

UNIVERSIDAD CONTINENTAL
ESCUELA PROFESIONAL DE INGENIERÍA



CONSTRUCCIÓN DE SOFTWARE

TRABAJO FINAL
DESARROLLO DE UN SISTEMA INTELIGENTE CONVERSACIONAL
PARA LA GENERACIÓN DE
ITINERARIOS PERSONALIZADOS

NRC:

17200

DOCENTE:

HUGO ESPETIA HUAMANGA

ESTUDIANTES:

**Roberto Israel
Meza Oroe**

**Diego Alejandro
De La Flor
Parhuayo**

**Frank Luis Gomez
Cornejo**

**Jhojan Vidal
Seguil Osorez**

**Airton Waldir
Nina Meza**

Cusco - Perú

2025

INTRODUCCIÓN	3
Alcance del segundo Sprint	4
Funcionalidades mínimas viables:	4
METODOLOGÍA Y TECNOLOGÍAS	5
Enfoque metodológico:	5
Modelos de Inteligencia Artificial utilizados:	5
Tecnologías y herramientas empleadas:	6
Lenguajes de programación:	6
DATOS (SI APLICA)	6
Descripción del conjunto de datos utilizado	7
ARQUITECTURA PRELIMINAR	7
RESULTADOS PRELIMINARES Y DESAFÍOS	7
Resultados alcanzados hasta la fecha:	8
Principales desafíos y abordaje:	8
CONCLUSIONES PRELIMINARES	8
Aplicación Beta Funcional (Código y Demostración)	9

INTRODUCCIÓN

La seguridad ciudadana es un desafío creciente en entornos urbanos y privados. La vigilancia tradicional, basada en la supervisión manual de cámaras, presenta limitaciones importantes: es costosa, demandante de recursos humanos y su efectividad puede disminuir por la fatiga o distracción de los operadores. Además, la detección de conductas sospechosas puede retrasarse, afectando la capacidad de responder a incidentes a tiempo.

En este contexto, el presente proyecto desarrolla un Detector de Actividades Sospechosas con IA utilizando YOLOv8, un modelo avanzado de visión computacional. Al emplear una aplicación web que permite la carga de imágenes desde dispositivos locales, buscamos ofrecer una solución accesible y versátil para usuarios que no disponen de infraestructura de videovigilancia en tiempo real. Esta herramienta identifica rápidamente elementos sospechosos —como personas encapuchadas, cascos, armas u otros comportamientos inusuales— y genera alertas visuales y sonoras, contribuyendo a una vigilancia más efectiva, oportuna y económica.

El enfoque adoptado permite no sólo automatizar los procesos de detección, sino también reducir significativamente el tiempo de respuesta frente a situaciones potencialmente peligrosas, democratizando el acceso a sistemas inteligentes de seguridad para instituciones públicas y privadas.

Alcance del segundo Sprint

Funcionalidades mínimas viables:

Durante el segundo sprint, se profundizó en la integración funcional del sistema de detección, logrando un mayor nivel de automatización, estabilidad y precisión. Las funcionalidades trabajadas fueron:

- **Mejora del modelo YOLOv8** mediante reentrenamiento con nuevas imágenes y ajuste fino sobre el archivo best.pt, mejorando la precisión y reduciendo falsos positivos.
- **Optimización del backend en Flask** para procesar imágenes de mayor tamaño y mantener la velocidad de respuesta.
- **Mejora de la interfaz web:** se agregaron elementos visuales más claros (bordes de detección, etiquetas dinámicas), mejorando la experiencia del usuario.
- **Validación de múltiples clases sospechosas** en una sola imagen (detección simultánea de varios objetos).
- **Pruebas de robustez** con imágenes variadas, simulando distintos escenarios (interiores, exteriores, personas aisladas o en grupo).
- **Incorporación de mensajes dinámicos de estado**, mostrando al usuario si hay una alerta activa o si la imagen es considerada segura.

Objetivos Específicos Logrados:

- Se mejoró la precisión del modelo con un nuevo set de imágenes, alcanzando mejores resultados en detección de "arma", "casco" y "encapuchado".
- Se estabilizó el entorno Flask para evitar reinicios del servidor al procesar imágenes grandes o múltiples solicitudes.
- Se ajustó el sistema de alertas visuales, mostrando resultados más detallados y claros en la imagen procesada.
- Se validó la compatibilidad del sistema en diferentes navegadores (Chrome, Edge y Firefox) con comportamiento estable.
- Se documentó el flujo interno del procesamiento de imágenes y se organizó el código base para facilitar futuras integraciones.

METODOLOGÍA Y TECNOLOGÍAS

Enfoque metodológico:

El desarrollo del proyecto se basa en la metodología **CRISP-DM (Cross-Industry Standard Process for Data Mining)**, ampliamente utilizada en proyectos de ciencia de datos e inteligencia artificial.

Esta metodología permitió estructurar el trabajo en fases claras e iterativas:

1. **Comprensión del problema:** Se identificó la necesidad de detectar actividades sospechosas en imágenes como una herramienta de apoyo a la seguridad.
2. **Comprensión de los datos:** Se seleccionaron datasets con imágenes etiquetadas de objetos sospechosos, evaluando su calidad y formato.
3. **Preparación de los datos:** Se organizó el conjunto de datos, se etiquetaron manualmente nuevas imágenes y se convirtió todo al formato YOLO.
4. **Modelado:** Se entrenó un modelo YOLOv8 personalizado para detectar clases específicas como "arma", "capucha", "casco", etc.
5. **Evaluación:** Se realizaron pruebas controladas con imágenes nuevas, evaluando la precisión, recall y detección efectiva.
6. **Despliegue:** Se integró el modelo en una aplicación web accesible mediante un backend en Flask.

Modelos de Inteligencia Artificial utilizados:

El sistema utiliza el modelo de detección de objetos YOLOv8 (You Only Look Once versión 8), desarrollado por Ultralytics, que permite procesar imágenes rápidamente con alta precisión. Las principales características aplicadas fueron:

- Modelo personalizado: Entrenado con imágenes específicas del dominio de vigilancia (más de 50 épocas).
- Transfer learning: Se partió de un modelo base (yolov8n.pt) y se reentrenó con un dataset personalizado.
- Clases detectadas: Se entrenó para detectar las clases "suspicious", "helmet".

Tecnologías y herramientas empleadas:

Lenguajes de programación:

- Python 3.11: Lógica del backend, entrenamiento del modelo y procesamiento de imágenes.
- HTML5, CSS, JavaScript: Desarrollo de la interfaz web.

Frameworks y librerías:

- Flask: Framework ligero de Python para desarrollo del servidor web.
- Ultralytics YOLOv8: Entrenamiento, predicción y manejo del modelo.
- OpenCV: Para lectura y manipulación de imágenes.
- Roboflow: Para anotación y exportación del dataset en formato YOLO.

Entorno de entrenamiento:

- Google Colab: Entrenamiento del modelo en GPU de forma gratuita.

Control de versiones y despliegue:

- Git y GitHub: Gestión del código fuente y versiones del proyecto.
- Live Server (VS Code): Para pruebas locales de la interfaz.

(Futuro) Base de datos:

- Se proyecta utilizar SQLite o Firebase en fases posteriores para almacenar alertas detectadas y gestionar historial de imágenes procesadas.

DATOS

El conjunto de datos utilizado fue descargado desde la plataforma **Roboflow**, Se seleccionaron datasets que incluían clases relevantes para el contexto del proyecto, como:

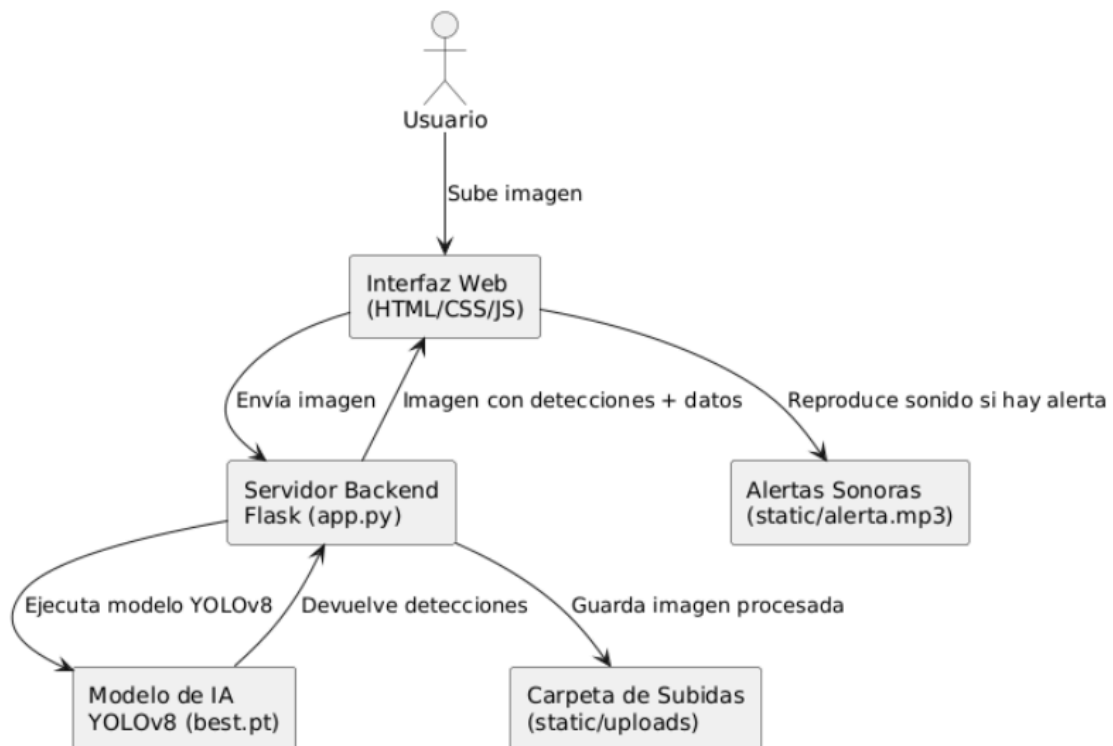
- **“suspicious”** (persona con actitud o vestimenta sospechosa)
- **“helmet”** (casco)
- **“hood”** (capucha)
- **“gun”** (arma)

En algunos casos se complementó con imágenes adicionales subidas manualmente al entorno de Roboflow para aumentar la variedad y precisión del modelo entrenado.

Descripción del conjunto de datos utilizado

- **Cantidad total de imágenes:** Aproximadamente **500 imágenes**
- **Formato de las imágenes:** JPG/PNG
- **Formato de anotación:** YOLOv8 (bounding boxes en archivos .txt)
- **Estructura de carpetas:** Separación en train, val y test
- **Variables:**
 - Coordenadas de detección (x_center, y_center, width, height)
 - Clases (etiquetas numéricas mapeadas a: suspicious, gun, helmet, hood)

ARQUITECTURA PRELIMINAR

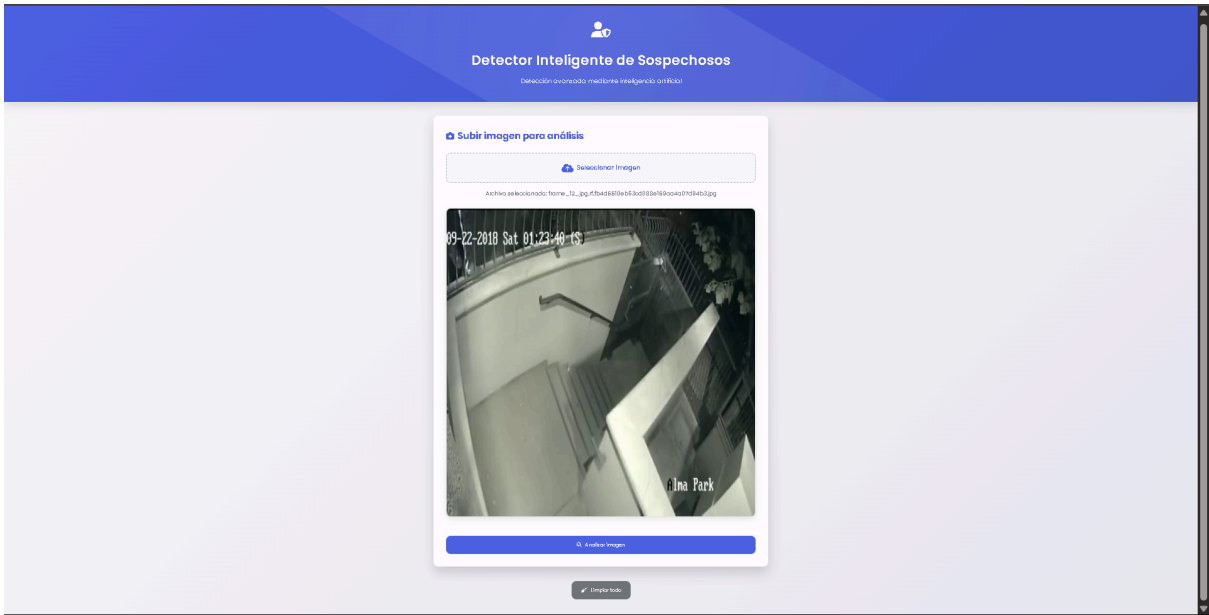


El proceso inicia con el usuario cargando una imagen a través de una interfaz web. Esa imagen es enviada al servidor backend (desarrollado en Flask), donde el modelo de inteligencia artificial YOLOv8 analiza su contenido. Si se detectan objetos o comportamientos sospechosos (como armas, capuchas o cascos), el sistema genera una respuesta con alertas visuales y sonoras que se devuelven al usuario en la interfaz.

Interfaz web:



Subida de imagen:



Resultados:

127.0.0.1:3000


Resultados del análisis

2

Objetos detectados

normal 0.66.57

09-12-2018 Sat 01:23:46 (S)



Alma Park

Elementos detectados

normal x1

suspechoso x1

¡Alerta de seguridad!

Se ha detectado a: sospechoso

RESULTADOS PRELIMINARES Y DESAFÍOS

Resultados alcanzados hasta la fecha:

Durante los dos primeros sprints del proyecto se han logrado avances importantes en cuanto a funcionalidad del sistema y desempeño del modelo:

- Se entrenó un modelo **YOLOv8** con un dataset específico que incluye clases como "suspicious", "gun", "helmet" y "hood", alcanzando **más de 50 épocas** con resultados satisfactorios.
- El modelo fue integrado exitosamente en una **aplicación web desarrollada con Flask**, permitiendo a los usuarios subir imágenes desde su dispositivo y obtener detecciones automáticas.
- Se implementaron **alertas visuales** (marcado de objetos detectados con bounding boxes y etiquetas).
- El sistema fue probado con imágenes externas (no pertenecientes al dataset de entrenamiento), logrando una detección efectiva en varios escenarios.

Principales desafíos y abordaje:

Desafío	Descripción	Solución propuesta
Falsos positivos	El modelo detecta elementos como “casco” o “arma” en imágenes que no los contienen realmente.	Reentrenamiento del modelo con más imágenes reales, revisión de anotaciones y refinamiento de clases.
Precisión limitada en fondos complejos	La detección pierde exactitud en imágenes con múltiples personas, sombras o fondos desordenados.	Aumentar el tamaño del dataset con variedad de escenarios y condiciones. Aplicar técnicas de aumento de datos.
Interfaz no responsive aún	La visualización del sistema en móviles o pantallas pequeñas no está optimizada.	Rediseñar la interfaz con CSS adaptable en el Sprint 3.
Procesamiento de imágenes grandes	El sistema demora en responder si se suben imágenes de gran resolución.	Agregar redimensionamiento automático antes del análisis.

CONCLUSIONES PRELIMINARES

En esta primera etapa del proyecto se han cumplido los principales objetivos técnicos: se entrenó un modelo funcional con YOLOv8 y se integró exitosamente en una aplicación web que permite a los usuarios subir imágenes y detectar elementos sospechosos como armas, capuchas o cascos.

El sistema responde de forma estable, emitiendo alertas visuales y sonoras, y ha sido probado con éxito en distintos escenarios. Si bien se identificaron algunos desafíos, como falsos positivos o limitaciones en la precisión, estos serán abordados en los próximos sprints mediante la mejora del dataset, la optimización del modelo y el rediseño de la interfaz.

Se confirma la viabilidad de aplicar inteligencia artificial en tareas de seguridad a través de una solución accesible, portable y de bajo costo. El desarrollo futuro se centrará en robustecer el sistema, documentarlo correctamente y dejarlo preparado para una posible implementación en entornos reales.

Aplicación Beta Funcional (Código y Demostración)

Entrenamiento con Google Colab:

<https://colab.research.google.com/drive/1H36SemrZrD9OY-uw0lY1jJIRzfcxiPjZ?usp=sharing>

GitHub: <https://github.com/DiegoDeLaFlor/Proyecto-Detecci-n-de-Actividades-Sospechosas.git>

GitLab: [Diego De La Flor / Proyecto-Detecci-n-de-Actividades-Sospechosas · GitLab](#)