“Desarrollo de un sistema para el envío automático de información de alerta en eventos de asalto en el interior de automóviles de transporte público”

Gutiérrez Valverde, Jorge Sebastián

Departamento de Eléctrica, Electrónica y Telecomunicaciones

Carrera de Ingeniería en Electrónica, Automatización y Control

Trabajo de titulación, previo a la obtención del título de Ingeniero en Electrónica, Automatización y Control

Ing. Silva Tapia, Rodrigo

03 de agosto del 2022

COPYLEAKS

Trabajo_Titulacion_JGutierrez_Final.pdf

Scanned on: 19:44 August 3, 2022 UTC

Identical Words	375
Words with Minor Changes	16
Paraphrased Words	103
Omitted Words	0

Firmado electrónicamente por:
RODRIGO SILVA TAPIA

Ing. Rodrigo Silva Tapia
C.I 0602199523

Departamento de Eléctrica, Electrónica y Telecomunicaciones

Carrera de Ingeniería en Electrónica, Automatización y Control
Certificación

Certifico que el trabajo de titulación: "**Desarrollo de un sistema para el envío automático de información de alerta en eventos de asalto en el interior de automóviles de transporte público"** fue realizado por el señor **Gutiérrez Valverde, Jorge Sebastián** 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 analizado 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.

Sangolquí, 03 de agosto de 2022

Firmado electrónicamente por:
RODRIGO SILVA TAPIA

.....

Ing. Silva Tapia, Rodrigo

C. C: 0602199523

Departamento de Eléctrica, Electrónica y Telecomunicaciones

Carrera de Ingeniería en Electrónica, Automatización y Control

Responsabilidad de Autoría

Yo, **Gutiérrez Valverde, Jorge Sebastián**, con cédula de ciudadanía n°1726609769, declaro que el contenido, ideas y criterios del trabajo de titulación: **"Desarrollo de un sistema para el envío automático de información de alerta en eventos de asalto en el interior de automóviles de transporte público"** 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.

Sangolquí, 03 de agosto del 2022

Firmado electrónicamente por:
JORGE SEBASTIAN GUTIERREZ VALVERDE

.....

Gutiérrez Valverde, Jorge Sebastián

C.C.: 1726609769

Departamento de Eléctrica, Electrónica y Telecomunicaciones

Carrera de Ingeniería en Electrónica, Automatización y Control

Autorización de Publicación

Yo **Gutiérrez Valverde, Jorge Sebastián**, con cédula de ciudadanía n°1726609769 autorizo a la Universidad de las Fuerzas Armadas ESPE publicar el trabajo de titulación: **Título: "Desarrollo de un sistema para el envío automático de información de alerta en eventos de asalto en el interior de automóviles de transporte público"** en el Repositorio Institucional, cuyo contenido, ideas y criterios son de mi responsabilidad.

Sangolquí, 03 de agosto del 2022

Firmado electrónicamente por:
JORGE SEBASTIAN
GUTIERREZ
VALVERDE

.....

Gutiérrez Valverde, Jorge Sebastián

C.C.:1726609769

Dedicatoria

Para todas aquellas personas que me acompañaron durante todo el desarrollo del presente trabajo de investigación, tanto mi familia como amigos que estuvieron allí en todo mi proceso de aprendizaje, crecimiento y desarrollo tanto personal como profesional. Este trabajo es el fruto del esfuerzo, acompañamiento de múltiples personas que durante mis años de estudio estuvieron allí conmigo para guiarme, apoyarme y estar presentes en todo momento.

Jorge Sebastián Gutiérrez Valverde

Agradecimiento

Mi gratitud por todo el apoyo y guía en el desarrollo de este proyecto al Ing. Rodrigo Silva, al departamento de Eléctrica, Electrónica y Telecomunicaciones de la Universidad de las Fuerzas Armadas “ESPE” y su profesorado, por todo el conocimiento, tiempo y herramientas que me fueron ofrecidos. Un muy especial agradecimiento y un inmenso cariño a mis padres Jorge y Rocío por ser la piedra angular de mi vida y por enseñarme con amor lo valioso del trabajo honesto y de la disciplina. Agradezco también a todas las personas con las que conviví estos años de carrera, por todas las experiencias que pude vivir y que atesoraré por siempre.

Jorge Sebastián Gutiérrez Valverde

Índice de Contenido

Índice de Tablas

Índice de Figuras

Resumen

En los últimos años, la seguridad ciudadana ha sido objeto de importantes progresos a nivel de tecnología. El desafío de desarrollar tecnologías de detección y envió automático de notificaciones, en caso de asaltos con armas, cobra mayor importancia a la luz de los índices de criminalidad. Sobre la base de situaciones reales, se han desarrollado herramientas que ayuden a un control efectivo y autónomo, en situaciones de asalto en el interior de vehículos de transporte público; esto, con el fin de fortalecer la seguridad de la ciudadanía al interior de aquellos espacios. El presente trabajo se centra en el desarrollo de un pequeño prototipo de sistema, para el envío automático de alertas en caso de eventos de asalto en el interior de vehículos automóviles, utilizados en servicio de transporte público. En este contexto, el prototipo utiliza hardware Nvidia Jetson Nano equipado con webcam, para la implementación de un algoritmo detector de objetos SSD MobileNet, entrenado para detectar de manera local armas portables tipo pistolas o cuchillos en posición amenazante por parte del pasajero. También se utiliza un módulo externo GPS SIM7600G-H conectado a la tarjeta Jetson para la transmisión de datos móviles con las coordenadas de ubicación del vehículo y el envío en tiempo real de notificaciones de alerta tanto a dispositivos móviles vía SMS como también a un servidor web, en el cual se registran y se almacenan todos los eventos detectados en el interior del vehículo. El servidor web es implementado mediante el framework de desarrollo web Django y desplegado en un ambiente de producción a través del servicio de Heroku.

Palabras Claves: Detección de armas, visión por computador, seguridad ciudadana, Mobilenet, Gps, servidor web

Abstract

In recent years, citizen security has undergone significant progress in terms of technology. The challenge of developing technologies for detecting and sending automatic notifications, in case of assaults with weapons, becomes more important in light of crime rates. Based on real situations, there have been developments of tools for an effective and autonomous control in situations of assault inside public transport vehicles. The purpose is to strengthen the security of citizens security inside those spaces. The present work focuses on the small system prototype development for the sending automatic alerts in assault events inside automobile vehicles used in public transport service. In this context, the prototype uses an Nvidia Jetson Nano card equipped with a webcam, for implementation of the MobileNet SSD object detector algorithm, trained to locally detect portable weapons such as guns or knives in a threatening position by the passenger. Also an external module GPS SIM7600G-H are joined to Jetson card for mobile data transmission of the vehicle's location coordinates and send real-time alert notifications to mobile devices via SMS as well as to a web server, in which are recorded and stored all data of events detected in the vehicle inside. The web server is implemented using the Django web development framework and a production environment through the Heroku service.

Key words: Weapon detection, computer vision, citizen security, Mobilenet, GPS, web server

Capítulo 1: Introducción

Antecedentes

La cada vez mayor expansión de la población en el país, provoca que los problemas a enfrentar sean de mayor complejidad. Entre los problemas que se enfrenta es el inquietante aumento de la inseguridad ciudadana dado que esta se caracteriza por el aumento de delitos. En el país la ola delictiva ha ocasionado un impacto importante en el índice de robos vehiculares, según el INEC (Instituto Nacional de Estadísticas y Censos) tan solo en los meses de enero-noviembre de 2020 a enero-noviembre de 2021 la variación en incidentes de esta índole ha aumentado un 48%. La segunda ciudad más afectada por este tipo de delitos es la ciudad de Quito. (INEC, 2019)

En el año 2017, el número de vehículos que brindan el servicio de transporte público en la capital alcanzó una cifra de 29347 unidades, de los cuales 55% se encuentran reguladas mientras que los demás se encuentran distribuidos entre aplicaciones de transporte como: Cabify, Uber, Didi. Dado la gran cantidad de vehículos que prestan su servicio para el transporte de la ciudadanía, estos se han visto asediados por la delincuencia por lo que se han creado múltiples campañas de transporte seguro, donde por lo general dotan de un botón de auxilio y cámaras de video.

Los sistemas de monitoreo existentes envían la notificación a las autoridades de control al momento de que se accione el botón de auxilio, esto provoca que muchas veces el operador del vehículo ponga en riesgo su vida para conseguir aplastar el botón y recibir ayuda en caso de un evento de robo. Una vez que la alerta es enviada el sistema guarda en sus registros lo ocurrido. Sin embargo, ya que estos sistemas son pasivos la mayoría de estos eventos no reciben la ayuda necesaria.

Los actos delictivos con armas cada vez son más frecuentes, por lo que muchas veces se busca que la tecnología brinde un soporte para mitigar estos eventos. La detección de

armas es una posible solución. Sin embargo, a pesar de que se han visto avances en el campo de la visión artificial y en modelos propuestos para la detección de armas.

En la actualidad se plantea el uso de sistemas basados en visión artificial dado que en la última década han tenido una gran relevancia en el mundo tecnológico al poder ser incorporados dispositivos. Con el uso de técnicas de deep learning se consigue se logra crear aplicaciones de detección de imágenes permitiendo el desarrollo de aplicaciones como la detección de armas puedan tener buenos resultados incluso con imágenes de baja resolución.

Otra de las maneras de lograr mitigar el uso de armas en actos delictivos y él envió oportuno de alertas es mediante el uso del modelo de detección YOLO (You Only Look Once) dado que este al se encuentra diseñado para obtener una alta precisión y una velocidad rápida de detección es apto para el uso en tiempo real (Hanan Ashraf, 2022). La incorporación de este modelo en sistemas de vigilancia y seguridad modernos permite reducir el riesgo para la vida humana un ejemplo de la implementación de esto en la empresa Actuate que brinda el servicio de implementación AI para la detección de armas en tiempo real (Actuate, 2021).

La detección de objetos es uno de los problemas más desafiantes en la visión por computador, dado la combinación de localización y clasificación dentro de un frame. Debido a esto se han creado sistemas embebidos de alto rendimiento, bajo consumo capaces de correr algoritmos de inteligencia artificial en tiempo real (Lee, 2021).

Justificación

La implementación de sistemas que incorporan visión por computador e inteligencia artificial constituyen una solución atractiva para industrias que cuentan con procesos de inspección de calidad o clasificación de objetos, debido a que permite automatizar tareas manuales, reduciendo riesgos por error humano, considerando que las decisiones tomadas por un operario se ven afectadas por factores psicológicos como puede ser la fatiga, estrés u otros, más aún cuando la tarea a realizar es repetitiva, de apreciación o percepción.

En el estudio realizado el personal fue sometido a pruebas para comparación de parámetros tales como forma, tamaño y color de varias manzanas, el resultado fue la incapacidad de una correcta selección de la fruta bajo los requerimientos solicitados. Es así como características de los sistemas de clasificación como, la objetividad y consistencia en largos periodos de tiempo, se convierten en ventajas dentro una línea de producción.

Bajo el estudio de trabajos realizados de manera previa, ha quedado demostrado que un sistema de clasificación puede ser usado para tareas como las de separar tipos de café, seleccionar huevos en buen estado, detectar tipos de flores, y más, mejorando así la calidad del producto y aumentando el volumen de producción. Si bien la implementación de sistemas de automatización permite mejoras en las líneas de producción, estas serán aún mayores si se añaden el uso de herramientas como lo son los servicios web, los cuales son eficaces por su capacidad de conexión con la nube y el almacenamiento o descarga de información desde cualquier lugar, por lo que, esta herramienta da una versatilidad a un sistema clasificador.

El deseo de generar un sistema de clasificación configurable que permita cambiar el tipo de objeto a clasificar empleando una misma máquina, puede llegar a requerir características altas de hardware, siendo una alternativa para reducir este requerimiento el uso de servicios con procesamiento en la nube.

La violencia con armas se ha convertido en una de las principales preocupaciones en la sociedad en los últimos años. La mayoría de los asaltos, homicidios y agresiones son cometidos con el uso de armas. El número global de muertes debido al uso de armas puede llegar a 1000 muertos por día (United Nations Office on Drugs and Crime, 2019).

En Ecuador entre los meses de enero a noviembre del año 2019 se puedo observar un alza en la tasa de robos de vehículos, donde el mes de junio ocurrieron la mayor cantidad de este tipo de robos con un total de 507. Entre los meses mencionados se contabilizaron cerca de 5094 robos a vehículos (INEC, 2019).

En el país las autoridades han tratado de reducir este problema de seguridad con campañas de transporte seguro, otorgando a las unidades de transporte de un sistema de monitoreo, sin embargo, estos sistemas al ser pasivos no consiguen funcionar de una manera óptima, ya que la persona tendría que accionar el botón de auxilio para enviar la señal de alerta a las autoridades. Cuando una persona entra en situación de riesgo tiende a ponerse nerviosa por lo sus reflejos bajan y tiende a tener problemas al presionar el botón. Además, el botón de auxilio tiene que presionar durante al menos tres segundos para realizar él envió de la alerta y según el estudio realizado por el “Ministerio de Trabajo y Asuntos Sociales España” determinó que el tiempo de reacción de una persona en situaciones de emergencia es de 1 segundo, tiempo el cual puede marcar la diferencia.

En la actualidad los sistemas de seguridad instalados en vehículos en la campaña de transporte seguro, una vez accionado el botón de auxilio el operador de monitoreo es el encargado de decidir las acciones a ejecutarse sobre el evento, sin embargo, como lo muestra en (Tiwari, 2015).

Los sistemas de detección de armas son un desafío dado que requieren la detección con una alta precisión logrando reducir la tasa de falsos positivos y falsos negativos. Además, debido a la naturaleza crítica de la detección se requiere que el tiempo de procesamiento y respuesta sea rápido.

El uso de nuevos avances tecnológicos es una alternativa viable para esto, el deep learning al ser una herramienta que dado a sus capacidades de lograr replicar como un cerebro humano procesaría la información, brinda una ventaja muy importante al momento del reconocimiento de objetos es que estos algoritmos pueden ser ejecutados en dispositivos de Edge computing el mismo que consiste en el procesamiento de datos más al borde de la red, logrando un procesamiento más eficiente de los datos (Shi, Dustdar, & Wien, 2016). En

palabras más simples los dispositivos de Edge Computing son capaces de procesar los datos lo más cerca de la fuente de generación de los mismos.

La detección de objetos tiene como fin identificar y localizar objetos específicos dentro de una imagen, en deep learning existen diversas técnicas para esto como lo son: SDD, Faster RCNN, los mismos que utilizan pipeline para realizar la detección provocando que el proceso sea lento y difícil de optimizar, mientras que YOLO (You Only Look Once) trabaja con una sola red neuronal consiguiente que tenga un buen rendimiento en detecciones en tiempo real (Sánchez, 2020).

Los dispositivos de Edge computing permiten llegar la AI a estos dispositivos logrando potenciar el procesamiento, estos dispositivos pueden ser de diferentes marcas y con diferentes características y propósitos. Entre las marcas más que ofrecen los dispositivos de Edge computing tenemos: Raspberry, Arduino, Nvidia.

Debido a las consideraciones mencionadas este trabajo el brindar una solución innovadora en el ámbito de detección de armas en vehículos de transporte público es de gran impacto dentro del país, dado que permitiría a las autoridades de control un mayor control delictivo y una acción oportuna dado el caso de presentarse una situación que involucre armas. Además, el trabajo al hacer uso de inteligencia artificial brinda al país un nuevo horizonte al uso de nuevas tecnologías en el ámbito de seguridad ciudadana.

Alcance

En el presente proyecto se plantea realizar un sistema para la detección de armas dentro vehículos y el envío automático de notificaciones de alerta, entre las armas a detectar tenemos armas de fuego tipo pistolas y armas blancas tipo cuchillos. El hardware del proyecto cuenta con una cámara de video conectada a un dispositivo Edge computing el mismo que permitirá la ejecución del algoritmo de detección de armas desarrollado en el lenguaje de programación Python utilizando librerías de visión por computador.

El sistema al contar con un dispositivo Edge computing su procesamiento será de local, haciendo uso de un algoritmo existente de detección el cual contará con un dataset personalizado de imágenes de armas tipo pistolas y cuchillos para su entrenamiento. La detección se la realizará en tiempo real.

Para entrenar los pesos necesarios para la detección se pretende crear un dataset de imágenes personalizado de armas en las que incluye: pistolas, cuchillos. Una vez creado el dataset se procederá a entrenar haciendo uso de la plataforma web de Google Colab.

Para observar la detección de manera remota se plantea mediante la implementación de un servidor web, el cual permita receptar los datos de detección para mostrar el resultado en tiempo real. Además de mostrar una alerta en caso de detectar un arma.

Objetivos

General

Desarrollar un sistema capaz de enviar automáticamente información de alerta en eventos de asalto en el interior de automóviles de transporte público

Específicos

- Desarrollar el subsistema de detección personas con armas (cuchillos o pistolas)
- Desarrollar el subsistema para el envío de información de ubicación y las imágenes del evento.
- Integrar el servidor web con el subsistema de detección y envío de información.
- Realizar pruebas del sistema en distintas condiciones para verificar su funcionamiento y desempeño.

Capítulo 2: Marco Teórico

Edge Computing

Con el desarrollo de la sociedad y con el continuo aumento de las necesidades tecnológicas de las personas, el número de dispositivos inteligentes conectados a internet ha tenido un incremento exponencial y con esto un incremento a larga escala de información. El aumento de la cantidad de información produce múltiples problemas entre ellos tenemos: baja seguridad, ancho de banda limitado, y una baja seguridad en los modelos de procesamientos basados en la nube (Khan, 2019).

La tecnología emergente de Edge Computing es un nuevo paradigma que hace énfasis en rendimiento de procesamiento al borde de la red, al presentar un modelo cerrado entre el usuario y la fuente de la información, al estar al borde la red es son dispositivos con alto uso en casas inteligentes y vehículos inteligentes. Tan solo en el 2016 según Global Cloud Index se tuvo 17.1 billones de dispositivos conectados a internet generando cerca de 10.4 Zettabyte de información, de los cuales el 45% de estos fueron almacenados, procesados y analizados en dispositivos de Edge Computing (Cao, 2020).

Arquitectura

En el contexto del mundo actual donde el desarrollo del Internet Of Things (IoT) y de la inteligencia artificial (AI) es cada vez mayor, el procesamiento al borde se podría considerar como una extensión de los servicios en la nube dado que ambos servicios pueden llegar a ser complementarios (Cao, 2020). El Edge Computing se basa en tres capas las cuales son:

Capa de dispositivos

Consiste en la conexión de dispositivos a la red de borde, donde se incluyen, dispositivos móviles, sensores, cámaras, actuadores. En esta capa se reduce el tiempo en la recepción de información sin procesar.

Capa de Borde

Esta capa es el núcleo del procesamiento en la cual debido a su cercanía con la fuente de información es la más adecuada para el procesamiento inteligente y en tiempo real de los datos, logrando un manejo de más eficiente y seguro si lo comparamos con los servicios de cloud computing.

Capa de Nube

La capa de nueve consta de servidores y dispositivos de almacenamiento con altas prestaciones, con la capacidad de almacenar de forma permanente todos los datos que provengan de la capa de borde. Esta capa brinda la capacidad de ajustar dinámicamente de acuerdo a las necesidades el algoritmo de procesamiento de la capa anterior.

Características del edge computing

Distribución Geográfica

El despliegue de múltiples plataformas de Edge computing a lo largo de un área específica, brinda la capacidad de realizar mantenimiento y facilitar los servicios de movilidad basados en la ubicación sin necesidad de tener que atravesar toda la WAN (Wide Area Network).

Proximidad

Al estar los recursos y servicios computacionales cerca de los usuarios brinda la capacidad de tomar decisiones de uso del servicio de una manera mucho más rápida y directa, así como analizar el comportamiento, mejorar servicios y gestionar la asignación de recursos de consumo del dispositivo.

Baja Latencia

El paradigma de Edge Computing al acercar los servicios y recursos al usuario, permite reducir la latencia a estos por el usuario, Esta característica permite la ejecución de aplicaciones demandantes de recursos y sensibles al retardo en él envió de información.

Conciencia de Ubicación

Conocer la ubicación donde se encuentra el dispositivo Edge Computing permite a los usuarios acceder a los servicios desde el dispositivo Edge más cercano a la ubicación física del usuario. Esta característica se utiliza en servicios de telefonía, GPS o acceso a inalámbricos, seguridad vehicular, gestión de desastres, etc.

Aplicaciones Web

Una aplicación web es aquella que se almacena en un servidor remoto y que puede ser accesible mediante un navegador web para lo cual necesita internet. Una aplicación web consta de dos partes: front-end y back-end.

Es muy común el uso de bibliotecas que faciliten el desarrollo del front-end, una de las bibliotecas más comunes para la estilización de páginas web es Bootstrap. Un sitio web que solo posee la parte de front-end se lo considera estático dado que el contenido siempre será el mismo.

El componente crucial de una aplicación web es el back-end, dado que permite la dinamización de los sitios web permitiendo que el sitio cambie dependiendo de las preferencias de quien interactúa con la aplicación o del usuario. Además, el back-end permite el desarrollo de múltiples funcionalidades como pueden ser el envío de notificaciones, mails o sms según los requerimientos del desarrollo.

Para realizar la comunicación entre el front-end y el back-end es necesario utilizar peticiones HTTP (Protocolo de transferencia de hipertexto) estas peticiones dependen de cómo el usuario interactúa con el sitio web. Son peticiones con una URL que los identifica. Entre las peticiones más comunes tenemos:

- GET: Es un método que solicita un recurso específico y solo devuelve datos.
- POST: Es aquel método utilizado para enviar datos y usualmente causa cambios en el servidor.

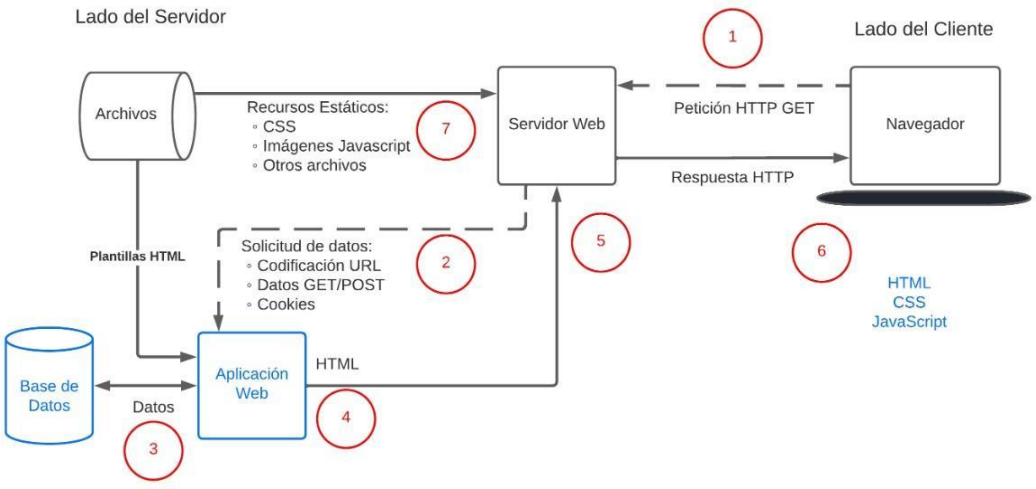
- PUT: Es un método que permite la creación de un recurso o si es el caso sobrescribir dicho recurso.
- DELETE: Nos permite eliminar un recurso específico.

Sitio web Dinámico

En un sitio web dinámico uno de los componentes principales es el desarrollo del back-end dado que se necesita la creación de contenido personalizado de acuerdo a usuarios, preferencias o uso de la aplicación web. En la actualidad es muy común el uso de plantillas para maquetar el contenido HTML y sobre esta montar la información proveniente de una base de datos que se conecta a través del back-end. El back-end es el encargado de realizar las operaciones que se ejecutan y retornar la información al lado del front-end. En la Figura 1 se muestra el proceso por el cual debe pasar una aplicación web para que un usuario la pueda utilizar en un navegador web.

Figura 1

Arquitectura sitio web dinámico



Django

Es un framework de código abierto creado en uno de los lenguajes con mayor crecimiento en los últimos años como lo es Python. Una de las razones principales para utilizar este framework es el ágil desenvolvimiento de APIs (Aplication Programming Interface) y sus herramientas de alto nivel. Fue creado en 2003 y es un framework que soporta el desarrollo rápido de aplicaciones web con una cantidad reducida de código (Muittar, 2020).

Muchos sitios populares de la actualidad se encuentran desarrollados en Django entre los cuales tenemos Instagram, Pinterest, National Geographic.Django cuenta con una larga cantidad de librerías y herramientas que pueden ser usadas de a acuerdo a los requerimientos de la aplicación, permitiendo que la aplicación sea escalable a medida que los requerimientos aumentan.

Deep Learning

El deep learning en los últimos años se está volviendo cada vez más indispensable en la vida de las personas y en el desarrollo de nuevas tecnologías. El desarrollo del deep learning viene de la mano con la gran cantidad de datos que se encuentran disponibles y el aumento de poder de procesamiento de los dispositivos actuales, estos avances han permitido crear aplicaciones de análisis de datos, detección de objetos, reconocimiento de voz, entre muchas otras más. Deep learning es usado particularmente en contextos donde la información es compleja y está disponible una gran cantidad de datos.

En la actualidad muchas de las compañías tecnológicas son consumidores de esta tecnología. Un ejemplo de esto es Facebook que usa aprendizaje profundo con el fin de analizar textos en las conversaciones de los usuarios. Deep learning es un algoritmo de machine learning caracterizado por el uso de múltiples capas de redes neuronales con la capacidad de extraer progresivamente una gran cantidad de características sin procesar directamente los datos (Xie, 2020).

El Deep learning nace de investigaciones de inteligencia artificial y de machine learning. En la Figura se ilustra la relación que existe entre estos.

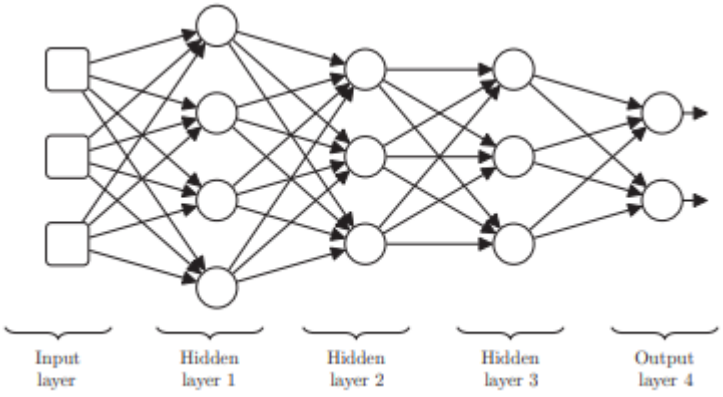
Redes Neuronales

Son modelos computacionales que se encuentran inspirados por la estructura del cerebro humano. La capacidad que poseen las redes neuronales para modelar relaciones complejas no viene dada de modelos matemáticos complicados, sino que surge del gran conjunto de interacciones que neuronas simples ejecutan entre sí (Kelleher, 2019).

En la Figura 2 se observa la estructura de una red neuronal usada en deep learning, el estándar de estructura de esta red neuronal está compuesta por 5 capas: una capa de entradas, tres capas que se encuentran ocultas y por último una capa de salida. Una de las características que posee el deep learning es que sus redes neuronales presentan múltiples capas ocultas, considerando que mínimo debe haber dos capas ocultas.

Figura 2

Topología red neuronal simple



Nota. Figura tomada de (Kelleher, 2019)

Redes Neuronales Convolucionales (CNNs)

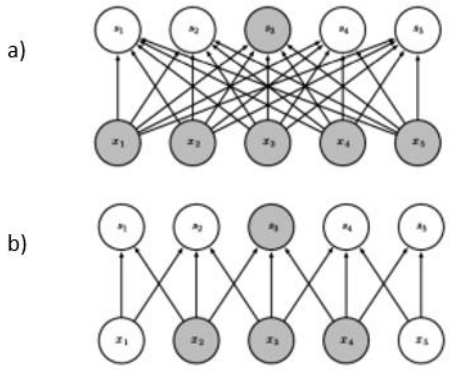
Las redes neuronales convolucionales fueron en un principio diseñadas para tareas de reconocimiento de imágenes y en sus inicios fueron aplicadas para resolver el reconocimiento

de dígitos escritos a mano. El diseño de las redes neuronales convolucionales es la creación de una cada de red neuronal que extrae las características visuales y las neuronas consiguientes combinaría estas características para obtener características mucho más complejas.

Las CNNs usadas para el reconocimiento de objetos tan solo son redes especializadas que tienen como principal característica que su topología es similar a una cuadrícula, donde las redes utilizan convolución en vez de la multiplicación de matrices en al menos una de sus capas (Kelleher, 2019). En la Figura 3 se ilustra la diferencia entre una red totalmente conectada (a) y una red convolucional (b) que extraen información y reduce las operaciones realizadas.

Figura 3

a) Red totalmente conectada vs b) red convolucional



Nota. Figura tomada de (Kelleher, 2019)

La convolución presenta tres características indispensables para la mejora de sistemas de visión por computador, dichas características son:

- Interacciones Dispersas: Las imágenes de entrada por lo general suelen tener miles o inclusive millones de píxeles. Sin embargo, es posible detectar características significativas con núcleos con poca cantidad de píxeles.

- Parámetros Compartidos: En una red neuronal convolucional los datos son aprendidos en conjunto para cada ubicación.
- Representaciones Equivalentes: Esta es una de las características fundamentales dado que significa que si la entrada cambia por ende la salida cambia de igual forma, esto se representa: $f\big(g(x)\big) = g(f(x))$

Detección de Objetos

La detección de objetos es un área de estudio de la visión por computadora, estudia la construcción de sistemas inteligentes que sean capaces de procesar, percibir y razonar sobre imágenes o videos (Amit, 2020).

En una imagen se puede percibir un número limitado de objetos, pero existe una cantidad muy alta de posibles ubicaciones y escalas donde este pueda ser detectado. Por lo general la detección otorga información de la ubicación y escala, un cuadro delimitador (bounding box) o una máscara de segmentación.

La detección de objetos es un reto complejo dado que las imágenes de objetos de cierta clase son altamente volátiles, dado que puede verse afectada por cambios de iluminación, posición de la cámara, resolución, filtros de digitalización, esto ocasiona que la apariencia de la imagen cambie. Para la evaluación y entrenamiento cada.

SSD-Mobilnet V2

Es un modelo desarrollado para ser utilizado en dispositivos de bajo consumo, de esto viene su de mobilenet, en los últimos años ha ganado gran relevancia por utilizar un tipo de convoluciones profundas,

La red Mobilenet fue desarrollada pensando en utilizarla en aplicaciones de tiempo real de deep learning y en dispositivos con limitaciones de hardware. Está diseñado para reducir el

número de parámetros necesarios para la detección sin tener que sacrificar la precisión en la detección.

Mientras que SSD (Single Shot Detector) tiene como característica fundamental que realiza la detección en un solo paso, ahorrando un tiempo considerable en la detección. Para lograr que la precisión sea alta el modelo lo que hace es producir predicciones a diferentes escalas tomando como base el mapa de características en escalas distintas y separar las predicciones.

Métricas de Evaluación

Las métricas de evaluación es un punto de referencia para determinar el funcionamiento correcto del detector de objetos en distintas circunstancias. El objetivo principal de dichas métricas es estimar la precisión que tendrá un modelo.

Intersection over Union (IoU)

La intersección sobre unión (IoU), es por defecto usada en detección de objetos, es una medida de similitud entre dos cuadros delimitadores dado que representa la intersección entre 2 cuadros delimitadores. Dicha relación se establece como:

$$IoU = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

Iou, también es llamada como semejanza Jaccard o índice Jaccard. La intersección sobre la unión (IoU) codifica alturas, anchos y ubicaciones de dos cuadros delimitadores, con esta información se calcula la medida normalizada que se enfoca su área. Los cuadros delimitadores a comparar son el de la ubicación real del objeto y el de la detección por lo que según (Barba, 2021), el valor obtenido de IoU determina que tan buena o mala es la detección. En la Figura 4 se observa una precisión decente, buena y excelente.

Figura 4

Predicción IoU: Decente, buena y excelente

Considerando que:

$$IoU > 0.5 \; es \; "regular"$$

$$IoU > 0.7 \; es \; "buena"$$

$$IoU > 0.9 \;\; es \; "excelente"$$

Mean Average Precision(mAP)

Determina la precisión de los detectores de objetos, estima el área bajo la curva de precisión vs recall. La curva de precisión vs recall se la llega a considerar como la compensación para diferentes valores de confianza que se asocian con el detector.

Esta métrica se calcula con la siguiente fórmula:

$$mAp = \frac{1}{N} \sum_{i=1}^{N} \quad AP_i$$

Donde N es el total de clases a evaluar.

Ubicación GPS

En un mundo cada vez más complejo, donde las ciudades han tenido un crecimiento considerable la seguridad de la ciudadanía ha generado que exista una necesidad de que los mapas sean cada vez más precisos y muestran una mayor cantidad de información.

Obtener la ubicación de un objeto se ha tornado un rato y cada vez se pretende aumentar la precisión y con esto aumentar la calidad y funcionalidad de los mapas. El GPS consiste en 24 satélites que orbitan a 20mil metros sobre el globo terrestre y a 55 grados de inclinación respecto al ecuador.

El GPS hace uso de un modelo de diferencia de tiempos de llegada para ello utilizad relojes atómicos en cada satélite con esto logra tener la posición precisa del satélite y crea mensajes de navegación que son transmitidos constantemente. Cada uno de los satélites presenta un código individual y único con el fin de que el receptor pueda conocer qué satélite proviene la señal y con esto calcular la posición (Mårten, 2011).

Los sistemas de posicionamiento constan de: satélites, red de control y el usuario. Donde la red de control es la encargada de corregir errores eventuales de los relojes, define las órbitas y actualiza la información que se transmite. Los satélites que son el componente indispensable son los encargados de transmitir dos tipos de señales L1 (1575,42 MHz) y L2 (1227.6 MHz) las cuales son las utilizadas para la navegación (McNeff, 2002).

El usuario para obtener la señal necesita de al menos tres satélites para definir la posible ubicación, mientras que si se agrega la señal de un cuarto satélite se reduce las posibles ubicaciones y con esto aumentando la precisión de la detección.

Características

Entre las características más importantes que ofrece la ubicación por GPS presenta como principales características:

- Presenta una cobertura mundial.
- Capacidad de usuarios ilimitada.
- Utiliza el sistema de coordenadas Geodésico Mundial.
- Está constituido de tres segmentos: Del espacio, de control y del usuario.
- Utiliza 24 satélites y 3 de respaldo.

Ventajas y Desventajas

El sistema de posicionamiento global brinda la capacidad de realizar la localización de objetos, lugares y personas. Permitiendo integrar esta herramienta en sistemas de rastreo, localización, sistemas antirrobos y un sinnúmero de aplicaciones más que cada día se van creando en este mundo digitalizado.

Sin embargo, una de las desventajas es que el sistema no siempre se encuentra disponible en su totalidad, debido a que es susceptible de interferencias, obstáculos o el objeto se encuentra en un lugar donde el acceso a la señal de los satélites se vea mermado. Esto genera problemas en encontrar la ubicación y puede generar problemas dependiendo del entorno en el que se esté utilizando este sistema de posicionamiento global.

Trabajos relacionados

En los últimos años la tendencia al aumento de crímenes ha experimentado un rápido crecimiento, debido a esto la seguridad ciudadana es un tema vital que debe ser atendido por los países. La seguridad ciudadana es definida como el proceso de proteger, establecer un orden democrático, reduciendo las amenazas de violencia en la ciudadanía para con esto lograr una convivencia segura y pacífica (Suárez, 2020).

Uno de los mecanismos para mejorar la seguridad ciudadana es la creación y desarrollo de políticas públicas que hagan uso de herramientas tecnológicas. A continuación, se brinda una revisión breve de las tecnologías utilizadas para que esto se logre (Lasso, 2021).

- El proyecto Maven, desarrollado por el Departamento de Defensa de Estados Unidos, procesa vídeo capturado desde drones en tiempo real con el fin de identificar actividad inusual y peligrosa. Además de ayudar a los operadores de sistemas de video vigilancia a tomar más eficientes y oportunas decisiones en situaciones de emergencia.

- Sistema autónomo de armas letales (LAWS), es un sistema utilizado por el ejército de los Estados Unidos, el cual usa algoritmos complejos de deep learning para clasificar un objeto como peligroso, tomar una decisión y decidir el usar o no el arma. Este sistema brinda la capacidad de funcionamiento en lugares peligrosos o difíciles de acceder.
- Proyecto Patternizr, desarrollado por la policía de Nueva York como una herramienta al apoyo en la toma de decisiones que busca patrones en actos delictivos, ayuda a que la decisión tomada sea más eficiente y eficaz. El sistema fue entrenado por más de 10 años de manera manual. Donde en las pruebas de rendimiento obtuvo un 80% de aciertos al probarlo con eventos sucedidos en el pasado.
- Zeroeyes, con el lema de reduce el tiempo, salva vidas. Es un proyecto basado en inteligencia artificial que se integra en sistemas de videovigilancia pasivos para lograr detener tiroteos y eventos delictivos que intervengan armas. El departamento de seguridad de los Estados Unidos lo usa como un multiplicador de esfuerzos dado que este sistema a través de análisis avanzados de detección envía una alerta logrando una respuesta rápida y precisa durante una situación de un tirador activo (ZeroEyes, 2022).

Capítulo III. Diseño del Sistema

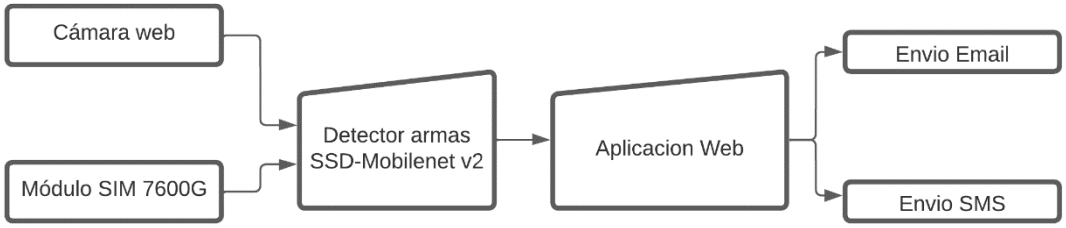
Conceptualización del sistema

La idea de la detección de armas y él envió automático de notificaciones de interés en el ámbito investigativo, ya que al aumento en el uso de armas en los actos delincuenciales y a la ola de violencia que se vive alrededor del mundo. Con los avances de la tecnología y la investigación sobre la inteligencia artificial de los últimos años ha permitido que sea una herramienta para combatir tipo de eventos, el desarrollo del deep learning ha permitido que las aplicaciones orientadas a la seguridad ciudadana sean cada vez más y más necesarias para las ciudades con un alto índice poblacional.

La Figura 5 muestra el diagrama de bloques del funcionamiento, este diagrama de bloques existe dos componentes principales: el detector de armas y la aplicación web.

Figura 5

Diagrama de bloques del sistema



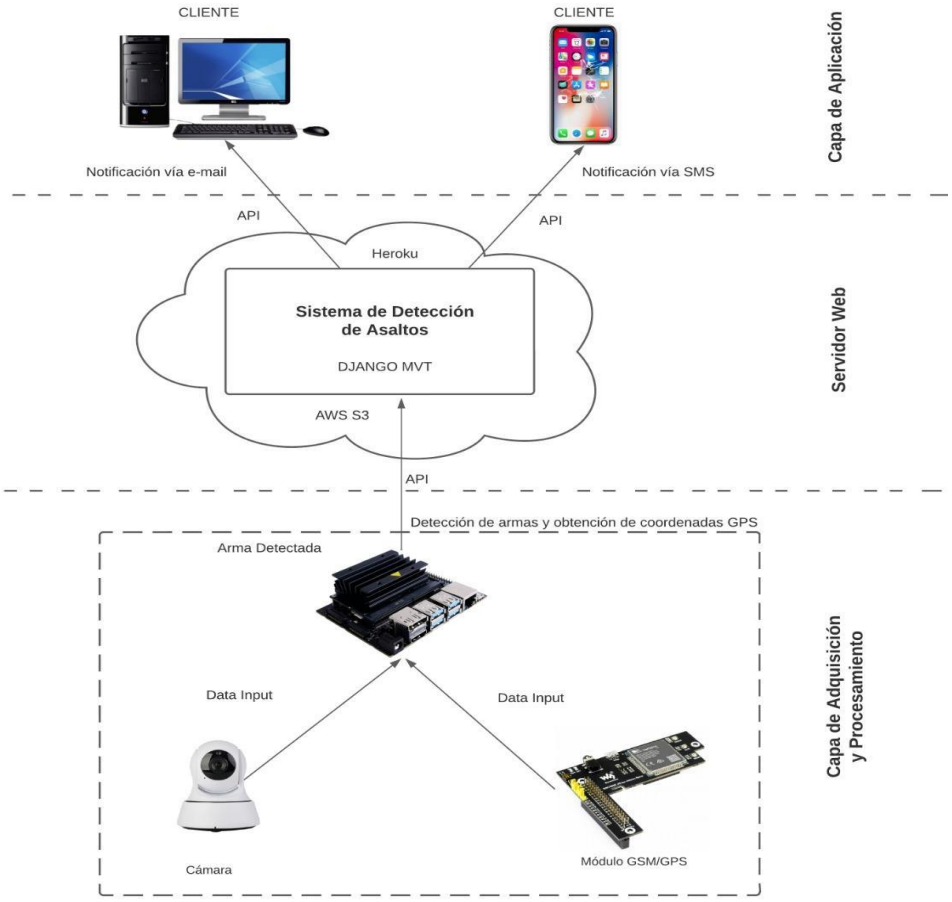
Mientras que en la Figura 6 se observa la estructura del sistema más detallada, donde se observa la capa de adquisición la cual está compuesta por una tarjeta Edge computing Jetson nano. La tarjeta Jetson nano tiene dos señales de entrada, una de ellas la cámara de video para lo cual se utilizará una cámara web usb, esta cámara es la encargada de capturar la imagen para su procesamiento y detección de armas en tiempo real. La detección de pistolas y

cuchillos se realiza mediante el algoritmo Mobile SSD, la otra entrada es la información de coordenadas que proviene del módulo SIM7600G-H.

Además, el módulo SIM7600G-H permite la conexión 4G a la tarjeta jetson nano y gracias a esta conexión el dispositivo es capaz de conectarse al servidor y enviar la información del evento.

Figura 6

Desarrollo sistema para el envío automático de información de alerta en eventos de asalto en el interior de automóviles de transporte público.



Una vez detectada el arma, se procede a enviar la información de lo ocurrido al servidor web haciendo uso de una API. La alerta del asalto está compuesta por: imagen de la detección,

localización, fecha completa y a quien fue enviada la alerta. Esta información es enviada utilizando el método POST al servidor web desarrollado en DJANGO. Una vez recibida la alerta el servidor almacena la información en una base de datos sql y envía la notificación mediante SMS o E-mail.

En la Tabla 1 se presenta la configuración detallada de hardware y software con una breve descripción de cada uno de ellos.

Tabla 1

Configuración del sistema: hardware y software

	Componente	**Descripción**
Jetson	Modelo	Jetson Nano 2GB
	Sistema Operativo	JetPack 5.0.1
	CPU	Quad core ARMA57 1.43 GHz
	GPU	128-core NVIDIA
	RAM	2 GB 64-bit
	SD Card	16Gb microSD
MPC01 Web Camera	Modelo	MPC01
	Resolución	5 MP
	Modo Video	1080p30
	Rango de Imagen	70-90cm
Software Tools	Visual Studio Code	Editor de Código Fuente
	Twilio	Plataforma de comunicaciones
	Heroku	Plataforma como servicio de computación en la Nube
Lenguajes de Programación	Python	Python 3.7
	Django	Framework de desarrollo Web en python
	HTML	Maquetado de interfaces de aplicaciones web
	CSS	Estilizar interfaces de aplicaciones web
Librerías	OpenCv	Biblioteca libre de visión artificial
	Mobile-SSD Algoritmo	Implementación algoritmo Mobile-SSD
	SMTP	Protocolo para transferencia simple de correo
Cloud	Amazon Web Service	Amazon Simple Storage Service
	Sistema Operativo	Ubuntu 16.04 LTS (HVM)
	CPU	1 vCPUs 2.5 GHz Intel Xeon family
	Memoria	1 GB

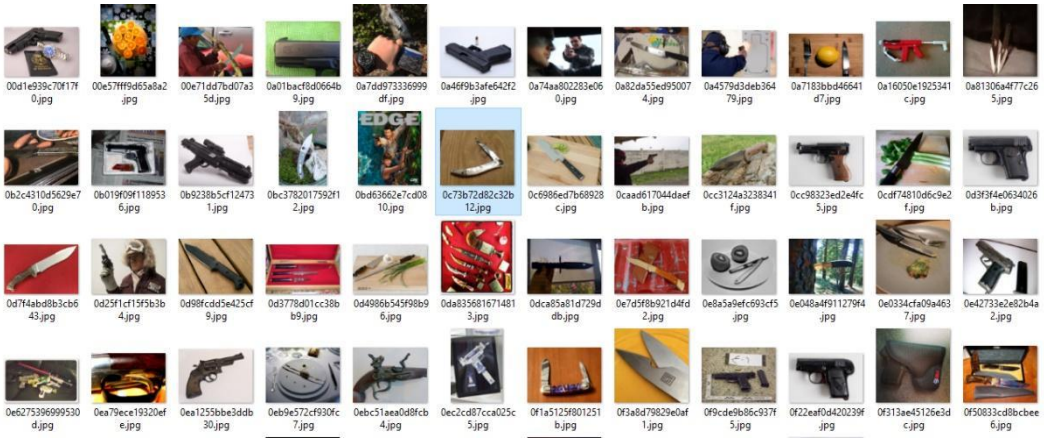
Modelo detección de armas

Obtención del dataset para entrenamiento del modelo

La búsqueda de imágenes que se ajusten a nuestra necesidad es de gran relevancia dado que la creación de un correcto dataset influye en el desempeño del detector. Para este trabajo de investigación los objetos de interés a detectar son dos: pistolas y cuchillos. Para la creación del dataset de imágenes se puede usar bases de datos existentes o buscarlos de forma manual. Una de las principales consideraciones para la creación correcta se debe buscar imágenes que contengan los objetos interesados y que presenten diversas condiciones de luz. En la Figura 7 se muestra la recolección de imágenes, las imágenes fueron netamente adquiridas en búsqueda de imágenes de Google.

Figura 7

Dataset de imágenes.



Herramientas de etiquetado de imágenes

Con el dataset de imágenes creado, es necesario etiquetar cada imagen y asignar la clase específica. Este proceso se lo puede realizar de distintas maneras, la más común y más sencilla es utilizar un software libre como lo es LabelImg. LabelIg es una herramienta de anotación gráfica que nos permite ubicar las los cuadros delimitadores y la clase de objeto de la

imagen. La Figura 8 representa el proceso de etiquetado de una imagen con la herramienta LabelImg, este proceso se lo debe repetir con cada una de las imágenes que componen el dataset.

Figura 8

Imagen etiquetada en la herramienta LabelImg

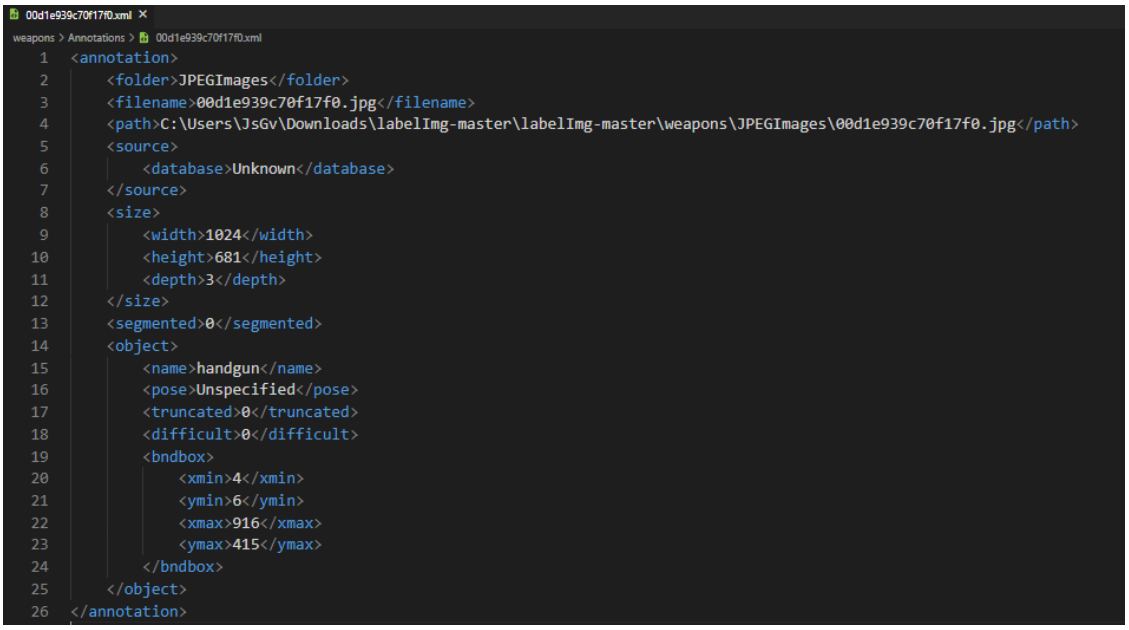
Una vez etiquetadas todas las imágenes, la herramienta crea una carpeta de nombre “Annotations” la cual contiene un archivo .XML por cada imagen. El archivo generado contiene información relevante del objeto de interés, la información de cada imagen es usada para realizar el entrenamiento del modelo de detección. Este archivo generado sirve como un mapa de las coordenadas donde se ubica el cuadro delimitar y su correspondiente clase. Para el caso particular del presente trabajo de investigación las clases son:

- Handgun
- Knife

En la Figura 9 el resultado de etiquetar la imagen en la herramienta LabelImg donde se detalla la información de la imagen en formato XML (Extensible Markup Languaje) .

Figura 9

Información de la imagen etiquetada en la herramienta LabelImag



```
00d1e939c70f17f0.xml
weapons > Annotations > 00d1e939c70f17f0.xml
<annotation>
    <folder>JPEGImages</folder>
    <filename>00d1e939c70f17f0.jpg</filename>
    <path>C:\Users\JsGv\Downloads\labelImg-master\labelImg-master\weapons\JPEGImages\00d1e939c70f17f0.jpg</path>
    <source>
        <database>Unknown</database>
    </source>
    <size>
        <width>1024</width>
        <height>681</height>
        <depth>3</depth>
    </size>
    <segmented>0</segmented>
    <object>
        <name>handgun</name>
        <pose>Unspecified</pose>
        <truncated>0</truncated>
        <difficult>0</difficult>
        <bndbox>
            <xmin>4</xmin>
            <ymin>6</ymin>
            <xmax>916</xmax>
            <ymax>415</ymax>
        </bndbox>
    </object>
</annotation>
```

En la Tabla 2 el detalle de las etiquetas más importantes del archivo.

Tabla 2

Descripción etiquetas archivo .XML

Etiqueta	Descripción
<path></path>	Ubicación archivo
<width/>	Ancho imagen
<height/>	Altura imagen
<name/>	Clase del objeto
<bndbox></bndbox>	Coordenadas x1, x2, y1, y2 del cuadro delimitador de la detección

Entrenamiento del modelo

Antes de comenzar con el entrenamiento del modelo, debemos tomar en cuenta que se debe dividir el dataset. Para la división de este se emplea comúnmente el principio de Pareto, esta regla establece que el 80% de los datos para entrenamiento, 10% validación y 10% test. Para acelerar el entrenamiento de los datos se optó por utilizar Google Colab, dado que este

servicio de Google ejecuta el código en los servidores de Google proporcionando la capacidad de aprovechar de un hardware sofisticado. En la Tabla 3 se detallan las características de hardware proporcionados por Google Colab para el entrenamiento.

Tabla 3

Características hardware de Google Colab

Parámetro	Descripción
CPU Modelo	Intel® Xeon®
CPU Frecuencia	2.30GHz
RAM Disponible	12GB
Espacio Disco	25GB
GPU	Nvidia K80/T4
GPU Memoria	12/16 GB
RAM GPU	12GB
Rendimiento	4.1 TFLOPS/ 8.1 TFLOPS

Pasos para el Entrenamiento

Como paso inicial para comenzar el entrenamiento debemos crear un nuevo cuaderno de trabajo en Google Colab, una vez creado debemos configurarlo para que funcione con GPU dado que si no activamos esta configuración por defecto Google Colab entrega una máquina virtual solo con CPU.

Además, tenemos que subir las imágenes a utilizar para el entrenamiento a Google Drive para poder brindarle a Google Colab los permisos necesarios para poder acceder a la información almacenada en esta.

Antes de comenzar el entrenamiento debemos tomar en cuenta que la versión gratis dispone de un tiempo de uso limitado de la GPU por lo que se aconseja realizar todo el entrenamiento hasta el final, para evitar problemas de desconexión de la plataforma y contar con una conexión estable de internet.

Descarga de Paquetes Necesarios

Este paso consiste en la descarga y compilación de todas las librerías necesarias para la ejecución del entrenamiento.

```
-git clone https://github.com/dustynv/jetson-inference
-cd jetson-inference
```

Creación de archivos de entrenamiento y validación

Se crean los archivos de train.txt y trainval.txt que son requisitos indispensables para proceder al entrenamiento. Además, se accede a la carpeta de “TrainingTools” que viene pse crea al clonar el repositorio Git en el paso anterior.

```
!mkdir Imagenes
!mkdir Imagenes /Main
!python3 readImage.py
```

Descarga del Modelo Pre-entrenado

El modelo pre-entrenado del modelo de detección Mobile-SSD, es el conjunto base de pesos sobre el cual se ejecuta el nuevo entrenamiento. Esto debido a que cuenta con la detección de múltiples clases por defecto.

```
!mkdir modelos
!mkdir modelos/Capstone
!wget
https://nvidia.box.com/shared/static/djf5w54rjvpqocsiztzaandq1m3avr7c.pth -O
models/mobilenet-v1-ssd-mp-0_675.pth
```

Entrenamiento

Este paso consiste en ejecutar el entrenamiento, pero para que este funcione se debe agregar la información de:

- Data: Este parámetro se refiere a la ubicación en donde se encuentra nuestro dataset, el cual debe estar debidamente etiquetado y dividido.
- Model: Define la ubicación donde se va a guardar cada iteración.
- Batch: Es la cantidad de imágenes que se va a utilizar por época.
- Workers: es el número máximo de procesos para activar cuando se utilizan subprocesos.
- Epochs: Número de épocas del entrenamiento (se aconseja que sea igual o mayor de 50)

El cuaderno de trabajo para realizar el entrenamiento de cualquier con cualquier dataset se encuentra disponible, para libre uso en el apartado de anexos del presente trabajo de titulación.

Resultados del Entrenamiento del Modelo de Detección

El valor de pérdida determina cómo se encuentra funcionando el modelo después de cada iteración, cuanto menor sea esta pérdida el modelo será mejor. La pérdida es la suma de los errores cometidos por cada uno de los conjuntos de entrenamiento.

Debemos tomar en cuenta que el tiempo de entrenamiento depende de la cantidad de imágenes que proporcionamos a la red. Dado que al aumentar la cantidad de datos las funciones de la red neuronal también se ven afectadas y aumentan. Esto produce que el modelo sea más o menos complejo.

En la Tabla 4 se observa cada una de las épocas del entrenamiento como su valor de pérdida respectivo. De acuerdo a esta tabla se escoge como peso del modelo la época con menor valor en este caso la época 48.

Tabla 4

Épocas vs Valor de pérdida

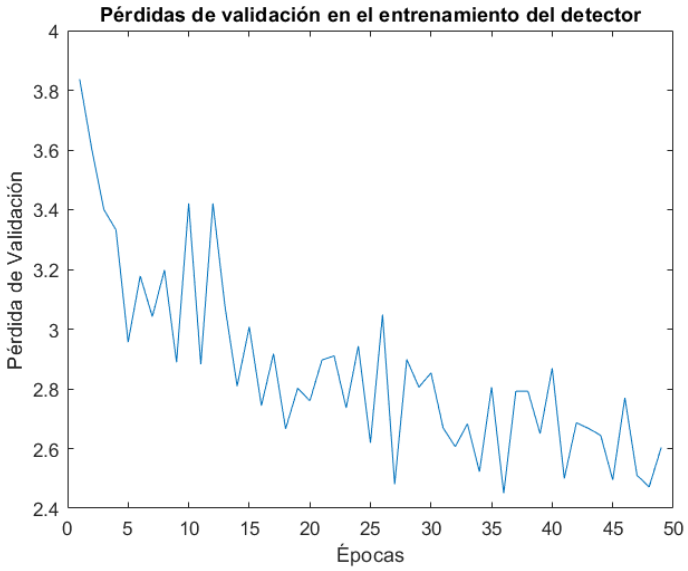
Época	Pérdida de Validación	Época	Pérdida de Validación
0	4.3853	25	2.6194
1	3.8370	26	3.0485
2	3.6034	27	2.4805
3	3.4004	28	2.8983
4	3.3323	29	2.8055
5	2.9562	30	2.8535
6	3.1777	31	2.6689
7	3.0418	32	2.6062
8	3.1979	33	2.6827
9	2.8891	34	2.5227
10	3.4210	35	2.8055
11	2.8822	36	2.4504
12	3.4210	37	2.7920
13	3.0745	38	2.7920
14	2.8091	39	2.6502
15	3.0076	40	2.8689
16	2.7438	41	2.4999
17	2.9177	42	2.6864
18	2.6661	43	2.6674
19	2.8021	44	2.6443
20	2.7598	45	2.4949
21	2.8965	46	2.7699
22	2.9105	47	2.5101
23	2.7363	48	2.4716
24	2.9428	49	2.6036

En la Figura 10, muestra como esta fluctúa la perdida de validación en cada una de las épocas del entrenamiento. Además, se observa que la variación en la perdida de validación llega un momento en el que es muy pequeña, esto se debe a que la cantidad de imágenes utilizadas para el entrenamiento no puede reducir la perdida de validación.

Una de las maneras de reducir la pérdida de validación es aumentar el número y variedad de imágenes de cada clase que se desee entrenar.

Figura 10

Pérdidas de validación en el entrenamiento del detector



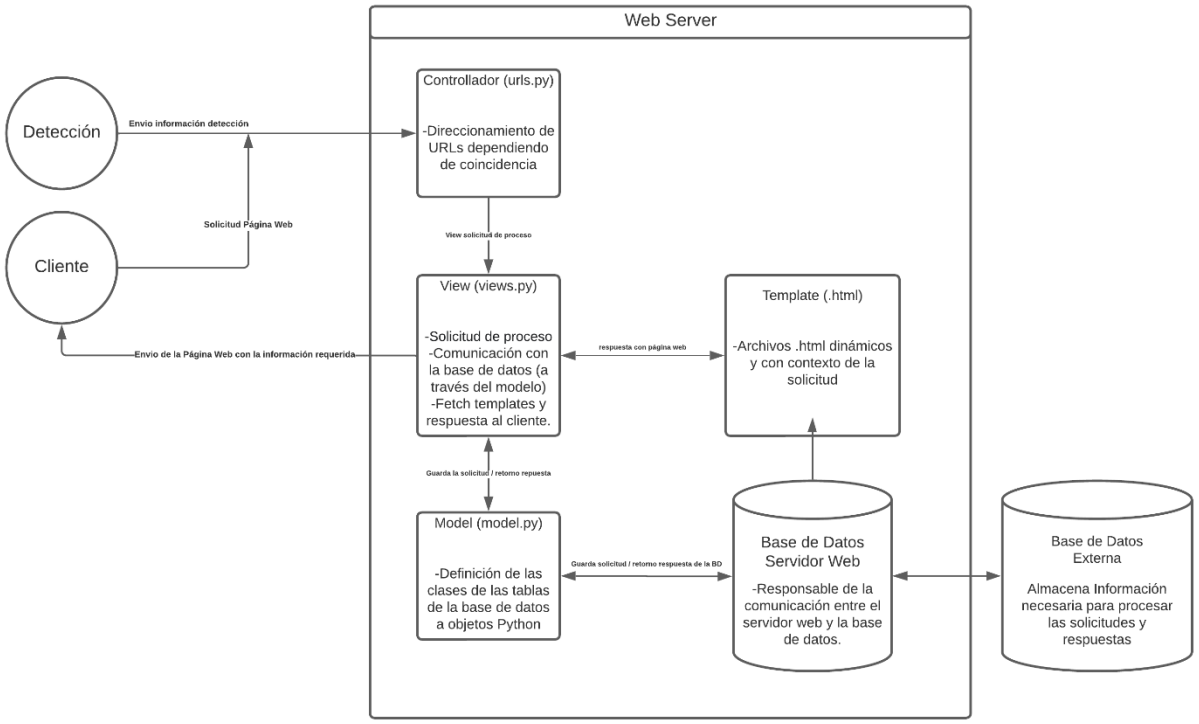
Diseño de la Aplicación Web

Una aplicación web tiene back-end, el cual corre en el servidor y el front-end el cual es ejecutado únicamente para el cliente. La comunicación entre el backend, frontend y la tarjeta jetson nano es mediante RESTful API. En la Figura 11 se aprecia la arquitectura del servidor web, el servidor se encuentra desarrollado en Django al ser este un framework de Python de alto nivel brinda la ventaja de desarrollar de manera limpia, usar el patrón de diseño modelo-vista-template (MVT).

En este patrón de diseño el apartado de modelo tiene la capacidad de acceder a la base de datos, también contamos con vista que es en donde se encuentra la lógica de funcionamiento de la aplicación y templates que es la capa de presentación y en donde se toma la decisión de que página o documento se muestra.

Figura 11

Arquitectura Servidor Web MVT (Model-View-Template)



Consideraciones

Las consideraciones para el desarrollo la aplicación web se centran el usuario administrador de la aplicación, por lo que los requisitos son:

- Registrar: El usuario puede registrarse en el sistema, para esto debe ingresar la información necesaria para el registro. El registro se debe realizar por una sola ocasión.
- Ingresar: Si el usuario se encuentre registrado, este debe poder proporcionar sus credenciales válidas para el inicio de la sesión, estas credenciales al ser autenticadas permiten el ingreso al dashboard principal del servidor web.
- Envío de Notificaciones: Con la detección del arma tipo pistola o cuchillo desde la tarjeta de desarrollo jetson nano, el servidor es el encargado de alertar mediante SMS o Email de lo sucedido.

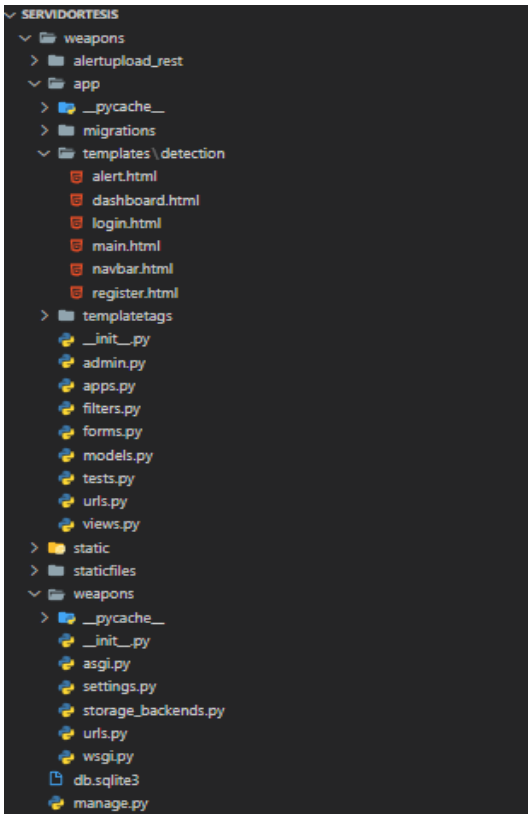
- Alertas: Es la pantalla donde se observa los detalles del evento, tales como: la imagen, ubicación y fecha.

Estructura del Proyecto

El proyecto tiene diversos directorios, donde cada uno de ellos cumple ya sea colectivamente o individuales roles para lograr el funcionamiento general. En la siguiente figura se ilustra la estructura del proyecto.

Figura 12

Estructura de archivos del proyecto



.

El directorio del proyecto contiene diferentes carpetas con varios archivos que componen la aplicación completa. A continuación, se explica un breve de estos:

Modelo de Datos

Las aplicaciones desarrolladas en el framework Django son capaces de acceder y administrar los datos a través de objetos de Python dado que los modelos definen la estructura de los datos que se van a almacenar. En los modelos se detallan los atributos de cada campo.

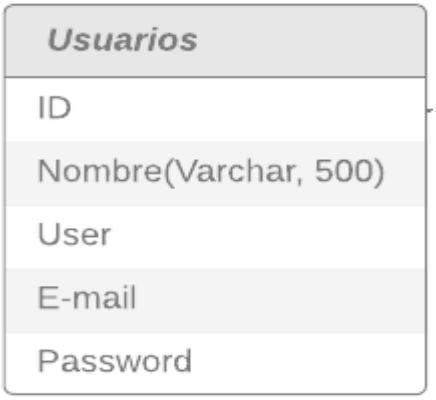
Modelos

Para el desarrollo del sistema se han utilizado dos modelos para llevar a cabo la aplicación:

- Usuarios: Es necesario establecer los campos de información que va a utilizarse en la aplicación. En la Figura 13 se detalla los campos que la aplicación guardará cada vez que un usuario se registre.

Figura 13

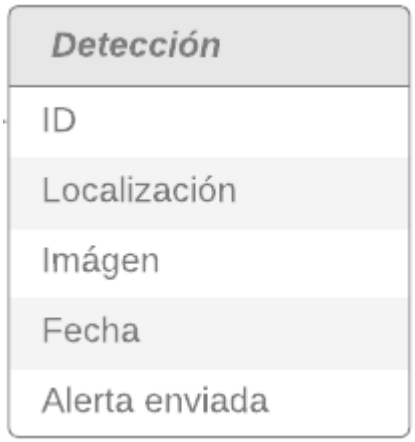
Elementos clase Usuarios



- Detección: El modelo de detección únicamente se utilizará para almacenar la información proveniente de la detección del dispositivo Edge Computing. En la Figura 14 se muestra la información de la detección que se va a almacenar en la base de datos.

Figura 14

Elementos clase Detección



Amazon S3

Dado a que el sistema guarda las capturas del evento se decidió optar por un servicio de Amazon para el almacenamiento de las imágenes. Amazon S3 es un servicio que permite el almacenamiento de objetos que presenta las siguientes características:

- Escalable
- Disponibilidad de los datos
- Seguridad
- Alto rendimiento

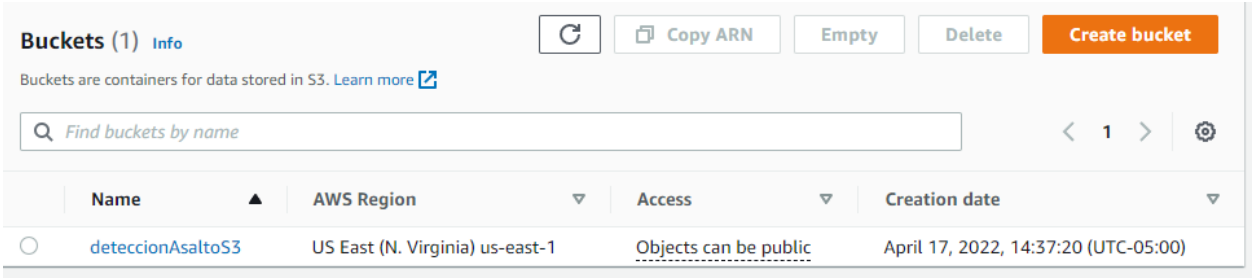
Como paso inicial antes de empezar a almacenar los datos se debe crear una cuenta en AWS y configurarla de acuerdo a los requerimientos. Considerando que Amazon proporciona una versión gratuita es la que se va a utilizar en el presente desarrollo del sistema. Una vez creada la cuenta necesitamos crear un usuario dado que para permitir la conexión necesitamos el token de acceso el cual es único para cada usuario.

Para permitir el almacenamiento de las imágenes debemos crear un bucket. Un bucket es un contenedor de objetos en el cual se van a cargar los objetos en este caso los objetos a cargar son las imágenes de la detección.

Una vez creado el bucket como se muestra en la Figura 15, se debe considerar que el acceso sea público y crear las carpetas necesarias para él envió de la información.

Figura 15

Bucket del sistema

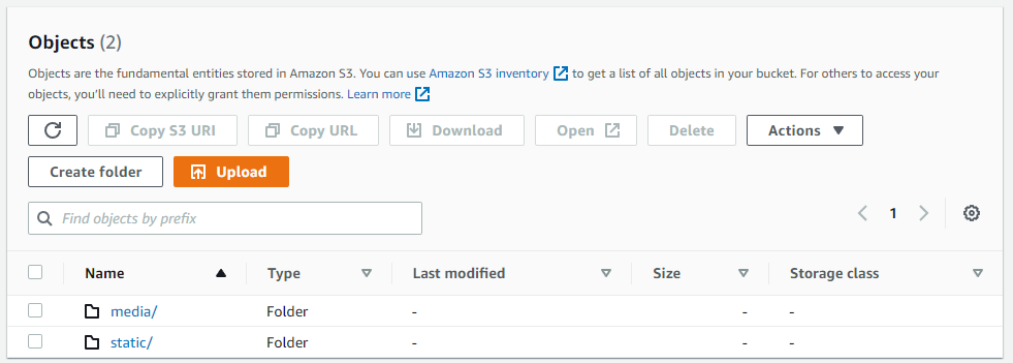


Amazon S3 proporciona buckets globales, por esta razón el nombre que se decide para el bucket debe ser único. Además, por motivos de accesibilidad debemos escoger la zona esta depende de en qué parte del mundo nos encontremos, para mantener una buena velocidad de acceso a la información se recomienda seleccionar la región más próxima a donde va a ser utilizada la aplicación web. A pesar de que un bucket es capaz de almacenar cualquier tipo de dato debemos tener presente que la versión gratuita de AWS presenta limitaciones a las cuales tenemos que regirnos.

Se aconseja que para el almacenamiento de imágenes se cree una carpeta con el nombre media/ como se ilustra en la Figura 16 y para los archivos de imágenes y fuentes del diseño del front-end una carpeta de nombre static/.

Figura 16

Objetos del Bucket deteccionAsaltoS3



Vistas

En el framework Django, la parte de métodos que son necesarios para realizar las acciones lógicas de la aplicación se deben localizar en el archivo views.py. Estos métodos toman una solicitud web y devuelven una respuesta. Esta respuesta puede ser un contenido HTML, una redirección, errores, entre muchas cosas más.

A continuación, haciendo uso de diagramas de flujo se describirán las principales funciones de la aplicación web:

- Registrar
- Iniciar
- Principal
- Envió de Notificaciones

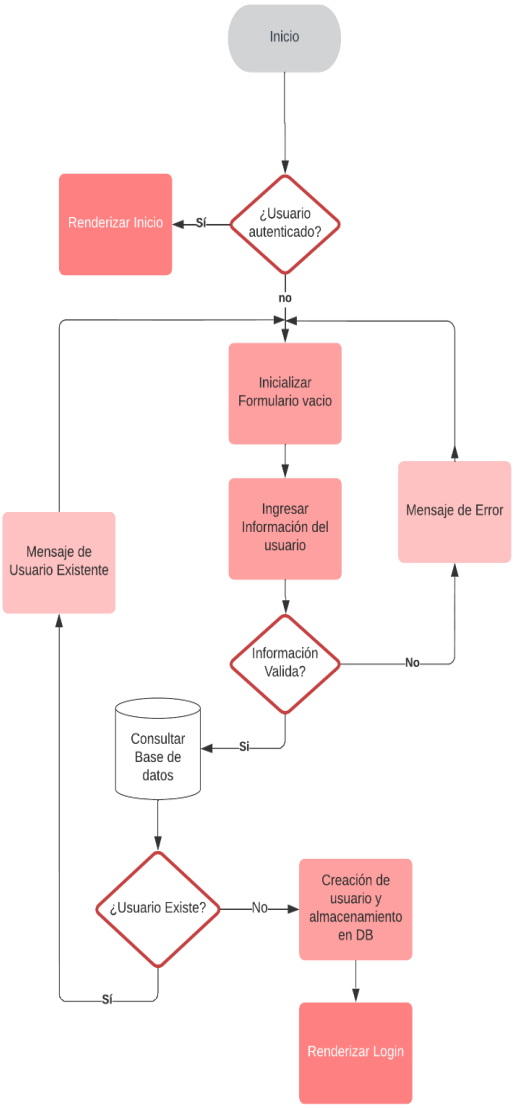
Registrar

La función registrar recibe los parámetros usuario, email, contraseña y confirmación de contraseña. Estos campos son enviados mediante una petición post a ser validados. Una vez

validados se comprueba su existencia en los registros de la base de datos, si este usuario no existe se procede a guardarlo y renderizar la pantalla de Login. Previo a todo este procedimiento se valida de que no existe una sesión iniciada, si esta existe se renderiza la pantalla de Inicio. En la Figura 17 se puede observar el funcionamiento de la vista registrar mediante la representación de un diagrama de flujo.

Figura 17

Diagrama de flujo vista registrar



Una vez con la lógica del funcionamiento de la función se realiza la implementación de código la misma que quedaría de la siguiente manera:

Figura 18

Método registrar

```
def registrar(request):
    if request.user.is_authenticated:
        return redirect('home')
    else:
        formulario = CreateUserForm()
        #verificamos que se reciba un metodo post
        if request.method == 'POST':
            formulario = CreateUserForm(request.POST)
            if formulario.is_valid():
                formulario.save()
                user = formulario.cleaned_data.get('username')
                messages.success(request, 'La cuenta ha sido creada exitosamente' + user)
                return redirect('login')
        context = {'form':form}
        return render(request, 'detection/register.html', context)
```

En la construcción del método registrar, debemos considerar que usamos un formulario dado que esta es una herramienta que recolecta datos desde una entrada. Para validar el formulario se debe ejecutar un post dado que se debe realizar la consulta a la BD de si el usuario existe o no.

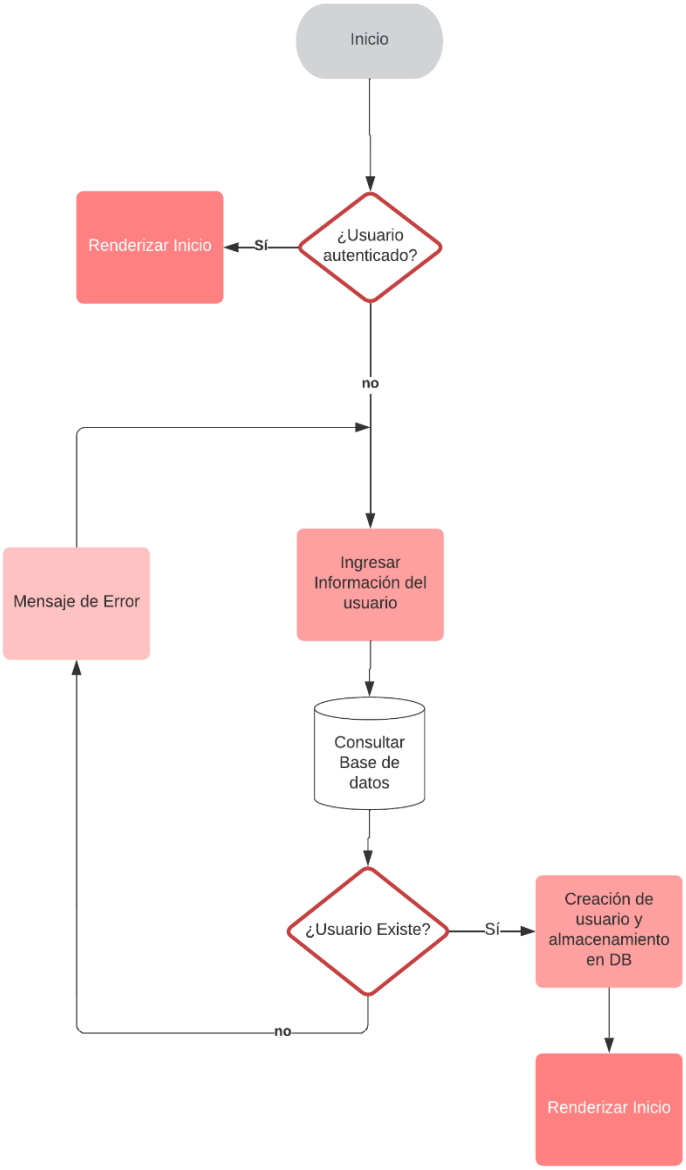
En el código también se observa que el primer paso es comprobar si existe una sesión inicializada en el navegador. Si existe la sesión se procede a renderizar el inicio caso contrario se procede a mostrar la pantalla de registro, una vez que se ha completado el ingreso de los datos y estos sean dados como válidos se muestra el mensaje de que se almacenaron los datos correctamente y se muestra la pantalla de ingresar donde se debe introducir los datos de acuerdo con el registro anterior.

Ingresar

La función ingresar toma los valores de usuario y contraseña, Si los datos coinciden con los de la base de datos se renderiza la pantalla de inicio. En la Figura 19 se observa la representación de la función ingresar de la aplicación web.

Figura 19

Diagrama de flujo vista ingresar



En el método de ingresar se requiere realizar una validación del formulario adecuada y segura puesto que en este se ingresa la contraseña, la cual debe estar cifrada para evitar el robo de la información proporcionada al sistema.

Una vez con la lógica del funcionamiento de la función se realiza la implementación de código la misma que quedaría de la siguiente manera:

Figura 20

Método ingresar

```
def ingresar(request):
    if request.user.is_authenticated:
        return redirect('home')
    else:
        if request.method == 'POST':
            nombreUsuario = request.POST.get('username')
            passwordUsuario =request.POST.get('password')

            user = authenticate(request, username= nombreUsuario, password= passwordUsuario)
            if user is not None:
                login(request, user)
                return redirect('home')
            else:
                messages.info(request, 'Usuario o contraseña son incorrectos')
```

Para el método de ingresar al igual que en el método anterior, primero verificamos la existencia de una sesión iniciada en el navegador, de encontrarse se redirige directamente al inicio caso contrario se despliega la pantalla de ingreso.

El uso de la función ingresar necesitamos el proporcionar el usuario y su respectiva contraseña, estos campos son enviados al servidor y autenticados en caso de que esto se cumpla se despliega la pantalla de inicio, caso contrario se muestra un mensaje de error debido a que los datos proporcionados no cumplen la autenticación.

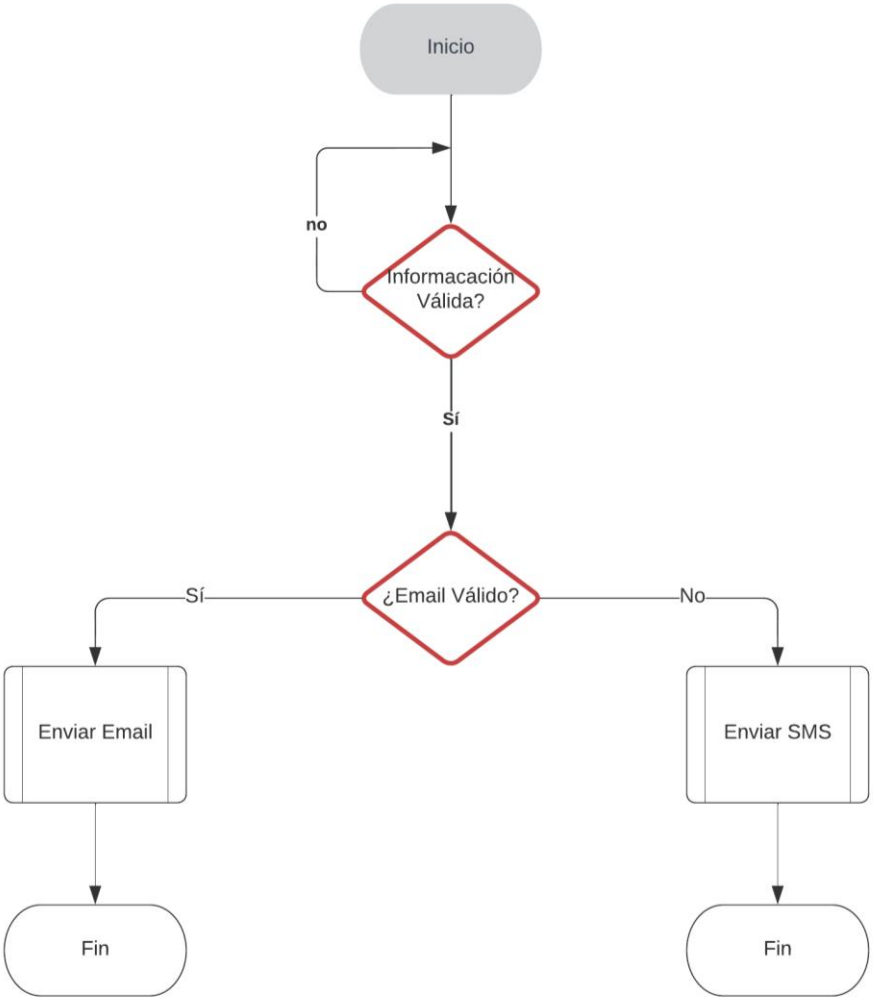
Envió de Notificaciones

La función de envió de notificaciones recibe los campos detallados en la Figura 12. Estos campos son enviados mediante una petición post a ser almacenados en la base de

datos. Una vez almacenados se procede al envío de la notificación creando un cuerpo del mensaje y se envía la notificación vía email usando un servidor SMTP o vía SMS usando la plataforma Twilio. En la Figura 21 se observa el diagrama de flujo.

Figura 21

Diagrama de flujo método envió de notificaciones



Para realizar un código limpio se optó por utilizar funciones para el envío de SMS y email, para si es el caso en un futuro escalar la aplicación y reutilizar el código.

Una vez con la lógica del funcionamiento de la función se realiza la implementación de código la misma que quedaría de la siguiente manera:

Figura 22

Método de envío de alerta

```
def postAlerta(request):
    serializer = UploadAlertSerializer(data=request.data)
    if serializer.is_valid():
        serializer.save()
        sendSms(serializer)
        sendEmail(serializer)
```

Envío SMS

Para el envío de la notificación por sms se utiliza el servicio de Twilio.Twilio es una plataforma de comunicación basada en la nube que permite una fácil integración con Django. Además, permite el envío a nivel global de SMS y con una versión de prueba que permite el desarrollo de las pruebas. Para hacer uso del servicio de Twilio es registrarnos y activar el número proporcionado por la plataforma para pruebas con sus respectivas credenciales de seguridad para poder usarlo sin inconvenientes. El método desarrollado para él envió se muestra a continuación en este se observa que existe un método que prepara el mensaje a enviar. Este mensaje puede ser modificado de acuerdo a las necesidades del sistema. Para este caso contiene el link en donde se muestra la información del evento.

Figura 23

Método de envió de SMS

```
def sendSms(serializer):
    client = Client(settings.TWILIO_ACCOUNT_SID, settings.TWILIO_AUTH_TOKEN)
    message = client.messages.create(body=prepare_alert_message(serializer),
                                     from_=settings.TWILIO_NUMBER,
                                     to=serializer.data['alert_receiver'])
```

Envió Email

Para realizar el envío vía mail de la notificación del evento, se decidió utilizar el protocolo de transferencia simple de correo (SMTP) dado que este nos permite el envío de mails de manera sencilla. Para el desarrollo se utiliza el servicio SMTP otorgado por Gmail que para hacer uso de esta herramienta simplemente debemos configurar la seguridad de nuestra cuenta de Google, obtener las credenciales necesarias y el puerto de comunicación que por default es el puerto 25.

Figura 24

Método de envío de Email

```
def send_email(serializer):
    send_mail('Asalto Detectado!',
    prepare_alert_message(serializer),  'jsgv1405@gmail.com',
    [serializer.data['alert_receiver']],
    fail_silently=True,)
def prepare_alert_message(serializer):
    uuid_with_slashes = split(serializer.data['image'], ".")
    uuid = split(uuid_with_slashes[3], "/")
    url = 'https://servidortesis.herokuapp.com/alert/' + uuid[2]
    return 'Asalto detectado! ver alerta en ' + url
```

Templates

En el directorio Template van todos los archivos HTML, que se emplearán en la aplicación. Para eso se utiliza un motor de plantillas guarda cierta estructura y solo cambie la información que en esta se despliega. Para estilizar los templates se utilizó CSS el mismo que nos asigna distintos estilos a nuestra aplicación. Además, se utiliza la biblioteca de Boostrap para estilizar de manera mucho más rápida las vistas que componen la aplicación.

Capítulo IV. Desarrollo e Implementación del Sistema

Una pieza clave en el desarrollo del sistema para él envió automático de información de alerta en eventos de asalto es el uso de un hardware potente, especializado y de un tamaño moderado, estas características son vitales debido a que del hardware depende el rendimiento y eficiencia en la ejecución del software.

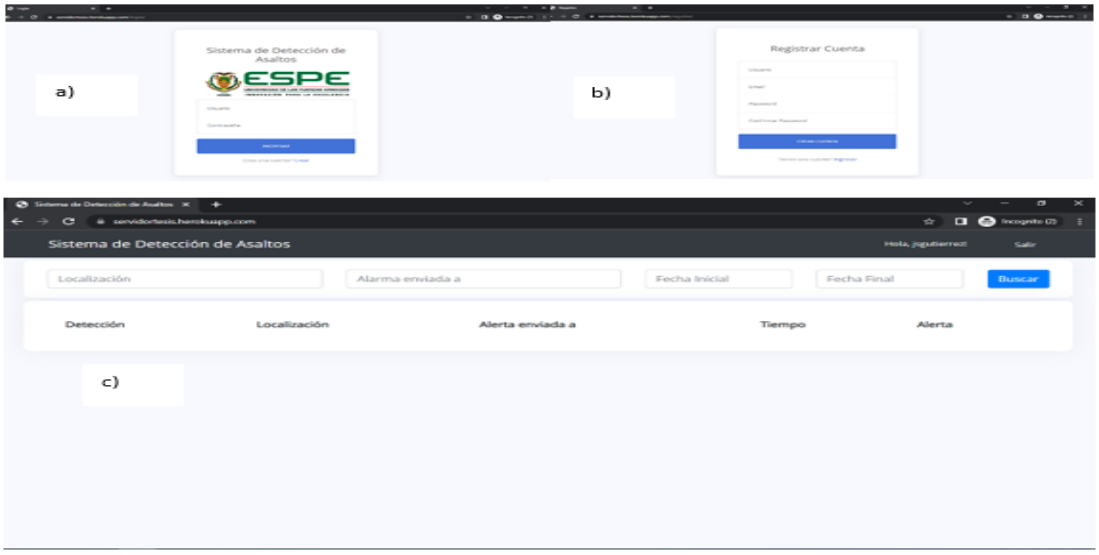
En la actualidad existen múltiples dispositivos de Edge computing con características que permiten un alto nivel de procesamiento. En los últimos años Nvidia ha lanzado múltiples dispositivos enfocados en la inteligencia artificial y a un precio asequible. Para la implementación del sistema se decidió optar por el uso de la tarjeta de desarrollo Nvidia Jetson Nano 2Gb.

Servidor Web

El despliegue en producción de la aplicación web se la realizó utilizando el hosting de Heroku. La sección de anexos contiene los links relacionados con el desarrollo del sistema. En la Figura 25 se ilustra el servidor desplegado con sus distintas vistas.

Figura 25

Servidor web desplegado con las vistas: a) Ingresar, b) Registrar y c) Inicio



Nvidia Jetson Nano 2Gb

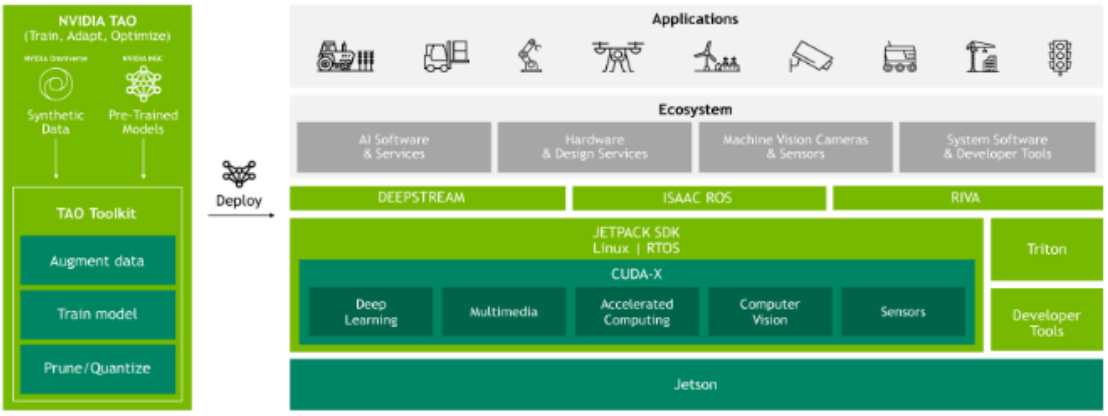
La tarjeta de desarrollo Nvidia Jetson Nano 2GB es un dispositivo pequeño, con un poderoso poder de procesamiento capaz de correr múltiples redes neuronales en paralelo para aplicaciones como clasificación de imágenes, segmentación, detección de objetos, procesamiento de voz (Nvidia, 2021).La jetson nano incluye múltiples librerías, APIS, módulos que vienen por default entre estas las más importantes para el desarrollo de este sistema tenemos:

- Computer Vision
- CUDA
- cuDNN
- OpenCV
- TensorRT

En la Figura 26 se ilustra todas las características que incluye el dispositivo.

Figura 26

Características Jetpack



Nota. La figura representa las características que integra la tarjeta Jetson Nano. Tomado de (Nvidia, 2021)

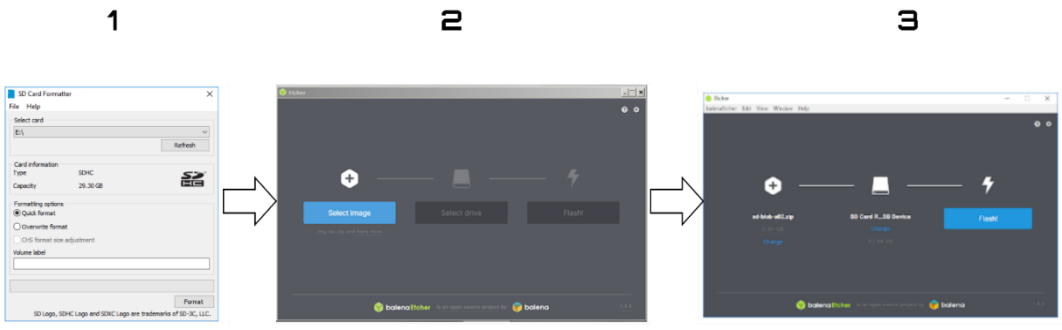
Instalación de sistema operativo

El dispositivo Edge computing Jetson Nano usa una tarjeta microSD como dispositivo de arranque y almacenamiento principal. Para instalar el sistema operativo se debe seguir los siguientes pasos:

1. Descargar la Imagen SDK: La imagen oficial para el uso de la Jetson Nano es la NVIDIA Jetpack que se encuentra desarrollada para la creación de aplicaciones de inteligencia artificial. La versión a utilizar es la 5.0.1, la cual cuenta con un kernel de Linux 34.1.1. La descarga se la realiza del sitio oficial de Nvidia.
2. Grabar la imagen en la microSD: Para lograr grabar la imagen en la microSD debemos seguir los pasos que se pueden observar en la Figura 27. En donde como primer paso tenemos el formateo de la microSD, posteriormente con el uso del software Balena Etcher procedemos a grabar la imagen.

Figura 27

Grabado imagen SDK en una microSD



Nota. La figura muestra los pasos para grabar la imagen es adaptada de (Nvidia, 2021)

3. Configuración Inicial: Con la imagen en la microSD se procede a insertar la microSD en la Nvidia Jetson Nano, encenderla y configurarla. Una vez realizado estos pasos tenemos la tarjeta lista para usar como se observa en la Figura 28.

Figura 28

Primer encendido Jetson Nano

Nota.

Adaptada de (Nvidia, 2021)

Conexión 4G

En los últimos cincuenta años el internet y las comunicaciones móviles han visto un desarrollo significativo. Este desarrollo se da por la revolución digital que se vive en la actualidad donde la información se genera todo el tiempo en todas partes.

Debido al incremento en la telefonía móvil han permitido un desarrollo de redes móviles confiables y de calidad. Existen muchos operadores en el país que brindan los servicios de voz y datos, para el desarrollo del sistema se utilizará la operadora Tuenti.

Módulo SIM7600G-H

Este es un módulo diseñado y comercializado por la empresa WaveShare Electronics. El módulo SIM7600G-H se encuentra optimizado para ser usado con la tarjeta Jetson Nano admite comunicación 4G/3G/2G global y posicionamiento GNSS (Global Navigation Satellite System) (Waveshare, 2021). En la Figura 29 se aprecia el módulo colocado en la tarjeta Jetson Nano. Este dispositivo permite habilitar funciones tales como:

- Conexión 4G
- Llamadas telefónicas

- Posicionamiento Global
- Envió de SMS

Figura 29

Instalación módulo SIM 7600G-H en la tarjeta Jetson Nano

Nota. Figura tomada de (Waveshare, 2021)

Configuración Software

Una vez colocado el módulo en la Jetson Nano se necesita realizar los siguientes pasos para que el módulo funcione correctamente.

1. Abrir el terminal e instalar las librerías de pyserial la cual permite obtener el acceso al puerto serial, Jetson.GPIO la misma que permite obtener acceso a los pines de entrada y salida que posee la tarjeta de desarrollo y minicom que permite la comunicación serial con dispositivos a través de puertos seriales.

2. Descargar el compilador sincom_wwan el cual es un controlador de la red NDIS de Linux, esto se lo consigue mediante los siguientes comandos.
3. Una vez el archivo makefile fue modificado procedemos a compilarlo e instalarlo con los comandos mostrados a continuación.

Una vez completados estos pasos el dispositivo está listo para poder obtener conectividad 4G. Los pasos para esta configuración se encuentran en su totalidad en el repositorio de GitHub con el nombre de simcom_wwan-setup (Stearns, 2019).

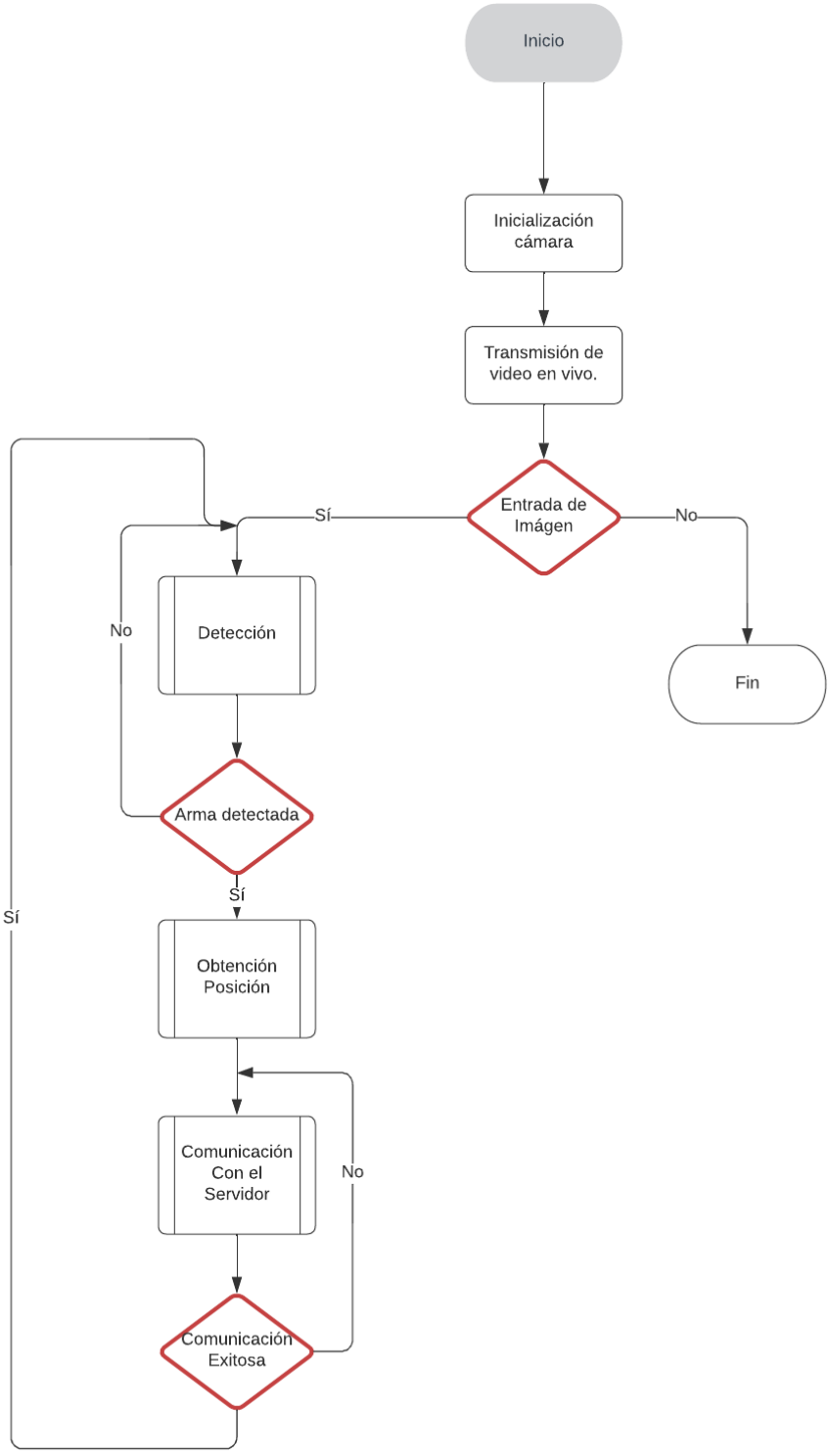
Implementación Detector

Para la implementación del detector en la tarjeta de desarrollo Jetson Nano se requiere realizarlo mediante un Docker. Un Docker permite la creación, testear e implementar aplicaciones de software en contenedores, estos son unidades estandarizadas que contienen librerías, herramientas logrando crear aplicaciones escalables y con la capacidad de ejecutarlas en cualquier entorno.

En la Figura 30, se ilustra el diagrama de flujo del detector donde podemos determinar tres componentes claves que son: la detección, obtención de la información y comunicación con el servidor web.

Figura 30

Diagrama de flujo detección



Implementación Algoritmo de detección

La detección de objetos es una aplicación de la visión por computador que está estrechamente relacionada con el procesamiento de imágenes puesto que detecta clases específicas dentro de una imagen. En la actualidad existen muchos modelos de detección para la implementación de este sistema se usará el modelo SSD-MobileNet, el cual es el modelo por defecto del Docker instalado en la tarjeta Jetson Nano.

El modelo de detección propuesto al tener objetos atípicos se necesitó re-entrenarlo, por lo que para su implementación se utilizará los pesos generados en el capítulo anterior. Este peso cuenta con dos clases las cuales son:

- Handgun
- Knife

Para poder realizar la detección se realiza el código Python con el uso de las librerías adecuadas y ejecutado sobre el Docker para obtener un rendimiento óptimo en la ejecución. En el apartado de anexos del presente trabajo se encuentra el enlace al repositorio de Github donde se puede hallar el código desarrollado para la detección.

Obtención de la ubicación

La integración del módulo SIM7600G-H permite el posicionamiento GNSS (Global Navigation Satellite System), para lo cual es necesario utilizar comandos AT los cuales son cadenas de datos que presentan como prefijo “AT” y se utilizan para enviar peticiones de comunicación hacia un módulo, estos comandos son enviados a través de una interfaz serial asíncrona.

En la Tabla 5, se detallan los comandos utilizados para poder realizar la comunicación entre el módulo SIM7600G-H y la jetson nano. El módulo se comunica con la tarjeta jetson

nano mediante comunicación serial, por lo que es indispensable que comprobemos que la comunicación serial del dispositivo esté activada.

Tabla 5

Comandos AT para obtención de ubicación

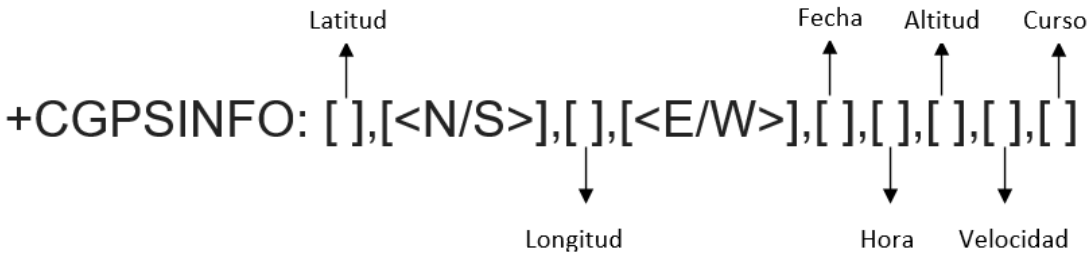
Comandos AT	Función
AT+CGPS=1	Inicializar GPS
AT+CGPSINFO	Obtención arreglo de la posición.
AT+CGPS=0	Detener GPS

Una vez obtenido el arreglo de la posición está compuesto por diferente información, en la Figura 31 se muestra la información que se incluye en este arreglo.

Figura 31

Arreglo información GPS (CPSINFO)

La información tanto de latitud y longitud debe ser decodificada antes de poder utilizarla en el



sistema para lo cual se debe aplicar la siguiente fórmula para la latitud:

$$latitud_{dec} = \frac{latitud}{100} * sentido$$

Donde

$sentido = 1$ sí “N”

$sentido = -1$ sí “S”

De la misma forma tenemos que decodificar la longitud:

$$longitud_{dec} = \frac{longitud}{100} * sentido$$

Donde

$sentido = 1$ sí “E”

$sentido = -1$ sí “W”

Comunicación con el servidor web

Para realizar la comunicación con el servidor web se la realiza mediante el uso de endpoints de la API (Interfaz de Programación de Aplicaciones) dado que estas sirven para comunicar dos aplicaciones dado un conjunto de reglas. En la Tabla 6 se observan los endpoints, una breve descripción de su función en el sistema comunicación con el servidor, el método HTTP correspondiente para su uso y la información necesaria para su funcionamiento.

Tabla 6

EndPoints de Comunicación con el servidor web

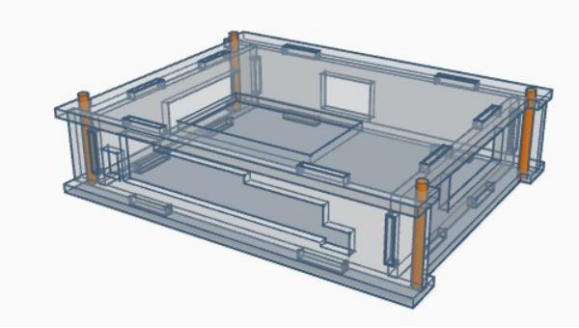
endPoints	Método HTTP	Data	Descripción
api/get_auth_token/	POST	{‘usuario’,’contraseña’}	Inicia la sesión en el servidor web y obtiene el token de acceso.
api/images/	POST	{‘token’,’imagen’,’localización’,’alerta_recibida’}	Envía la información correspondiente a la detección hacia el servidor.

Montaje e instalación del sistema

El sistema al incorporar un dispositivo de edge computing como lo es la tarjeta Jetson Nano, requiere la incorporación de una carcasa para la cual evite el deterioro de los componentes electrónicos, que permita la fácil conexión de los periféricos, sea resistente y permita la correcta ventilación del dispositivo. Dado estas consideraciones se eligió realizar un diseño asistido por computador (CAD) el mismo se muestra en la Figura 32.

Figura 32

Diseño Carcasa del Sistema



Para la fabricación de la carcasa se optó por utilizar tecnologías de impresión 3D, por lo que se seleccionó un filamento ABS debido a que su presenta una resistencia significativa. Además, es un material que puede ser reciclable. En la Figura 32 se ilustra la instalación de la tarjeta Jetson Nano en la carcasa.

Figura 33

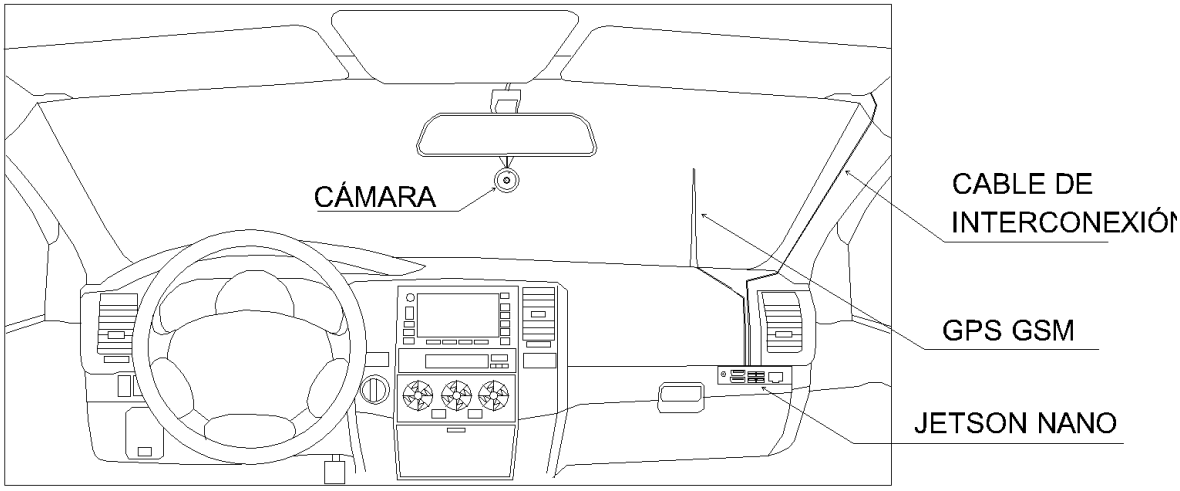
Instalación Carcasa

Una de las principales ventajas de los dispositivos de Edge computing es que brindan la versatilidad de instalación, el sistema debido a que está pensado en la detección de asaltos con el uso de armas tipo pistolas o cuchillos dentro de vehículos debe ser instalado en el que la cámara disponga de un rango alto de visión por lo que se optó que el lugar de instalación sea como se muestra en la Figura 34.

Con esta ubicación se logra un amplio campo de visión y un lugar para evitar inconvenientes de desconexión involuntaria del sistema. Además, se consigue que la parte crucial del sistema como lo es la jetson nano al estar en la guantera es un lugar discreto y difícil de que lo puedan manipular.

Figura 34

Forma de instalación del sistema



Para la instalación del sistema en un ambiente de prueba se consideró que la fuente de alimentación para la tarjeta Jetson Nano provenga del vehículo mediante la conexión de un cable tipo C y un adaptador de cargador para auto. Finalizando todo el proceso de desarrollo tanto de software como de instalación del hardware en el entorno de pruebas, se obtiene el sistema instalado como se muestra en la Figura 35.

Figura 35

Sistema Instalado en vehículo

Capítulo V. Pruebas y Resultados

En el presente capítulo se muestran tanto los resultados del funcionamiento como la evaluación del modelo de detección y la obtención de las coordenadas de la ubicación que son los elementos principales de la información que proporciona el sistema. Para la realización de las pruebas, la señal de vídeo se toma en tiempo real por lo que el proceso de detección se desarrolla a 35 FPS (frames per second).

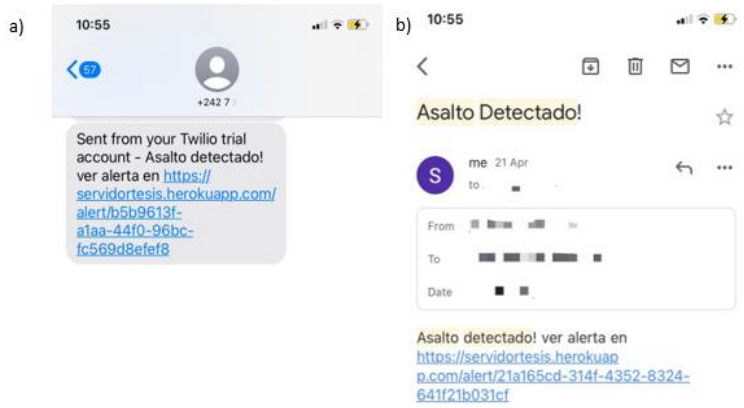
Con el fin de comprobar el correcto funcionamiento del prototipo tanto del envío de notificaciones como de la detección de armas tipo pistola o cuchillo, se simuló un evento de asalto dentro de un vehículo haciendo un recorrido de calles en una ruta específica.

Pruebas de funcionamiento

Con el sistema en funcionamiento, se procede a simular un evento de asalto para el cual se usa 2 tipos de pistolas de juguete y 3 tipos distintos de cuchillos. El sistema una vez detecta cualquiera de estas dos clases procede al envío de la notificación a través de SMS como se ilustra en la Figura 36 a) o mediante email como se muestra en la Figura 36 b). Esta notificación incluye un enlace con la información detallada del evento.

Figura 36

Notificaciones SMS o email del sistema

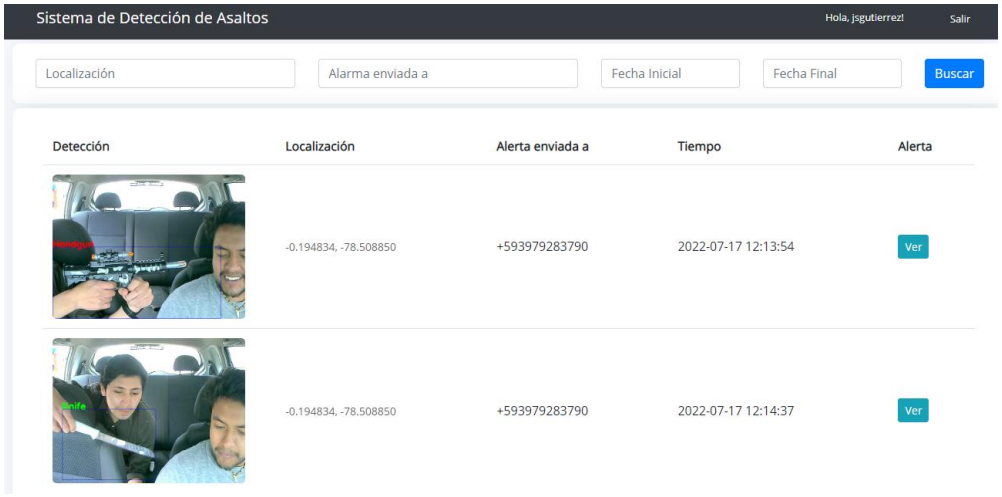


Una vez obtenida la notificación de la detección en el enlace se muestra la información del evento con la siguiente:

- Imagen de la detección
- Coordenadas de la ubicación del vehículo con hipervínculo a Google maps.
- Número de teléfono o email a quien fue enviada la alerta
- Fecha y hora de la detección.

Figura 37

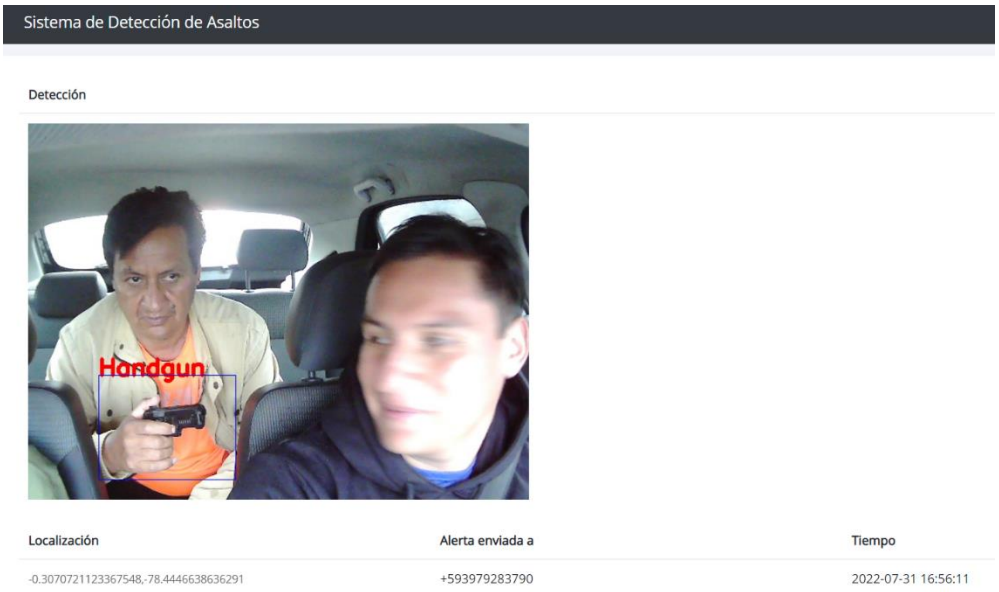
Funcionamiento Servidor Web



El servidor, además, dispone la posibilidad de visualizar una alerta en específico en la cual se puede ver la información más detalladas y la localización es un hipervínculo hacía la aplicación de Google Maps para la visualización del punto donde se realizó la detección. En la Figura 38 muestra la detección de una simulación de un evento de asalto en el interior de un vehículo donde se observa la imagen de la detección, las coordenadas de la ubicación donde se realizó la detección, la hora en la que se realizó la misma y a quien fue dirigida la alerta.

Figura 38

Visualización alerta del Servidor Web



Evaluación detección

El sistema al estar diseñado para enviar notificaciones de eventos en los cuales una persona está en potencial peligro, el margen de error en la detección debe ser lo más reducido posible, dado que una falsa alarma puede resultar en serio problema. Para la evaluación de detección se tiene como núcleo principal el uso de la matriz de confusión debido a que esta otorga la comparativa entre la predicción y la realidad.

Figura 39

Matriz de Confusión

		Clase de Predicción	
		Positivo	**Negativo**
Clase real	**Positivo**	Positivo Verdadero (PV)	Falso Positivo (FP)
	Negativo	Falso Negativo (FN)	Verdadero Negativo (VN)

Partiendo de la matriz de confusión se desarrollan distintas métricas para medir el desempeño de un detector, la Tabla 7 se muestran las más importantes, con su fórmula de cálculo y descripción.

Tabla 7

Métricas de rendimiento de la matriz de confusión

Métrica	Fórmula	Descripción
Precisión	$Acc = \frac{PV + VN}{PV + FP + FN + VN}$	Calcula el porcentaje de ejemplos positivos que realmente fueron positivos.
Tasa de Verdaderos Positivos (Recall)	$Recall = \frac{PV}{PV + FN}$	Determina el porcentaje de verdaderos positivos entre todos los elementos positivo positivos.
Tasa de Verdaderos Negativos (Miss Rate)	$MissRate = \frac{FN}{PV + FN}$	Calcula el porcentaje de ejemplos falsos negativos de todos los ejemplos positivos

Para lograr el desarrollo correspondiente a la etapa de pruebas de una manera adecuada se debe considerar el funcionamiento integral del sistema. Dado esto las pruebas de rendimiento del detector se ejecutan en base al funcionamiento general del sistema, se planteó dos escenarios, la diferencia entre ellos es la hora en la que se realizó las pruebas. Esto con el objetivo de conseguir dos tipos de iluminación que ingresa al interior del vehículo y afectó directamente a la detección.

Escenario 1

El escenario de pruebas 1 corresponde al sistema ejecutándose en el interior del vehículo tal y como se muestra en la Figura 37, la hora de ejecución de este escenario se desarrolló de 11h00 a 11h30.

En el intervalo de tiempo simulado se simularon eventos de asalto con el uso de armas de juguete de tipo pistolas y cuchillos para conseguir que el sistema detecte el evento. La Figura 40 muestra algunas de las detecciones realizadas.

Figura 40

Detecciones Escenario 1

Las consideraciones fundamentales en los escenarios son el horario y el nivel de luminosidad. Bajo estas consideraciones el escenario 1 se desarrolló en un horario en el que las condiciones climáticas de la ciudad de Quito presentan una luz de sol intensa y por ende una luminosidad al interior del vehículo alta.

Para conseguir que la detección se la realice en un entorno más real posible al de uno presentado en un vehículo de transporte público de la capital, se decidió realizar con dos personas: un chofer que en este caso será el afectado por el hecho a ocurrir y un pasajero que intentará asaltarlo.

Una vez con el entorno establecido se procede a simular un asalto con el uso de armas de tipo pistolas y cuchillos. Por lo que durante la ejecución del sistema se procede a realizar movimientos bruscos y lentos para ver la reacción del sistema. Dado que el sistema tiene la capacidad de almacenar el frame de la detección realizada se lleva a cabo la construcción de la matriz de confusión, la misma que se muestra en la Figura 41. Dicha matriz está compuesta por 120 detecciones.

Figura 41

Matriz de confusión Escenario 1

		Clase de Predicción	
		Pistola o Cuchillo	**No Pistola o Cuchillo**
Clase real	**Pistola o Cuchillo**	86	17
	No Pistola o Cuchillo	12	5

Con los resultados obtenidos de las detecciones se determina que los verdadero positivos son más, por lo que se puede determinar que el sistema está funcionando de una manera correcta en un horario y unas condiciones climáticas que permitan el ingreso de una gran cantidad de luz en el interior del vehículo. Sin embargo, en la Figura 41 de la matriz de confusión también se nota que la cantidad de falsos positivos (FP) también es alta y esto podría ocasionar problemas al utilizar el sistema en un ambiente real.

Escenario 2

Bajo las consideraciones tomadas en el escenario 1, se planteó el presente escenario bajo un horario de ejecución de 17h30 a 16h00. En el escenario 2 las condiciones climáticas de la ciudad de Quito se vieron afectadas y se observó un clima nublado en el cual la entrada de luz natural al interior del vehículo se vio drásticamente afectada si lo comparamos con el escenario anterior.

Al tener menor iluminación, la detección se vuelve un reto mayor para el modelo de detección, esto debido a que el a pesar de que la cámara de web utilizada en el sistema presenta una buena resolución el diafragma de esta al ser fijo no se puede controlar la cantidad de luz que entra al obturador de la cámara web. Además, una de las consideraciones adicionales que se tomó para esto es realizar la misma cantidad de detecciones enviadas hacia el servidor y su correspondiente envío de notificaciones.

En el escenario 2 se realizaron 120 detecciones, las cuales fueron entre pistolas y cuchillos. Un parámetro importante para realizar esta prueba fue que al igual que en la anterior la fiabilidad de la detección de cada uno de los tipos de objetos debe ser mayor al 80% para que esta sea válida y enviada al servidor web implemento.

A continuación, se muestran las detecciones realizadas y su distribución en la matriz.

Figura 42

Matriz de Confusión Escenario 2

		Clase de Predicción	
		Pistola o Cuchillo	**No Pistola o Cuchillo**
Clase real	**Pistola o Cuchillo**	72	14
	No Pistola o Cuchillo	28	6

En la matriz de confusión expuesta del presente escenario se observa una disminución notoria en la cantidad de verdaderos positivos (VP) alcanzada, esto debido a que al

encontrarse en un lugar con menor luminosidad el desempeño del detector se ve mermado. Además, se observa que la cantidad de falsos negativos (FN) es alta y esto es debido principalmente a que el sistema al no tener una cantidad de luz suficiente se le dificulta la tarea de realizar la detección.

Comparativa Escenarios

Una de las maneras de realizar la comparativa de cómo la luz afecta el desempeño del detector, es las utilizaciones de las métricas mostradas en la Tabla 7. Estas métricas determinan el desempeño en base a las matrices de confusión encontradas.

Las métricas escogidas para el análisis son:

- Precisión: La precisión busca llevar un seguimiento en el rendimiento de un modelo en base a la cantidad de verdaderos positivos (VP).
- Recall: Es una métrica conocida como la tasa de acierto, describe la probabilidad que tiene el detector de poseer valores verdaderos positivos (VP) genuinos.
- Miss Rate: Es una métrica que busca determinar la proporción de cuando un caso positivo es detectado como un falso negativo (FN).

En la Figura 43 se observa los resultados de calcular las métricas mencionadas en los dos escenarios planteados.

Figura 43

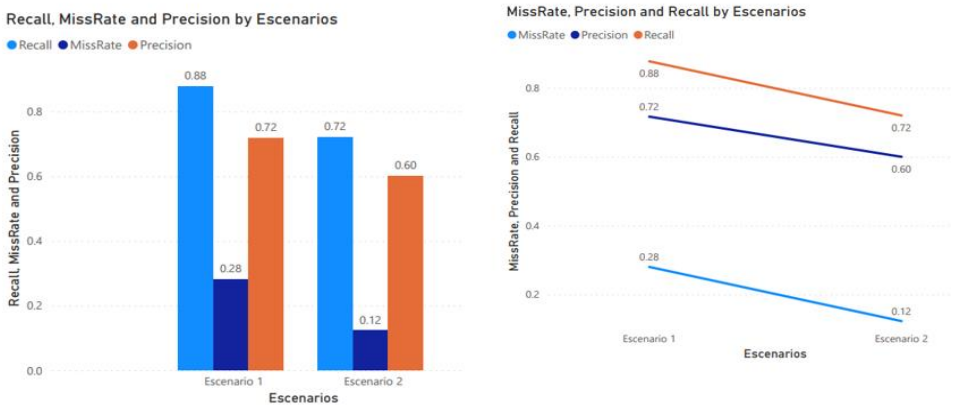
Métricas de Evaluación

	Precision	Recall	MissRate
Escenario 1	71.67%	87.76%	28.00%
Escenario 2	60.00%	72.00%	12.24%

Para observar cómo estos parámetros fluctúan de escenario a escenario se decidió representar los resultados mediante gráficos estadísticos en donde podemos ver en la Figura 44 como todas las métricas presentan una baja en su rendimiento.

Figura 44

Comparativa Métricas



De los gráficos comparativos de las métricas en ambos escenarios se puede observar que el recall es la más alta por lo que el rendimiento del detector es bueno, puesto que la tasa de aciertos es alta. Sin embargo, se observa que entre los dos escenarios el rendimiento se ve altamente afectado por la disminución de iluminación en el interior del vehículo, debido al horario y condiciones climáticas al momento de realizar las pruebas del sistema dentro del vehículo donde se instaló.

Además, de las métricas podemos destacar que la precisión que se presentan en ambos escenarios presenta valores considerablemente buenos dado el modelo de detector utilizado y de que se está utilizando un dispositivo Nvidia Jetson Nano el cual tiene limitaciones de hardware considerables, que pueden llegar a limitar el desempeño del detector.

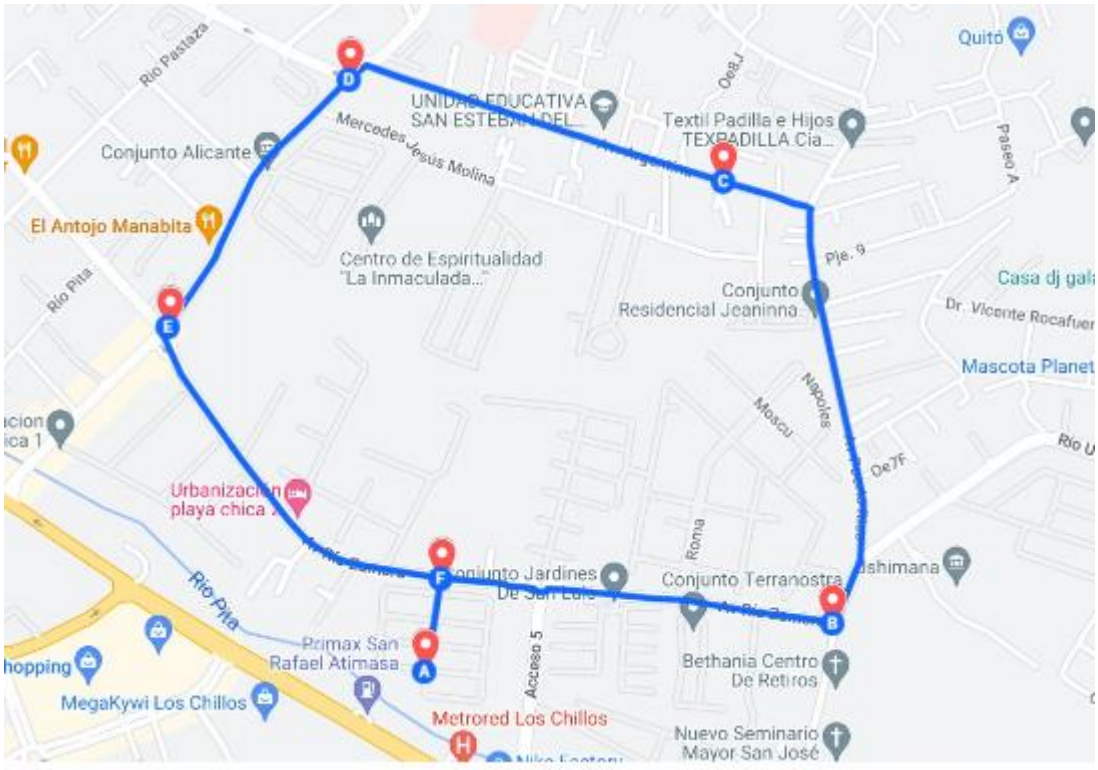
Pruebas ubicación GPS

La ubicación de la detección es de vital importancia dado que esta información puede ser utilizada para en caso de que el evento sea real poder llegar con ayuda o brindar un aviso a las autoridades de control. Para control de la ubicación se decidió realizar una ruta establecida

e ir simulando un evento de asalto. El servidor proporciona las coordenadas cada vez que se realiza la detección del arma por lo tanto se compara con la ruta seguida y vemos la similitud entra ambas. En la Figura 45 se observa la ruta recorrida por el vehículo y los marcadores donde el sistema obtuvo la coordenada del acontecimiento.

Figura 45

Ruta recorrida y marcadores de detección



En base a los resultados obtenidos de los puntos donde se realizó la detección del evento se comprueba que dichos puntos se acercan a la ruta establecida por lo que el sistema de localización se encuentra funcionando correctamente.

Para determinar la precisión en la obtención de las coordenadas de ubicación GPS, se consideró el auto parqueado y realizar simulaciones de detecciones para obtener las coordenadas de este.

En esta prueba se decidió tomar 10 muestras de las detecciones enviadas en las pruebas de detección y compararlas con las coordenadas de ubicación otorgada por la aplicación de Google Maps, obteniendo los siguientes resultados ilustrados en las dos siguientes tablas. Para las pruebas de ubicación se colocó el GPS en un lugar en el cual la recepción de la señal sea la mejor posible.

Tabla 8

Latitud

	Latitud Real	Latitud Obtenida	Error Relativo
Detección 1	-0.194834	-0.190831	2.055%
Detección 2	-0.194834	-0.193833	0.514%
Detección 3	-0.194834	-0.190933	2.002%
Detección 4	-0.194834	-0.194436	0.204%
Detección 5	-0.194834	-0.194933	0.051%
Detección 6	-0.194834	-0.192429	1.234%
Detección 7	-0.194834	-0.191133	1.900%
Detección 8	-0.194834	-0.193134	0.873%
Detección 9	-0.194834	-0.196836	1.028%
Detección 10	-0.194834	-0.191827	1.543%
		Promedio Error	**1.140%**

Tabla 9

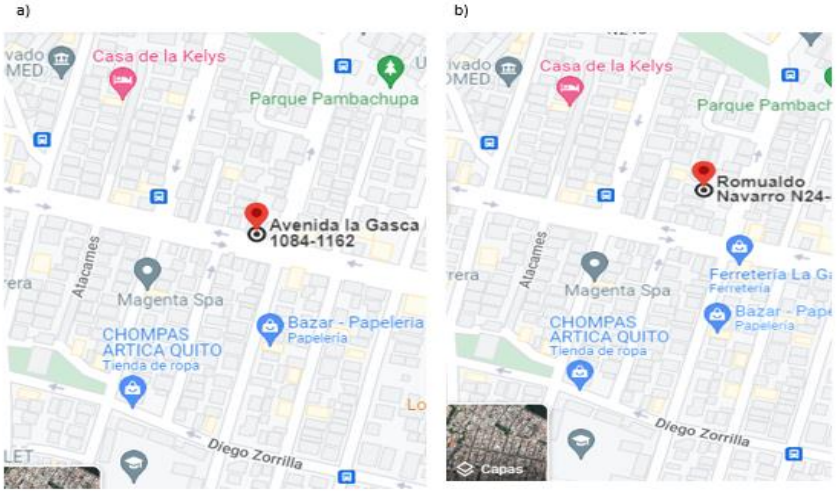
Longitud

	Longitud Real	Longitud Obtenida	Error Relativo
Detección 1	-78.50885	-78.50883	0.00003%
Detección 2	-78.50885	-78.50883	0.00003%
Detección 3	-78.50885	-78.50884	0.00001%
Detección 4	-78.50885	-78.50888	0.00004%
Detección 5	-78.50885	-78.50879	0.00008%
Detección 6	-78.50885	-78.50888	0.00004%
Detección 7	-78.50885	-78.50875	0.00013%
Detección 8	-78.50885	-78.50884	0.00001%
Detección 9	-78.50885	-78.50883	0.00003%
Detección 10	-78.50885	-78.50883	0.00003%
		Promedio Error	0.00004%

Dado el bajo error encontrado tanto en la latitud como en la longitud de la obtención de las coordenadas en el sistema se llega a la conclusión que el módulo SIM 7600 es confiable dado que nos otorga valores muy cercanos a los otorgados por la aplicación de Google Maps. Para ilustrar la distancia del máximo error en la obtención de las coordenadas tomemos la detección 3, la cual tiene uno de los errores más altos. En la Figura 46 a) se tiene el punto de la ubicación real tomado con el GPS de un dispositivo móvil, en este caso un Iphone 11 y en la Figura b) tenemos el punto proporcionado por el sistema.

Figura 46

a) Ubicación real vs b) ubicación obtenida con mayor porcentaje de error



Si tomamos como referencia en las imágenes la ferretería la Gasca observamos que entre ambos puntos existe una diferencia notoria, uniendo las ubicaciones con la ayuda de la aplicación tenemos que la distancia es de al menos 20m, lo que provocaría serios problemas si se tratase de una notificación real de un evento de asalto. Conseguir la ubicación precisa del vehículo se vuelve un reto, dado que el receptor GPS que se utiliza que el módulo SIM 7600G-H es muy susceptible a perturbaciones por lo que se decidió colocar la antena en un lugar en el cual exista la menor cantidad de obstáculos y permita a su vez una mayor cobertura.

Capítulo VI. Conclusiones, Recomendaciones y Trabajos Futuros

Conclusiones

- El prototipo del sistema para el envío automático de alertas en situaciones de asalto en el interior de automóviles, presenta una confiabilidad mayor al 80% considerando la tasa de aciertos relacionada directamente con la detección de armas.
- El prototipo del sistema puede ser instalado fácilmente en el interior de un automóvil, permitiendo discreción y sin afectar el entorno visual tanto del pasajero como del conductor.
- El detector de objetos implementado con el algoritmo SSD-MobileNet V2 en la tarjeta Nvidia Jetson Nano, trabaja en tiempo real con una tasa de vídeo de 35 FP. El algoritmo fue entrenado con un dataset personalizado de imágenes de armas tipo pistolas o cuchillos, logrando un buen desempeño cuya precisión está en el orden del 72% en condiciones favorables de iluminación dentro del vehículo.
- La información de coordenadas GPS entregadas por el módulo SIM 7600G-H, presenta una leve desviación respecto de la ubicación real con un error relativo de 1.14%.
- La integración entre el servidor web para desplegar la información de notificación de alerta y las coordenadas de ubicación del vehículo se realizó mediante API’s para el envío de los datos. El servidor web al estar en un ambiente de producción es accesible a través del navegador web.

Recomendaciones

- Para crear de manera adecuada el dataset de entrenamiento y validación del detector de armas, es necesario disponer de cientos o miles de imágenes claras y de diferentes ángulos con el fin de lograr una mejor confiabilidad en la detección.

- La tarjeta jetson nano está disponible en diversos modelos y con distintas características de hardware incorporadas a un rango similar de precio. Por lo que antes de optar por una versión de esta se debe tomar muy en cuenta la complejidad de recursos que consume el algoritmo de AI a ejecutarse.
- Para la instalación del dispositivo, se debe tomar en cuenta la ubicación tanto de la antena GSM como de la GPS, puesto que de estas depende la conexión 4G que permite el correcto envío de la información hacia el servidor web y la obtención de las coordenadas de la ubicación del vehículo.
- La implementación del sistema al estar conectado al vehículo debemos tener las precauciones de que el auto no se quede sin batería.

Trabajos Futuros

- Utilizar un receptor GPS con mayores prestaciones que el módulo SIM 7600G, con el fin de obtener la ubicación con mayor exactitud de la ubicación del evento. Además, una de las mejoras sustanciales del sistema puede ser la incorporación de una cámara web usb infrarroja que permita realizar la detección en condiciones de iluminación baja. Sin embargo, este tipo de cámaras en el mercado ecuatoriano no se encuentran y tocaría importarlas.
- Con los datos de las coordenadas del GPS realizar un tracking del vehículo durante un intervalo de tiempo a partir de donde se detectó el evento. Con esta funcionalidad el sistema es capaz de tener mayor integración con el vehículo e incluso llegar a bloquearlo si el caso ameritara.
- Experimentar más con el modelo de detección utilizado y evaluar su potencial en el uso de otro tipo de aplicaciones. Además, es importante evaluar diferentes configuraciones de hardware con mayores prestaciones de procesamiento y de software con diversas

versiones del modelo de detección, diversos entrenamientos para explorar el comportamiento y eventualmente mejorar el sistema.

- Recrear el sistema con una empresa local para poder evaluar factores más allá de los técnicos como lo son las barreras culturales, sociales y económicas que podrían llegar a afectar o entorpecer el desarrollo de la implementación del sistema en el entorno de las industrias del país.
- Utilizar técnicas basadas en datos sintéticos para aumentar la cantidad de datos. Dado que es una la detección de objetos un limitante es la creación de un dataset representativo de objetos a clasificar, por lo tanto, los datos sintéticos podrían usarse para entrenar el modelo de manera más poderoso inclusive crear un entorno no supervisado con la capacidad de que las imágenes del dataset puedan etiquetarse de manera automática.

Acrónimos

- HTTP: Hypertext Transfer Protocol
- API: Aplication Programming Interface
- AI: Artificial Intelligence
- SMTP: Simple Mail Transfer Protocol
- CSS: Cascading Style Sheets
- HTML: HyperText Markup Language
- XML: Extensible Markup Language

Bibliografía

Actuate. (2021). *Detection, Actuate Ai Gun*. Obtenido de Actuate: https://actuate.ai/ai-security/gun-detection

Amit, Y. (2020). Computer Vision. Chicago: Springer International Publishing.

Barba, L. (2021). *Uso de técnicas deep learning para reconocimiento de objetos en áreas rurales.* Madrid: Universidad Politécnica de Madrid.

Cao, K. (2020). An Overview on Edge Computing Research. *IEEE Access, 8*, 85714-85728.

Hanan Ashraf, A. (2022). Weapons Detection for Security and Video Surveillance Using CNN and YOLO-V5s. *Computers, Materials & Continua, 70*(2), 2761-2775.

INEC. (2019). *Estadísticas de Seguridad Integral-Delitos de mayor connotación psicosocial*. Obtenido de INEC: https://insightcrime.org/wp-content/uploads/2020/01/112019_Cifras_Seguridad.pdf

Kelleher, J. (2019). *Deep Learning.* The MIT Press: Boston.

Khan, W. Z. (2019). Edge computing: A survey. *Future Generation Computer Systems, 97*, 219-235.

Lasso, L. (2021). Technological trends: a focus. *Ingeniería Solidaria, 17*(1), 1-28.

Lee, J. (2021). YOLO with adaptive frame control for real-time object detection applications. *Multimedia Tools and Applications*.

Mårten, B. (2011). *THE UTILIZATION OF GPS IN ORIENTEERING MAPPING IN URBAN HELSINKI AND RURAL KENYA.* HELSINKI: UNIVERSITY OF HELSINKI.

McNeff, J. (2002). The global positioning system. *IEEE Transactions on Microwave Theory and Techniques, 50*(3), 645-652.

Muittar, J. (2020). *Modern Web Back-End.* Oulu: Oulu University of Applied Sciences.

Nvidia. (2021). *Jetson Nano Developer Kit and Module*. Obtenido de Jetson Nano Nvidia: https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-nano/

Sánchez, J. (2020). *Evaluación de algoritmos de detección de objetos basados en deep learning para detección de incidencias en carreteras.* Valladolid.: Universidad de Valladolid. Escuela Técnica Superior de Ingenieros de Telecomunicación.

Shi, W., Dustdar, S., & Wien, T. (2016). CLOUD COVER WHY DO WE NEED EDGE COMPUTING? The Promise of Edge Computing. *Computer*, 78-81.

Stearns, P. (2019). *Simcom Wwan Setup*. Obtenido de Github: https://github.com/phillipdavidstearns/simcom_wwan-setup

Suárez, J. (2020). *Arquitectura de detección de actividades criminales basada en análisis de vídeo en tiempo real.* Valencia: Escuela Técnica Superior de Ingeniería de Telecomunicación Departamento de Comunicaciones Universitat Politècnica de València.

Tiwari, R. K. (2015). A Computer Vision based Framework for Visual Gun Detection Using Harris Interest Point Detector. *Procedia Computer Science, 54*, 703-71.

United Nations Office on Drugs and Crime. (2019). *Global Study On Homicide*. Obtenido de United Nations Office on Drugs and Crime: https://www.unodc.org/documents/data-and-analysis/gsh/Booklet1.pdf

Waveshare. (2021). *SIM7600G-H 4G / 3G / 2G / GNSS Module for Jetson Nano, LTE CAT4, Global Applicable*. Obtenido de Waveshare: https://www.waveshare.com/sim7600g-h-4g-for-jetson-nano.htm

Xie, T. (2020). *Deep Learning Methods for the Design and Understanding of Solid Materials.* Shanghai: University of Shanghai.

ZeroEyes. (2022). *ZeroEyes: Save Times. Save Live*. Obtenido de ZeroEyes: https://zeroeyes.com/

Apéndices

- Enlace GitHub repositorio del sistema

  https://github.com/JsGv1405/deteccionAsaltos