UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO
Facultad de Ingeniería Civil, Sistemas y de Arquitectura
Escuela Profesional de Ingeniería de Sistemas

TESIS

Aplicación Web para la Implementación de un Modelo Basado en Redes Neuronales Convolutivas (Yolo) para la Clasificación y Conteo de Vehículos

Presentada para optar por el Título Profesional de:

INGENIERO DE SISTEMAS

Autor:

Bach. Jara Huaman, Carlos Daniel

Asesor:

Dr. Ing. Villegas Cubas, Juan Elías

Lambayeque - Perú
Marzo 2026

UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO
Facultad de Ingeniería Civil, Sistemas y de Arquitectura

Escuela Profesional de Ingeniería de Sistemas

TESIS

Aplicación Web para la Implementación de un Modelo Basado en Redes Neuronales Convolutivas (Yolo) para la Clasificación y Conteo de Vehículos

Presentada para optar por el Título Profesional de Ingeniero de Sistemas:

Aprobada por:

Dr. Ing. Luis Alberto Luis Dávila
(Presidente)

Msc. Ing. Omar Wilton Saavedra Salazar
(Secretario)

(Vocal)
Msc. Ing. Oscar Efraín Capuñay Ayala

Lambayeque - Perú

Marzo 2026

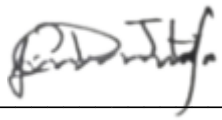
UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO
Facultad de Ingeniería Civil, Sistemas y de Arquitectura
Escuela Profesional de Ingeniería de Sistemas

TESIS

Aplicación Web para la Implementación de un Modelo Basado en Redes Neuronales Convolutivas (Yolo) para la Clasificación y Conteo de Vehículos

Presentada para optar por el Título Profesional de Ingeniero de Sistemas:

Autores:



Bach. Jara Huaman Carlos Daniel

(Autor)

Dr. Ing. Juan Elías Villegas Cubas
(Asesor)

Lambayeque - Perú

Marzo 2026

UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO

FACULTAD DE INGENIERÍA CIVIL DE SISTEMAS Y DE ARQUITECTURA

UNIDAD DE INVESTIGACIÓN

ACTA DE SUSTENTACIÓN N°637-2026-UI-FICSA

Siendo las 12:00 m horas del día 2 de marzo del 2026, se reunieron de manera presencial los miembros de jurado de la tesis titulada: : **"APLICACIÓN WEB PARA LA IMPLEMENTACIÓN DE UN MODELO BASADO EN REDES NEURONALES CONVOLUTIVAS (YOLO) PARA LA CLASIFICACIÓN Y CONTEO DE VEHÍCULOS" con código N° IS_V_2024_026,** designado por **Resolución Decanal N° 61-2026-UNPRG-FICSA** con la finalidad de Evaluar y Calificar la sustentación de la tesis antes mencionada, conformado por los siguientes docentes:

DR. ING. LUIS ALBERTO DAVILA HURTADO	**PRESIDENTE**
MSC. ING. OMAR WILTON SALAZAR SAAVEDRA	**SECRETARIO**
MSC. ING. OSCAR EFRAIN CAPUÑAY UCEDA	**VOCAL**

Asesorado por DR. ING. JUAN ELIAS VILLEGAS CUBAS.

El acto de sustentación fue autorizado por **OFICIO VIRTUAL N° 43-2026-UIFICSA**, la tesis fue presentada y sustentada por el Bachiller: **CARLOS DANIEL JARA HUAMAN,** tuvo una duración de 80 minutos. Después de la sustentación, y absueltas las preguntas y observaciones de los miembros del jurado; se procedió a la calificación respectiva:

	NUMERO	LETRAS	CALIFICATIVO
CARLOS DANIEL JARA HUAMAN	18	DIECIOCHO	MUY BUENO

Por lo que queda **APTO** para obtener el Titulo Profesional de **INGENIERO DE SISTEMAS** de acuerdo con la Ley Universitaria 30220 y la normatividad vigente de la Facultad de Ingeniería Civil De Sistemas y de Arquitectura de la Universidad Nacional Pedro Ruiz Gallo.

Siendo las 13:20 Se dio por concluido el presente acto académico, dándose conformidad al presente acto, con la firma de los miembros del jurado.

DR. ING. LUIS ALBERTO DAVILA HURTADO
PRESIDENTE

MSC. ING. OMAR WILTON SALAZAR SAAVEDRA
SECRETARIO

MSC. ING. OSCAR EFRAIN CAPUÑAY UCEDA
VOCAL

DR. ING. JUAN ELIAS VILLEGAS CUBAS
ASESOR

JHFF/bcm

CONSTANCIA DE VERIFICACIÓN DE ORIGINALIDAD DE TESIS

Yo, Dr. Ing. Juan Elias Villegas Cubas**, asesor de Tesis del bachiller**:

Jara Huaman Carlos Daniel

TITULADA:

"Aplicación Web para la Implementación de un Modelo Basado en Redes Neuronales Convolutivas (Yolo) para la Clasificación y Conteo de Vehículos"

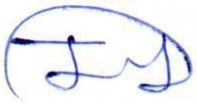
Luego de la revisión exhaustiva del documento constato que la misma tiene un índice de similitud de 13% verificable en el reporte de similitud del programa TURNITIN.

El suscrito analizó dicho reporte y concluyó que cada una de las coincidencias detectadas NO CONSTITUYEN PLAGIO. A mi leal saber y entender la Tesis cumple con todas las normas para el uso de citas y referencias establecidas por la Universidad Nacional Pedro Ruiz Gallo.

Se expide la presente según lo dispuesto en la Resolución Nº 659-2020-R, de fecha 8 de setiembre de 2020 formativa para la obtención de Grados y Títulos de la UNPRG:

Lambayeque, 06 de febrero del 2026

ATENTAMENTE,



Dr. Ing. Juan Elias Villegas Cubas
DNI. 80103991

Se adjunta:
Recibo digital de Turnitin
Revisión de informe en Turnitin

Aplicacion Web para la Clasificación y Conteo de Vehículos

INFORME DE ORIGINALIDAD

13%	12%	3%	5%
INDICE DE SIMILITUD	FUENTES DE INTERNET	PUBLICACIONES	TRABAJOS DEL ESTUDIANTE

FUENTES PRIMARIAS

1	docs.ultralytics.com Fuente de Internet	3%
2	hdl.handle.net Fuente de Internet	1%
3	repositorio.unprg.edu.pe Fuente de Internet	1%
4	Submitted to udep Trabajo del estudiante	<1%
5	oa.upm.es Fuente de Internet	<1%
6	Submitted to Instituto Superior de Artes, Ciencias y Comunicación IACC Trabajo del estudiante	<1%
7	iieta.org Fuente de Internet	<1%
8	docplayer.es Fuente de Internet	<1%
9	repositorio.uleam.edu.ec Fuente de Internet	<1%
10	laccei.org Fuente de Internet	<1%

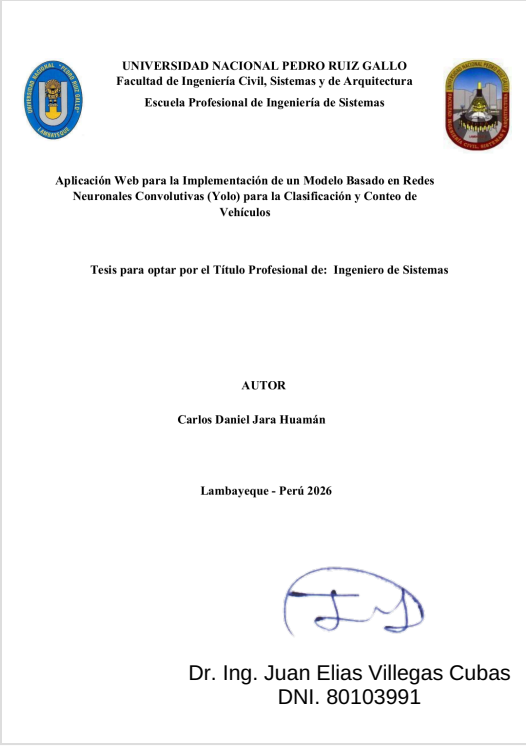
Dr. Ing. Juan Elias Villegas Cubas
DNI. 80103991

Recibo digital

Este recibo confirma quesu trabajo ha sido recibido por Turnitin. A continuación podrá ver la información del recibo con respecto a su entrega.

La primera página de tus entregas se muestra abajo.

Autor de la entrega:	Carlos Daniel Jara Huaman
Título del ejercicio:	Quick Submit
Título de la entrega:	Aplicacion Web para la Clasificación y Conteo de Vehículos
Nombre del archivo:	Tesis_Carlos_Jara.pdf
Tamaño del archivo:	2.68M
Total páginas:	67
Total de palabras:	11,576
Total de caracteres:	68,373
Fecha de entrega:	06-feb-2026 02:05p. m. (UTC-0500)
Identificador de la entrega:	2872713549



UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO
Facultad de Ingeniería Civil, Sistemas y de Arquitectura
Escuela Profesional de Ingeniería de Sistemas

Aplicación Web para la Implementación de un Modelo Basado en Redes Neuronales Convolutivas (Yolo) para la Clasificación y Conteo de Vehículos

Tesis para optar por el Título Profesional de: Ingeniero de Sistemas

AUTOR

Carlos Daniel Jara Huamán

Lambayeque - Perú 2026

Dr. Ing. Juan Elias Villegas Cubas
DNI. 80103991

DEDICATORIAS

A Dios por permanecer en nuestras mentes y nuestros corazones, a mis Madre Raquel por haber el ejemplo y amor que siempre necesitaba en los momentos difíciles, a mi Abuelo Gilberto y abuela Armandina por haberme dado todo ese soporte desinteresado que inspira y motiva, a mis hermanos Steven y Kristel considerados por mis grandes talentos y por supuesto buenas personas que muchos desearían disponer del soporte y motivación que ellos otorgan, tanto como mi papá Elar, mis tíos Walter y Amparo, por bajo las sombras también depositar confianza en mi.

A mis amigos Jair, Fabián, Jhonatan, Fabián, Héctor Miguel, William, Rubén, Julio, Carlos Yahveh, Brayan Scott, Héctor Thomas que en aquellos momentos de última instancia estuvieron ahí otorgando ayuda creativa y motivacional, tanto como para el presente como para futuro, y que cada grano de apoyo siempre será recodado y dejar sustento que esta tesis también hay algo de cada uno de su pragmatismo y determinación.

El tesista

AGRADECIMIENTOS.

A Dios por ser el más grande amigo y guía de todos, desde luego que a nuestras familias soporte emocionales a lo largo de todo este camino, las personas a las que se unieron esto no hubiese posible sin la ayuda de amigos, las personas que estuvieron ahí dando el empuje, gente que tiene increíbles historias detrás y dignas de seguir, a nuestro asesor que fue el puente para abrirnos interés sobre lo que es la investigación y poder desempeñar de manera adecuada y organizada en este rumbo, gracias a todos.

Apreciamos el desempeño y la enseñanza de nuestros grandes mentores investigadores por compartir sus estrategias y conocimientos, esa guía y dedicación constante basado en la excelencia ética que dejan huella en cada pisada que van dejando dicha que fue fundamental para la realización de nuestra tesis.

Finalmente, declaramos nuestro grato agradecimiento a nuestras amistades que, con su incondicional apoyo moral, y puntuales golpes de ánimo, nos inspiraron a continuar adelante sin freno. Agradecemos desde lo más profundo su confianza en nosotros y esas palabras de aliento que marcan un antes y después en nuestro andar.

RESUMEN

Esta investigación tiene como matiz principal la implementación de una aplicación web basada en un modelo de aprendizaje profundo (Deep Learning) utilizando Redes Neuronales Convolutivas (YOLO) tanto como clasificar y contar clases de vehículos. Este enfoque busca automatizar el proceso de recopilación de datos vehiculares, superando las limitaciones de los métodos manuales convencionales que son laboriosos, propensos a errores humanos y poco eficientes para el análisis a gran escala.

En la actualidad, la contabilización vehicular se realiza mayormente mediante procedimientos manuales que requieren una gran inversión de tiempo y no permiten una respuesta en tiempo real. Esta limitación resulta problemática en entornos urbanos congestionados, donde la gestión eficiente del tráfico es fundamental para optimizar la señalización, la iluminación y otras infraestructuras viales. Por tanto, surge la necesidad de desarrollar un sistema automatizado que brinde precisión y rapidez en la obtención de datos.

Para abordar este problema, nos centramos en el desarrollo de una aplicación web acompañado de un módulo backend basado en un modelo preentrenado YOLO (You Only Look Once) para la detección, clasificación y conteo de vehículos según las categorías establecidas por el Ministerio de Transportes y Comunicaciones. El desarrollo del sistema se basa en cinco etapas: La creación de un conjunto de imágenes categorizadas, el procesamiento de estas imágenes, el entrenamiento de modelo custom, la evaluación del desempeño, y el desarrollo del aplicativo web con Python.

A partir de las pruebas realizadas en distintas carreteras del país, se concluye que el desempeño del sistema depende en gran medida de factores como el entrenamiento del modelo y la correcta configuración en campo. Para lograr una detección más estable en el tiempo, es recomendable entrenar con conjuntos de datos que incluyan condiciones reales de ruido visual, así como optimizar la ubicación de la línea de conteo según la posición de la cámara. En conjunto, los resultados obtenidos en distintos escenarios muestran niveles de precisión del conteo consistentemente altos, lo que respalda la viabilidad de la solución en aplicaciones reales.

Palabras clave: YOLO, aprendizaje profundo, clasificación de vehículos, conteo de vehículos, aplicación web, gestión de tráfico, redes neuronales convolucionales.

ABSTRACT

The primary focus of this research is the implementation of a web application based on a Deep Learning model using Convolutional Neural Networks (YOLO) to both classify and count various types of vehicles. This approach aims to automate the vehicular data collection process, overcoming the limitations of conventional manual methods which are labor-intensive, prone to human error, and inefficient for large-scale analysis.

Currently, vehicle counting is mostly conducted through manual procedures that require a significant time investment and do not allow for real-time responses. This limitation is particularly problematic in congested urban environments, where efficient traffic management is essential for optimizing signaling, lighting, and other road infrastructure. Consequently, there is a pressing need to develop an automated system that provides both precision and speed in data acquisition.

To address this problem, we focus on the development of a web application paired with a backend module based on a pre-trained YOLO (You Only Look Once) model for the detection, classification, and counting of vehicles according to the categories established by the Ministry of Transport and Communications. The system development is structured into five stages: the creation of a categorized image dataset, image processing, custom model training, performance evaluation, and the development of the web application using Python.

Based on tests conducted on various highways across the country, it is concluded that the system's performance largely depends on factors such as model training and proper field configuration. To achieve more stable detection over time, it is recommended to train the model with datasets that include real-world visual noise conditions, as well as to optimize the location of the counting line based on the camera's position. Overall, the results obtained across different scenarios demonstrate consistently high levels of counting accuracy, supporting the viability of this solution for real-world applications.

Keywords:

YOLO, Deep Learning, vehicle classification, vehicle counting, web application, traffic management, convolutional neural networks.

INDICE GENERAL

DEDICATORIAS **4**

RESUMEN **6**

ABSTRACT **8**

INTRODUCCIÓN **12**

CAPÍTULO 1: DISEÑO TEÓRICO **14**

1.1 Antecedentes de la investigación 14

1.2 Bases teóricas 20

1.2.1 Categorización vehicular utilizada por el Ministerio de transportes y Comunicaciones 20

1.2.2 Deep Learning y Visión Artificial 23

1.2.3 Clasificación y Conteo Vehicular 23

1.2.4 Herramientas Tecnológicas 24

1.3. Categorización de las variables 24

1.3.1. Variable independiente 24

1.3.2 Variable dependiente 24

1.3.3. Definiciones conceptuales 25

1.3.4. Definiciones operativas 33

CAPÍTULO 2: DISEÑO METODOLÓGICO 34

2.1. Tipificación de la investigación 34

2.1.1. Justificación 34

2.2 Población y muestra 34

2.3 Metodología y recolección de datos 35

2.4. Aspectos Éticos de la Investigación 36

CAPÍTULO 3: DESARROLLO DEL PROYECTO 37

3.1 Adquisición de imágenes y videos de prueba 37

3.1.1 Metodología de Extracción de imágenes: 37

3.2. Preprocesamiento de imágenes 41
3.2.1. Preparar el entorno de trabajo. 41
3.2.2. Refinamiento de los datasets: 41
3.2.3 Equilibrado de clases y creación de labels: 41
3.2.4 Aumentación de imágenes 42
3.3. Entrenamiento 43
3.3.1 Entrenamiento 1: 44
3.3.2 Entrenamiento 2: 46
3.4. Evaluación del modelo 48
3.4.2 Configuración del entorno 50
3.5. Flujo de trabajo del proyecto 52
3.5.1 Herramientas de desarrollo en PyCharm 52
3.5.2 Pruebas en entornos de trabajo 52
3.5.3 Validación y ejecución 53
CAPÍTULO 4: EVALUACIÓN DE LOS RESULTADOS 54
4.1 Escenario 1: Av. Brasil 54
4.2 Escenario 2: Av. Salaverry (Puente) 55
4.3 Escenario 3: Carretera Interestatal 56
4.4 Escenario 4: Av. Balta con Leguía (Escenario con Ruido) 57
CAPÍTULO 5: DISCUSIÓN DE RESULTADOS **59**
CAPÍTULO 6: CONCLUSIONES **61**
CAPITULO 7: RECOMENDACIONES **62**
8. REFERENCIAS **63**
9. APÉNDICES **66**

INDICE DE TABLAS

Tabla 1: Operacionalización de variables……...25

Tabla 2: Cantidad de datos por Categorías…….38

Tabla 3: Cantidad de datos aumentados por Categoría……..……43

Tabla 4: Comparación modelos de Deep Learning……..43

Tabla 5: Métricas de entrenamiento 1……..45

Tabla 6: Métricas de entrenamiento 2…….47

Tabla 7: Cuadro comparativo entrenamiento 1 y entrenamiento 2…….48

Tabla 8: Cuadro Resumen Evaluación de Resultados…….58

INTRODUCCIÓN

En el contexto actual, la gestión del tránsito vehicular se ha convertido en un desafío relevante debido al crecimiento sostenido del parque automotor y a la necesidad de contar con información precisa para la planificación y toma de decisiones en infraestructura vial. Actualmente, la recopilación de datos para la clasificación y el conteo vehicular se realiza mayoritariamente mediante métodos manuales convencionales, los cuales demandan un alto esfuerzo operativo, consumen considerable tiempo y se encuentran expuestos a errores humanos. Esta situación limita la obtención de información confiable y oportuna, especialmente en estudios preliminares orientados a proyectos de señalización, iluminación y gestión del tráfico, tanto en vías urbanas como interurbanas.

Esta problemática se agrava en escenarios donde se requiere analizar grandes volúmenes de datos o contar con resultados en tiempo casi real, condición que los métodos tradicionales no logran satisfacer de manera eficiente. En ciudades con alta congestión vehicular, la falta de inmediatez y escalabilidad en la recopilación de datos afecta directamente la capacidad de respuesta de las entidades responsables de la planificación y control del tránsito, evidenciando la necesidad de adoptar soluciones tecnológicas innovadoras alineadas con las actuales líneas de investigación en inteligencia artificial y visión computacional.

En este marco, la presente investigación se justifica por su aporte a la automatización de los procesos de clasificación y conteo vehicular mediante el uso de técnicas de Deep Learning, las cuales permiten analizar imágenes y videos capturados en carreteras para identificar y contabilizar unidades vehiculares con mayor precisión y eficiencia. El estudio adopta un enfoque aplicado y experimental, dado que busca resolver un problema concreto del entorno real mediante el diseño, implementación y evaluación de un modelo de aprendizaje profundo, cuyas métricas son validadas sobre un conjunto de datos previamente procesado.

El objetivo general de este trabajo es implementar una aplicación basada en Deep Learning para la clasificación y conteo vehicular en carreteras. Para ello, se plantean como objetivos específicos la elaboración de un conjunto de datos conforme a la categorización vehicular establecida por el Ministerio de Transportes y Comunicaciones, el

procesamiento y transformación de dichos datos, el entrenamiento del modelo de aprendizaje profundo, la evaluación de los resultados obtenidos en distintos escenarios y el desarrollo de una aplicación funcional que integre el modelo para su uso práctico.

El informe se estructura en seis capítulos. El primero desarrolla el diseño teórico, abordando los antecedentes de la investigación, las bases conceptuales y la categorización de variables. El segundo capítulo presenta el diseño metodológico, donde se detallan el tipo de investigación, la población y muestra, la metodología y los aspectos éticos. El tercer capítulo describe el desarrollo del proyecto, incluyendo la adquisición y preprocesamiento de datos, el entrenamiento del modelo y el flujo de trabajo de la aplicación. El cuarto capítulo expone la evaluación de los resultados obtenidos en distintos escenarios de aplicación. El quinto capítulo corresponde a la discusión de los resultados, y finalmente, el sexto capítulo presenta las conclusiones derivadas del estudio, seguidas de las referencias y apéndices correspondientes.

CAPÍTULO 1: DISEÑO TEÓRICO

1.1 Antecedentes de la investigación

Lin et al. (2022) presentan un marco de trabajo basado en aprendizaje profundo para el conteo de vehículos a partir de videos. La investigación propone un enfoque que utiliza cámaras de video como una alternativa a los métodos tradicionales de conteo, como los sensores en carretera. El sistema se divide en tres etapas: construcción de un conjunto de datos, construcción de un modelo de detección de vehículos y conteo de vehículos. Para abordar la falta de datos anotados, se utiliza el aprendizaje por transferencia, lo que permite construir un modelo de detección de vehículos de alto rendimiento rápidamente. En la etapa de conteo, se fusionan áreas de detección virtual y seguimiento de vehículos, implementando módulos para suprimir alarmas de detección errónea y mejorar la precisión del conteo. Los resultados muestran que el marco propuesto puede alcanzar una precisión de conteo de vehículos de hasta el 99%, ofreciendo una solución eficiente para la gestión del tráfico urbano y la mejora de la movilidad.

Kosuri, N.B. y Manne, S. (2023). Presentan un sistema automático de monitoreo de asistencia estudiantil que utiliza un enfoque integrado de HAAR Cascade y redes neuronales convolucionales (CNN) para el reconocimiento facial, incluso con el uso de mascarillas. La investigación aborda la dificultad de marcar la asistencia manualmente, especialmente durante la pandemia, donde el uso de mascarillas es obligatorio. El sistema propuesto combina algoritmos de detección de objetos para reconocer múltiples rostros simultáneamente, extrayendo características clave de las áreas visibles del rostro, como los ojos y la parte superior de la nariz. Este enfoque innovador permite marcar la asistencia de los estudiantes de manera eficiente y precisa, superando las limitaciones de los métodos tradicionales y adaptándose a las nuevas normativas de salud pública.

La tesis de Coanqui et al. (2022) presentan un sistema automático para el conteo de vehículos y la detección de congestión de tráfico utilizando procesamiento de imágenes. La investigación propone el uso de cámaras de video como una alternativa más eficiente a los métodos tradicionales, como los sensores en carretera. El sistema se basa en técnicas de visión por computadora, incluyendo la

segmentación de fondo y la detección de objetos, para contar vehículos y medir el nivel de congestión en tiempo real. Esto proporciona una herramienta más efectiva para la gestión del tráfico urbano y la mejora de la movilidad. Los resultados obtenidos con el modelo YOLOv3 demuestran una alta precisión del 95,33%, evidenciando la efectividad del enfoque desarrollado en la tesis.

Huang, Y. (2024) presenta un estudio sobre la aplicación de modelos YOLO-NAS y YOLOv8 en sistemas de seguimiento de múltiples objetos. La investigación explora la combinación de estos modelos de detección de última generación con técnicas avanzadas de seguimiento para mejorar la precisión y eficiencia en la detección y seguimiento de objetos en tiempo real. Utilizando conjuntos de datos públicos como MOT17 y MOT20, el sistema demuestra un rendimiento superior en comparación con métodos tradicionales, logrando una evaluación más efectiva del seguimiento de objetos en diversos entornos. Los resultados indican que la configuración del sistema y la elección de parámetros son fundamentales para optimizar el rendimiento, destacando la capacidad de los modelos YOLOv8 y YOLO-NAS para adaptarse a diferentes escenarios con una alta precisión.

Redmill et al. (2023). Presentan un sistema automático para el conteo de vehículos utilizando cámaras montadas en autobuses de tránsito. La investigación propone operacionalizar la vigilancia del tráfico aprovechando los sensores de percepción y localización ya instalados en estos vehículos. El sistema utiliza un modelo de aprendizaje profundo para la detección de objetos y el método SORT para el seguimiento de vehículos, logrando contar vehículos bidireccionalmente y distinguir entre vehículos estacionados y en movimiento. Los resultados demuestran que el sistema puede alcanzar altos niveles de precisión en el conteo de vehículos, lo que representa una herramienta efectiva para la planificación y gestión del tráfico urbano, aumentando así la cobertura de datos de tráfico en la red vial.

Rajasekhar et al. (2023). Presentan un sistema de detección de crímenes que utiliza aprendizaje automático mediante OpenCV, YOLO y CNN. La investigación propone el uso de un sistema de CCTV distribuido como una alternativa a los métodos tradicionales de vigilancia. El sistema aprovecha técnicas de visión por computadora, incluyendo reconocimiento facial con el algoritmo OpenFace, detección de objetos con YOLOv3 y reconocimiento de gestos con CNN, para identificar sospechosos y analizar escenas del crimen en tiempo real. Esto

proporciona una herramienta más efectiva para la seguridad pública y la prevención del crimen, logrando una mejora significativa en la precisión de identificación y una reducción en las falsas alarmas, con un enfoque en la automatización del proceso de detección y el envío de informes a las autoridades.

Kranthi et al. (2021). Presentan un sistema de identificación de rostros criminales utilizando un algoritmo de red neuronal de cascada multitarea (MTCNN). La investigación propone el uso de este sistema para la detección y reconocimiento automático de rostros de criminales en tiempo real, utilizando solo una imagen del sospechoso, lo que se conoce como aprendizaje de un solo disparo. El sistema tiene como objetivo mejorar la identificación de criminales y facilitar la labor de las fuerzas del orden al enviar notificaciones con información relevante. Se logró una precisión del 86% en un conjunto de datos de 50 registros, destacando la capacidad del modelo para operar en situaciones del mundo real y su potencial para prevenir delitos antes de que ocurran.

Chowdhury et al. (2022). Presentan un sistema automático para la detección y clasificación de vehículos utilizando redes neuronales profundas. La investigación propone el uso de modelos de aprendizaje profundo, como VGG16, VGG19 y YOLOv5, como una alternativa a los métodos tradicionales de clasificación. El sistema utiliza un conjunto de datos de imágenes de vehículos de Bangladesh y aplica técnicas de aumento y preprocesamiento de datos mediante el paquete Keras "ImageDataGenerator". Los resultados muestran que el modelo YOLOv5 logra una alta precisión del 83,02% para la clasificación de vehículos en tiempo real, proporcionando una herramienta eficiente para la gestión del tráfico y la mejora de la seguridad vial.

Martínez Samper, A.J. (2023). Presenta un sistema automatizado para el conteo y clasificación de vehículos en los accesos de aeropuertos, utilizando inteligencia artificial y procesamiento de imágenes. La investigación propone el uso de algoritmos avanzados, como YOLO (You Only Look Once), como alternativa a los métodos manuales tradicionales. El sistema implementa técnicas de visión por computadora para detectar y clasificar vehículos en tiempo real, facilitando la toma de decisiones en la gestión del tráfico aeroportuario y optimizando la ocupación de las instalaciones. Los resultados preliminares indican una mejora significativa en

la eficiencia del conteo y clasificación de vehículos, contribuyendo a una mejor planificación y respuesta ante la demanda en los aeropuertos.

Gómez, J. y Ramos, P. (2023). Se presenta un enfoque de estrategias de preprocesado y postprocesado en 17os17 learning aplicado a problemas multiclase en el ámbito de la seguridad y la biodiversidad. La investigación propone técnicas avanzadas para la clasificación de objetos en imágenes complejas, abordando desafíos como la variabilidad de las clases y la mejora en la precisión de los modelos. Utilizando métodos como el filtrado de ruido y la normalización de datos, se logró mejorar la precisión de los modelos de 17os17 learning en más del 90%, optimizando su rendimiento en la detección de objetos relacionados con la biodiversidad y la seguridad.

Reddy, S.A. (2023). Presentan un sistema automático para el conteo de vehículos y la detección de congestión de tráfico utilizando procesamiento de imágenes. La investigación propone el uso de cámaras de video como una alternativa a los métodos tradicionales, como sensores en carretera. El sistema aprovecha técnicas de visión por computadora, incluyendo segmentación de fondo y detección de objetos, para contar vehículos y medir el nivel de congestión en tiempo real, proporcionando una herramienta más eficiente para la gestión del tráfico urbano y la mejora de la movilidad, demostrándose así resultados del modelo YOLOv3 con una alta precisión del 95,33%.

Kotsoglou, K.N. y Oswald, M. (2020). El artículo examina el uso de la Reconocimiento Facial Automatizado (AFR) en el contexto de la intervención policial y su validez como evidencia en procedimientos penales. Los autores argumentan que, aunque AFR tiene el potencial de revolucionar la identificación de individuos y facilitar la detección de delitos, su implementación plantea importantes desafíos legales y éticos. A través del análisis de un caso reciente de la Policía de Gales del Sur, se discute la necesidad de un marco conceptual sólido que regule el uso de AFR, destacando que las decisiones deben ser tomadas por humanos y no por algoritmos. El artículo subraya que, a pesar de la aparente objetividad de AFR, su naturaleza probabilística y los riesgos asociados requieren una cuidadosa consideración para garantizar el respeto a los derechos humanos y la justicia en el proceso penal.

Romero et al. (2020). Proponen un sistema basado en visión artificial para el reconocimiento, clasificación y conteo de vehículos en tiempo real, utilizando técnicas de aprendizaje profundo como YOLOv3. El sistema integra cámaras para captar imágenes del tráfico y emplea redes neuronales convolucionales para clasificar vehículos en ligeros y pesados, proporcionando una alternativa más eficiente y de menor costo frente a los sensores tradicionales. Los resultados experimentales muestran una alta precisión del 70% al detectar vehículos, mientras que otras arquitecturas como R-CNN lograron un 80%.

Barba Guamán, L.R. (2021). Este estudio propone el uso de redes neuronales profundas, específicamente YOLOv3 y SSD MobileNet v2, para detectar objetos en zonas rurales con condiciones complejas de iluminación y ausencia de marcas viales. Se utilizó el dispositivo embebido NVIDIA Jetson Nano para ejecutar los modelos. Los resultados muestran que no existe un modelo ideal, pero YOLOv3 logró una precisión promedio del 76% para detección de vehículos, mientras que SSD MobileNet v2 fue más eficiente en procesamiento.

Gaur, K. et al. (2023). Presentan un sistema de visión por computadora para la detección, seguimiento y conteo de vehículos durante la temporada de lluvias nocturnas en India. La investigación propone el uso de técnicas de aprendizaje profundo, específicamente el algoritmo YOLOv8 para la detección de vehículos y el algoritmo DeepSORT para el seguimiento de múltiples objetos. Este sistema aborda los desafíos asociados con las condiciones climáticas adversas, demostrando una notable precisión en la identificación y conteo de vehículos, así como en la estimación de velocidades. Los resultados indican que el sistema es eficaz para el análisis del tráfico, contribuyendo a la gestión del transporte y la mejora de la movilidad en situaciones difíciles, con un enfoque en la recopilación de datos en tiempo real. Se reportó que el modelo YOLOv4, utilizado como referencia, alcanzó una precisión promedio (AP) del 82.08% en la detección de vehículos bajo condiciones de lluvia nocturna.

Shanthi et al. (2021) presenta un sistema de reconocimiento facial en tiempo real utilizando drones, diseñado para ayudar a las fuerzas de tarea en la identificación de criminales, personas desaparecidas y para fines de vigilancia. La investigación propone el uso de un vehículo aéreo no tripulado (UAV) equipado con una cámara que se conecta a un software de reconocimiento facial. Este sistema logra una

precisión del 98.6% cuando el ángulo de la cámara se encuentra dentro de los 37 grados, lo que indica su capacidad para adaptarse a diferentes posiciones del dron. Las aplicaciones del sistema incluyen asistencia a las autoridades policiales en investigaciones, rescates, análisis de incidentes y monitoreo de multitudes, demostrando así su potencial en la mejora de la seguridad pública y la gestión de emergencias.

Abuelgasim S. et al. (2024) presenta un sistema innovador para el conteo de vehículos en tiempo real, utilizando un algoritmo de detección de objetos personalizado basado en YOLOv8n y el algoritmo de seguimiento DeepSORT. Este sistema está diseñado para ser implementado en dispositivos de computación de borde con recursos limitados, eliminando la dependencia de recursos de computación en la nube. La investigación destaca la importancia de las soluciones basadas en visión por computadora, que ofrecen una alternativa más eficiente a los métodos tradicionales de conteo de vehículos, como los sensores instalados en las carreteras. El sistema propuesto logra una precisión media de detección (mAP) del 97.5% y una precisión en el conteo de vehículos del 96.8%, lo que demuestra su efectividad en la gestión del tráfico y la mejora de la seguridad vial. Además, el enfoque de computación en el borde proporciona beneficios en términos de seguridad y latencia, al mantener el procesamiento de datos más cerca de la fuente.

Neamah, S.B. et al. (2023). Presentan un sistema de monitoreo de tráfico en tiempo real basado en técnicas de aprendizaje profundo utilizando el algoritmo YOLOv8. La investigación propone el uso de cámaras de video como una alternativa a los métodos tradicionales de monitoreo, proporcionando una solución más eficiente para la gestión del tráfico urbano. El sistema aprovecha técnicas de visión por computadora para ofrecer información crucial, incluyendo el conteo de vehículos, la clasificación, la estimación de velocidad y la estimación del tamaño de los vehículos. Se destaca la capacidad del sistema para adaptarse a diferentes configuraciones de hardware y condiciones ambientales, logrando resultados de alta precisión y bajos. Para la estimación del tamaño de los vehículos, se obtuvo una precisión promedio del 87.28% con un error promedio del 12.72%.

Xie, H. et al. (2023). Presentan el modelo PVNet, diseñado para la detección, seguimiento y conteo de vehículos y peatones en entornos de tráfico complejos. Este modelo, basado en una red mejorada de YOLO y utilizando el algoritmo

Bytetrack, optimiza la extracción de características, reduciendo la complejidad computacional y los parámetros del sistema. Los resultados experimentales muestran que PVNet logra una precisión media promedio (mAP) de 0.952, utilizando solo el 32% de los parámetros en comparación con YOLOv8s. Esto demuestra su efectividad en la detección de vehículos y peatones, abordando desafíos como detecciones perdidas y falsas alarmas, lo que lo convierte en una herramienta valiosa para sistemas de transporte inteligente y la prevención de colisiones en vehículos autónomos.

AforosDron. (2023). Presentan un sistema automático para el conteo de vehículos y la detección de congestión de tráfico utilizando tomas aéreas realizadas por drones. La investigación propone el uso de esta tecnología como una alternativa a los métodos tradicionales, como el conteo manual en intersecciones. El sistema aprovecha técnicas de visión por computadora, incluyendo flujo óptico y detección de objetos, para contar vehículos y medir el nivel de congestión en tiempo real. Se demuestra que el modelo YOLOv3 alcanza una alta precisión en la detección, con un porcentaje de precisión del 100% para motocicletas, un recall del 84.61% para buses y un 86.79% para camiones. Esto proporciona una herramienta más eficiente para la gestión del tráfico urbano y la mejora de la movilidad, optimizando así los recursos y abriendo nuevas oportunidades de mercado para la empresa.

1.2 Bases teóricas

1.2.1 Categorización vehicular utilizada por el Ministerio de transportes y Comunicaciones

Tenemos una clasificación vehicular de 4 categorías, los vehículos de carga liviana se definen según su peso bruto vehicular (Directiva N° 002-2006-MTC/15, MTC, 2006, p. 3-4).

- **Categoría M: Vehículos automotores de 4 ruedas o más diseñados para el transporte de personas**

Figura 1

Vehículo categoría M

- M1: De hasta 9 asientos (incluido el conductor).
- M2: De más de 9 asientos (incluido el conductor) y PBV de hasta 5 toneladas.
- M3: De más de 9 asientos (incluido el conductor) y PBV mayor a 5 toneladas.

- **Categoría N: Vehículos automotores de 4 ruedas o más diseñados para el transporte de mercancías**

Figura 2

Vehículo categoría N

- N1: De PBV de 3,5 tn. O menos.

- N2: De PBV mayor a 3,5 tn. Hasta 12 tn.
- N3: De PBV mayor a 12 tn

- **Categoría O: Remolques y Semiremolques**

  - O1: Remolques

  Figura 3

  Vehículo categoría O

- **Categoría L: Vehículos automotores con menos de 4 ruedas**

  Figura 4

  Vehículo categoría L

  - L1: Dos ruedas, hasta 50 cm3 y Velocidad máxima de 50 km/h.
  - L2: Tres ruedas, hasta 50 cm3 y Velocidad máxima de 50 km/h.
  - L3: Dos ruedas, más de 50 cm3 o Velocidad mayor a 50 km/h.
  - L4: Tres ruedas asimétricas al eje longitudinal, más de 50 cm3 ó Velocidad mayor a 50 km/h.
  - L5: Tres ruedas simétricas al eje longitudinal, más de 50 cm3 o Velocidad mayor a 50 km/h y PBV menor a 1 tonelada.

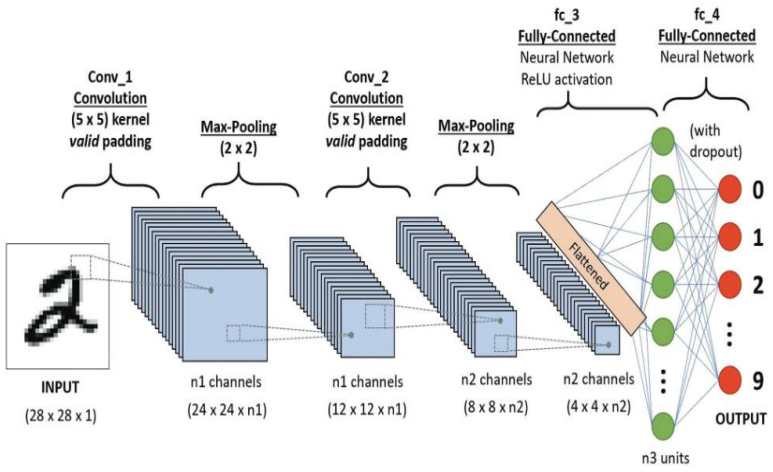
1.2.2 Deep Learning y Visión Artificial

El deep learning se ha consolidado como la técnica más avanzada para el reconocimiento de patrones en imágenes, gracias al uso de redes neuronales convolucionales (CNN). Estas arquitecturas han demostrado un rendimiento sobresaliente en tareas como clasificación de objetos, segmentación y detección.

En el caso de la visión artificial aplicada al tránsito, las CNN permiten analizar secuencias de video e identificar vehículos con gran precisión, incluso bajo condiciones complejas como variaciones de luz, clima o ángulos de cámara. El aporte clave está en su capacidad de aprender representaciones jerárquicas, desde bordes simples hasta formas complejas de vehículos.

Figura 5

Modelo CNN tradicional



Fuente: Obtenido de (Neamah, 2023)

1.2.3 Clasificación y Conteo Vehicular

El conteo vehicular basado en IA ofrece ventajas sobre métodos tradicionales (sensores de bucle inductivo o conteo manual). Con modelos de detección, se pueden obtener métricas en tiempo real como flujo vehicular, densidad y composición de flota. Estos indicadores son cruciales para la planificación urbana y la toma de decisiones en gestión de tráfico.

El Ministerio de Transportes y Comunicaciones (MTC) establece tipologías de vehículos (M, N, L, O) que deben ser reconocidas. Por ello, el dataset preparado responde directamente a la normativa nacional, garantizando pertinencia y aplicabilidad en el contexto peruano.

Figura 6

System's vehicle classification

Fuente: Obtenido de (Neamah, 2023)

1.2.4 Herramientas Tecnológicas

El proyecto emplea el framework YOLOv8 (You Only Look Once, versión 8), que es un modelo state of the art para detección rápida y precisa. Se ha complementado con:

- Python 3.13+ como lenguaje base.
- PyTorch como backend para entrenamiento.
- Ultralytics como librería de implementación.
- Dataset propio balanceado con imágenes recolectadas y aumentadas, siguiendo lineamientos del MTC.
- Duplicate Photos Fixer Pro

1.3. Categorización de las variables

1.3.1. Variable independiente

Modelo custom de deep learning (YOLOv8).

1.3.2 Variable dependiente

Nivel de precisión en la clasificación y conteo vehicular

VARIABLE	DIMENSIÓN	INDICADORES	ESCALA DE MEDICIÓN
Variable Independiente Modelo de *Deep Learning* (YOLOv8)	Precisión del modelo	- Valor de *mAP50* y *mAP50-95* obtenido en la validación del modelo.- Precisión (P) y *Recall* ® promedio en la detección.	Confianza y probabilidad
Variable Dependiente Nivel de precisión en la clasificación y conteo vehicular	Métricas de desempeño	Diferencia porcentual entre el conteo real y el conteo estimado por el modelo.	Confianza y probabilidad

Tabla 1: Operacionalización de variables

1.3.3. Definiciones conceptuales

Disponemos de ciertas métricas que, llevados de la mano con la precisión, de igual manera tienen un punto crítico como medida estadística en entrenamiento como en evaluación y pruebas de resultados, se escogieron las principales tomando como referencia algunas de estas métricas de Ultralytics (Ultralytics, 2023):

- **IoU (Intersection over Union)**

  Es la métrica base. Mide la superposición entre el cuadro predicho (Bp) y el real (Bg).

$$IoU = \frac{Area(Bp \cap Bg)}{Area(Bp \cup Bg)}$$

- **Precision (Precisión)**

  Mide la calidad de las predicciones positivas. Responde a: "¿Qué porcentaje de lo que detecté es realmente correcto?"

$$Precision = \frac{TP}{(TP + FP)}$$

- TP (True Positives): Detecciones correctas (IoU > umbral).
- FP (False Positives): Detecciones erróneas o en lugares donde no hay nada.

- **Recall (Exhaustividad)**

  Mide la capacidad del modelo para encontrar todos los objetos. Responde a: "¿Qué porcentaje de los objetos reales logré capturar?"

$$Recall = \frac{TP}{(TP + FN)}$$

- FN (False Negatives): Objetos reales que el modelo ignoró por completo.

- **F1-Score**

  Es el balance ideal (media armónica). Se usa porque la media aritmética simple puede ser engañosa si la precisión o el recall son muy bajos.

$$F1 = 2 * \frac{(Precision * Recall)}{(Precision + Recall)}$$

- **AP (Average Precision) y mAP**
  - El AP es el área bajo la curva de Precision-Recall.
  - El mAP (mean Average Precision) es el promedio de esa área para todas las clases (N).

$$mAP = \frac{1}{N} * \sum_{i=1}^{N} APi$$

- mAP@50: Precisión media cuando el IoU es mayor al 50%. Es la métrica estándar de “éxito”
- mAP@50-95: Promedio de la precisión en varios niveles de IoU (del 50% al 95%). Es mucho más exigente y premia la precisión exacta de los cuadros.

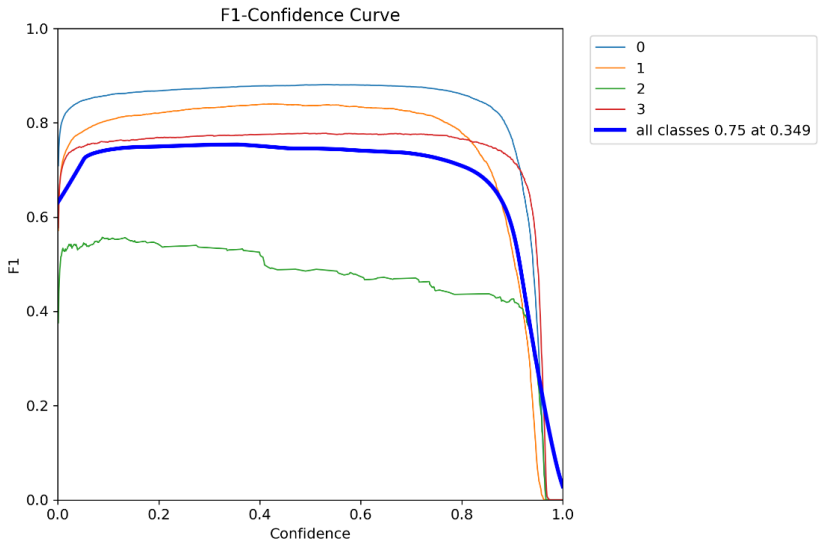
- **Salidas visuales**

La función model.val(), además de producir métricas numéricas, también produce salidas visuales que pueden proporcionar una comprensión más intuitiva del rendimiento del modelo. Aquí hay un desglose de las salidas visuales que puede esperar, según la información de Ultralytics (Ultralytics, 2023):

- Curva de Puntuación F1 (F1_curve.png): Esta curva representa el Puntuación F1 a través de varios umbrales. La interpretación de esta curva puede ofrecer información sobre el equilibrio del modelo entre falsos positivos y falsos negativos en diferentes umbrales.

Figura 7

F1 - Confidence Curve

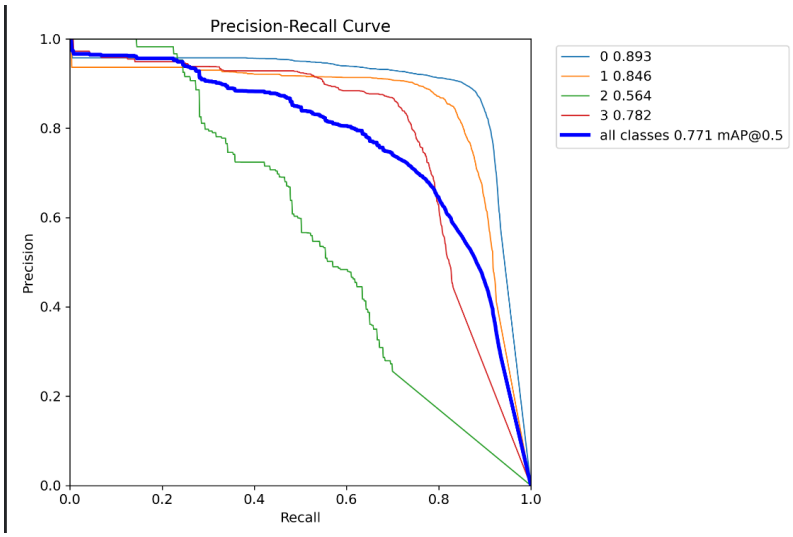


Fuente: Fuente propia

- Curva Precisión-Recall (PR_curve.png): Una visualización integral para cualquier problema de clasificación; esta curva muestra las ventajas y desventajas entre precisión y exhaustividad a diferentes umbrales. Se vuelve especialmente significativo cuando se trata de clases desequilibradas.

Figura 8

Precision – Recall Curve

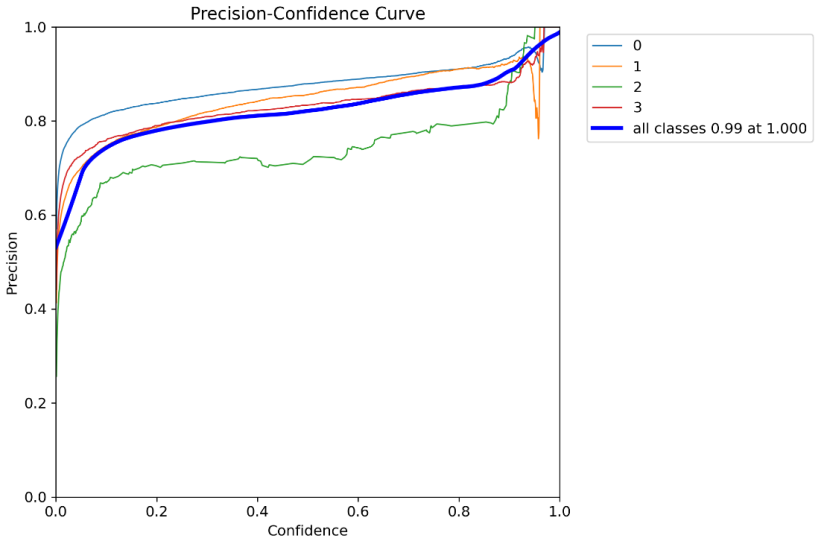


Fuente: Fuente propia

- Curva de Precisión (P_curve.png): Representación gráfica de los valores de precisión en diferentes umbrales. Esta curva ayuda a comprender cómo varía la precisión a medida que cambia el umbral.

Figura 9

Precision – Confidence Curve

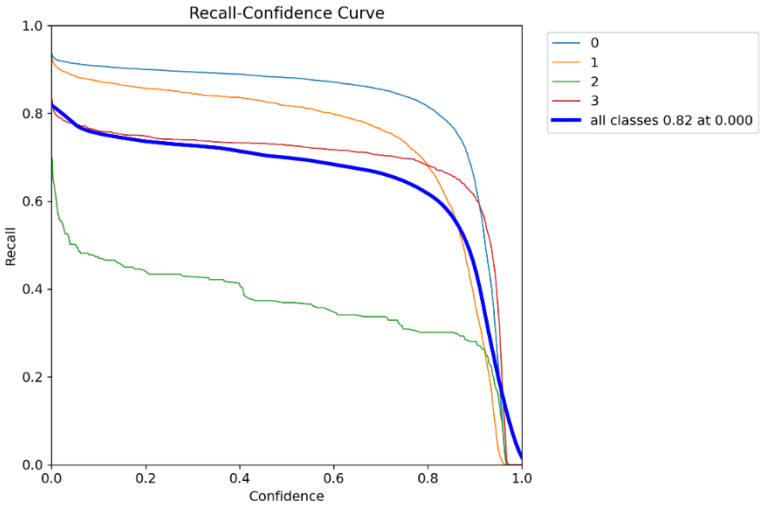


Fuente: Fuente propia

- Curva de Recall (R_curve.png): De manera correspondiente, este gráfico ilustra cómo cambian los valores de exhaustividad en los diferentes umbrales.

Figura 10

Recall-Confidence Curve

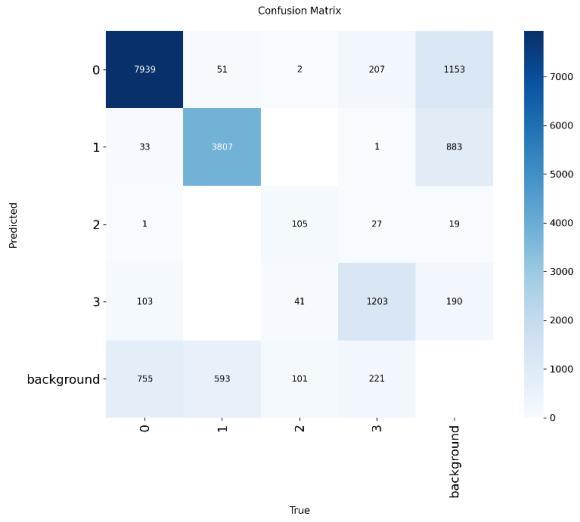


Fuente: Fuente propia

- Matriz de Confusión (confusion_matrix.png): La matriz de confusión proporciona una vista detallada de los resultados, mostrando los recuentos de verdaderos positivos, verdaderos negativos, falsos positivos y falsos negativos para cada clase.

Figura 11

Confusion - Matrix

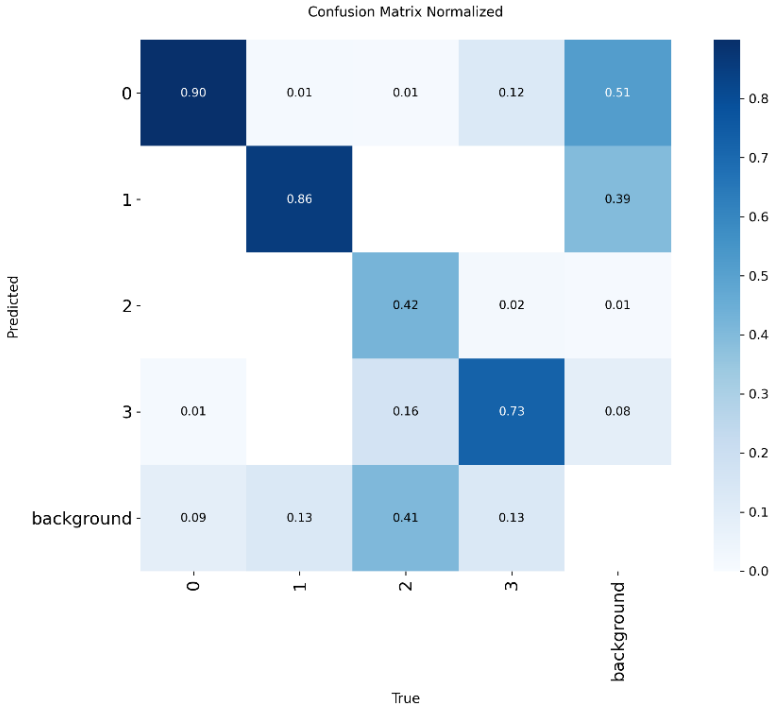


Fuente: Fuente propia

- Matriz de confusión normalizada (confusion_matrix_normalized.png): Esta visualización es una versión normalizada de la matriz de confusión. Representa los datos en proporciones en lugar de recuentos brutos. Este formato facilita la comparación del rendimiento entre clases.

Figura 12

Confusion Matrix Normalized

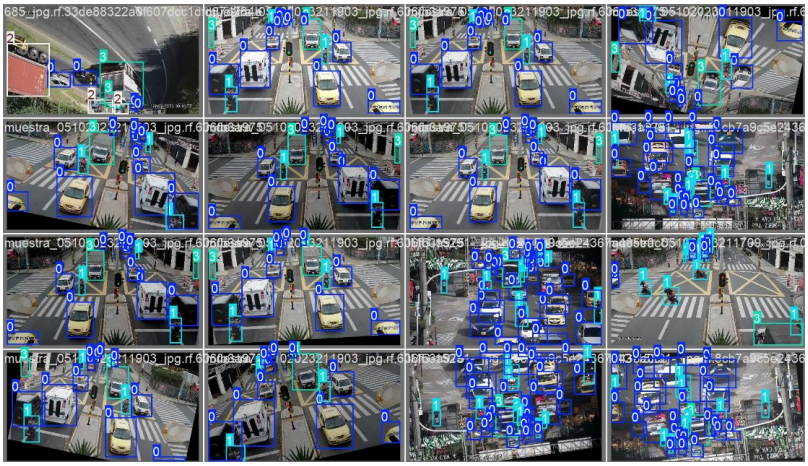


Fuente: Fuente propia

- Etiquetas del lote de validación (val_batchX_labels.jpg): Estas imágenes representan las etiquetas de verdad fundamental para distintos lotes del conjunto de datos de validación. Proporcionan una imagen clara de cuáles son los objetos y sus respectivas ubicaciones según el conjunto de datos.

Figura 13

Val batch X labels.jpg

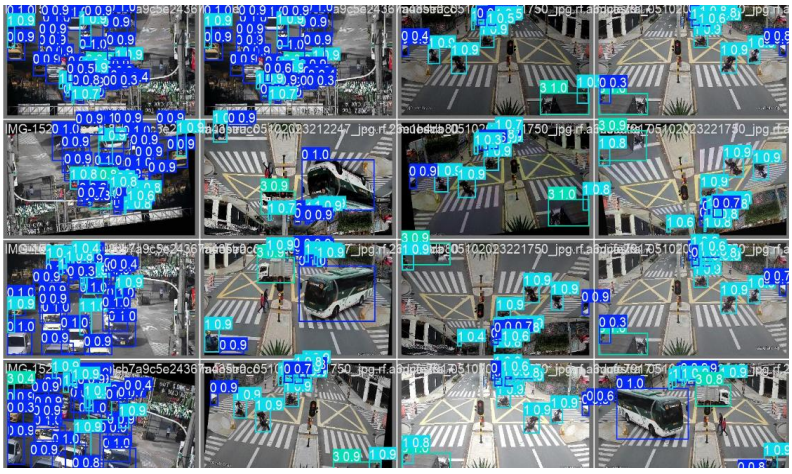


Fuente: Fuente propia

- Predicciones del lote de validación (val_batchX_pred.jpg): En contraste con las imágenes de etiquetas, estas visualizaciones muestran las predicciones realizadas por el modelo YOLO para los lotes respectivos. Al compararlas con las imágenes de etiquetas, se puede evaluar fácilmente qué tan bien el modelo detecta y clasifica visualmente los objetos.

Figura 14

Val batch X pred.jpg



Fuente: Fuente propia

1.3.4. Definiciones operativas

- **Preprocesamiento (Preparación de datos)**

  Antes de que el modelo aprenda, los datos se "estandarizan" para que las redes neuronales operen de forma eficiente, según la información de Ultralytics (Ultralytics, 2023):

  - Normalización: Proceso de escalar los valores de los píxeles (usualmente de 0-255 a 0-1) para estabilizar el aprendizaje.
  - Data Augmentation (Aumento de datos): Creación de variaciones artificiales de las imágenes (rotación, brillo, flips, 33os33ic) para evitar el sobreajuste y mejorar la generalización.
  - HSV Augmentation: Modificación del tono (Hue), saturación (Saturation) y valor (Value) para que el modelo sea robusto ante cambios de iluminación.
  - Bounding Box Scaling: Ajuste matemático de las coordenadas de las etiquetas cuando la imagen se redimensiona (ej. De 4000px a 640px).

- **Entrenamiento (Optimización y Aprendizaje)**

  Términos que definen cómo el modelo "ajusta" sus pesos internos según la documentación oficial de Ultralytics (Ultralytics, 2023):

  - Epochs (Épocas): El número total de veces que el modelo verá el dataset completo.
  - Batch Size: Cantidad de imágenes procesadas antes de que el modelo actualice sus parámetros.
  - Learning Rate (Tasa de aprendizaje): Qué tan grandes son los pasos que da el modelo al corregir sus errores.
  - Loss Functions (Funciones de pérdida)**:** Cuantifican el error. YOLOv8 usa:
    - **box_loss:** Error en la ubicación del cuadro delimitador.
    - **cls_loss:** Error en clasificar el objeto correctamente (ej. Confundir un gato con un perro).
    - **dfl_loss:** (Distribution Focal Loss) Mejora la precisión en los bordes de los objetos.

CAPÍTULO 2: DISEÑO METODOLÓGICO

Ya conociendo los entornos, tecnologías, y datos que vamos a emplear en el proyecto procedemos a compartir la metodología experimental, primero inicializamos viendo el modelo más adecuado para este tipo de situaciones vehiculares, procedemos a evaluar la situación problemática y de qué manera estos datos se trabajaran, si es por clasificación, segmentación, etc. Posterior a ello seleccionamos características específicas de cada clase de vehículos, y de qué manera estos van a ser identificados, finalmente se procesan esos datos, y tenemos como resultado el modelo ajustado que utilizaremos para hacer las inferencias VS el escenario ideal.

2.1. Tipificación de la investigación

2.1.1. Justificación

Se adopta un diseño **aplicado y experimental**. Aplicado porque busca resolver un problema concreto de clasificación y conteo vehicular; experimental porque se construye y acoge de un modelo de Deep learning con métricas evaluadas sobre un conjunto de validación.

2.2 Población y muestra

La **población** objetivo para este estudio se compone de las 17 clases que el MTC considera, por supuesto incluyendo en la totalidad todos los subtipos de clases que estas albergan; para clase M (M1, M2, M3), para clase N (N1, N2, N3), para clase L (L1, L2, L3, L4, L5, L6, L7), y la clase O (O1, O2, O3, O4).

La **muestra** vendría a ser representada por las 4 categorías que en este caso representan las clases de las etiquetas de nuestro modelo de entrenamiento (M, N, L, O).

2.3 Metodología y recolección de datos

La metodología planteada se estructura de forma secuencial para asegurar un desarrollo sólido y coherente del sistema de detección y conteo vehicular. En primer lugar, se realiza la adquisición de imágenes, donde se recopilan videos provenientes de cámaras CCTV ubicadas en ciudades con altos niveles de congestión o en vías con tránsito pesado; esta etapa es fundamental, ya que la representatividad de los escenarios reales garantiza que el modelo aprenda patrones acordes a condiciones de tráfico reales.

En el segundo paso, correspondiente al procesamiento de imágenes, se extraen fotogramas relevantes a partir de los videos y se aplican técnicas de aumento de datos, lo cual permite incrementar la diversidad del conjunto y mejorar la capacidad de generalización; además, los datos se organizan en conjuntos de entrenamiento y prueba para asegurar una evaluación objetiva.

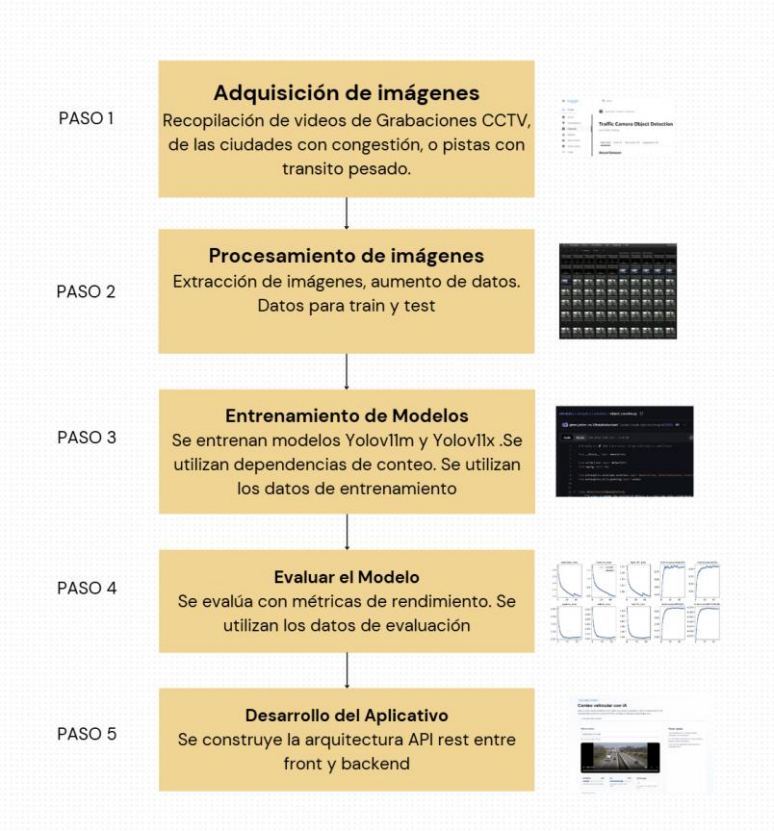
Posteriormente, en la fase de entrenamiento de modelos, se entrenan arquitecturas YOLOv11m y YOLOv11x utilizando los datos preparados, incorporando dependencias de conteo que permiten no solo detectar vehículos, sino también cuantificarlos de manera precisa en distintos contextos de tráfico.

A continuación, se lleva a cabo la evaluación del modelo, donde se emplean métricas de rendimiento para medir su precisión, estabilidad y capacidad de generalización, utilizando exclusivamente los datos de evaluación para evitar sesgos.

Finalmente, se desarrolla el aplicativo, en el cual se implementa una arquitectura basada en una API REST que integra el frontend y el backend, permitiendo que el modelo entrenado sea consumido, y podamos tener una interfaz iterativa para realizar inferencias especializadas.

Figura 15

Metodología secuencial Flujo Web + Python



Fuente: Fuente Propia

2.4. Aspectos Éticos de la Investigación

- Integridad y originalidad académica: El diseño metodológico incluye una validación de originalidad mediante el programa Turnitin, asegurando que el índice de similitud (18%) no constituya plagio y que se respeten estrictamente las normas de citación y referencias institucionales.
- Responsabilidad y transparencia científica: La metodología incorpora la evaluación del desempeño en cuatro escenarios reales con distintas condiciones (como ruido visual o congestión) para garantizar la veracidad de los datos y reportar con honestidad las limitaciones de precisión del modelo.

CAPÍTULO 3: DESARROLLO DEL PROYECTO

3.1 Adquisición de imágenes y videos de prueba

Teniendo como perspectiva a futuro el lugar donde se están evaluando esta investigación se tomó como referencia imágenes un video de la Av Brasil con el objetivo de adaptar al contexto Peruano, como referencia de entrenamiento del modelo, y se procederá a tomar como referencias de pruebas las Av. Salaverry con José Leonardo Ortiz. Los frames tomadas del video, se tomaron en resolución 1920x1080 y otro Split de 960x540 y se aproximó una perspectiva de 50-60 metros de distancia de detección desde el lugar de grabación.

Este dataset está enfocado en detección de vehículos y transportes, basadas en escenarios específicos en este caso Avenidas públicas de distancias de cámaras estáticas para distancias no mayores a 150m desde la perspectiva de la cámara.

Ya que mencionamos el término videos, tenemos que hacer un Split de datos basados en esos videos entonces tuvimos que hacer una conversión de data de video a imágenes, de que manera se podría decir, digamos que basados en las características de los videos tomamos para evitar la sobreestimulación de data tomamos cierta proporción de frames de los videos, si por un video duraba 49 segundos y tenía 30fps hacemos la conversión siguiente:

$$N\ total\ de\ imágenes\ tomadas\ en\ el\ video\ = \frac{\#total\ de\ segundos\ x\ total\ de\ frames}{\#imagenes\ extraídas\ x\ frame)}$$

$$49\ x\ 30/\ 6\ =\ 245\ imágenes$$

Tuvimos un total de 2820 imágenes

3.1.1 Metodología de Extracción de imágenes:

1. Encontrar una librería que realice el Split de frames (Ffmpeg)
2. Encontrar la proporción de imágenes extraídas por cada cierto número de frames recorridos fps = 6
3. Registrar o crear el directorio a donde estas imágenes van a ir.

4. Definir un nombre específico por cada Split de imágenes para evitar duplicados.
5. Posterior a la extracción de las carpetas frames por cada video, unirlas todas las imágenes a una carpeta única para realizar la dispersión del dataset estándar 80%/10%/10%
6. Como paso adicional también podemos optar por hacer un Split automático apoyándonos de los splits de datos que podemos hacer desde Roboflow.
7. De total de las categorías señaladas, se ha optado por considerar a manera general por clase, la clasificación M,N,L,O.

Imágenes	**Total de imágenes**	**Clases (Labels)**			
		Categoría M	**Categoría N**	**Categoría L**	**Categoría O**
Train (80%)	2261	3800	764	310	8
Val (10%)	280	475	96	39	1
Test (10%)	280	475	96	39	1
Total	2821	4750	956	388	10

Tabla 2: Cantidad de datos por Categorías

- A continuación, se muestran imágenes muestrales por categorías:

Categoría M (Transporte de personas):

Figura 16

Van

Figura 17

Auto Sedan

Figura 18

Camioneta

Fuente: Kaggle - Vehicular traffic video database (2022)

- Categoría N (Transporte de cargas – camión)

Figura 19

Camión mediano

Fuente: Kaggle - Vehicular traffic video database (2022)

- Categoría L (Vehículos de transporte con 2 y 3 ruedas):

Figura 19

Moto Lineal (Vista Frontal)

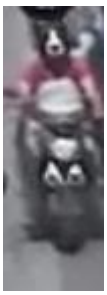


Figura 20

Moto Lineal (Vista Posterior)

Figura 21

Moto taxi

Fuente: Kaggle - Vehicular traffic video database (2022)

- Categorìa O (Trailers con remolque y semirremolque):

Figura 22

Camión con Volquete

Figura 23

Trailer con remolque

Figura 24

Trailer con remolque

Fuente: Kaggle - Vehicular traffic video database (2022)

3.2. Preprocesamiento de imágenes

3.2.1. Preparar el entorno de trabajo.

El entorno de trabajo del proyecto fue diseñado para garantizar un desarrollo eficiente, organizado y escalable. Se utilizó PyCharm y Visual Studio Code como Github como entornos de desarrollo integrado (IDE) debido a su robustez y basado en experiencia como herramientas de depuración avanzadas y su capacidad para manejar proyectos complejos con múltiples módulos y dependencias. A continuación, se profundiza en los aspectos clave del entorno de trabajo, incluyendo la estructura del proyecto, la configuración del entorno, la integración con herramientas externas y el flujo de trabajo.

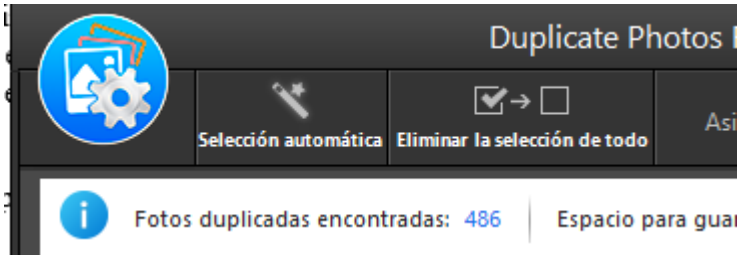
3.2.2. Refinamiento de los datasets:

Conociendo que partimos de un dataset que nace de los frames y objetos asociados a un frame, tenemos que aplicar ciertas técnicas de preprocesado una de ellas es la eliminación de algunos

Path: C:\mtc-multiclase\data\train\images

Figura 25

Duplicate Photos – Delete duplicated photos section



Fuente: Duplicate Photos

3.2.3 Equilibrado de clases y creación de labels:

Basados en lo que tenemos y la cantidad de imágenes, lo que procedimos hacer es preparar un pequeño proceso de autolabel, para estos casos se puede optar por utilizar herramientas como autolabel Robloflow, lo que hace esta herramienta es ayudarnos a las

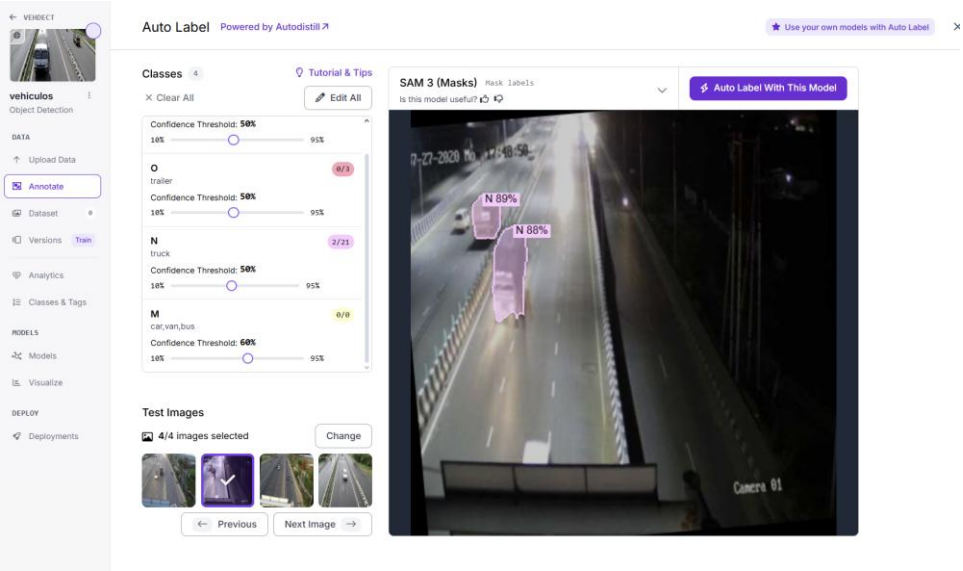
propiedades de los objetos de datasets del modelo y datasets de Ultralytics por ejemplo COCO.

Se organizaron estas relaciones asociadas a propiedades que nuestras clases tienen:

- 0 -> M-#car, #van
- 1 -> L-# motorcycle,
- 2 -> N-# bus, #truck,
- 3 -> O-#trailer, #remolque

Figura 26

Duplicate Photos – Delete duplicated photos section



3.2.4 Aumentación de imágenes

Basados en el dataset base que tenemos con las clases equilibradas ahora, realizamos una aumentación de x3 al dataset original con métricas de preprocesamiento basadas en rotación a 15º, escala de grises, saturación.

Por lo tanto, nuestro dataset quedaría 8000-9000 imágenes previo al entrenamiento:

Imágenes	Total de imágenes	Clases (Labels)			
		Categoría M	Categoría N	Categoría L	Categoría O
Train -80%	6783	11400	2292	930	24
Val -10%	840	1425	288	117	3
Test -10%	840	1425	288	117	3
Total	8463	14250	2868	1164	30

Tabla 3: Cantidad de datos aumentados por Categorías

3.3. Entrenamiento

- Basándonos en un modelo preentrenado bastante popular haciendo referencia a los antecedentes:

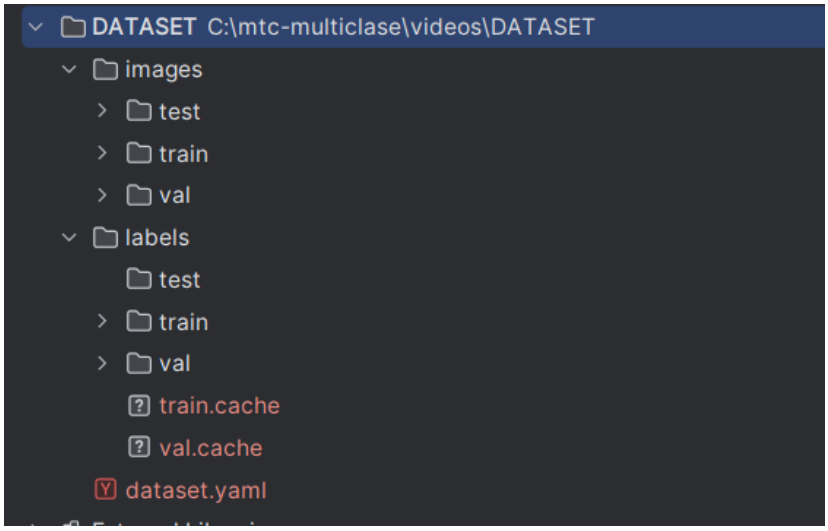
Característica	YOLO (ej. YOLOv8/v11)	SSD (Single Shot MultiBox Detector)	Faster R-CNN
Velocidad de Inferencia	**Muy Alta** (ideal para tiempo real)	Alta (buen balance)	Baja a Moderada (más lento)
Precisión (mAP)	**Muy Alta** (comparable a los mejores)	Alta (ligeramente inferior a YOLO/FRCNN)	**Excelente** (máxima precisión)
Complejidad Arquitectura	Moderada	Moderada	Alta (requiere generación de propuestas)
Uso Industrial Típico	**Inspección en línea de producción**, sistemas de vigilancia, robótica autónoma	Detección en dispositivos móviles o con recursos limitados, donde se requiere un balance	**Aplicaciones críticas de precisión** (ej. análisis médico, inspecciones de alta regulación)

Tabla 4: Comparación modelos de Deep Learning

- Llegados aquí podemos justificar que el uso cometido del modelo Yolo se debe a que está enfocado al tiempo real, practicidad, uso de inferencias, a lo que estamos apuntando como situación problemática, a continuación, los pasos que se utilizaron para preparar el entrenamiento:
  - Primero identificamos el tipo de entrenamiento que vamos a realizar, en este caso utilizamos la Detección de Objetos, como acción train, asignar la ruta a nuestro dataset:

Figura 27

Estructura Dataset



Fuente: Elaboración Propia

- Se utilizó el framework **Ultralytics** para entrenar el modelo YOLO en el conjunto de datos anotado.
- El entrenamiento se realizó en una GPU NVIDIA con soporte para CUDA, lo que aceleró significativamente el proceso.
- Durante el entrenamiento, se ajustaron hiperparámetros como la tasa de aprendizaje, el tamaño del batch y el número de épocas para optimizar el rendimiento del modelo tales como: imgsz=1024 epochs=100 device=0
- **Workers y batch:** Se configure un total de 4 cores, y un batch de 8 para no saturar la gráfica ya que tenemos como limite usar los 12 GB de VRAM de la gráfica en cuestión
- **Tamaño de la imagen de entrada**: Se configuró el modelo para procesar imágenes de 1920x1080 píxeles, lo que permitió una detección precisa en videos de alta resolución.
- **Clases de interés**: Se seleccionaron las clases relevantes para el proyecto, como automóviles, camiones, motocicletas y autobuses. Esto se hizo mediante el parámetro classes=[4] en el dataset.yaml

3.3.1 Entrenamiento 1:

Tuvimos el escenario que la clase 3 (remolques y vehículos de carga) no tiene un performance por encima de al menos 0.5 que es lo mínimo que se requiere

en la métrica de mAP, por lo que tuvimos que nutrir nuevamente a nuestro set de datos con nuevas imágenes y obtener un mejor performance.

Los recomendable aquí es ajustar el archivo. yaml a un sobreajuste de pesos para que a partir de nuestro best.pt (archivo de entrenamiento resultante) priorize el aprendizaje de esa clase con este parámetro: cls_weights: [1.0, 1.0, 1.0, 3.0], utilizando 100 épocas y yolov8m.pt uno estándar respecto a consumo energético vs tiempo de entrenamiento.

Resultados:

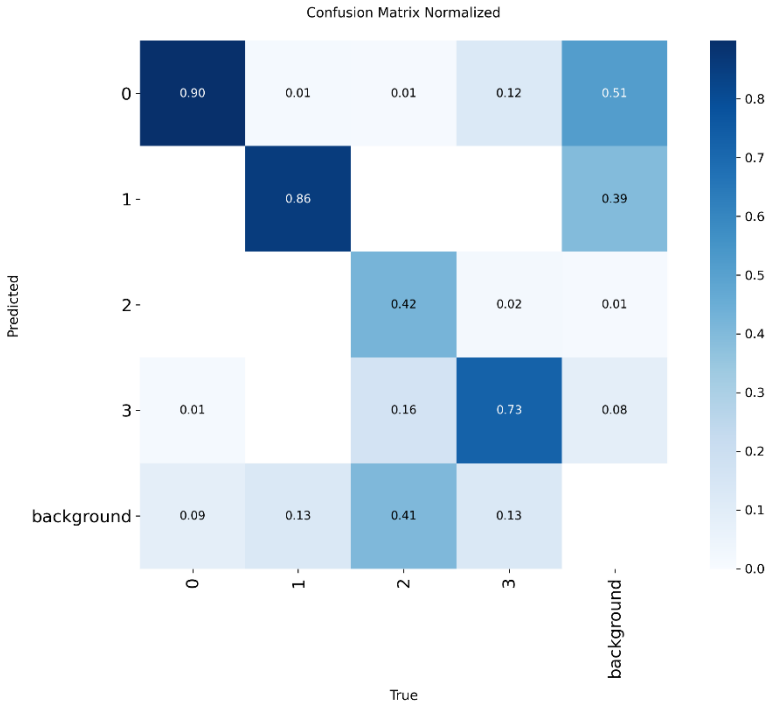
Class	Images	Instances	Box(P)	R	mAP50	mAP50-95
All	1425	15190	0.806	0.722	0.771	0.595
0	1395	8831	0.861	0.892	0.893	0.72
1	1169	4451	0.831	0.84	0.846	0.562
2	219	249	0.716	0.422	0.564	0.454
3	1035	1659	0.815	0.735	0.782	0.644

Tabla 5: Métricas de entrenamiento 1

- Podemos tomar 2 perspectivas de los problemas al enfrentarnos a un modelo de entrenamiento basado en 2 clases (camión y remolque – tráiler) debido a que el tráiler es un subtipo de camión por lo que damos por lo que podríamos tomar en este caso un MAP – H, tomando 0, 1, 3, que corresponde a una clase única de vehículos de carga pesada haciendo subir nuestro **mAp a 0.84**, y otra es la tradicional tomando transparentemente el promedio de todos los acuracy de todas las clases sería un resultante de 0.771.
- Tomamos como punto de mejora y/o críticos para este tipo de entrenamientos la tabla de normalización:

Figura 28

Matriz de Confusión Entrenamiento 1



Fuente: Elaboración Propia

- Como mejora critica puede apreciar que en la clase 2 tenemos una aversión de errores que muestra confusión con fondo quiere decir falsos negativos con respecto a otras clases, una recomendación sería una nueva aumentación en esa clase especifica o también el uso de un umbral de confianza específico en lugar de uno global que esta puesto de 0.35 por Yolo, reducirlo de 0.1 a 0.2.

3.3.2 Entrenamiento 2:

Tenemos un planteamiento diferente, en este caso para mejorar nuestro performance, mejoramos nuestro set de datos basado en uno de Roboflow con labels mejor equilibrados, utilizando el modelo yolov11x.pt, en este caso el estándar recomendable para detección de objetos actual, y además el más pesado con respecto a rendimiento, apuntando a tener los mejores resultados posibles desde la parte del modelo de reentrenado, por lo que realizamos una apuesta por 250 épocas:

Class	Images	Instances	Box(P)	R	mAP50	mAP50-95
all	1425	15190	0.837	0.762	0.798	0.639
0	1395	8831	0.856	0.893	0.894	0.737
1	1169	4451	0.828	0.875	0.875	0.607
2	219	249	0.863	0.506	0.617	0.529
3	1035	1659	0.803	0.774	0.805	0.684

Tabla 6: Métricas de entrenamiento 2

- Utilizando la perspectiva de las clases padres. Acumulando las mejores performances basadas en características de entidades similares, tendríamos como promedio resultante de las clases 0, 1 y 3: un mAP – H de 0.86 haciéndolo más competitivo con respecto al entrenamiento anterior
- Desde luego que un entrenamiento como este también tendría una mejora donde nos enfocamos en la clase 2, podríamos también hacer una evaluación con respecto a tiempo de entrenamiento entre ambas, desde luego que una aumentación específica para la clase 2 nos daría un mejor performance tanto en Recall (instancias de esta clase), como las otras métricas.

3.3.3. Análisis Comparativo entre Entrenamiento 1 vs Entrenamiento 2

Métrica	Entrenamiento 1	Nuevo Entrenamiento	Mejora/Diferencia
mAP50 (global)	0.771	0.798	**+0.027**
mAP50-95 (global)	0.595	0.639	**+0.044**
Precisión (P) global	0.806	0.837	**+0.031**
Recall (R) global	0.722	0.762	**+0.040**
mAP-H (global)	0.84	0.86	**+0.020**

Tabla 7: Cuadro comparativo entrenamiento 1 y entrenamiento 2

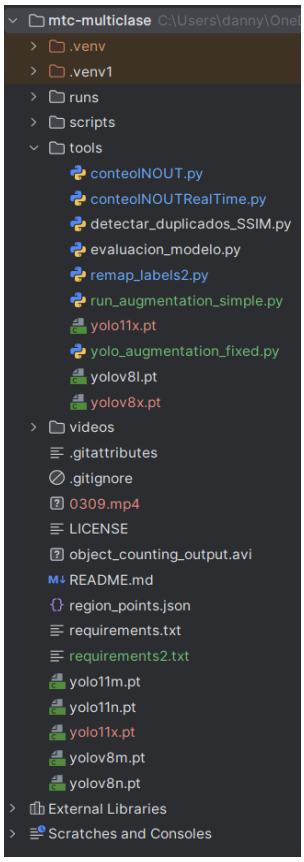
3.4. Evaluación del modelo

3.4.1. Módulos del proyecto:

- Backend Python del proyecto:

Figura 29

Estructura Backend Python del proyecto

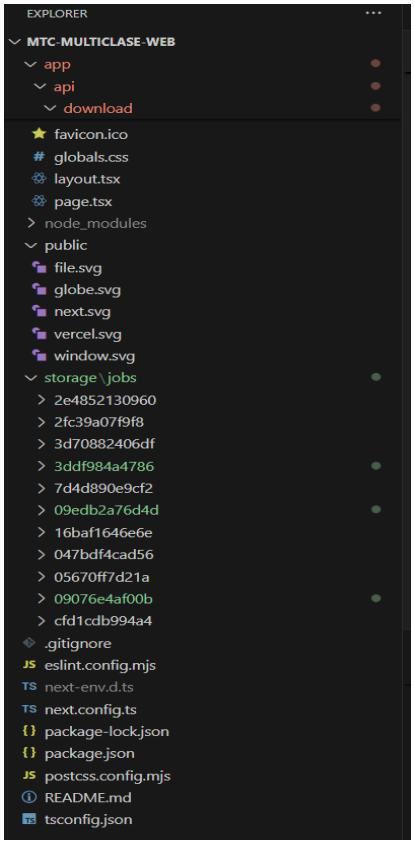


Fuente: Elaboración Propia

- Frontend del Node JS:

Figura 30

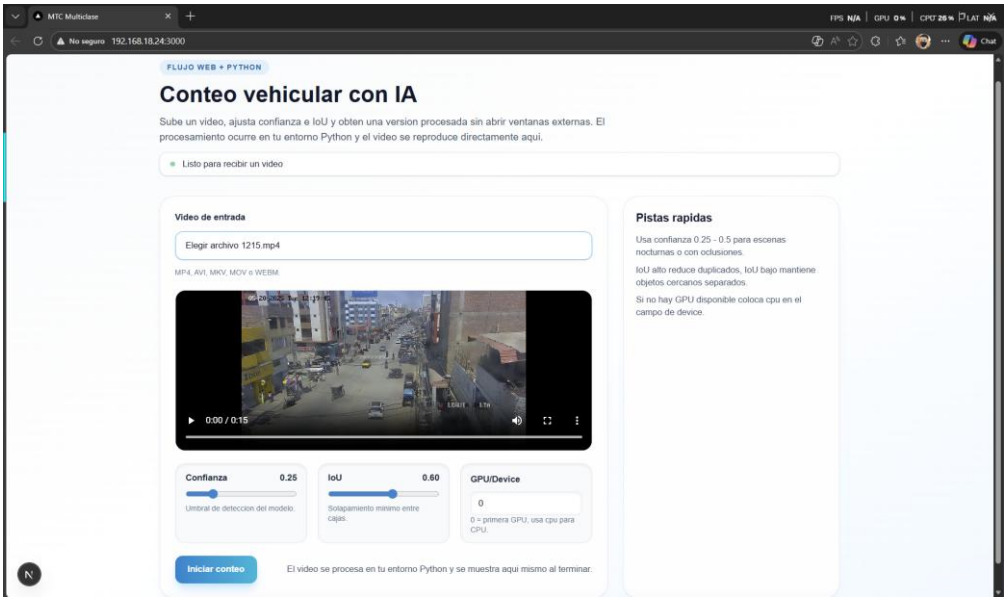
Estructura Frontend del proyecto



- Interfaz de la app web:

Figura 31

Vista de la Interfaz gráfica Web



Fuente: Elaboración Propia

Cada módulo tiene una responsabilidad específica, lo que permite una fácil actualización y depuración. Por ejemplo, route.ts se encarga de la lógica de seguimiento y conteo de vehículos por frames, mientras que page.tsx maneja la interfaz gráfica y la interacción con el usuario.

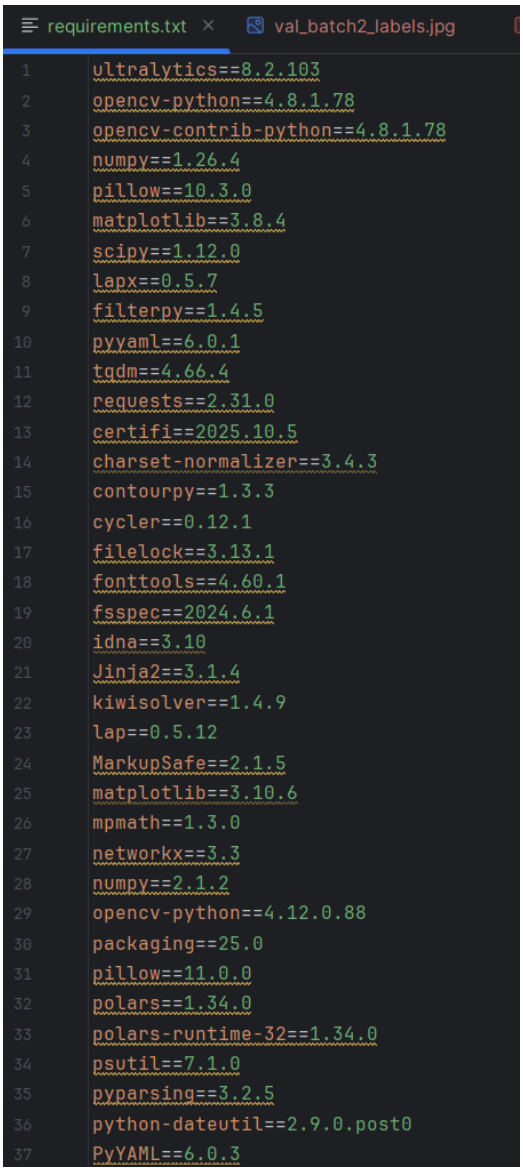
3.4.2 Configuración del entorno

Para garantizar que el proyecto funcione correctamente en diferentes entornos, se configuraron las siguientes herramientas y dependencias:

1. **Python y librerías**:
    - Se utilizó Python tanto 3.11 y 3.13 debido a su soporte para las últimas versiones de las librerías necesarias.
    - Las dependencias se gestionaron mediante un archivo requirements.txt, que incluye:

Figura 32

Dependencias y librerías necesarias en entorno virtual



- Para instalar las dependencias, se ejecutó el comando:

```
install -r requirements.txt
```

2. **Entorno virtual:**
   - Se creó un entorno virtual utilizando venv para aislar las dependencias del proyecto:

```
venv\Scripts\activate
```

- Esto aseguró que las versiones de las librerías fueran consistentes en todos los entornos de desarrollo.

3.5. Flujo de trabajo del proyecto

El flujo de trabajo del proyecto se dividió en varias etapas, cada una con un propósito específico:

3.5.1 Herramientas de desarrollo en PyCharm

PyCharm fue el IDE principal para el desarrollo del proyecto, aprovechando sus características avanzadas:

1. **Depuración**:
   - Se utilizó el depurador integrado de PyCharm para identificar y corregir errores en el código.
   - Los puntos de interrupción (breakpoints) permitieron inspeccionar variables y flujos de ejecución en tiempo real.
2. **Control de versiones:**
   - Se integró Git en PyCharm para gestionar el control de versiones.
   - Cada cambio importante se registró en un repositorio Git, permitiendo un seguimiento detallado del desarrollo.
3. **Gestión de dependencias:**
   - PyCharm facilitó la instalación y actualización de dependencias mediante su interfaz gráfica.
   - También permitió la creación y activación de entornos virtuales directamente desde el IDE.
   - También se configuró para trabajar con ReportLab y OpenCV, proporcionando autocompletado y documentación en línea.

3.5.2 Pruebas en entornos de trabajo

El entorno de trabajo incluyó un proceso riguroso de pruebas y validación:

1. **Pruebas unitarias**:
   - Se realizaron pruebas unitarias para cada módulo (detección, seguimiento, generación de reportes).

- Esto garantizó que cada componente funcionara correctamente antes de integrarse en el sistema completo.

2. **Pruebas de integración:**
   - Se realizaron pruebas de integración para verificar la interacción entre los módulos.
   - Por ejemplo, se verificó que los datos de detección se almacenaran correctamente en la base de datos y que los reportes se generaran sin errores.
3. **Pruebas de rendimiento:**
   - Se evaluó el rendimiento del sistema en diferentes hardware, ajustando parámetros como el tamaño de los fotogramas y la frecuencia de detección.
   - El sistema logró procesar 6-30 FPS en hardware moderado, cumpliendo con los requisitos de tiempo real.

3.5.3 Validación y ejecución

El modelo se validó exhaustivamente para garantizar su precisión y robustez en diferentes escenarios:

1. **Pruebas en videos de tráfico real**:
   - Se probó el modelo en videos de tráfico real en diferentes condiciones (día, tarde mediana iluminación).
   - El modelo demostró ser robusto, detectando vehículos con alta precisión incluso en condiciones adversas.
2. **Pruebas de rendimiento**:
   - Se evaluó el rendimiento del modelo en diferentes hardware, desde GPUs de gama alta hasta CPUs de gama media.
   - El sistema logró procesar 15-30 FPS en hardware moderado, cumpliendo con los requisitos de tiempo real.

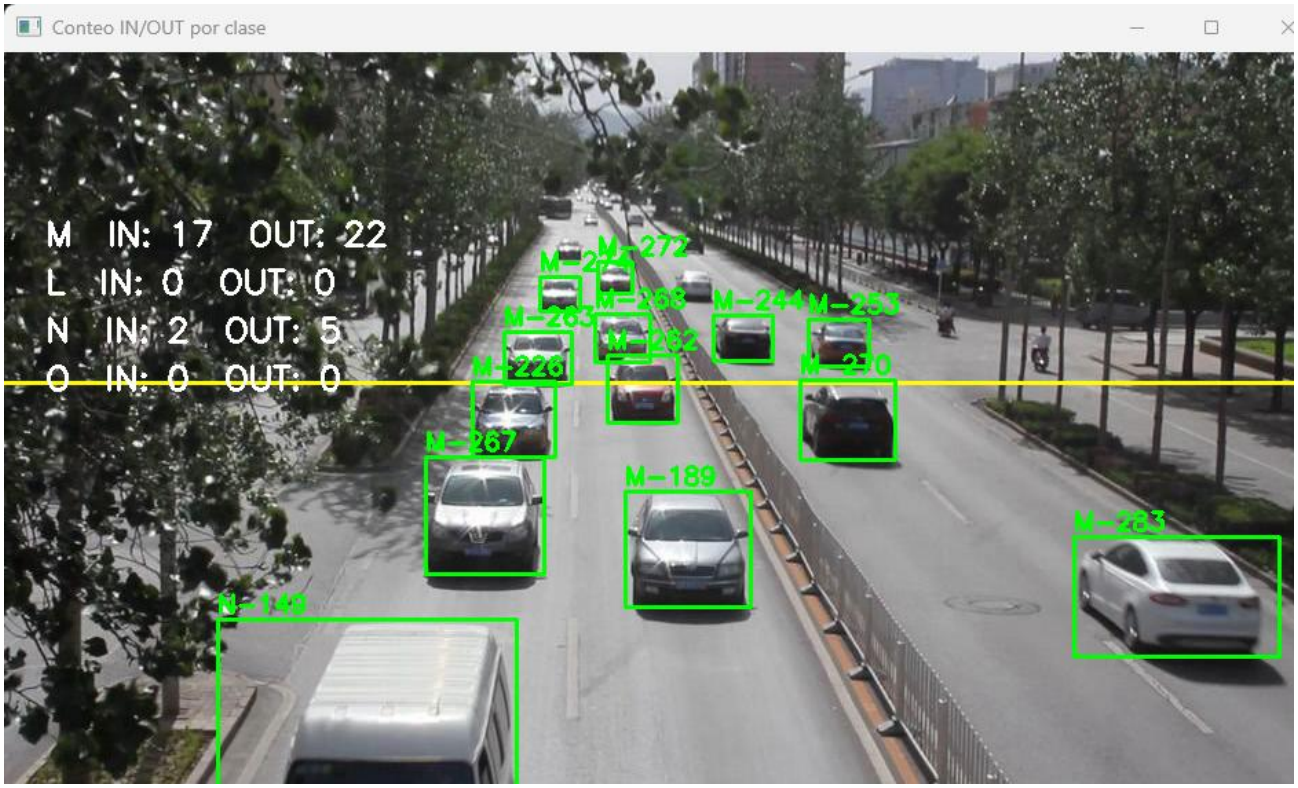
CAPÍTULO 4: EVALUACIÓN DE LOS RESULTADOS

Para evaluar los resultados definimos un video de la Av, Salaverry en el cruce del puente del Parque Infantil, y otros videos de Avenidas con tránsito pesado. Todos ajustados a una resolución de 720p

4.1 Escenario 1: Av. Brasil

Figura 33

Prueba Av. Brasil – Video 2 tramos – 49 segundos



Fuente: Elaboración Propia

Eje Contador (y): 450

4.1.1. Resultados obtenidos

Trazada la línea de referencia el modelo realiza la contabilidad por clase teniendo estos resultados:

Real:

Clase,	IN	OUT
M	18	23
L	1	0
N	2	5
O	0	0

Predicción con modelo:

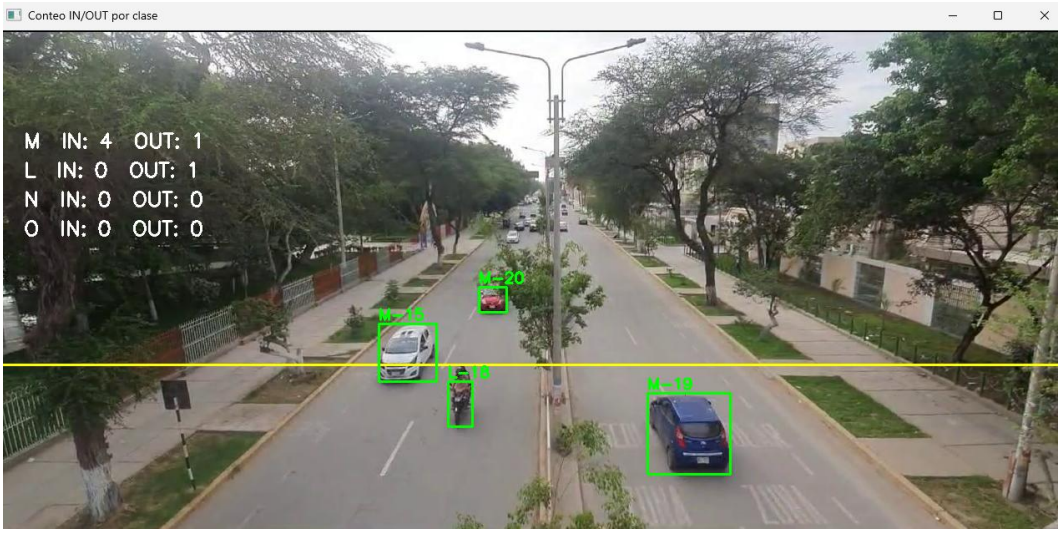
Clase,	IN	OUT
M	18	23
L	0	0
N	2	5
O	0	0

- Basado en la observación las clases en la que el modelo tiene mejor desempeño son las M y N, por tercera la L, que se detectó mas no se pudo contabilizar. Teniendo una precisión del 0.97 del contador con respecto al total de vehículos registrados entre el total de ellos contabilizados de un total de 48 coches **registrados** y un total de 47 **contabilizados**.

4.2 Escenario 2: Av. Salaverry (Puente)

Figura 34

Prueba Av. Salaverry – Video 2 tramos – 49 segundos



Fuente: Elaboración Propia

Eje Contador (y): 500

Trazada la línea de referencia el modelo realiza la contabilidad por clase teniendo etos resultados:

Real:

Clase	IN	OUT	TOTAL CONT/ CLASE
M	13	8	21
L	0	1	1
N	0	0	0
O	0	0	0
TOTAL	13	9	22

Predición con modelo:

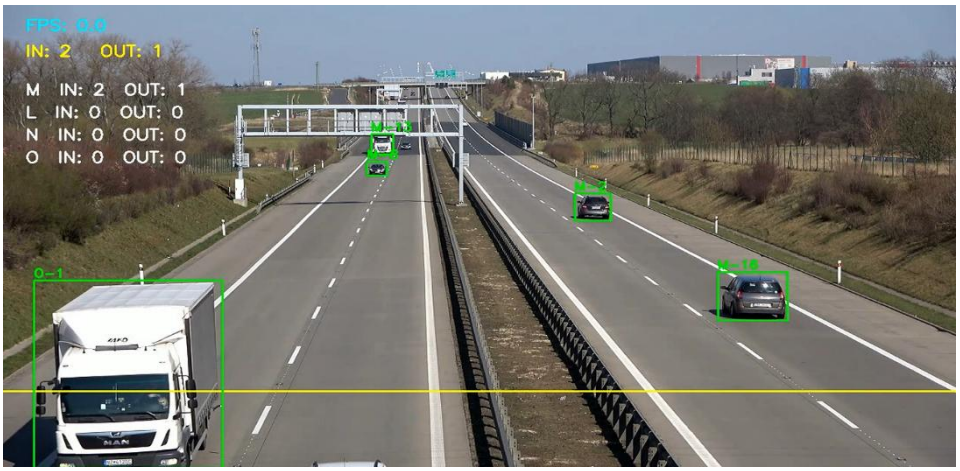
Clase	IN	OUT	TOTAL CONT/ CLASE
M	13	8	21
L	0	1	1
N	0	0	0
O	0	0	0
TOTAL	13	9	22

- Basado en la observación las clases en la que el modelo tiene mejor desempeño son las M y N, por tercera la L, que logró detectar y contabilizar 1. Teniendo una precisión del 0.99 del contador con respecto al total de vehículos registrados entre el total de ellos contabilizados de un total de 22 coches **registrados** y un total de 22 **contabilizados**

4.3 Escenario 3: Carretera Interestatal

Figura 35

Prueba Av. Interestatal – Video 2 tramos – 49 segundos



Fuente: Elaboración Propia

Eje Contador (y): 600

Real:

Clase	IN	OUT	TOTAL CONT/ CLASE
M	5	2	7
L	0	0	0
N	0	0	0
O	2	0	2
TOTAL	7	2	9

Predicción con modelo

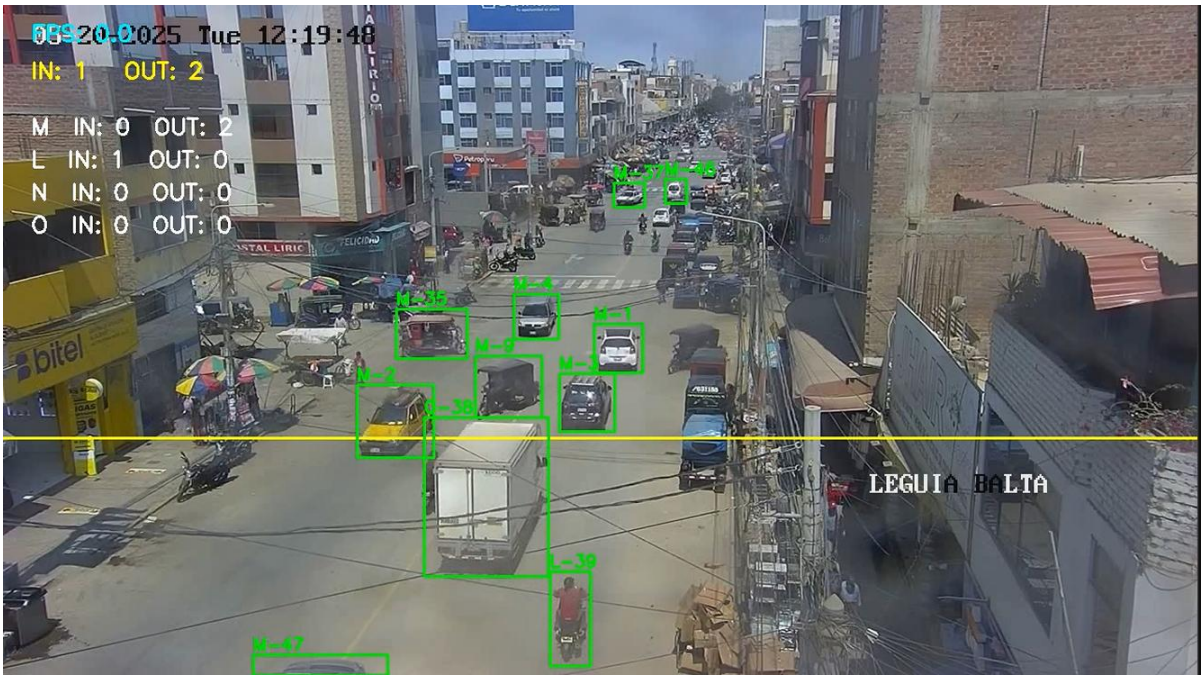
Clase	IN	OUT	TOTAL CONT/ CLASE
M	5	2	7
L	0	0	0
N	0	0	0
O	2	0	2
TOTAL	7	2	9

- Basado en la observación las clases en la que el modelo tiene mejor desempeño son las M y L, por tercera la O, que se detectó mas no se pudo contabilizar. Teniendo una precisión del 0.99 del contador con respecto al total de vehículos registrados entre el total de ellos contabilizados, basados en un total de 9 coches **registrados**, 9 coches **contabilizados.**

4.4 Escenario 4: Av. Balta con Leguía (Escenario con Ruido)

Figura 36

Prueba Av. Balta con Leguía – Video 2 tramos – 49 segundos



Fuente: Elaboración Propia

Eje Contador (y): 465

Real:

Clase	IN	OUT	TOTAL CONT/ CLASE
M	2	4	6
L	3	2	5
N	0	0	0
O	0	1	1
TOTAL	5	7	12

Predicción con modelo

Clase	IN	OUT	TOTAL CONT/ CLASE
M	2	4	6
L	2	1	3
N	0	0	0
O	0	1	1
TOTAL	4	6	10

- Basado en la observación las clases en la que el modelo tiene mejor desempeño son las M y L, aunque L tuvo cierta carencia en su contabilizacion, por tercera la O que el camión que fue registrado también contabilizado. Teniendo una precisión del 0.886 con un total de 12 vehículos **registrados**, sobre un total de 10 **contabilizados**.

Escenario	Ubicación / Condición	Total vehículos reales	Total vehículos contados	Diferencia	Precisión del contador	Clases con mejor desempeño	Observaciones relevantes
1	Av. Brasil	48	47	-1	**0.97**	M, N	La clase L fue detectada pero no contabilizada correctamente. Buen desempeño general en tránsito urbano fluido.
2	Av. Salaverry (Puente)	22	22	0	**0.99**	M, L	Conteo exacto. Escenario controlado con bajo flujo vehicular y buena visibilidad.
3	Carretera Interestatal	9	9	0	**0.99**	M, L	Alto desempeño en vía rápida. La clase O fue detectada y contabilizada
4	Av. Balta con Leguía (Ruido)	12	10	-2	**0.886**	M,O	Escenario complejo por ruido visual, congestión y oclusiones. Dificultad de detección por ruido en la clase L

Tabla 8: Cuadro Resumen Evaluación de Resultados

- Por lo tanto, se puede decir que se evaluaron y estudiaron los resultados en 4 escenarios tanto en avenidas como la Salaverry de la ciudad Chiclayo como en la Avenida Brasil de la ciudad de Lima para cerciorar la precisión del modelo y la eficiencia de su clasificador.

CAPÍTULO 5: DISCUSIÓN DE RESULTADOS

Tenemos entonces definidos como alcances los resultados basados en las pruebas que se realizaron a lo largo de las carreteras del País, podemos definir que para una mejor detención a lo largo del tiempo se requieren o un entrenamiento con los sets de datos incluyendo ruido (cables, charcos, refracción de luz), de otro modo podemos también jugar con la línea de conteo, posicionando esta misma en una posición especifica basada en la locación de la cámara, otro punto también a tomar en cuenta es que se requiere para carreteras interestatales o de largo recorrido, como punto estratégico un posicionamiento de la cámara en medio de estas carreteras, mientras que en las vías de las ciudades si se puede apreciar un mejor desempeño, teniendo como precisión del conteo desde el 0.83 al 0.97 entre 4 pruebas en 4 diferentes escenarios y zonas que se utilizaron como aplicación en campo.

Abuelgasim S. et al. (2024) *Real-time vehicle counting using custom YOLOv8n and DeepSORT for resource-limited edge devices.* Los autores basados en predicción de la calidad del aire hasta la optimización de sistemas energéticos y el conteo de vehículos en tiempo real utilizaron el modelo preentrenado YOLOv8n, el sistema propuesto logra una precisión media de detección (mAP) del 97.5% y una precisión en el conteo de vehículos del 96.8%, debido a la calidad de luz y vista de la carretera, y la disposición en las zonas correspondientes. Lo que comparado a la presente investigación también enfrentada a la misma problemática de conteo el sistema logro una performance de (mAP) de 0.86 en clases M,N,L y O, si bien en el papel el mAP es menor, también esta tesis plantea una clasificación basada en rendimientos, basadas en metodologías + Datasets Customizados, Clases Padres – Hijas, posterior a ello basado en cuatro escenarios teniendo un rendimiento promedio del 97%

Neamah et al. (2023) *Real-time Traffic Monitoring System Based on Deep Learning and YOLOv8.* Los autores presentan un sistema avanzado de monitoreo de tráfico en tiempo real basado en técnicas de aprendizaje profundo, utilizando el algoritmo YOLOv8. Donde el sistema logra una precisión promedio del 96.58% utilizando técnicas de recalibración y preprocesamiento con un sistema que detecta, clasifica, cuenta y estima la velocidad y el tamaño de los vehículos orientado a cámaras de detección y seguimiento, 97.54% en

conteo de vehículos, muy comparable a nuestra investigación que posee una precisión del 97% realizado en el escenario de la avenida Salaverry del con respecto a contabilización.

Chowdhury et al. (2022) *Vehicle Detection and Classification Using Deep Neural Networks*
El autor presenta un sistema automático para la detección y clasificación de vehículos en entornos urbanos, utilizando redes neuronales profundas. Este enfoque responde a la necesidad de gestionar el tráfico de manera eficiente, el sistema emplea modelos de deep learning como VGG16, VGG19 y YOLOv5, aplicando técnicas de transferencia de aprendizaje para mejorar la identificación y clasificación de vehículos en imágenes teniendo como mAP máximo del modelo Yolov5 con un valor del 83,02% efectuándolos con 15 clases, situándonos así en un escenario de comparativo vs el nuestro que tiene un mAP del 84,01%

CAPÍTULO 6: CONCLUSIONES

- Se elaboraron conjuntos de datasets basados en imágenes de Kaggle, Roboflow y Videos de Prueba tomados en las avenidas de la ciudad de Chiclayo, tanto como pruebas en otras ciudades de la Nación como la Av. Brasil ubicada en Lima. Unas 2821 imágenes de un dataset personal resultantes de la metodología de extracción de imágenes con videos mp4.
- Se procesó y entrenaron dando como mejores métricas de rendimiento el modelo yolov11x con técnicas de preprocesado como aumento de contraste, giro de -15°, eliminación de duplicidad de imágenes, balanceo de clases, tanto como la relación entre el escenario establecido como lo es la clasificación del MTC en donde se llegan a relacionar estas detecciones.
- Se realizaron 2 escenarios de entrenamientos con los modelos yolov11 y yolov8, con una serie de parámetros, basándonos en las arquitecturas yolov8m y yolov11x; donde yolov11 dió los mejores resultados con un 0.86 de mAP frente a un 0.84 del yolov8, optando a nuestro favor utilizar parámetros con mejor calidad de imagen (512px vs 1024px) y cantidad de épocas (100 vs 200) respectivamente por entrenamiento y device CUDA, basado en la gráfica Nvidia 12GB DE VRAM.
- Se desarrolló una aplicación Web basada en Next Js y Python que contiene el script del proyecto relacionada a una interfaz web que me permite tales acciones como subir Archivo, descargar Inferencia-Detección y Descargar CSV de Resultados en la detección data.

CAPITULO 7: RECOMENDACIONES

- Como posibles mejoras se recomienda en nuevos eventos de entrenamiento aplicar eliminación de ruido, como de cables antenas, además en las inferencias se puede implementar ajuste de hiperparámetros, IoT y confianza para una precisión de contador mucho más alta frente a escenario desfavorables, a su vez, implementación manual y operativa de muchos labels de clases de motos y camiones de carga para una distinción muy minúscula, para mejorar diferenciación de su respectiva función (transporte de cargas) de sus características.
- Asu vez una alternativa para poder mejorar el performance de las métricas en el segundo entrenamiento como ya lo habíamos dicho sería aumentar el set de datos de la clase N(Camiones), ya que la similitud con la clase remolques es muy similar ya que vienen a formar parte de un tráiler por lo que, esta similitud requeriría tomas aéreas incluso con drones para evaluar características mínimas entre ambas clases.

8. REFERENCIAS

Barba Guamán, L. R. (2021). Uso de técnicas deep learning para reconocimiento de objetos en áreas rurales [Tesis doctoral, Universidad Politécnica de Madrid]. Escuela Técnica Superior de Ingeniería de Sistemas Informáticos. Doctorado en Ciencias y Tecnologías de la Computación para Smart Cities.

Chowdhury, S., Ifty, J., & Khan, R. (2022). Detección y clasificación de vehículos mediante redes neuronales profundas. Conferencia Internacional sobre Ingeniería de Software y Ciencia de Servicios. https://doi.org/10.1109/IEIT56384.2022.9967885

Coanqui Apaza, F. Y., Estofanero Yanapa, R. F., & Mamani Condori, H. W. (2022). Aplicación de inteligencia y visión artificial para la obtención del aforo vehicular. Universidad Peruana Unión.

Gaur, K., Dhakar, J., Singh, S., & Khosla, A. K. (2023). Nighttime rainy season traffic analysis: Vehicle detection, tracking, and counting with YOLOv8 and DeepSORT. Journal of Innovative Image Processing, 5(3), 214-228. https://doi.org/10.36548/jiip.2023.3.001

Gómez, J., & Ramos, P. (2023). Estrategias de pre y postprocesado en deep learning para problemas multiclase en el ámbito de la seguridad y la biodiversidad. Universidad de Granada.

Huang, Y. (2024). Aplicación de modelos YOLO-NAS y YOLOv8 en sistemas seguimiento de múltiples objetos (Trabajo Fin de Máster). Universidad Politécnica de Madrid.

IBM. (2021, abril 12). ¿Qué es Deep Learning? https://www.ibm.com/es-es/topics/deep-learning

Kosuri, N. B., & Manne, S. (2023). An automatic student attendance monitoring system using an integrated HAAR cascade with CNN for face recognition with mask. Traitement du Signal, 40(2), 743-749. https://doi.org/10.18280/ts.400234

Kotsoglou, K. N., & Oswald, M. (2020). The long arm of the algorithm? Automated facial recognition as evidence and trigger for police intervention. Forensic Science

International: Synergy, 2, 86-89. https://doi.org/10.1016/j.fsisyn.2020.01.002

Kranthi Kumar, K., Kasiviswanadham, Y., Indira, D. V. S. N. V., Palesetti, P. P., & Bhargavi, C. V. (2021). Criminal face identification system using deep learning algorithm multi-task cascade neural network (MTCNN). Materials Today: Proceedings. https://doi.org/10.1016/j.matpr.2021.06.373

Lin, H., Yuan, Z., He, B., Kuai, X., Li, X., & Guo, R. (2022). A deep learning framework for video-based vehicle counting. Frontiers in Physics, 10, 829734. https://doi.org/10.3389/fphy.2022.829734

Martínez Samper, A. J. (2022). Diseño e implementación de una metodología basada en inteligencia artificial capaz de contar y clasificar vehículos en los accesos de aeropuertos y de reconocer vehículos aeroportuarios (Trabajo de fin de máster, Universidad Politécnica de Valencia). Recuperado de https://m.riunet.upv.es/handle/10251/184351

Montoya Correa, O. (2020). Proyecto de automatización de aforamiento vehicular. Universidad de Antioquia, Facultad de Ingeniería, Departamento de Ingeniería Electrónica y Telecomunicaciones.

Neamah, S. B., & Karim, A. A. (2023). Real-time traffic monitoring system based on deep learning and YOLOv8. ARO: The Scientific Journal of Koya University, 11(1), 141-148. http://dx.doi.org/10.14500/aro.11327

Redmill, K. A., Yurtsever, E., Mishalani, R. G., Coifman, B., & McCord, M. R. (2023). Automated traffic surveillance using existing cameras on transit buses. Sensors, 23(11), 5086. https://doi.org/10.3390/s23115086

Reddy, N. V., Priya, P., Aryan, S., & Ajay, P. (2023). Crime detection system with machine learning using OpenCV, YOLO and CNN. Proceedings of the 14th International Conference on Computing Communication and Networking Technologies (ICCCNT), 1-7. https://doi.org/10.1109/ICCCNT56998.2023.10306367

Romero Perdomo, Y. A., & Rojas Beltrán, M. S. (2020). Visión artificial para el reconocimiento del tráfico vehicular [Tesis de pregrado, Universidad Piloto de Colombia]. Facultad de Ingeniería, Programa de Ingeniería Mecatrónica.

Saadeldin, A., Rashid, M. M., Shafie, A. A., & Hasan, T. F. (2024). Real-time vehicle counting using custom YOLOv8n and DeepSORT for resource-limited edge devices. TELKOMNIKA Telecommunication Computing Electronics and Control, 22(1), 104-112.

Shanthi, K. G., Sivalakshmi, P., & Sesha Vidhya, S. (2021). Smart drone with real time face recognition. Materials Today: Proceedings. https://doi.org/10.1016/j.matpr.2021.07.214

Xie, H., Xiao, Z., Liu, W., & Ye, Z. (2023). PVNet: A used vehicle pedestrian detection tracking and counting method. Sustainability, 15(14326). https://doi.org/10.3390/su151914326

Ultralytics. (2023). *Entrenamiento de modelos con Ultralytics YOLO – Train settings*. En *Documentación de Ultralytics YOLO*. Recuperado el 5 de febrero de 2026, de https://docs.ultralytics.com/es/modes/train/#train-settings

Ultralytics. (2023). *Performance metrics deep dive*. Documentación de Ultralytics YOLO. Recuperado el 5 de febrero de 2026, de https://docs.ultralytics.com/guides/yolo-performance-metrics/#introduction

Ultralytics. (2023). *Performance metrics deep dive: Visual outputs*. Documentación de Ultralytics YOLO. Recuperado el 5 de febrero de 2026, de https://docs.ultralytics.com/es/guides/yolo-performance-metrics/#visual-outputs

9. APÉNDICES

APENDICE A: Código de la estructura Backend – MTC multiclase
GitHub: https://github.com/CarlosJaradev/mtc-multiclase2

APENDICE B: Codigo de la estructura Fronted -- MTC-multiclase-web
GitHub: https://github.com/CarlosJaradev/mtc-multiclase