenido relevante para el usuario interesado. Además, incluye una espera de 10 segundos antes de cortar el video al no detectar ningún evento, ahorrando con ello el almacenamiento y simplificando la revisión de las detecciones.

El otorgar autenticación con Firebase en el inicio de sesión para el acceso a las detecciones de imágenes y videos en tiempo real en el aplicativo web con Angular, agrega una capa de seguridad en relación al contenido que se está abordando, gestionando el acceso y rol del usuario autenticado. Pues al agregar la creación de cuentas supone personalizar futuramente la experiencia a un contenido especifico según los privilegios del usuario auditado.

V. RECOMENDACIONES

Se recomienda el uso de los modelos YOLO, ya que su tecnología integra versiones adaptables a procesadores con bajos recursos computacionales que pese a no integrar una GPU avanzada como sugiere su efectividad y precisión, las mismas no se ven afectadas como es el caso de este proyecto, pues al tener un procesador Intel, se requirió la versión optimizada del modelo de YOLO en Openvino para su aplicación.

Se sugiere aplicar y profundizar en los modelos de estimación de pose, ya que sugiere una mejor precisión al integrar puntos clave del objeto en cuestión para su entrenamiento, destacando un punto a favor de la visión artificial y futuras investigaciones o aplicaciones.

Usar el servidor Firebase para el almacenamiento de las detecciones ofrece facilidad de integración en herramientas con Python y Angular, además de ser gratuita, facilitando su acoplamiento y mejora.

Aplicar las notificaciones en un Bot y un grupo privado en Telegram ofrece una versatilidad para el envío de imágenes y video en tiempo real, además de agregar un plus en la privacidad de la información que se está manejando.

REFERENCIAS

[1] B. Montenegro and M. F. Calero, “Pedestrian detection at daytime andnighttime conditions based on YOLO-v5,” *INGENIUS*, 2021, doi: 10.17163/ings.n27.2022.03.

[2] Ultralytics Inc, “Inicio - Ultralytics YOLOv8 Docs.” Accessed: Feb. 28, 2024. [Online]. Available: https://docs.ultralytics.com/es

[3] J. Redmon, S. Divvala, R. Girshick, and 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, Dec. 2016, doi: 10.1109/CVPR.2016.91.

[4] G. Karimi, “Introduction to YOLO Algorithm for Object Detection - Webscale’s Engineering Education.” Accessed: Feb. 28, 2024. [Online]. Available: https://www.webscale.com/engineering-education/introduction-to-yolo-algorithm-for-object-detection/

[5] B. Kwon and T. Kim, “Towards an Online Continual Learning Architecture for Intrusion Detection of Video Surveillance,” *IEEE Access*, 2022, doi: 10.1109/ACCESS.2022.3201139.

[6] Y. Guo, Y. Chen, J. Deng, S. Li, and H. Zhou, “Identity-Preserved Human Posture Detection in Infrared Thermal Images: A Benchmark,” *Sensors 2023, Vol. 23, Page 92*, vol. 23, no. 1, p. 92, Dec. 2022, doi: 10.3390/S23010092.

[7] K. Abhishek and S. B. ud din Tahir, “Human Verification over Activity Analysis via Deep Data Mining,” *Computers, Materials and Continua*, vol. 75, no. 1, pp. 1391–1409, 2023,

doi: 10.32604/cmc.2023.035894.

[8] L. Liu, Y. Jiao, and F. Meng, “Key Algorithm for Human Motion Recognition in Virtual Reality Video Sequences Based on Hidden Markov Model,” *IEEE Access*, vol. 8, pp. 159705–159717, 2020, doi: 10.1109/ACCESS.2020.3020591.

[9] FGE, “Fiscalía General del Estado | Analítica cifras de robo.” Accessed: Feb. 29, 2024. [Online]. Available: https://www.fiscalia.gob.ec/analitica-cifras-de-robo/

[10] A. Banafa, *Introduction to Internet of Things (IoT)*. 2023. Accessed: Feb. 29, 2024. [Online]. Available: https://www.routledge.com/Introduction-to-Internet-of-Things-IoT/Banafa/p/book/9788770224451

[11] F. G. P. Márquez, *Internet of Things*, IntechOpen. London, 2021.

[12] L. Rouhiainen, “Artificial intelligence : 101 things you must know today about our future,” *Amacom*, p. 310.

[13] C. Janiesch, P. Zschech, and K. Heinrich, “Machine learning and deep learning,” *Electronic Markets*, vol. 31, no. 3, pp. 685–695, Sep. 2021, doi: 10.1007/S12525-021-00475-2/TABLES/2.

[14] S. Russell and P. Norvig, “Artificial Intelligence A Modern Approach Third Edition,” *Pearson*, 2010, doi: 10.1017/S0269888900007724.

[15] I. H. Sarker, “Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions,” *SN Comput Sci*, vol. 2, no. 6, pp. 1–20, Nov. 2021, doi: 10.1007/S42979-021-00815-1/FIGURES/11.

[16] R. Szeliski, “Computer Vision,” 2022, doi: 10.1007/978-3-030-34372-9.

[17] David. Forsyth and Jean. Ponce, “Computer vision : a modern approach,” p. 761, 2012.

[18] G. N. Elwirehardja, T. Suparyanto, and B. Pardamean, “Pengenalan Konsep machine Learning Untuk Pemula,” *Instiper Press*, pp. 1–19, 2022, Accessed: Feb. 26, 2024. [Online]. Available: https://www.manning.com/books/deep-learning-with-python

[19] N. Cristianini and J. Shawe-Taylor, “Support Vector Machines,” *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*, pp. 93–124, Mar. 2000, doi: 10.1017/CBO9780511801389.008.

[20] L. Breiman, “Random Forests,” *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[21] P. Viola and M. J. Jones, “Robust Real-Time Face Detection,” *Int J Comput Vis*, vol. 57, no. 2, pp. 137–154, 2004.

[22] S. Uma, S. B. J, B. R. Ramachandra, and A. professor, “HOG and CS-LBP Methods for Human Detection with Occlusion,” *International Journal of Computer Science and Information Technology Research*, vol. 3, pp. 692–698, Accessed: Feb. 27, 2024. [Online]. Available: www.researchpublish.com

[23] S. S. Sumit, D. R. A. Rambli, and S. Mirjalili, “Vision-Based Human Detection Techniques: A Descriptive Review,” *IEEE Access*, vol. 9, pp. 42724–42761, 2021, doi: 10.1109/ACCESS.2021.3063028.

[24] G. Van Rossum and Python development team, *Python Tutorial*, vol. Release 3.7.0. 2018.

[25] G. Bradski, “The OpenCV Library,” *Dr. Dobb’s Journal of Software Tools,* 2000.

[26] Z. R. Vargas Cordero, “La Investigación aplicada: Una forma de conocer las realidades con evidencia científica,” *Revista Educación*, vol. 33, no. 1, pp. 155–165, 2009.

[27] K. Beck, “Principles behind the Agile Manifesto.” Accessed: Dec. 11, 2023. [Online]. Available: https://agilemanifesto.org/principles.html

[28] Ultralytics, “Guía completa de Ultralytics YOLOv5 - Ultralytics YOLOv8 Docs,” Ultralytics YOLOv8 Docs. Accessed: Dec. 12, 2024. [Online]. Available: https://docs.ultralytics.com/es/yolov5/