UNIVERSIDAD NACIONAL SAN ANTONIO ABAD DEL CUSCO

Facultad de Ingeniería Eléctrica, Electrónica, Informática y Mecánica

Escuela Profesional de Ingeniería Informática y de Sistemas

Desarrollo de una Aplicación de Detección de Cascos con YOLOv8 y Flask

January 17, 2024

Curso: Deep Learning

Docente del curso:

Carlos Fernando Montoya Cubas

Presentado por:

Carmen Guadalupe Blanco Mozo (193027)

Mary Carmen Flores Castro ()

Anthony Mayron Lopez Oquendo ()

Harol Helbert Merma Quispe (130741)

Roy Marvin Mamani Tairo ()

Alex Harvey Pfoccori Quispe()

Abstract

This project introduces an advanced security system for warehouses, focusing on real-time helmet detection using YOLOv8 object detection technology. The system incorporates features such as automatic alarm activation and instant notification to security personnel. The YOLOv8 model is adapted to recognize individuals with and without helmets, ensuring precision and efficiency. The installation process involves meticulous steps, including Python and PyCharm setup. YOLOv8 installation and adaptation for helmet detection in images and webcams are detailed, followed by thorough testing. The web application development phase includes the creation of HTML and CSS pages, with a focus on video upload and detection functionalities. The abstract highlights the project's key components, theoretical framework, and development processes.

1 Introduction

El proyecto tiene como objetivo principal implementar un sistema de seguridad avanzado para un almacén, centrándose en la detección en tiempo real de personas que ingresan sin casco. Para lograr esto, se empleó la tecnología de detección de objetos YOLOv8, que se basa en una red neuronal convolucional profunda. A lo largo del desarrollo, nos enfocamos en garantizar la precisión y eficiencia del sistema, así como en integrar funciones adicionales, como la activación de una alarma y la notificación al personal de seguridad.

2 Marco Teorico

2.1 Principio de Funcionamiento

YOLOv8 es un modelo de detección de objetos en tiempo real que utiliza una única red neuronal para predecir bounding boxes y clasificaciones de objetos en una imagen completa. A diferencia de métodos más lentos que operan en múltiples etapas, YOLOv8 realiza estas predicciones de manera eficiente y precisa.

2.2 Aplicación a Cascos de Protección

El modelo YOLOv8 ha sido adaptado para reconocer específicamente personas con y sin casco en un entorno de almacén, permitiendo la identificación instantánea de individuos que no cumplen con los requisitos de seguridad.

2.3 Activación Automática

La detección en tiempo real de personas sin casco desencadena automáticamente una alarma sonora en el área del almacén. Este enfoque proactivo permite una respuesta inmediata ante posibles riesgos de seguridad.

2.4 Notificación al Personal de Seguridad

El sistema va más allá de la alarma local, enviando notificaciones instantáneas al personal de seguridad a través de correo electrónico o mensajes de WhatsApp. Esto asegura una respuesta rápida y eficaz ante situaciones de incumplimiento.

2.5 Uso de Cascos

El proyecto se alinea con los estándares de seguridad industrial que enfatizan la importancia del uso de equipos de protección personal, como los cascos, para prevenir lesiones en entornos laborales. Automatización para el Cumplimiento: La automatización de la detección y alarma contribuye al cumplimiento de normativas de seguridad laboral, aliviando la carga de supervisión manual y reduciendo el riesgo de accidentes. Evaluación y Mejora Continua:

2.6 Pruebas y Validación

El sistema se someterá a pruebas exhaustivas para evaluar su rendimiento en diferentes condiciones de iluminación y escenarios de almacén. Ciclo de Mejora Continua: Se establecerá un ciclo de mejora continua basado en la retroalimentación del desempeño del sistema, permitiendo ajustes y actualizaciones para abordar posibles limitaciones y mejorar la eficacia a lo largo del tiempo.

3 Instalación de Python y PyCharm

La instalación de Python y PyCharm se realizó meticulosamente. La elección de la versión 3.8 de Python se basó en su compatibilidad con las bibliotecas clave, como YOLOv8. La creación de una carpeta específica para el proyecto en PyCharm proporcionó un entorno de desarrollo organizado y dedicado.
Luego se procedió a la creación de un entorno virtual en PyCharm para aislar las dependencias del proyecto. La instalación de paquetes, especialmente "ultralytics," se llevó a cabo de manera cuidadosa para garantizar una configuración correcta y sin conflictos. Se destaca la importancia de instalar el paquete "ultralytics" para la detección de objetos mediante YOLOv8

4 Data Augmentation

Para incrementar el conjunto de datos, se empleó la plataforma de Roboflow debido a su conjunto integral de herramientas necesarias. El proceso de aumento de datos se llevó a cabo de la siguiente manera:

1. Se realizó un "fork" al conjunto de datos, lo que permitió crear una copia independiente. Una vez completada esta acción, se habilitaron las modificaciones directas en el conjunto de datos, lo que incluyó la capacidad de generar nuevas divisiones para entrenamiento, prueba y validación, así como realizar el preprocesamiento y la ampliación de datos.

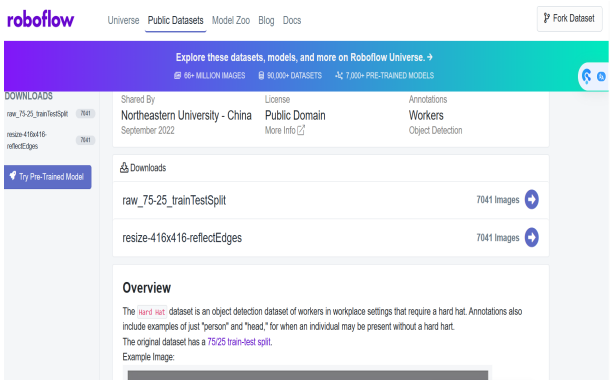

Figure 1: Fork del Dataset

2. Se describen detalladamente las modificaciones realizadas durante el aumento de datos, destacando los cambios específicos implementados en el proceso.

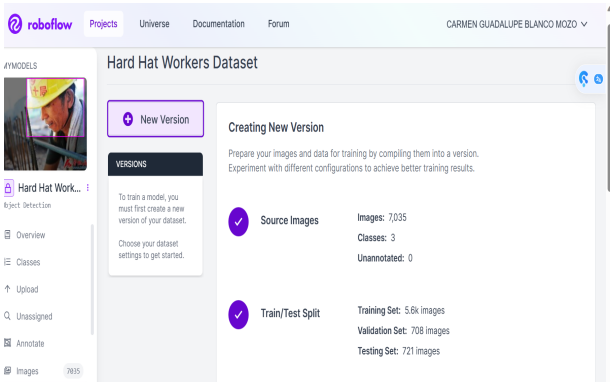

Figure 2: Preprocesamiento y partición

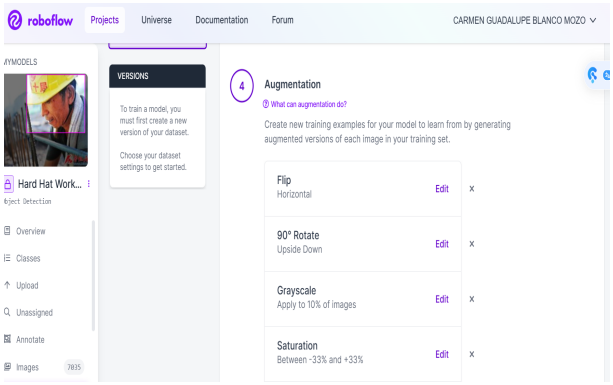

Figure 3: Especificación de las caracterististicas de data augmentation

3. Se generaron nuevos datos mediante el procedimiento de aumento de datos, destacando la creación de imágenes adicionales.

4. Se incrementó el número total de imágenes en el conjunto de datos, contribuyendo así a una mayor diversidad y enriquecimiento del conjunto de datos original.

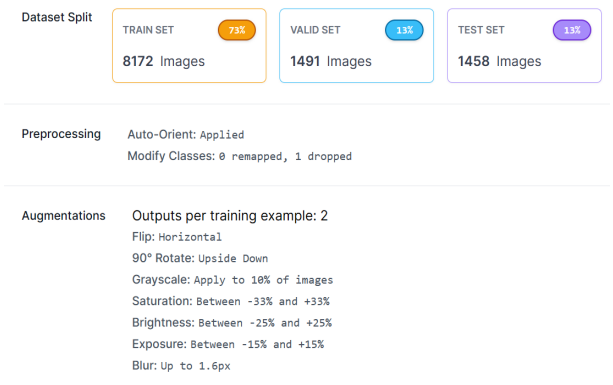

Figure 4: Dataset aumentado

5 Instalación de YOLOv8 y Detección de Cascos en Imágenes

La instalación de YOLOv8 se realizó mediante el comando "pip install ultralytics". Se adaptaron los scripts para enfocarse específicamente en la detección de cascos en lugar de objetos genéricos. Se ajustaron los parámetros del modelo para optimizar la identificación precisa de cascos en imágenes.

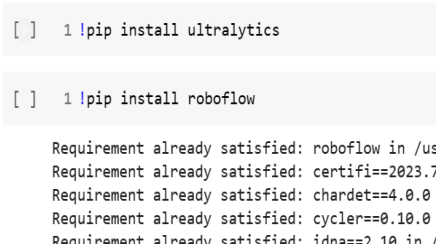

Figure 5: Instalación YOLO8

Figure 6: Configuración para la detección de cascos

5.1 Pruebas en Imágenes y Webcamss

Las pruebas se llevaron a cabo de manera extensiva en nuevas imágenes utilizando el dataset proporcionado en clase y se aumentaron datos adicionales, y seleccionamos el modelo "YOLOv8x" para obtener resultados más precisos en la detección de cascos. La adaptación del modelo a la detección en tiempo real en una cámara web implicó la configuración de un nuevo directorio y la validación de resultados.

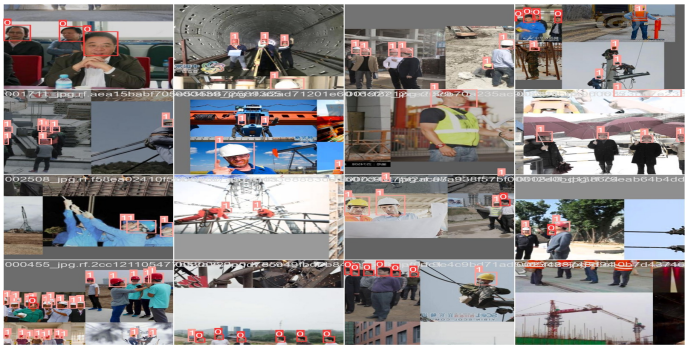

Figure 7: Prueba en imágenes

6 Métricas

Las métricas obtenidas después del proceso de entrenamiento son altamente positivas. El modelo fue entrenado a lo largo de 20 épocas, revelando los siguientes resultados:

```
Model summary: 218 layers, 25840918 parameters, 0 gradients, 78.7 GFLOPs
               Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 45/45 [00:27<00:00,  1.62it/s]
                 all       1413       5252      0.932      0.928      0.966      0.672
                head       1413       1339      0.901      0.939      0.957      0.664
              helmet       1413       3913      0.963      0.917      0.974      0.681
Speed: 0.2ms pre-process, 7.1ms inference, 0.0ms loss, 1.8ms post-process per image
Saving runs/detect/train2/predictions.json...
```

Figure 8: Metricas del entrenamiento

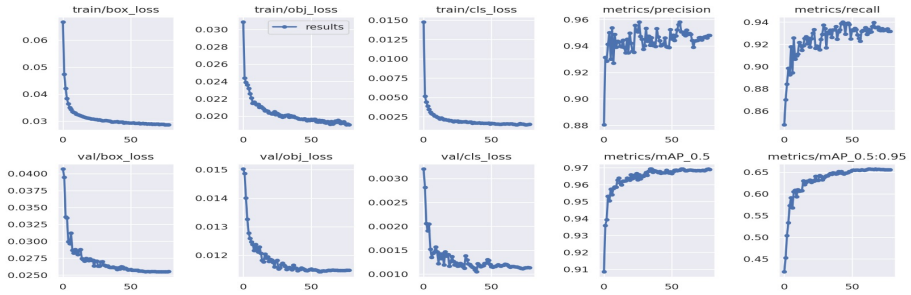

Figure 9: Metricas del entrenamiento - gráfica

En este contexto, se observa que las métricas de Box Precision, Recall y Max superan el umbral de 0.90, lo que sugiere que el modelo exhibe un desempeño robusto y preciso en la tarea asignada.

7 Creación de Páginas HTML y CSS:

Durante la fase de desarrollo de la aplicación web, se dedicó tiempo significativo a la creación de páginas HTML y sus estilos correspondientes en CSS. Se diseñaron tres páginas principales: "Inicio", "Video" y "Transmisión de Cámara Web en Vivo". Cada página fue cuidadosamente estructurada para proporcionar una experiencia de usuario intuitiva y atractiva. El archivo principal de Flask, "app.py", se encarga de manejar la lógica subyacente de estas páginas HTML.

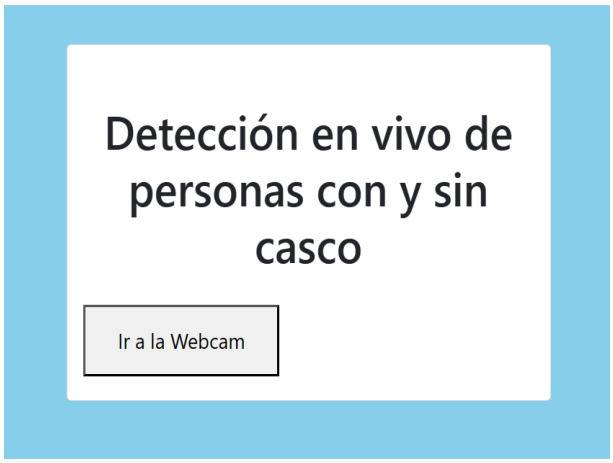

Figure 10: Prueba de interfaz - inicio

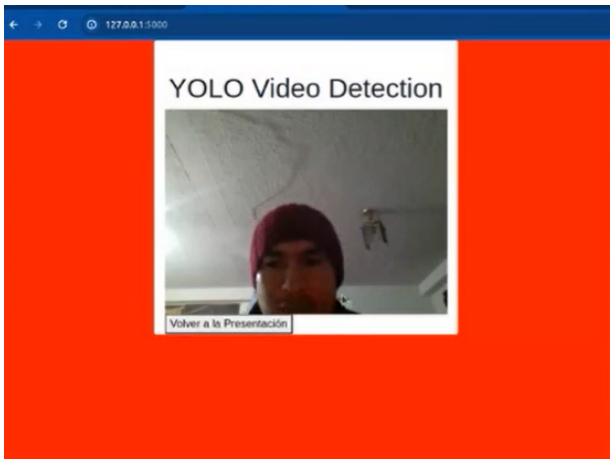

Figure 11: Prueba de interfaz -video

7.1 Detección en Tiempo Real en Webcams

El desarrollo de un script denominado "YOLOv8webcam.py" permitió la ejecución de YOLOv8 en una cámara web en tiempo real. Se prestaron especial atención a los parámetros de la cámara y se realizaron pruebas exhaustivas para asegurar la funcionalidad robusta de la detección en tiempo real. Para mejorar la presentación visual, se implementó la conversión de resultados de tensores a enteros y la creación de rectángulos alrededor de los cascos detectados. Esto facilitó la interpretación de los resultados y mejoró la claridad de la información presentada.

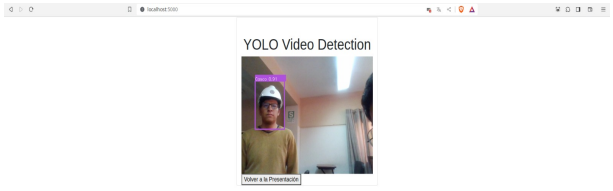

Figure 12: Prueba en cámara 1

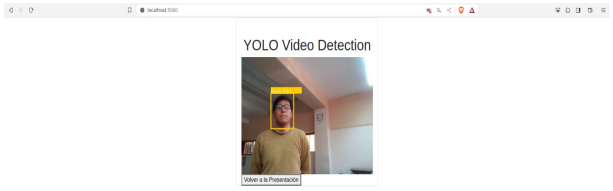

Figure 13: Prueba en cámara 2

Figure 14: Prueba en cámara 3

7.2 Función de Detección y Manejo de Sesiones

La función de detección, centralizada en el archivo "YOLODash.py", fue diseñada para proporcionar un mecanismo eficiente y preciso. Se destacó la importancia del manejo de sesiones para permitir la detección en nuevos videos, evitando cualquier interferencia de videos procesados anteriormente. Esta optimización garantiza un rendimiento constante y preciso, incluso en casos donde múltiples detecciones son realizadas en sesiones consecutivas.

La interfaz que implementamos tiene la propiedad de mandar un correo o mensaje de whatsapp al personal de seguridad con una foto de la persona sin casco y un informe de ocurrencia.

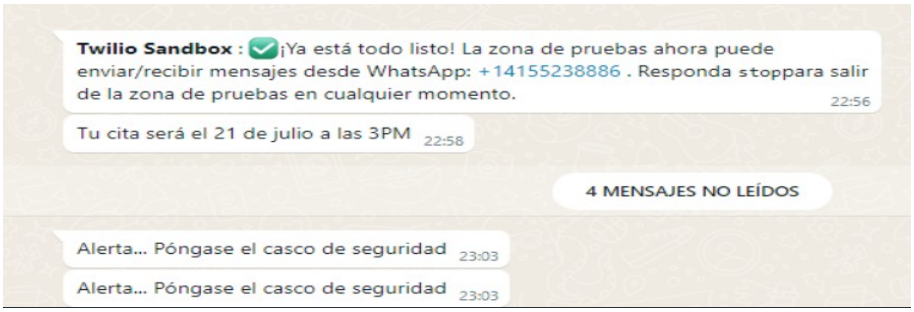

Figure 15: Mensaje de whatsapp enviado

8 Conclusión

Este proyecto, basado en los principios y pasos del Curso YOLOv8, ha logrado implementar con éxito un sistema de detección de cascos en tiempo real en un entorno de almacén. La combinación de YOLOv8 y Flask ha demostrado ser efectiva para desarrollar aplicaciones web de detección de objetos, adaptándola a nuestras necesidades específicas de seguridad laboral. La detección en tiempo real de cascos proporciona un componente valioso para mejorar la seguridad en el almacén, garantizando que todos los trabajadores estén debidamente equipados.

9 Referencias

Detección de cascos de seguridad en tiempo real mediante modelos de Deep Learning, Belén, A., Arias, A. (n.d.). UNIVERSIDAD POLITÉCNICA DE CARTAGENA *Escuela Técnica Superior de Ingeniería Industrial*. Retrieved January 16, 2024 sample