UNIVERSIDAD SANTO TOMÁS
SECCIONAL TUNJA

FACULTAD DE INGENIERÍA ELECTRÓNICA

APLICACIÓN DE MACHINE LEARNING EN SISTEMAS DE MENSAJERÍA VARIABLE (SMV) CON VISUALIZACIÓN DINÁMICA DE SEÑALES DE TRÁNSITO BASADAS EN DETECCIÓN DE VEHÍCULOS

Trabajo de grado que presenta

MARCO RODRIGO PÉREZ RAMÍREZ

Para obtener el título profesional de

INGENIERO ELECTRÓNICO

Tunja Boyacá- Colombia
Julio - 2025

UNIVERSIDAD SANTO TOMÁS
SECCIONAL TUNJA

FACULTAD DE INGENIERÍA ELECTRÓNICA

APLICACIÓN DE MACHINE LEARNING EN SISTEMAS DE MENSAJERÍA VARIABLE (SMV) CON VISUALIZACIÓN DINÁMICA DE SEÑALES DE TRÁNSITO BASADAS EN DETECCIÓN DE VEHÍCULOS

Trabajo de grado que presenta

MARCO RODRIGO PÉREZ RAMÍREZ

Director del trabajo de Grado:
Ingeniero Camilo Ernesto Pardo Beainy

Codirector del trabajo de Grado:
Ingeniero Edgar Andrés Gutiérrez Cáceres

Tunja Boyacá - Colombia
Julio - 2025

Exoneración de Responsabilidades

Declaro que el contenido del presente trabajo de grado, titulado "APLICACIÓN DE MACHINE LEARNING EN SISTEMAS DE MENSAJERÍA VARIABLE (SMV) CON VISUALIZACIÓN DINÁMICA DE SEÑALES DE TRÁNSITO BASADAS EN DETECCIÓN DE VEHÍCULOS", es de mi total autoría y responsabilidad. Exonero de cualquier tipo de responsabilidad directa o indirecta a la Universidad Santo Tomás, a la Facultad de Ingeniería Electrónica, así como a sus docentes, directivos y asesores, por las ideas, resultados, desarrollos, interpretaciones o conclusiones aquí presentadas, dado que han sido construidas en el marco de mi pasantía empresarial con base en mi criterio técnico y académico como estudiante de Ingeniería Electrónica. Garantizo que este trabajo ha sido elaborado de forma íntegra y autónoma, siguiendo principios éticos y profesionales, y que el contenido aquí consignado responde exclusivamente al ejercicio académico que he realizado durante el desarrollo del proyecto.

Dedicatoria

Dedico este trabajo, con profunda gratitud y respeto, a quienes han sido parte fundamental de este camino académico, personal y profesional.

A mi familia, por ser mi mayor fortaleza. A mis padres, por su amor incondicional, sus sacrificios silenciosos y su apoyo incansable, por enseñarme que los sueños se alcanzan con esfuerzo, paciencia y valores firmes. A cada uno de mis seres queridos, gracias por creer en mí y por impulsarme siempre a dar lo mejor de mí. Este logro es tan suyo como mío.

A las instituciones que me acogieron y me permitieron crecer: a la Universidad Santo Tomás, por formarme como ingeniero con principios éticos, humanos y científicos; y a la empresa IT Vial S.A.S por brindarme el espacio, los recursos y la confianza para llevar este proyecto a la realidad. Su respaldo fue clave para la aplicación práctica de mis conocimientos y el desarrollo de esta propuesta.

A los docentes que dejaron huella en mi proceso formativo, no solo por sus enseñanzas técnicas, sino también por su calidad humana, por su orientación, exigencia y acompañamiento constante. A mis compañeros de carrera, por compartir no solo clases y retos, sino también aprendizajes, errores y momentos que recordaré por siempre.

finalmente, lo dedico a mí mismo. Por mi perseverancia, por superar cada obstáculo, por mantener la convicción incluso cuando el camino se hizo cuesta arriba, porque detrás de cada conocimiento adquirido, cada noche de desvelo y cada prueba fallida, hubo un compromiso conmigo mismo y con el sueño de convertirme en ingeniero electrónico.

Agradecimientos

Quiero expresar mi más sincero agradecimiento a todas las personas e instituciones que hicieron posible la realización de este proceso de formación como Ingeniero Electrónico.

A la Universidad Santo Tomás, en especial a la Facultad de Ingeniería Electrónica, por brindarme una formación sólida, integral y basada en principios éticos y humanistas.

A la empresa IT Vial S.A.S por abrirme las puertas, confiar en mis capacidades y facilitar los medios para llevar a cabo este proyecto. Su compromiso con la innovación tecnológica y su acompañamiento constante fueron fundamentales para el desarrollo y la validación del sistema propuesto.

Extiendo un agradecimiento especial a mis directores de proyecto, el Dr. Camilo Ernesto Pardo Beainy y el Dr. Edgar Andrés Gutiérrez Cáceres, por su orientación académica, su exigencia profesional y su disposición permanente para guiar cada etapa de este proceso con criterio y claridad.

Agradezco también a todos los docentes que, a lo largo de mi formación, aportaron sus conocimientos y experiencia para fortalecer mi perfil como ingeniero. Sus enseñanzas han sido clave en cada decisión técnica tomada durante el proyecto.

Finalmente, a mi familia. Gracias por su amor incondicional, su paciencia, su confianza en mí y por acompañarme en cada etapa de este proceso. Su apoyo constante ha sido el motor que me impulsó a avanzar con determinación.

Índice general

Índice de figuras

Resumen

En el contexto actual de las ciudades en Colombia, el auge acelerado del número de vehículos ha originado serios desafíos relacionados con la movilidad, un aumento en los accidentes de tráfico y deficiencias en la comunicación en las vías. Los sistemas de mensajería convencionales no pueden ajustarse de forma dinámica a las condiciones reales del tráfico, dado que carecen de mecanismos para la detección y análisis en tiempo real. Esta limitación reduce su eficacia y afecta la prevención de accidentes viales. Por lo tanto, se hace necesario adoptar tecnologías inteligentes que optimicen la señalización y la regulación del tráfico urbano, fundamentándose en datos reales del entorno.

Este proyecto presenta una solución innovadora a través de la fusión de visión artificial, con sistemas de mensajería variable inteligentes (SMV). Estos sistemas son capaces de identificar y clasificar vehículos en tiempo real mediante modelos de detección de objetos. Los datos recopilados se utilizan para modificar automáticamente los mensajes en los SMV, lo que permite mostrar señales preventivas, reglamentarias e informativas de manera dinámica, en función del comportamiento del tráfico. Además, se crea una interfaz gráfica de monitoreo que facilita la vigilancia en tiempo real del flujo vehicular, el rendimiento del sistema, la gestión de modelos de detección optimizados y la consulta del historial de señales mostradas.

La validación del sistema se lleva a cabo mediante pruebas de campo controladas, donde se evalúan parámetros como la precisión en la detección, la velocidad de respuesta del sistema y la eficacia en la proyección de los mensajes. Esta propuesta tiene un gran potencial para impactar positivamente en la seguridad vial y mejorar el manejo del tráfico. Asimismo, es escalable y puede implementarse en otras áreas urbanas del país, a fin de alinearse con las exigencias actuales de modernización tecnológica de la infraestructura vial en Colombia.

Palabras Clave: Machine Learning, Saneles de Mensajería Variable (SMV), Visión Artificial, Detección de Vehículos, Señalización Inteligente, Seguridad Vial, Interfaz Gráfica de Monitoreo.

Abstract

In the current context of Colombian cities, the rapid surge in the number of vehicles has led to serious challenges related to mobility, an increase in traffic accidents, and deficiencies in roadside communication. Conventional messaging systems cannot dynamically adjust to real-time traffic conditions due to their lack of mechanisms for real-time detection and analysis. This limitation reduces their effectiveness and negatively impacts the prevention of road accidents. Therefore, there is a clear need to adopt intelligent technologies that optimize urban traffic signaling and regulation, based on real-world environmental data.

This project presents an innovative solution through the fusion of computer vision with intelligent variable messaging systems (VMS). These systems are capable of identifying and classifying vehicles in real time using object detection models. The collected data is then used to automatically modify messages on the VMS, allowing for the dynamic display of preventive, regulatory, and informational signs based on traffic behavior. Additionally, a graphical monitoring interface is created to facilitate real-time surveillance of vehicular flow, system performance, management of optimized detection models, and consultation of the history of displayed signs.

System validation is carried out through controlled field tests, where parameters such as detection accuracy, system response speed, and message projection effectiveness are evaluated. This proposal has great potential to positively impact road safety and improve traffic management. Furthermore, it is scalable and can be implemented in other urban areas across the country, aligning with the current demands for technological modernization of road infrastructure in Colombia.

Keywords: Machine Learning, Variable Message Systems (VMS), Computer Vision, Vehicle Detection, Intelligent Signaling, Road Safety, Monitoring Interface.

Capítulo 1

Introducción

En la actualidad, las ciudades de Colombia enfrentan serios retos en cuanto a movilidad, seguridad en las vías y la eficiencia del manejo del tránsito. El rápido incremento del parque automotor, la expansión urbana desorganizada y la limitada infraestructura vial han llevado a un notable aumento en la congestión y los accidentes de tránsito. Estas cuestiones no solo afectan la calidad de vida de las personas, sino que también representan un peligro para la seguridad pública y originan altos costos tanto económicos como ambientales. En este marco, los sistemas de señalización vial convencionales han mostrado ser inadecuados para adaptarse a las cambiantes dinámicas urbanas. Los sistemas de mensajería tradicionales, ubicados en diferentes carreteras del país, funcionan con esquemas fijos o programación básica que no tienen en cuenta las condiciones de tráfico en tiempo real. Esta falta de flexibilidad limita su efectividad al momento de prevenir situaciones peligrosas o de informar de manera eficiente a conductores y peatones, afectando su función como herramienta de control y prevención vial. Ante esta situación, tecnologías como la inteligencia artificial, el aprendizaje automático y la visión artificial se presentan como elementos fundamentales para la modernización de la infraestructura vial, permitiendo la identificación de vehículos en tiempo real, facilitando así la creación de sistemas de señalización que sean inteligentes, dinámicos y adaptativos.

Este proyecto propone una solución innovadora que integra la visión artificial con sistemas de mensajería variable (SMV), los cuales tienen la capacidad de actualizar automáticamente los mensajes en función del comportamiento del tráfico. Adicionalmente, se implementa una interfaz gráfica de monitoreo que proporciona tanto a operadores como a usuarios una visualización clara del sistema en tiempo real, incluyendo la detección de vehículos, proyección de señales de tránsito, el desempeño del modelo de inteligencia artificial y un historial accesible de eventos registrados. Esta herramienta mejora significativamente la capacidad de gestión del sistema, aumentando su utilidad y valor como apoyo en la toma de decisiones en entornos de movilidad urbana, asimismo, la propuesta no solo tiene un valor técnico, sino que también se alinea con los objetivos de modernización vial en Colombia, siendo una solución que puede escalar, replicarse y que presenta un costo relativo bajo. Este documento detalla el desarrollo completo del sistema, desde la identificación del problema y los fundamentos teóricos hasta la metodología utilizada, los resultados alcanzados y las conclusiones, con el propósito de ofrecer una herramienta tecnológica efectiva para optimizar la movilidad urbana en el país.

1.1. Planteamiento del Problema

1.1.1 Contextualización del problema

En las últimas décadas, el rápido aumento del número de vehículos en las principales ciudades de Colombia ha modificado de manera significativa las dinámicas de la movilidad urbana. Este incremento continuo en la cantidad de automóviles, junto con procesos de urbanización caóticos y una infraestructura vial que no ha progresado a la misma velocidad, ha ocasionado un deterioro notable en la calidad del tránsito. Las congestiones urgen, el incremento en los tiempos de viaje, la contaminación ambiental y la alarmante frecuencia de accidentes viales son reflejos de un sistema de transporte abarrotado y ineficaz. En este panorama, la correcta señalización vial, que debe ser clara, propicia y flexible, se convierte en un elemento esencial para mejorar la seguridad en las carreteras y facilitar el tráfico vehicular en entornos urbanos que se tornan cada vez más complejos.

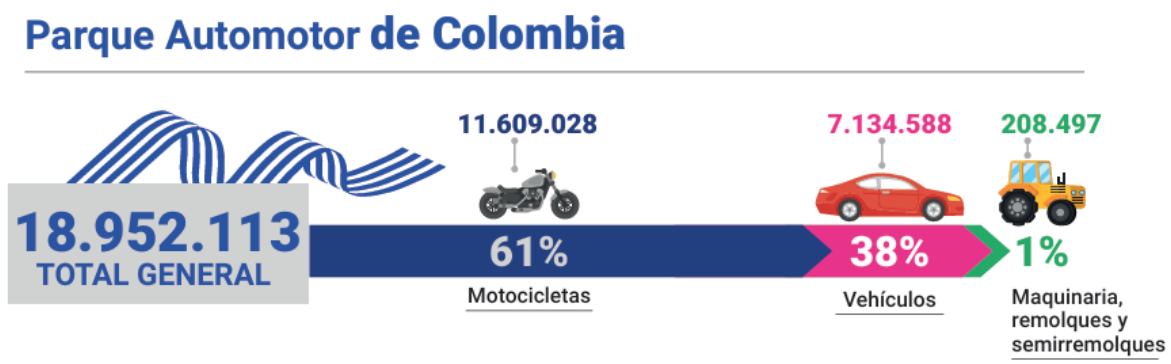

Figura 1.1.1: *Crecimiento Parque Automotor de Colombia, 2023*

1.1.2 Limitaciones de los sistemas actuales de señalización

En el contexto actual, los métodos tradicionales de señalización en las vías, como los paneles de mensajería variable típicos, se muestran inadecuados ante las dinámicas urbanas contemporáneas. Su naturaleza estática y la incapacidad de adaptarse en tiempo real restringen su eficacia para prevenir accidentes o informar adecuadamente a los conductores. Frecuentemente, estos dispositivos no incorporan tecnologías que les permitan percibir el entorno y así interpretar el tráfico en tiempo real. Esto limita la capacidad de difundir mensajes relevantes en situaciones tales como atascos, accidentes o comportamientos inusuales en la circulación. Esta falta de conexión entre la realidad del entorno y la información presentada disminuye la efectividad de la señalización y la confianza que los automovilistas depositan en estos sistemas. En consecuencia, su utilidad como recurso para mejorar la seguridad vial y optimizar la gestión del tráfico no se aprovecha plenamente.

Figura 1.1.2: *Panel Mensajeria Variable Tradicional*

1.1.3 Necesidad de tecnologías inteligentes

Frente a esta problemática, tecnologías emergentes como la inteligencia artificial, el aprendizaje automático (Machine Learning) y la visión por computador se consolidan como herramientas clave para modernizar la infraestructura vial. La aplicación de modelos de detección de objetos, especialmente mediante algoritmos como YOLO (You Only Look Once), permite identificar y clasificar vehículos y motocicletas con alta precisión, incluso en entornos complejos. Estos enfoques permiten analizar el comportamiento del tráfico en tiempo real y generar respuestas automáticas contextualizadas, que se pueden traducir en mensajes adaptativos proyectados en PMV inteligentes.

1.1.4 Delimitación del problema

Este proyecto se centra en el diseño e implementación de un sistema basado en visión artificial, que integre modelos de Machine Learning con sistemas de mensajería variable. Su objetivo es generar visualización dinámica de señales preventivas, reglamentarias e informativas, adaptadas a las condiciones del tráfico. El enfoque técnico se delimita a la detección y clasificación en tiempo real de vehículos y motocicletas, según lo permitido por los modelos de visión artificial utilizados. Se toma como base el contexto urbano colombiano, considerando la normatividad vial nacional del Ministerio de Transporte (1), Código Nacional de Tránsito (2) y alineándolo con estándares internacionales de señalización inteligente.

1.1.5 Impacto del problema

Si esta necesidad no es atendida, las consecuencias serán evidentes: se mantendrán elevados niveles de accidentalidad, desinformación vial, fallas en la eficiencia del tránsito y mayor dificultad para avanzar hacia modelos de ciudades inteligentes. La señalización fija o programada sin contexto real pierde valor frente a entornos altamente dinámicos, donde las decisiones deben tomarse en segundos. No contar con herramientas adaptativas disminuye la capacidad del sistema vial para responder a emergencias, gestionar el flujo y proteger la vida de los usuarios de la vía.

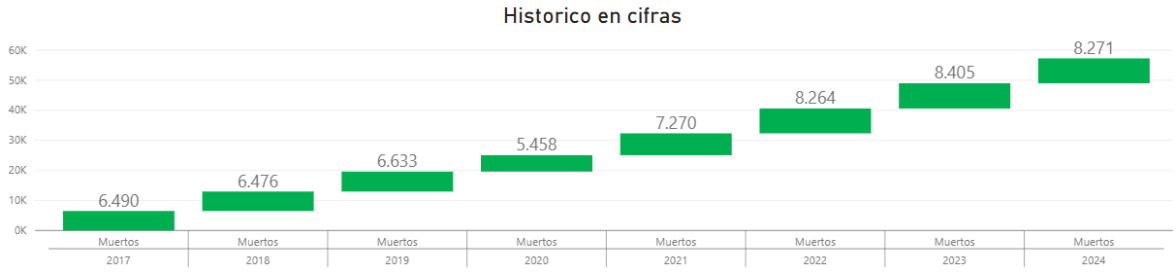

Figura 1.1.3: *Histórico Víctimas ANSV*

1.2. Formulación de Preguntas

¿Cómo mejorar la efectividad de la comunicación vial mediante el uso de SMV con tecnologías avanzadas?

¿Qué tan precisa puede ser la detección de objetos, vehículos y motocicletas con modelos de Machine Learning?

¿Cuál es el impacto en la movilidad urbana al integrar un SMV inteligente en la gestión del tráfico?

1.3. Justificación

El crecimiento acelerado del parque automotor en Colombia, especialmente en zonas urbanas, ha generado múltiples problemáticas asociadas con la movilidad, la seguridad vial, el impacto ambiental y la eficiencia del transporte. Ante este panorama, resulta indispensable el desarrollo de soluciones tecnológicas que permitan modernizar la infraestructura vial y responder de manera adaptativa a las condiciones cambiantes del tránsito. En este contexto, la aplicación de tecnologías como la inteligencia artificial y la visión por computador en sistemas de señalización dinámica ofrece una alternativa innovadora y de alto valor para la gestión del tráfico urbano.

1.3.1 Impacto social

La implementación de un sistema avanzado de mensajería variable, que tenga la capacidad de presentar mensajes en tiempo real según el flujo de vehículos, podría tener un impacto considerable en la disminución de accidentes de tráfico. La entrega oportuna y pertinente de señales reguladoras, de advertencia e informativas ayuda a guiar el comportamiento de los entornos viales, alertar anticipadamente sobre posibles peligros en la carretera y mitigar la incertidumbre durante los viajes. Esto no solo promueve una mayor seguridad en las vías, sino que también refuerza la percepción de los ciudadanos sobre la efectividad de las tecnologías implementadas en el entorno urbano.

1.3.2 Impacto económico

Una mejor organización del tránsito conlleva una disminución de los tiempos de viaje, la optimización del uso del espacio vial y la reducción del consumo de combustible, factores que se traducen en menores costos operativos tanto para los usuarios particulares como para los sistemas de transporte público. Asimismo, la implementación de soluciones tecnológicas avanzadas en la infraestructura vial fomenta la innovación en el sector del transporte, impulsa el desarrollo de nuevas aplicaciones en ingeniería y abre oportunidades para la integración de sistemas inteligentes en ciudades intermedias y capitales regionales, alineándose con los objetivos de modernización urbana del país.

1.3.3 Impacto medioambiental

Una circulación vehicular más fluida y eficiente reduce los niveles de emisión de gases contaminantes, especialmente CO2, óxidos de nitrógeno y material particulado, generados durante los períodos prolongados de espera o congestión. Al permitir un tránsito más ordenado, continuo y adaptativo, los SMV inteligentes aportan a la disminución de la huella de carbono asociada al transporte urbano, contribuyendo con ello a los compromisos ambientales del país en materia de sostenibilidad y mitigación del cambio climático.

1.4. Objetivos

1.4.1. Objetivo General

Desarrollar una solución basada en Machine Learning para la detección de vehículos mediante modelos de visión artificial, integrada con sistemas de mensajería variable (SMV) para la visualización dinámica de señales de tránsito.

1.4.2. Objetivos Específicos

- Implementar algoritmos de visión artificial para la detección y clasificación de vehículos en entornos urbanos.
- Automatizar la publicación dinámica de señales de tránsito en SMV según las condiciones del tráfico detectadas, asegurando la correcta visualización de señales preventivas (SP), reglamentarias (SR) y señales informativas (SI).

- Desarrollar una interfaz de monitoreo que facilite la supervisión del sistema y el análisis de su desempeño en contextos de movilidad urbana.

- Validar el sistema mediante pruebas controladas, evaluando indicadores como precisión de detección, tiempo de respuesta y visualización efectiva en los SMV, con el fin de ajustar los algoritmos y componentes de hardware.

1.5. Organización del Documento

Este documento ha sido organizado con el propósito de presentar de forma clara, coherente y sistemática el desarrollo del proyecto titulado "Aplicación de Machine Learning en sistemas de mensajería variable (SMV) con visualización dinámica de normas de tránsito basadas en detección de vehículos", en el contexto de una pasantía empresarial como opción para concluir la carrera. A lo largo de sus secciones, se aborda el contexto del problema, los fundamentos teóricos, la metodología empleada, el desarrollo técnico, los resultados alcanzados y las conclusiones que emergen del proceso.

En el Primer Capítulo se exponen los elementos fundamentales del proyecto, que incluyen la formulación del problema, la justificación, los objetivos y la motivación general detrás de la propuesta. Este capítulo establece la base del trabajo y determina la necesidad técnica y social de la implementación de sistemas inteligentes en la gestión del tránsito urbano.

El Segundo Capítulo se dedica al estado del arte y al marco de referencia. En este segmento se analiza la literatura científica y tecnológica vinculada con visión artificial, modelos para detección de objetos, señalización vial dinámica y experiencias similares tanto a nivel nacional como internacional. Además, se tratan los principios teóricos y normativos que respaldan la relevancia del sistema propuesto.

En el Tercer Capítulo se detalla la metodología utilizada para llevar a cabo el proyecto. Se describen los procesos de recolección de datos, la formación del modelo, la integración con el hardware de SMV, los protocolos de comunicación que se utilizaron y las pruebas funcionales del sistema. Este capítulo facilita una comprensión paso a paso del diseño e implementación del prototipo.

El Cuarto Capítulo presenta los resultados obtenidos junto con su análisis correspondiente. Se incluyen evidencias visuales del funcionamiento del sistema, métricas de rendimiento (como precisión y tiempo de respuesta) y una evaluación sobre la efectividad del sistema en situaciones controladas. También se discute la viabilidad del proyecto en contextos reales de movilidad urbana.

Por último, el Quinto Capítulo se dedica a las conclusiones y recomendaciones. En este, se reflexiona sobre los logros conseguidos, las limitaciones identificadas y las perspectivas futuras del sistema propuesto, tomando en cuenta su capacidad de escalar, sostenibilidad y su valor tecnológico dentro de la infraestructura vial en Colombia.

Capítulo 2

Estado del Arte y Análisis de Vigilancia Tecnológica

2.1. Estado del Arte

La seguridad vial y la gestión eficiente del tráfico son retos prioritarios ante el aumento de accidentes y la congestión en zonas urbanas. En este contexto, la aplicación de tecnologías como machine learning, visión por computadora y sistemas inteligentes de transporte (ITS) ha cobrado relevancia. El presente estado del arte y análisis de vigilancia tecnológica explora los avances en detección de objetos, reconocimiento de señales, sistemas de mensajería variable inteligentes (SMV), y normas técnicas internacionales. Asimismo, se destaca el impacto ambiental positivo de estas tecnologías, su integración con ciudades inteligentes y la importancia de la ciberseguridad en sistemas de transporte modernos.

2.1.1. Seguridad Vial y Gestión del Tráfico

La Organización Mundial de la Salud (OMS, 2022) ha señalado que cada año mueren aproximadamente 1.3 millones de personas por accidentes de tránsito, la mayoría en zonas urbanas con altos volúmenes de tráfico. Ante esta situación, las soluciones tecnológicas para la prevención, detección y actuación ante riesgos en la vía se consideran prioritarias en planes de desarrollo vial nacional e internacional (3). La gestión inteligente del tráfico, mediante redes neuronales, cámaras y SMV, permite informar a los actores viales en tiempo real sobre condiciones del tráfico, accidentes, obras, restricciones, zonas escolares y peatonales. Estas capacidades se ven reforzadas con la aplicación de modelos de machine learning (ML), que pueden anticipar condiciones y recomendar acciones preventivas.

2.1.2. Tecnología e Innovación en Movilidad

La aplicación de inteligencia artificial en el campo del transporte ha sido impulsada por la necesidad de mejorar la eficiencia y seguridad. Las tecnologías como la visión artificial, redes neuronales convolucionales (CNN) y modelos de detección de objetos han sido integradas a sistemas de transporte inteligentes (ITS) para la clasificación automática de vehículos y el conteo de flujo vehicular. El estudio de Li et al. (2021) en

la revista IEEE Transactions on Intelligent Transportation Systems demostró que el uso de modelos YOLOv4 para detección vehicular en tiempo real presentaba una precisión superior al noventa por ciento en condiciones urbanas complejas. Estas capacidades permiten automatizar la actualización de contenidos en sistemas de mensajería variable. Este enfoque no solo mejora la seguridad y eficiencia en la movilidad, sino que posiciona al proyecto como una solución moderna, escalable y replicable en diferentes regiones del país, integrando perfectamente la innovación tecnológica con la gestión vial. (4)

2.1.3. Señalización Vial y Seguridad del Trafico

Los sistemas de mensajería variable son dispositivos que muestran información dinámica al usuario vial, y forman parte esencial de los ITS modernos. Según el Manual de Señalización Vial del Ministerio de Transporte de Colombia (2020), estos deben cumplir con normativas de visibilidad, lectura y pertinencia. Estudios como el de Benchekroun proponen SMV inteligentes basados en arquitecturas edge computing que reciben datos de sensores de velocidad, condiciones climáticas y cámaras de vigilancia, adaptando los mensajes según el comportamiento detectado. El Manual on Uniform Traffic Control Devices (MUTCD) de EE.UU. regula el diseño de señales variables (VMS), mientras que la norma europea *UNE-EN 12966* establece requisitos técnicos para paneles de mensajería variable. A nivel internacional, ISO 39001:2012 proporciona directrices para reducir accidentes mediante sistemas inteligentes (5). En operación destacan sistemas como SCATS, que mediante sensores ajusta en tiempo real los semáforos para optimizar el flujo vehicular (6)

2.1.4. Normas de Tecnología e Innovación

El reconocimiento automatizado de señales es fundamental para la seguridad vial y sistemas de asistencia al conductor. Este campo se rige por estándares como ISO 14813 para sistemas de transporte inteligente (ITS), IEEE 802.11p (comunicación vehicular inalámbrica) e ISO 39001(Sistema de Gestión de la Seguridad Vial (RTS)). (5). En investigación destacan modelos avanzados como TSD-YOLO, que integra mecanismos de atención para detección robusta en condiciones adversas, y MDDFNet, que utiliza arquitecturas Mamba para identificar señales pequeñas manteniendo procesamiento en tiempo real (7)

2.1.5. Normas de Integración con Smart Cities

Los estándares para ciudades inteligentes y sostenibilidad implementan normas ISO 37120 que establecen indicadores clave para la gestión eficiente del tráfico en ciudades inteligentes, mientras que la norma española UNE 178104 regula la interoperabilidad de los sistemas de señalización en este contexto (5). En materia de impacto ambiental, los sistemas de transporte representan un desafío global debido a emisiones contaminantes, consumo de recursos y contaminación acústica. Los SMV inteligentes ofrecen soluciones concretas, como fomentar la movilidad sostenible (transporte público, bicicletas), reducen la congestión vehicular y emisiones de CO2 mediante la optimización de rutas y promueven prácticas responsables (vehículos eléctricos, carpooling) (8)

2.1.6. Machine Learning en la Detección de Objetos

La detección de objetos en tiempo real es fundamental para la gestión inteligente del tráfico. En este proyecto, se implementan algoritmos de visión por computadora para identificar vehículos, permitiendo que los Sistemas de Mensajería Variable Inteligente (SMV AI) adapten dinámicamente sus mensajes según el flujo vehicular. El modelo YOLOv11 destaca por su capacidad para clasificar vehículos en video en tiempo real, tomando decisiones autónomas sobre qué información mostrar según el tipo, cantidad y velocidad de los vehículos detectados. Sistemas similares, como el Dynamic Message Sign System de Florida (FDOT, 2022), ya utilizan visión artificial para alertar sobre incidentes, condiciones meteorológicas y congestiones.(9). Otras versiones de YOLO también han demostrado eficacia, tales como YOLOv5 donde se implementan ajustes de semáforos según la densidad vehicular detectada (6), YOLOv5 mejorando la precisión en el reconocimiento de matrículas y detección de vehículos (7)

2.1.7. Análisis Comparativo del Modelo de Detección de Vehiculos

Se llevó a cabo un análisis comparativo exhaustivo de las herramientas más relevantes en el campo de la visión por computador. La revisión se centró en métricas clave para aplicaciones en tiempo real, como la velocidad de inferencia, la precisión y la viabilidad de implementación en entornos con recursos computacionales limitados.Se analizaron modelos de arquitectura de detección de objetos de última generación como:

Faster R-CNN: Este modelo, aunque conocido por su alta precisión, presenta una velocidad de inferencia significativamente más baja debido a su proceso de dos etapas (generación de regiones de interés y clasificación). Su complejidad lo hace menos apto para la detección en tiempo real en un sistema dinámico como el propuesto en este proyecto.

SSD (Single Shot Detector): A diferencia de Faster R-CNN, SSD opera en una sola etapa, lo que mejora su velocidad. Sin embargo, en las comparativas de la literatura, versiones livianas como YOLOv11n demuestran un mejor equilibrio entre precisión y velocidad de inferencia, siendo más eficientes en el uso de memoria y procesamiento, lo que es vital para un prototipo.

EfficientDet: Es una familia de modelos escalables que ofrecen un rendimiento notable. No obstante, YOLOv11n fue elegido por su arquitectura optimizada para la detección en dispositivos con recursos limitados. Su rendimiento en cuanto a la relación entre precisión y velocidad, sumado a la extensa documentación y el soporte de la comunidad, lo convierte en una solución superior y más accesible para los fines de este proyecto de grado.

La elección de YOLOv11n se justifica plenamente al ofrecer la mejor combinación de velocidad de inferencia y precisión para una implementación en tiempo real, demostrando ser la herramienta ideal para un sistema que requiere eficiencia operativa y viabilidad técnica.

2.2. Análisis de Vigilancia Tecnológica

Se llevó a cabo una revisión exhaustiva de literatura científica y tecnológica en bases de datos especializadas como Scopus, IEEE Xplore, ScienceDirect y SpringerLink, con el fin de identificar tendencias actuales y desarrollos relevantes en el ámbito del transporte inteligente. La búsqueda se centró en temáticas como el uso de modelos de machine learning para la detección automática de vehículos, el reconocimiento visual de señales de tránsito y la visualización dinámica de información normativa a través de paneles de mensajería variable. Asimismo, se analizaron implementaciones recientes en infraestructura vial inteligente y se consideraron aspectos críticos de ciberseguridad aplicados a sistemas de gestión y comunicación en entornos urbanos. Esta revisión permitió consolidar un marco tecnológico actualizado que fundamenta y orienta el desarrollo de soluciones innovadoras en seguridad vial y movilidad inteligente.

2.2.1. Evolución tecnológica de los sistemas de mensajería variable (SMV)

Los sistemas de mensajería variable han experimentado una transformación radical al incorporar tecnologías y sistemas de percepción avanzada. Los SMV modernos ahora utilizan algoritmos de aprendizaje profundo para adaptar dinámicamente sus mensajes según las condiciones del tráfico en tiempo real, como demuestra el sistema FLIR TrafiBot AI (10), capaz de detectar incidentes y ajustar automáticamente la señalización vial. Esta capacidad se potencia mediante la integración de redes de sensores IoT y cámaras inteligentes que permiten un monitoreo preciso del flujo vehicular, tal como lo implementa Aleatica en sus soluciones de gestión de tráfico (11), donde los datos recopilados son procesados por IA para generar información contextualizada y tomar decisiones operativas optimizadas. Esta convergencia tecnológica ha convertido a los SMV en componentes inteligentes y autónomos de los sistemas de transporte modernos, marcando un hito en la evolución de la infraestructura vial hacia entornos más seguros y eficientes.

2.2.2. Detección de Vehículos mediante Machine Learning

La implementación de técnicas de aprendizaje profundo ha revolucionado la detección vehicular, permitiendo una gestión más eficiente del tráfico en tiempo real. Modelos avanzados como YOLOv5 (12) han demostrado capacidad para identificar vehículos con precisión, facilitando ajustes dinámicos en la señalización que mejoran significativamente la fluidez del tránsito. Un avance clave en este campo es la fusión sensorial (13), que combina datos de múltiples cámaras y sensores para incrementar la precisión en la detección de vehículos y obstáculos. Esta tecnología no solo optimiza la gestión del tráfico convencional, sino que constituye un pilar fundamental para el desarrollo de sistemas de conducción autónoma, donde la identificación precisa y en tiempo real de los elementos viales es crítica. La evolución continua de estos sistemas de detección inteligente está transformando los paradigmas de movilidad urbana, ofreciendo soluciones escalables y adaptables a distintos entornos viales.

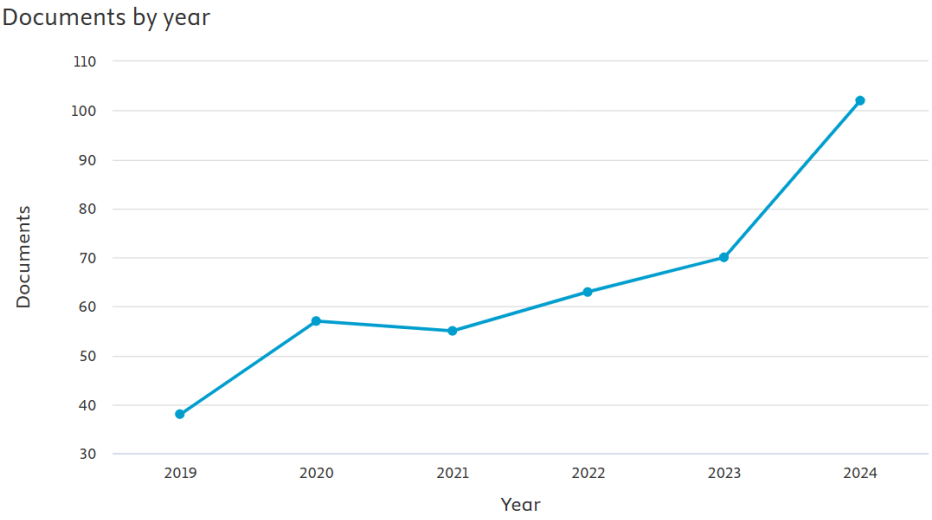

Figura 2.2.1: *Evolución Tecnológica de los Paneles de Mensajería Variable SCOPUS*

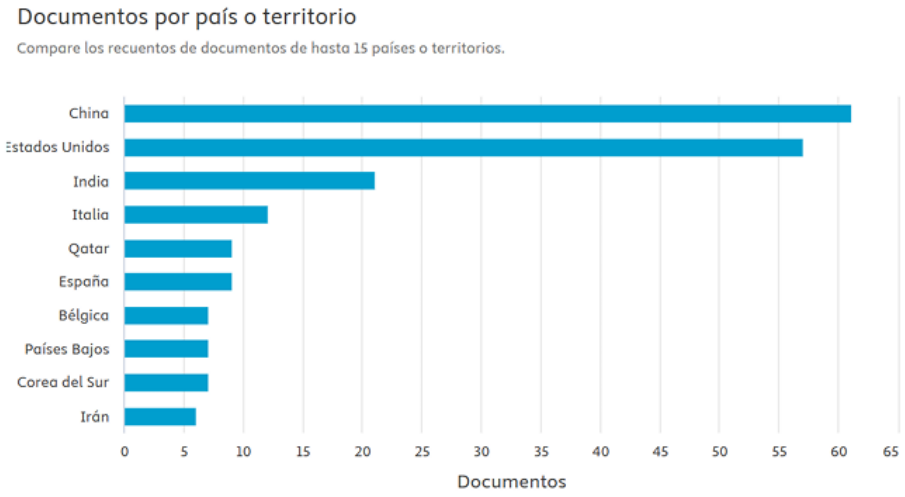

Figura 2.2.2: *Evolución Tecnológica de los Paneles de Mensajería Variable SCOPUS*

2.2.3. Reconocimiento de Señales de Tránsito

El reconocimiento automático de señales de tránsito se ha consolidado como un pilar fundamental para la seguridad vial y los sistemas avanzados de asistencia al conductor. Los avances en visión artificial han permitido desarrollar sistemas capaces de identificar y clasificar señales con alta precisión, mejorando tanto la capacidad de respuesta de los vehículos como de la infraestructura vial inteligente. Esta tecnología adquiere especial relevancia en el contexto de los vehículos autónomos, donde el reconocimiento preciso de señales es determinante para una navegación segura y adaptativa a las normativas viales en tiempo real (14). La evolución de estos sistemas refleja la convergencia entre inteligencia artificial, percepción computacional y movilidad autónoma, marcando un hito en el desarrollo de ciudades inteligentes y transporte del futuro.

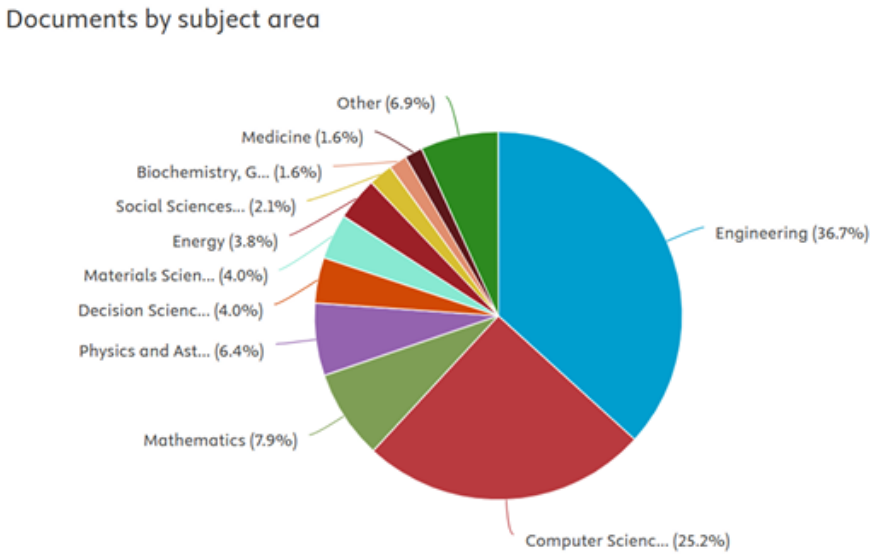

Figura 2.2.3: *Detección de Vehículos Mediante Machine Learning SCOPUS*

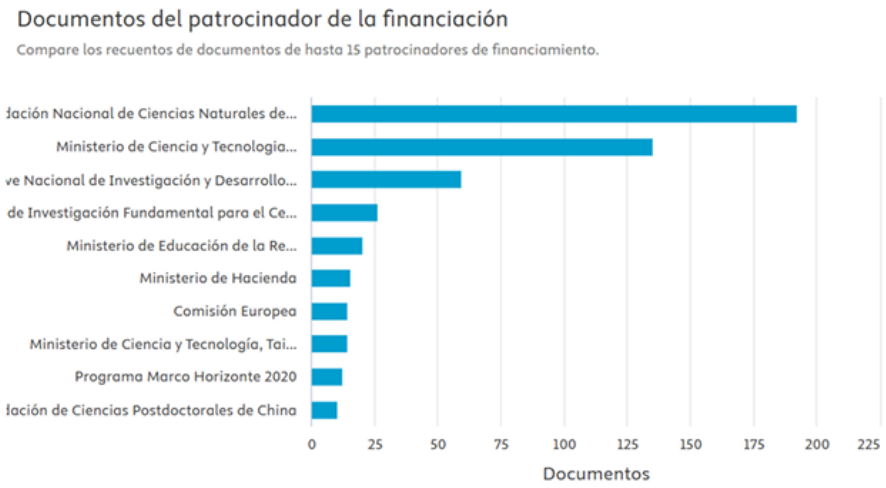

Figura 2.2.4: *Reconocimiento de Señales de Tránsito SCOPUS*

2.2.4. Implementaciones en Infraestructuras Críticas

Los sistemas inteligentes están transformando la gestión de infraestructuras clave como aeropuertos y vías urbanas. Investigadores de la Universidad de Málaga (15) han desarrollado un sistema de videovigilancia con visión por computadora que detecta automáticamente objetos y personas, mejorando la seguridad aeroportuaria sin supervisión constante. En movilidad urbana, el sistema Heimdall (16) utiliza IA para monitorear el tráfico y detectar anomalías en tiempo real, optimizando la respuesta ante incidentes. Estas soluciones demuestran el potencial de la visión artificial y el IoT para mejorar la eficiencia y seguridad en infraestructuras críticas.

2.2.5. Ciberseguridad en Sistemas de Transporte Inteligente

La protección de infraestructuras y datos en sistemas de transporte inteligente requiere enfoques robustos. Los Sistemas de Detección y Respuesta ante Intrusiones (IDPS) [20] monitorean y neutralizan amenazas cibernéticas en tiempo real, protegiendo componentes críticos como señales inteligentes y centros de control. Junto con protocolos avanzados de encriptación (17), garantizan la integridad de las comunicaciones entre dispositivos IoT y sistemas centrales. Estas medidas son especialmente relevantes para aplicaciones V2I (vehículo infraestructura) y sistemas de decisión automatizada, donde vulnerabilidades podrían comprometer la seguridad vial. La implementación combinada de IDPS y encriptación (17) no solo mitiga riesgos, sino que cumple con estándares como ISO 39001 para la gestión de la seguridad vial y sistemas de transporte.

Capítulo 3

Propuesta Metodológica

El proyecto detalla el procedimiento técnico y metódico utilizado para crear un sistema inteligente de señalización vial que se basa en visión artificial y aprendizaje automático. Su propósito es potenciar la eficacia de los sistemas de mensajería variable en el entorno urbano de Colombia. La estrategia implementada combina saberes de electrónica, inteligencia artificial, programación y comunicación industrial, estructurándose en fases que aseguran el seguimiento y la validación de cada parte del desarrollo del proyecto.

La creación del sistema se llevó a cabo en seis etapas secuenciales, cada una diseñada para alcanzar los propósitos generales y específicos del proyecto. Estas etapas se fundamentan en un enfoque de ingeniería aplicado que abarca desde la planificación y la recopilación de datos hasta la validación del sistema en ambientes controlados. El diagrama metodológico ilustra de manera gráfica el flujo de trabajo y su correspondencia con los objetivos establecidos.

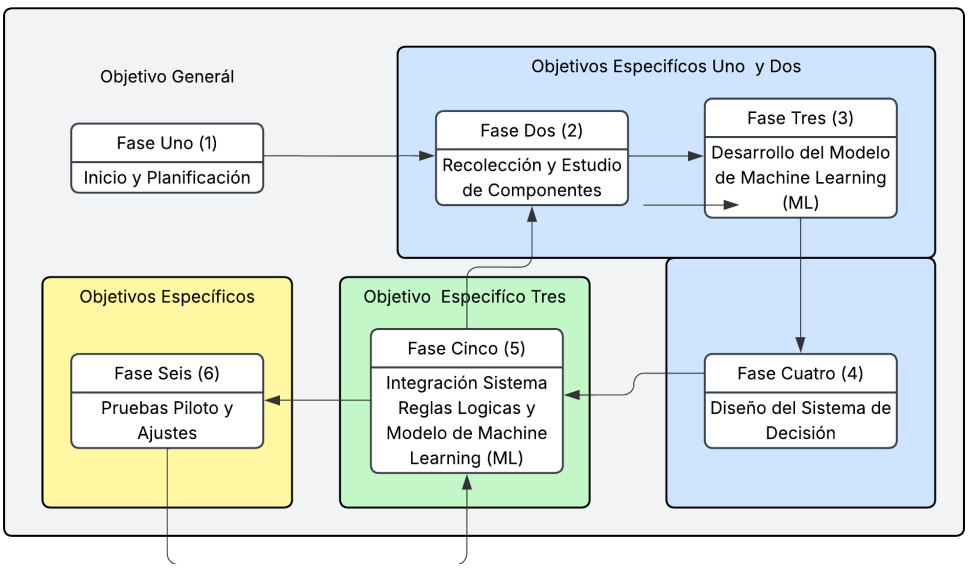

Figura 3.0.1: *Diagrama de Flujo*

3.0.1. Fase 1: Inicio y planificación

La fase inicial abarcó la creación del proyecto, el examen del entorno y la clarificación de los objetivos, el alcance y los requisitos técnicos. Se detectó el problema principal vinculado a la escasa flexibilidad de los SMV convencionales, y se sugirió una solución tecnológica que incluía procesamiento de imágenes en tiempo real, algoritmos para clasificar vehículos y control automatizado de la señalización de tránsito. Asimismo, se definieron los recursos esenciales, el calendario de implementación y la conexión con la empresa IT Vial S.A.S, donde se llevó a cabo la pasantía empresarial.

3.0.2. Fase 2: Recolección y estudio de componentes

Durante esta etapa, se llevó a cabo la elección y el análisis de los elementos de hardware y software necesarios para proceder con el diseño del sistema. Este proceso abarcó la selección de cámaras para la captura de imágenes, plataformas adecuadas para el procesamiento de datos, dispositivos de visualización compatibles con SMV y los protocolos de comunicación requeridos. A su vez, se comenzó el trabajo de recopilación y curaduría de datos visuales, creando un conjunto de datos que incluye imágenes de vehículos tomadas en entornos urbanos del país, las cuales son esenciales para el entrenamiento del modelo de aprendizaje automático.

3.0.3. Fase 3: Desarrollo del modelo de Machine Learning

Una vez estructurado el conjunto de datos, se llevó a cabo el proceso de entrenamiento de un modelo para la detección de objetos, empleando la arquitectura YOLOv11. Se ajustaron los hiperparámetros y se analizó el rendimiento del modelo a través de métricas como la precisión, el recall y el mAP, con el objetivo de lograr un equilibrio adecuado entre exactitud y eficiencia en el uso de recursos computacionales. El modelo final fue perfeccionado para su funcionamiento en tiempo real, garantizando su eficacia en entornos controlados.

3.0.4. Fase 4: Diseño del sistema de decisión y SMV

En esta fase, se estableció la lógica del sistema que facilita la conversión de las detecciones del modelo en acciones específicas sobre el monitor de mensajería. Se creó una arquitectura operativa que enlaza la entrada visual (cámara) con el sistema de procesamiento, las normas de decisión lógica y la visualización del SMV. Cada clase identificada (como coche, motocicleta, camión o autobús) se vincula automáticamente a una señal reglamentaria, preventiva o informativa que se proyecta en tiempo real en pantalla, de acuerdo con la normativa nacional.

3.0.5. Fase 5: Integración del sistema y reglas lógicas

Esta etapa reafirmó la unión de todos los subsistemas involucrados. Se desarrollaron las reglas lógicas requeridas para que el sistema pudiera interpretar de manera apropiada las salidas del modelo y activar las señales adecuadas. Asimismo, se estableció la comunicación entre el módulo de control y el SMV a través del protocolo de comunicación correspondiente (HDMI).

Se llevaron a cabo pruebas internas con el fin de garantizar la estabilidad, sincronización y eficacia del sistema en su totalidad.

Además, se creó una interfaz gráfica de monitoreo diseñada específicamente para respaldar el funcionamiento del sistema en tiempo real. Esta herramienta posibilita la visualización de la señal de video en directo, las detecciones realizadas por el modelo, las clases de vehículos reconocidas y las señales emitidas en el monitor de mensajería variable. La interfaz ofrece también métricas clave del sistema, tales como precisión, latencia, distribución de vehículos detectados, frecuencias y tendencias de proyección, e incluye un módulo de historial que documenta tanto las detecciones como las señales proyectadas, lo que facilita su análisis posterior y la trazabilidad operativa del sistema.

3.0.6. Fase 6: Pruebas piloto y ajustes

Con el propósito de verificar el correcto desempeño del sistema, se llevaron a cabo pruebas piloto en entornos controlados que replicaron situaciones urbanas habituales. Se evaluaron factores como el tiempo necesario para detectar, la rapidez en la actualización de los mensajes y la exactitud en la categorización de automóviles. Basándose en los hallazgos, se hicieron modificaciones técnicas y se elaboraron sugerencias para eventuales despliegues en el terreno. Esta etapa también sirvió para demostrar la factibilidad de expandir la solución a otras áreas urbanas del país.

Capítulo 4

Resultados y Discusión

4.1. Modelo De Detección De Vehículos

4.1.1. Descripción general del modelo entrenado

El fundamento de este sistema inteligente de señalización vial radica en un modelo de detección de vehículos de última generación, creado mediante métodos avanzados de visión artificial y aprendizaje profundo. Este elemento es fundamental para que el sistema pueda reconocer y responder de manera dinámica a las condiciones del tráfico. Para el modelo, se ha seleccionado la arquitectura YOLOv11n (You Only Look Once, versión 11, variante "nano"), que representa una decisión estratégica para lograr una óptima combinación entre la exactitud en la detección y la rapidez en la inferencia. Esta característica resulta especialmente útil para su aplicación en tiempo real en hardware que cuenta con recursos computacionales limitados, algo crucial en el ámbito de la infraestructura vial.

El modelo ha sido cuidadosamente entrenado para identificar con precisión cuatro categorías principales de vehículos que son predominantes en las ciudades colombianas. Su operación se integra directamente con el Sistema de Mensajería Variable (SMV), utilizando un conjunto de reglas lógicas preestablecidas que permiten activar señales en el monitor de manera dinámica, dependiendo del tipo de vehículo identificado. De esta manera, tras las detecciones en tiempo real, el modelo crea eventos clasificados que son convertidos al instante en decisiones automáticas de visualización en el SMV, lo que a su vez optimiza la gestión del tráfico y la seguridad en las vías.

La implementación técnica de este modelo se realizó en Python, aprovechando la efectividad de diversas librerías, y se ejecutó en un entorno Flask para facilitar su conexión con la interfaz gráfica de seguimiento. Además, el sistema está concebido para generar automáticamente archivos en formato JSON, que contienen información estructurada sobre cada detección, como la categoría del vehículo, el grado de confianza en la predicción y un timestamp preciso. Esta funcionalidad no solo resulta vital para el análisis en profundidad del flujo vehicular, sino que también establece las bases para mejorar y optimizar el modelo de manera iterativa en el futuro.

4.1.2. Formación del Conjunto de Datos (Dataset)

La robustez y el rendimiento de cualquier modelo de aprendizaje profundo dependen directamente de la calidad y diversidad del conjunto de datos (dataset) utilizado para su entrenamiento. En este proyecto, la creación del dataset fue un proceso meticuloso, diseñado para asegurar la capacidad del modelo de operar eficazmente en las condiciones variadas y complejas del tráfico urbano colombiano.

4.1.3. Adquisición y Etiquetado de Datos

El desarrollo del conjunto de datos comenzó con la recopilación de alrededor de 12,000 archivos, que incluían tanto imágenes fijas como fragmentos de video grabados en situaciones reales de tráfico. Esta etapa fue crucial para familiarizar al modelo con la variedad de contextos que podría encontrar. El conjunto de datos se estableció bajo diferentes condiciones de luz (diurna, nocturna, durante el amanecer y el atardecer), con diversas circunstancias climáticas (soleado, nublado, lluvioso) y desde distintos ángulos de toma, abarcando vistas elevadas, a nivel de la calle y en varias orientaciones con respecto al tráfico. Esta diversidad intencionada en la recolección fue esencial para asegurar la solidez y la capacidad de generalización del modelo en entornos operativos variados.

Después de reunir el material, se procedió a realizar el etiquetado manual de cada uno de los objetos de interés presentes en las imágenes y los fotogramas de video. Este proceso se llevó a cabo con mucha atención al detalle, marcando las áreas relevantes con cajas delimitadoras y asignando las etiquetas adecuadas a las cuatro categorías vehiculares preestablecidas: automóvil, motocicleta, camión y autobús. Para esta tarea de anotación, así como para la gestión y preprocesamiento del conjunto de datos, se utilizó la plataforma Roboflow. Roboflow no solo facilitó el etiquetado de manera eficiente, sino que también permitió una organización sistemática y la limpieza del conjunto de datos, garantizando la calidad y consistencia de las anotaciones. Un ejemplo visual de la interfaz de anotación en esta plataforma se puede observar en la figura 4.1.1, donde se muestran las cajas delimitadoras aplicadas a los vehículos.

4.1.4. Estructuración y Distribución del Dataset

Posterior a la fase de anotación en Roboflow, el dataset fue exportado y organizado en una estructura de directorios estandarizados para el entrenamiento del modelo de detección de objetos. Esta estructura resultante comprende tres conjuntos principales: entrenamiento (train), validación (valid) y prueba (test). La división de los datos se realizó siguiendo una proporción estratégica:

70 % para el conjunto de entrenamiento (train): Contiene la mayor parte de las imágenes y sus etiquetas correspondientes, sumando 8,400 imágenes. Este volumen considerable es crucial para que el modelo aprenda patrones complejos y características distintivas de cada clase vehicular.

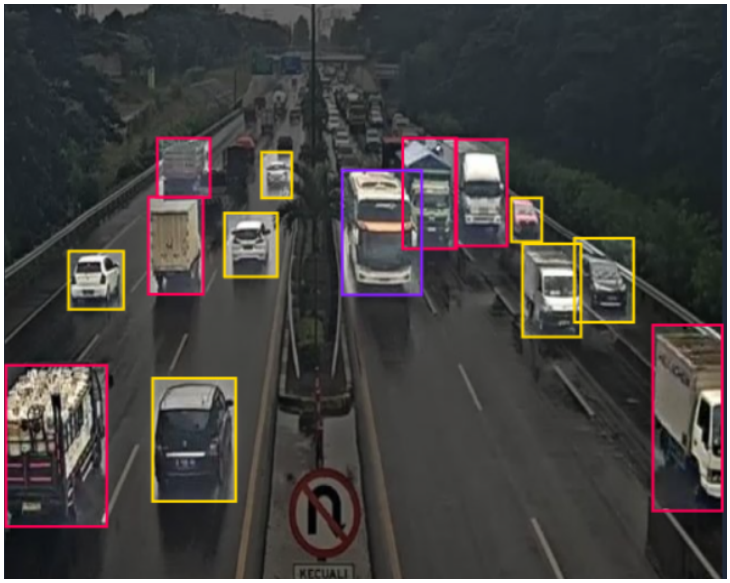

Figura 4.1.1: *Bounding Boxes Aplicados a los Vehículos*

15 % para el conjunto de validación (valid): Compuesto por 1,800 imágenes. Este subconjunto se utiliza durante el proceso de entrenamiento para monitorear el rendimiento del modelo en datos no vistos, permitiendo ajustar hiperparámetros y detectar fenómenos de sobreajuste (overfitting), garantizando que el modelo generalice adecuadamente.

15 % para el conjunto de prueba (test): Integrado por 1,800 imágenes. Este conjunto es fundamental para realizar una evaluación final e imparcial del rendimiento del modelo una vez completado el entrenamiento. Es vital que este subconjunto permanezca completamente aislado del proceso de entrenamiento y validación para proporcionar una métrica confiable de la capacidad del modelo en datos totalmente nuevos.

Cada uno de estas subcarpetas generadas (train, valid, test) contiene, a su vez, carpetas para las imágenes (images/) y sus respectivos archivos de etiquetas (labels/), además de un archivo crucial, data.yaml. La Figura 4.1.2 ilustra la estructura de carpetas del dataset después de la descarga, mostrando la organización jerárquica de imágenes y etiquetas para cada conjunto.

Justificación del Dataset y su Validación en Entorno Local: Los archivos de imágenes empleados para el entrenamiento y la demostración del modelo de detección de vehículos provienen de fuentes diversas y no exclusivamente de Colombia; su selección fue estratégica para validar la robustez y la adaptabilidad del algoritmo. El principal objetivo era asegurar que el sistema fuera capaz de identificar vehículos bajo una amplia gama de condiciones, tales como variaciones de luz, ángulos de cámara y tipos de vías, este enfoque fortalece la fiabilidad del modelo.

La validación específica para el contexto colombiano se realizó a través de pruebas de campo en un entorno real. Como se evidencia en el **Anexo Video SMV IA**, el sistema fue sometido a pruebas con material audiovisual grabado en Colombia, lo cual confirma su correcto funcionamiento y aplicabilidad en las condiciones de tráfico y entorno locales.

Esta prueba práctica demuestra que, a pesar de usar un banco de imágenes diverso, el modelo ha sido ajustado y verificado con éxito para cumplir con las necesidades específicas de la implementación de este prototipo.

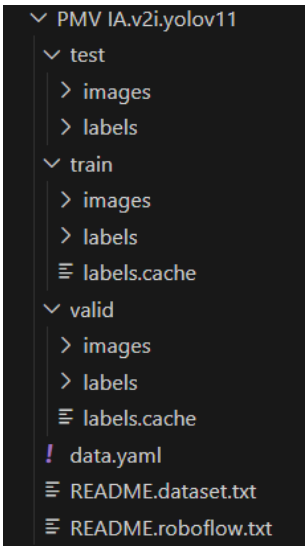

Figura 4.1.2: *Estructura de Carpetas del Dataset*

4.1.5. Gestión del Dataset Mediante Archivo de Configuración YAML

El archivo data.yaml sirve como un archivo de configuración central que establece las rutas relativas para cada conjunto de datos (entrenamiento, validación, prueba) y también incluye la lista de nombres de las clases (nombres). Un trozo de código diseñado se encarga de cargar y gestionar este archivo YAML, facilitando así que el sistema pueda acceder fácilmente a las imágenes y etiquetas requeridas en las diferentes etapas del proceso de entrenamiento, validación y prueba del modelo. En la Figura 4.1.3 se muestra un ejemplo del contenido del archivo data.yaml, subrayando su papel como guía para el modelo.

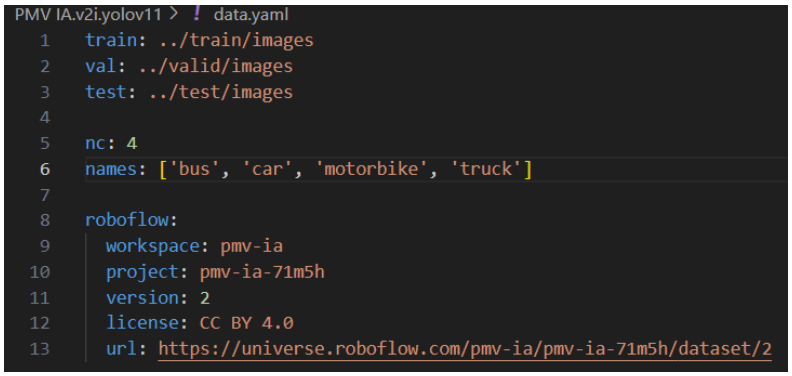

Figura 4.1.3: *Contenido Archivo data.yaml*

4.1.6. Verificación y Análisis del Balance de Clases

Con el objetivo de asegurar que las muestras de cada categoría sean distribuidas de manera justa y evitar sesgos en el proceso de entrenamiento, se diseñó un script que contabiliza las instancias de cada clase en el conjunto de entrenamiento. Al ejecutar este script, se obtuvieron los siguientes resultados sobre la distribución de las anotaciones por categoría:

Clase	Número de anotaciones	Nombre de la Clase
Clase 0	4486	bus
Clase 1	14495	car
Clase 2	4668	motorbike
Clase 3	6013	truck

Tabla 4.1.1: Distribución de Clases en el Conjunto *train*

Los resultados, observados en la Figura 4.1.4 mediante un gráfico de barras, corroboran que cada categoría está representada adecuadamente, contando con miles de anotaciones para cada una. Esto establece una base robusta para el desarrollo de un modelo que sea competente en identificar diferentes tipos de vehículos. A pesar de que hay diferencias en el número de anotaciones por categoría (siendo la clase 'Car' la más numerosa), esta variedad en las categorías es crucial para que el modelo adquiera la capacidad de generalizar y distinguir entre los distintos vehículos en diferentes escenarios.

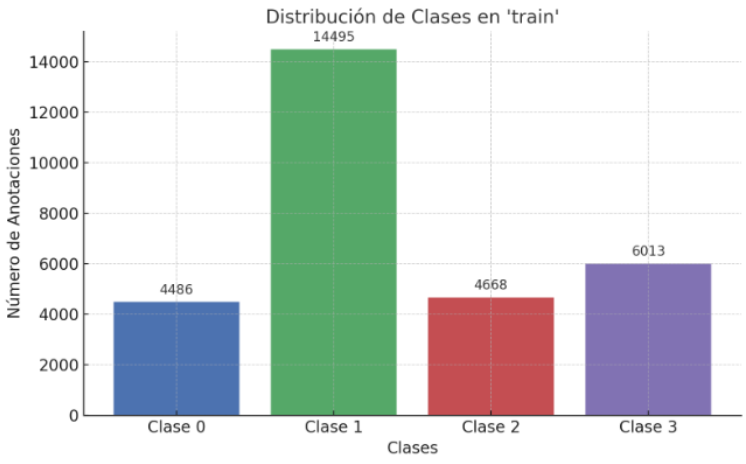

Figura 4.1.4: *Distribución Clases Train*

4.1.7. Visualización del Dataset

Para una inspección cualitativa del dataset y para confirmar la presencia de las imágenes, se realizó la visualización de muestras representativas del conjunto de entrenamiento. Mediante el uso de librerías como OpenCV (cv2) para la carga y manipulación de imágenes, y Matplotlib para la visualización, se implementó un

fragmento de código que permite:

1.Definir la ruta específica donde se encuentran las imágenes de entrenamiento.

2.Seleccionar una imagen particular (ej. la imagen número 6500) para su inspección.

3.Cargar la imagen y realizar la conversión del formato BGR (Blue-Green- Red), utilizado por OpenCV, al formato RGB (Red-Green-Blue), esperado por Matplotlib.

Figura 4.1.5: *Imagen de Entrenamiento 6500*

Esta visualización proporcionó una comprensión visual directa de la complejidad y variabilidad de los datos con los que el modelo fue entrenado. Un ejemplo de esta visualización se presenta en la Figura 4.1.5, donde se observa una imagen del conjunto de entrenamiento correspondiente al dataset de vehículos.

4.1.8. Preparación del Entorno de Entrenamiento

El entrenamiento del modelo de detección de objetos se llevó a cabo utilizando la arquitectura YOLOv11n (You Only Look Once, versión 11, variante "nano"), seleccionada por su eficiencia y su capacidad para operar en dispositivos con recursos computacionales limitados, manteniendo una velocidad de inferencia óptima. La flexibilidad propia de la librería Ultralytics de YOLOv11 permitió personalizar el proceso de entrenamiento, adaptándolo al dataset específico construido para este proyecto.

El comando de entrenamiento ejecutado fue:

```
yolo task=detect mode=train epochs=150
data="C:/Users/marco/Desktop/Object_Detec/PMV IA.v2i.yolov11/data.yaml"
model=yolo11n.pt imgsz=640 batch=2
```

Donde **task=detect** especifica que la tarea principal a realizar es la detección de objetos; **mode=train** indica que se ejecutará el proceso de entrenamiento del modelo; **epochs=150** define el número total de épocas de entrenamiento, lo cual permitió al modelo aprender adecuadamente las características distintivas de las clases vehiculares presentes en el dataset, maximizando su desempeño y reduciendo el riesgo de sobreajuste; **data** proporciona la ruta absoluta al archivo data.yaml, esencial para localizar los conjuntos de datos de entrenamiento, validación y prueba, así como los nombres de las clases a detectar; **model=yolo11n.pt** indica que se utilizará el modelo preentrenado yolo11n.pt como punto de partida, lo que permite aplicar técnicas de aprendizaje por transferencia para acelerar la convergencia y mejorar el rendimiento en datasets personalizados; **imgsz=640** establece que la resolución de entrada de las imágenes durante el entrenamiento será de 640x640 píxeles, balanceando el nivel de detalle con la eficiencia computacional; por ultimo **batch=2** define el tamaño de lote en 2, ajustado para un uso óptimo de los recursos de hardware disponibles.

La preparación cuidadosa del entorno de entrenamiento, junto con la configuración precisa de estos parámetros, fue determinante para lograr un proceso robusto y eficiente, orientado a la generación de un modelo confiable para su aplicación en sistemas de señalización vial inteligente.

4.1.9. Precisión del Modelo (Métricas de Evaluación)

La evaluación de la efectividad real del modelo entrenado se realizó mediante el análisis de diversas métricas de evaluación estándar en el campo de la detección de objetos. Estas métricas son fundamentales para cuantificar el rendimiento del modelo en datos no vistos (conjunto de validación), previniendo la confianza excesiva en los resultados de entrenamiento que podrían indicar un sobreajuste (overfitting). Además, permiten una comparación objetiva entre diferentes configuraciones o versiones del modelo, fundamentando las decisiones técnicas con criterios claros y objetivos.

Las métricas clave analizadas para evaluar el rendimiento global y por clase fueron:

Precisión (Precision, P): Mide la proporción de detecciones correctas respecto al total de detecciones realizadas por el modelo. Un valor alto indica una baja tasa de falsos positivos.

Recall (R): Mide la proporción de objetos reales que fueron detectados correctamente por el modelo. Un valor alto indica una baja tasa de falsos negativos.

mAP@0.5 (mean Average Precision at IoU=0.5): Es el promedio de la precisión promedio (AP) para cada clase, considerando un umbral de Intersección sobre Unión (IoU) de 0.5. Es una métrica global robusta que evalúa el rendimiento general del modelo en un umbral de superposición más permisivo.

mAP@0.5:0.95 (mean Average Precision across IoU thresholds from 0.5 to 0.95): Esta métrica más exigente calcula el promediando los umbrales de IoU desde 0.5 hasta 0.95 en pasos de 0.05. Ofrece una evaluación más completa del rendimiento, especialmente en la precisión de la localización en condiciones de superposición.

4.1.10. Resultados Globales de las Métricas de Evaluación

Los resultados del entrenamiento de nuestro modelo YOLOv11n en el conjunto de validación se resumen en la Tabla 4.1.2:

Métrica	**Valor**	**Interpretación**
`Box(P)` (Precisión promedio)	0.825	El 82.5 % de las veces que el modelo predice un objeto, acierta. **Muy buen valor**.
`R` (Recall promedio)	0.721	El 72.1 % de los objetos reales son detectados correctamente. **Hay margen de mejora.**
`mAP@0.5`	0.793	El modelo tiene una precisión promedio del 79.3 % considerando IoU $\geq$ 0.5. **Aceptable.**
`mAP@0.5:0.95`	0.500	Esta métrica más exigente (promedio de IoU desde 0.5 hasta 0.95) muestra una precisión **moderada**. Refleja que el modelo puede mejorar en **precisión fina** de cajas.

Tabla 4.1.2: *Resultados Obtenidos en la Evaluación del Modelo*

Estos valores muestran un desempeño general sólido, especialmente si consideramos que estamos usando una variante nano del modelo, que está optimizada para la velocidad. La Figura 4.1.6 presenta un gráfico que resume estas métricas, ofreciendo una visualización rápida y efectiva del rendimiento global de nuestro modelo.

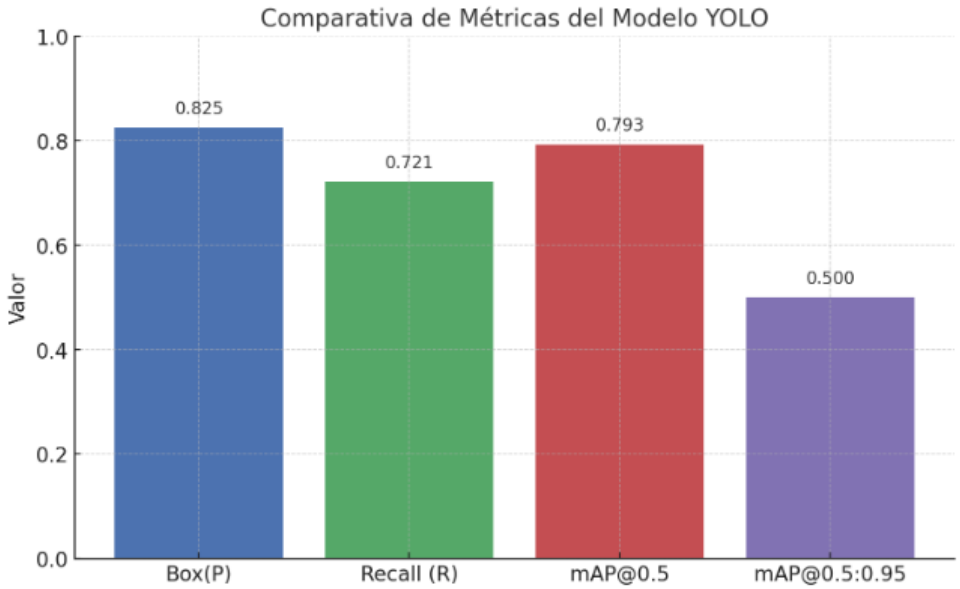

Figura 4.1.6: *Gráfico de Barras Comparativo de las Métricas*

4.1.11. Rendimiento por Clase

Un análisis más profundo nos permite ver cómo se desempeña el modelo específicamente para cada tipo de vehículo. Los resultados se muestran en la Tabla 4.1.3

Clase	Precisión	Recall	mAP@0.5	mAP@0.5:0.95	Observación
`bus`	0.827	0.691	0.748	0.607	Buen modelo, pero podría mejorar en recall.
`car`	0.869	0.690	0.814	0.476	Alta precisión pero bajo mAP@0.95: puede detectar bien, pero a veces coloca mal las cajas.
`motorbike`	0.812	0.721	0.784	0.405	Buen equilibrio, pero con menor precisión fina.
`truck`	0.790	0.784	0.825	0.512	Muy equilibrado: reconoce y detecta bien camiones.

Tabla 4.1.3: *Rendimiento por clase del modelo de detección vehicular*

Los resultados por clase son consistentes con nuestras métricas globales, mostrando tanto las fortalezas como las áreas donde podríamos mejorar en el futuro. Por ejemplo, aunque los automóviles (car) se detectan con muy alta precisión, la calidad de sus bounding boxes podría afinarse en escenarios más exigentes, mientras que los camiones (truck) muestran un rendimiento más uniforme en todas las métricas. La Figura 4.1.7 ilustra visualmente el rendimiento de cada clase a través de un gráfico comparativo de sus métricas.

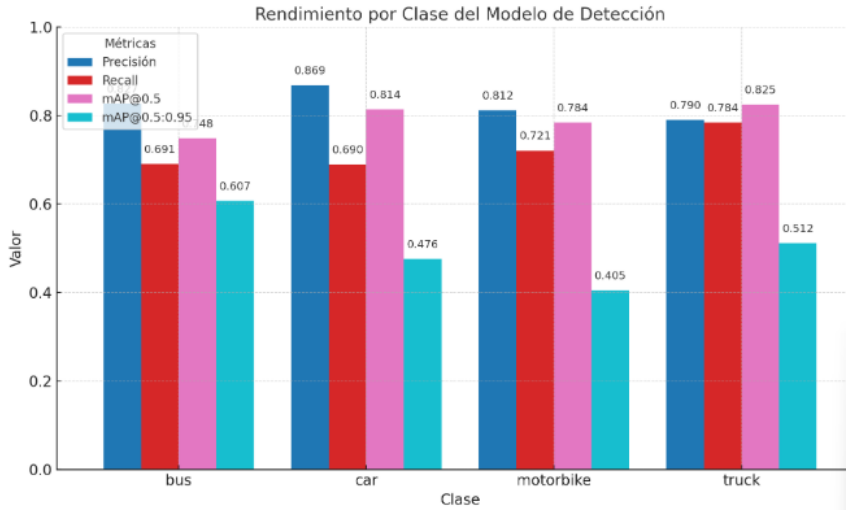

Figura 4.1.7: *Gráfico de Barras Comparativo de las Métricas por Clases*

4.1.12. Inferencia en Imágenes y Conteo de Detecciones

Para ver cómo funciona nuestro modelo en la práctica y entender su comportamiento en situaciones reales, hicimos inferencias sobre imágenes individuales. Este proceso no solo verifica la capacidad de detección, sino que también nos permite contar y registrar los objetos identificados de forma práctica.

Se desarrolló un código que toma una imagen, aplica nuestro modelo YOLOv11n entrenado y luego dibuja los resultados de la detección (la clase del vehículo, el nivel de confianza y las coordenadas del bounding box) directamente sobre la imagen original. Esto nos permite:

Verificación visual: Observar de forma directa cómo el modelo identifica y delimita los objetos en una escena. Es vital para detectar posibles errores como falsos positivos (detecciones incorrectas) o falsos negativos (objetos no detectados) en contextos específicos.

Cuantificación: Contar el número de detecciones por cada clase en una imagen, dándonos una medida objetiva de la presencia de vehículos.

Generación de evidencias: Crear imágenes anotadas que son muy útiles para nuestros informes, presentaciones o documentación técnica.

Durante las evaluaciones, el modelo logró identificar, por ejemplo, 17 vehículos en una determinada imagen de prueba. El tiempo total requerido para procesar esta imagen, que abarcó el preprocesamiento, la inferencia y el postprocesamiento, fue de cerca de 2 segundos. En términos más específicos, el tiempo de inferencia en sí fue de 113.3 milisegundos por imagen. Esta información es fundamental, ya que resalta la eficacia del modelo, lo que lo hace adecuado para aplicaciones en tiempo real que demandan respuestas rápidas. Se guardó la imagen que contenía las detecciones para revisiones futuras. Un ejemplo de estas detecciones se puede observar en la Figura 4.1.8.

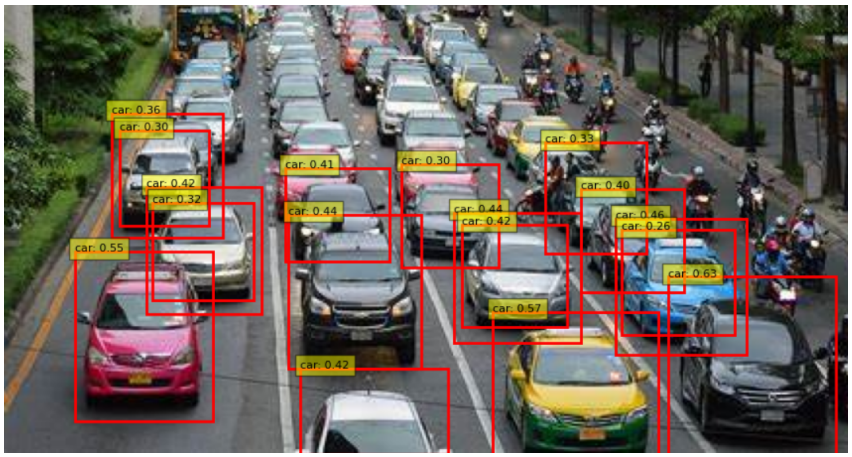

Figura 4.1.8: *Detecciones Imagen de Muestra*

4.1.13. Matriz de Confusión Normalizada

La matriz de confusión normalizada se presenta como una herramienta clave en el análisis del desempeño de un modelo de detección de objetos. Ofrece una representación tanto visual como numérica, en forma de proporciones o porcentajes, sobre la clasificación realizada por el modelo respecto a las instancias individuales. Esto ayuda a

desglosar los patrones de éxitos (verdaderos positivos) y fallos (falsos positivos y falsos negativos), sin que el tamaño absoluto de cada categoría en el conjunto de datos de validación influya en la interpretación.

La matriz de confusión normalizada generada para el modelo YOLOv11n, ilustrada en la Figura 4.1.9, posee las siguientes características estructurales:

Dimensiones: La matriz es de 5×5 dimensiones, representando las cuatro clases vehiculares de interés (bus, car, motorbike, truck) más la categoría background (fondo), tanto para las etiquetas reales (True) como para las predicciones del modelo.

Filas (True): Cada fila corresponde a la clase verdadera (Ground Truth) del objeto en la imagen. La suma de los valores en cada fila es aproximadamente 1 (o 100 %), lo que indica cómo las instancias de una clase real se distribuyeron entre todas las clases predichas.

Columnas (Predicted): Cada columna representa la clase que el modelo predijo para las instancias.

Diagonal Principal: Los valores ubicados en la diagonal principal de la matriz (superior izquierda a inferior derecha) representan la tasa de acierto (recall) para cada clase, es decir, la proporción de instancias de una clase real que fueron correctamente predichas por el modelo.

Elementos Fuera de la Diagonal Principal: Los valores fuera de la diagonal principal indican las proporciones de errores de clasificación.

• Un valor en la fila i y columna j significa que una proporción de instancias reales de la clase i fue predicha erróneamente como la clase j.

• La última columna, background (en Predicted), es particularmente importante. Los valores aquí indican la proporción de instancias reales de una clase vehicular que el modelo no detectó como un objeto, clasificándolas como parte del fondo. Esto representa una medida de los falsos negativos (objetos reales no detectados como tales) para cada clase de vehículo.

• La última fila, background (en True), indica la proporción de instancias que eran background en realidad, pero fueron erróneamente clasificadas como una de las clases vehiculares (un tipo de falso positivo para esas clases vehiculares).

4.1.14. Análisis Detallado de la Matriz de Confusión Normalizada

La matriz de confusión normalizada de la Figura 4.1.9 nos permite un análisis proporcional del desempeño del modelo por cada clase. A continuación, se detallan los resultados clave:

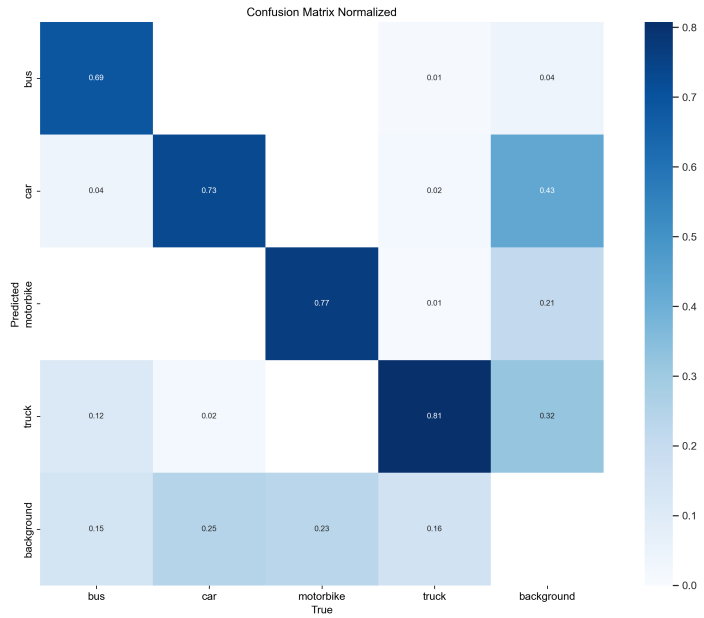

Figura 4.1.9: *Matriz Confusion Normalizada*

Rendimiento por Clase Real

Buses (Fila bus): El modelo identificó correctamente el 69 % de los buses reales. Las principales confusiones ocurrieron al clasificar el 4 % de los buses como truck y el 1 % como car. Un 15 % de los buses reales no fueron detectados, siendo erróneamente clasificados como background.

Automóviles (Fila car): El 73 % de los automóviles reales fueron correctamente detectados. Las confusiones con otras clases de vehículos fueron muy bajas (4 % con bus, 2 % con truck). Sin embargo, el 25 % de los automóviles no fueron detectados por el modelo, siendo clasificados como background.

Motocicletas (Fila motorbike): El modelo tuvo un buen desempeño, clasificando correctamente el 77 % de las motocicletas reales. Las confusiones con otras clases vehiculares fueron mínimas (1 % con car, 1 % con truck). La principal pérdida de detección se dio al clasificar el 23 % de las motocicletas como background.

Camiones (Fila truck): Esta fue la clase con el mayor recall, con un 81 % de camiones reales detectados correctamente. Se observaron confusiones, con el 12 % de los camiones siendo clasificados como bus y un 2 % como car. Un 16 % de los camiones reales fueron clasificados como background.

Falsos Positivos de Background La Fila background (última fila de la matriz) muestra la proporción de las veces que el modelo predijo un vehículo, cuando en realidad se trataba de background (fondo). Esto indica los falsos positivos:

El 1 % del background fue erróneamente clasificado como bus.
El 2 % del background fue erróneamente clasificado como car.
El 2 % del background fue erróneamente clasificado como motorbike.

El 3 % del background fue erróneamente clasificado como truck.
El 92 % alta precisión para no generar detecciones donde no hay vehículos.

En resumen, la matriz normalizada muestra que el modelo tiene un buen nivel de recall en general para todas las categorías de vehículos, siendo capaz de identificar la mayoría de los objetos que aparecen. Las confusiones entre las distintas clases de vehículos son bastante bajas. La principal área que necesita mejoras se encuentra en la disminución de los falsos negativos, ya que un porcentaje considerable de vehículos reales (entre el 15 % y el 25 %, en función de la clase) no fueron identificados y fueron considerados como parte del fondo. Por otro lado, la frecuencia de falsos positivos es baja, lo cual es un aspecto favorable, dado que el modelo rara vez indica la presencia de un vehículo donde realmente no existe.

4.1.15. Implementación del Modelo de Detección de Vehículos en Tiempo Real y Video MP4

La funcionalidad efectiva de nuestro modelo entrenado se llevó a cabo a través de un sistema de detección de objetos en tiempo real, creado para funcionar con secuencias de video. Esta implementación fue comprobada con una cámara web en vivo y también con videos pregrabados en formato MP4, lo que resalta la flexibilidad y capacidad de adaptación del modelo a diversas fuentes de entrada. La meta principal era reconocer y categorizar vehículos en cada fotograma, trazando rectángulos delimitadores y asignándoles etiquetas que indicaran la clase y el grado de confianza de la predicción.

Las características principales de esta implementación incluyen:

Detección en tiempo real: El sistema analiza los fotogramas de video capturados (ya sea en vivo o desde un archivo) y hace predicciones instantáneas sobre los objetos presentes. Esta capacidad es crucial para aplicaciones de monitoreo de tráfico y sistemas de señalización inteligente que requieren una respuesta inmediata.

Visualización interactiva y clara: Los resultados obtenidos durante la detección se superponen de manera directa al flujo de vídeo. Esto implica la creación de cuadros delimitadores de diferentes colores para cada tipo de vehículo, así como la inclusión de su etiqueta (automóvil, autobús, motocicleta, camión) y el nivel de confianza de la predicción. Esta forma de visualización permite a los operadores interpretar fácilmente la información y verificar cómo está funcionando el modelo. La Figura 4.1.10 presenta una imagen del sistema operando con una cámara web en tiempo real. Por otro lado, la Figura 4.1.11 muestra la detección realizada sobre un archivo de video MP4, evidenciando su habilidad para procesar contenido que ha sido grabado previamente.

Optimización de rendimiento: El sistema fue optimizado para lograr un procesamiento de imágenes con una eficacia destacable. En las evaluaciones realizadas, se obtuvo un tiempo medio de inferencia de 113.1 ms por cada imagen. Este logro resulta muy beneficioso para aplicaciones que requieren funcionamiento en tiempo real, garantizando una detección continua y una latencia reducida.

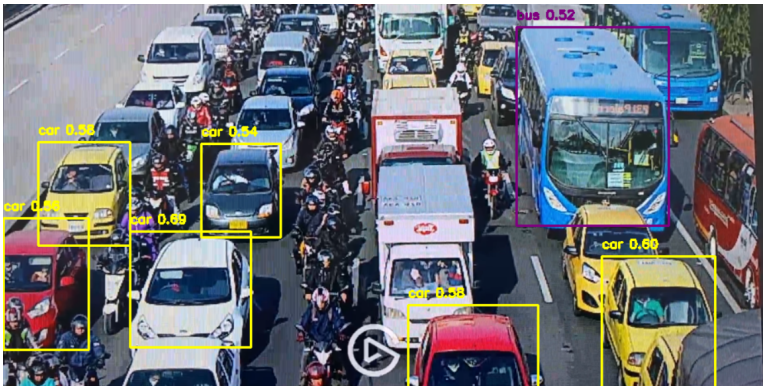

Figura 4.1.10: *Detección Vehículos en Tiempo Real*

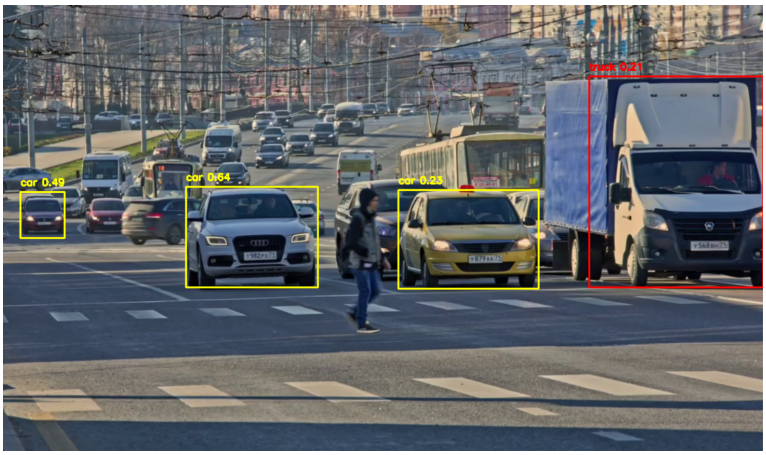

Figura 4.1.11: *Detección de Vehículos Archivo Mp4*

Escalabilidad y robustez: El diseño permite detectar múltiples clases de objetos simultáneamente, asignando un color distintivo a cada categoría, lo que mejora la claridad de las visualizaciones. Además, las pruebas confirmaron que la implementación funciona de manera consistente, incluso cuando no se detectan objetos en la escena, lo que demuestra la estabilidad del sistema.

El modelo de detección de objetos funcionó correctamente según lo planeado, mostrando un rendimiento eficiente tanto en precisión como en velocidad. Las detecciones fueron precisas, identificando correctamente las clases de objetos con alta confianza en las predicciones. Esto lo hace muy adecuado para su implementación en escenarios prácticos de señalización inteligente y monitoreo vehicular.

4.1.16. Modelo de Detección de Vehículos en Tiempo Real Integrado con un Sistema de Reglas Lógicas para la Proyección de Señales de Tránsito y el Almacenamiento de Datos en Archivos JSON

La funcionalidad del modelo de detección de vehículos trasciende la identificación visual de objetos. Se conecta de manera esencial con un sistema de decisiones que se

apoya en las regulaciones del tránsito. Esto se lleva a cabo al almacenar de forma organizada los datos de las detecciones en archivos con formato JSON (JavaScript Object Notation). El objetivo principal de esta integración es automatizar el proceso de recopilación, clasificación y registro de información pertinente sobre los vehículos identificados, lo que permite tanto decisiones automáticas en tiempo real como un análisis posterior de las tendencias del tráfico. El formato JSON se eligió por sus múltiples ventajas para la gestión de datos en aplicaciones de ingeniería, destacando aspectos como:

Ligereza y Legibilidad: JSON es un formato de texto muy eficiente, fácil de procesar por computadoras y legible para humanos, lo que simplifica su inspección y depuración.

Interoperabilidad Universal: Es un estándar global para el intercambio de datos en la web, compatible con la mayoría de los lenguajes de programación y sistemas. Esta compatibilidad permite una integración fluida con otras plataformas, como APIs (Interfaces de Programación de Aplicaciones) para servicios web o bases de datos NoSQL.

Registro Dinámico de Datos: Facilita el registro de información de manera ágil y ordenada en cada fase de detección. Para cada vehículo que se reconozca, el sistema produce un objeto JSON que alberga datos esenciales.

Cada entrada de detección incluye aspectos fundamentales para el estudio del tráfico, como la categoría del vehículo observado (por ejemplo, .ªutomóvil.º .ªutobús"), el grado de confianza asociado a la predicción del modelo expresado como un número decimal en el rango de 0 a 1, la marca de tiempo precisa del instante de detección (fecha y hora), que es vital para el seguimiento y análisis temporal de los eventos, así como las coordenadas del cuadro delimitador —normalmente en el formato [x_min, y_min, x_max, y_max]— que indican de forma exacta la posición del objeto dentro del marco.

Facilidad de Integración con Sistemas Superiores: Los archivos JSON generados se pueden integrar sin problemas con bases de datos relacionales o NoSQL, dashboards de visualización en tiempo real, o servicios web que necesiten procesar, almacenar o visualizar los datos de detección para análisis de tráfico, gestión de flotas o aplicaciones de seguridad vial.

La Figura 4.1.12 presenta un ejemplo esquemático de la estructura de un archivo JSON con datos de detección, ilustrando cómo se organiza la información relevante para su posterior consumo por otros sistemas.

Figura 4.1.12: *Esquema Formato JSON*

4.1.17. Evaluación de los Parámetros del Modelo YOLOv11n

Los resultados de los parámetros de evaluación del modelo de detección de objetos, con una precisión global del 82.5 % y una media de precisión promedio (mAP@0.5) del 79.3 %, se consideran aceptables y adecuados para la implementación de un prototipo funcional. Es crucial destacar que el modelo utilizado, YOLOv11 en su versión "nano", fue seleccionado por su alta adaptabilidad y eficiencia en dispositivos con restricciones de procesamiento, priorizando la viabilidad del sistema en su contexto de aplicación. Este valor demuestra un balance óptimo entre la precisión de las detecciones y la velocidad de inferencia, que es de 113.3 milisegundos por imagen, un aspecto crítico para un sistema que opera en tiempo real.

En correlación con la literatura académica y la industria, se evidencia que los modelos de detección de objetos que alcanzan una precisión superior al 90 % suelen ser arquitecturas más robustas y complejas, como las versiones "Large.º "X"de YOLO, o modelos como Faster R-CNN. Si bien estos modelos pueden ofrecer mayor precisión en entornos controlados, su complejidad y elevado requerimiento computacional pueden llevar a una latencia inaceptable para aplicaciones en tiempo real. Además, una precisión excesivamente alta en un conjunto de datos específico puede ser indicativa de sobreajuste, lo que resulta en un rendimiento deficiente cuando el modelo se enfrenta a variaciones inesperadas en un entorno no controlado.

El valor de mAP@0.5 del 79.3 % obtenido en este proyecto se encuentra dentro del margen de valores medios y aceptables para implementaciones en escenarios reales, demostrando que el modelo ha aprendido a generalizar de manera efectiva sin memorizar el conjunto de datos de entrenamiento. Específicamente, el rendimiento por clase muestra un comportamiento robusto:

- Bus: mAP@0.5 de 74.8 %
- Car: mAP@0.5 de 81.4 %
- Motorbike: mAP@0.5 de 78.4 %
- Truck: mAP@0.5 de 82.5 %

Si bien existe un margen de mejora en la velocidad de inferencia y en la reducción de falsos positivos y negativos, se considera que estos aspectos son parte de las direcciones futuras del proyecto. Por lo tanto, se recomienda que en implementaciones posteriores se exploren nuevas tecnologías o modelos más avanzados que garanticen resultados superiores, tal como se describe en el apartado de Conclusiones y Recomendaciones. La presente investigación, al centrarse en un prototipo funcional, demuestra de manera concluyente que la precisión lograda es la idónea para validar el concepto y la viabilidad técnica de la solución.

4.2. Reglas Lógicas Proyección Dinámica De Señales De Transito

En esta sección se examina el desarrollo y la ejecución del subsistema de reglas lógicas, que juega un papel crucial al dotar de inteligencia y flexibilidad al sistema de señalización de tráfico propuesto. Su principal responsabilidad es convertir en tiempo real las detecciones de vehículos, obtenidas a través del módulo de visión artificial, en una representación visual de señalización vial que sea dinámica y relevante para el contexto. Este subsistema completa el ciclo de percepción, decisión y acción, esencial para un sistema de transporte inteligente. Actualmente, su implementación se dirige hacia la proyección en un monitor HDMI, con la perspectiva futura de expandirse a Sistemas de Mensajería Variable (SMV) adecuados para entornos de movilidad vial.

4.2.1. Introducción General al Sistema de Reglas Lógicas

Se desarrolló e implementó un módulo de proyección lógica de señales viales como un elemento clave del sistema de visualización integral. Este módulo tiene como objetivo esencial crear una conexión en tiempo real entre los resultados obtenidos del modelo de detección de vehículos y señales de tráfico determinadas, que se presentan de forma dinámica en una pantalla secundaria (monitor HDMI). De esta manera se refleja el funcionamiento de un Sistema de Mensajería Variable inteligente. Es importante señalar que la versión actual se enfoca en la visualización en un monitor estándar a través del protocolo HDMI, actuando como un prototipo funcional que valida la lógica del sistema. Para el futuro, se plantea una integración con un panel de mensajería variable LED, capaz de adaptarse a diversas condiciones climáticas, lo que se considera una evolución para permitir la escalabilidad y la implementación real del proyecto en la infraestructura vial.

La versatilidad de este sistema se manifiesta en sus modos de operación:

Modo Automático: En este modo, el sistema opera de forma autónoma, leyendo las entradas de detección vehicular directamente desde un archivo vehiculos_detectados.json. Este archivo es el conducto de comunicación con el módulo de detección de objetos y contiene las clases de vehículos identificados en tiempo real.

Modo Manual (Simulación): Para fines de prueba, depuración, demostraciones en laboratorio o entornos académicos e institucionales, el sistema permite la entrada manual de clases vehiculares directamente a través de la consola. Esto facilita la simulación de escenarios específicos y la verificación del comportamiento de las reglas lógicas sin necesidad de un flujo de detección en vivo.

La integración de este sistema de reglas permite trascender de una pantalla estática a una interfaz vial activa y reactiva, capaz de comunicar mensajes relevantes para la seguridad y fluidez del tráfico, sentando las bases para una futura implementación robusta en paneles LED de gran escala.

4.2.2. Flujo de Ejecución del Sistema de Proyección de Señales

El sistema encargado de la proyección de señales fue completamente desarrollado en Python. Está organizado como un bucle continuo, diseñado para analizar nuevas detecciones de vehículos cada 0.5 segundos. Con el fin de asegurar una presentación continua y sin interrupciones ni demoras visibles en la proyección de las señales en la pantalla HDMI, se implementó de manera estratégica la programación multihilo. Esto facilita que la lógica de evaluación y toma de decisiones se realice al mismo tiempo que la proyección visual.

El proceso operativo esencial del sistema, desde su activación hasta la proyección dinámica de las señales, se puede resumir en la Tabla 4.2.1.

Paso	Descripción
1.Inicio del Sistema	Se ejecuta el programa y se imprime en consola: 'Sistema de señales iniciado...
2.Bucle Principal	El sistema entra en un bucle que se repite cada 0.5 segundos. Cada ciclo evalúa si hay nuevas clases detectadas.
3.Lectura Detecciones	Lee un archivo '.json' con los objetos detectados y la hora de detección ('timestamp'). Si está en modo simulación, solicita ingreso manual por consola.
4.Filtrado Tiempos	Solo se consideran válidas las detecciones más recientes. Se evita repetir señales si ya se mostraron en un ciclo anterior.
5.Determinación Clave	Evalúa las clases detectadas y decide qué conjunto de señales mostrar: una clase (CAR, BUS, etc.), varias clases (TODAS_VARIAS), o sin detección durante más de 2 segundos (NADA).
6.Cambio Señal	Si la nueva clase es diferente a la anterior ('ultima_clave_mostrada'), se detiene la proyección actual y se inicia una nueva.
7.Hilo Proyección	Se lanza un nuevo hilo ('threading.Thread') para mostrar las imágenes asociadas a la clase actual, una por una, en bucle.
8.Proyección Visual	Cada imagen se muestra por 5 segundos en pantalla. Si el usuario presiona 'ESC', el sistema finaliza.

Tabla 4.2.1: Flujo de ejecución del sistema de proyección de señales

Este flujo operativo, caracterizado por su evaluación periódica y el uso de hilos, permite que el sistema reaccione dinámicamente a los cambios en el entorno vial. La Figura 4.2.1 muestra una captura de pantalla de la consola durante la ejecución del sistema en modo simulación, evidenciando la interacción con el usuario y los mensajes de estado.

4.2.3. Reglas Lógicas de Señalización

El sistema de visualización avanzada utiliza una arquitectura de toma de decisiones compuesta por seis reglas lógicas fundamentales. La activación de estas reglas ocurre de manera exclusiva, dependiendo del tipo principal de detección de vehículos realizado por el módulo de visión artificial, o en su defecto, la falta de dicha detección. Cada una de estas reglas está relacionada de manera intrínseca con un conjunto específico y ordenado de imágenes digitales de señales de tráfico, que han sido creadas para ser proyectadas en el monitor de mensajería variable .

Las reglas lógicas implementadas son:

Regla 1: CAR: Se activa al detectar predominantemente automóviles.

Regla 2: TRUCK: Se activa al detectar principalmente camiones.

Figura 4.2.1: *Modo Simulación Proyección Señales*

Regla 3: MOTORBIKE: Se activa ante la detección significativa de motocicletas.

Regla 4: BUS: Se activa al detectar buses.

Regla 5: NADA: Se activa cuando no se detectan vehículos de interés durante un período prolongado (superior a 0.2 segundos), indicando un estado de baja o nula afluencia vehicular.

Regla 6: TODAS_VARIAS: Se activa ante la detección simultánea de múltiples clases de vehículos (por ejemplo, car y truck al mismo tiempo), indicando una condición de tráfico mixto.

Cada una de estas reglas lógicas se vincula a un grupo de imágenes de señales digitales en formato .png que han sido previamente definidas. Estas imágenes representan categorías estándar de señales viales, tales como Señales Preventivas (SP), Señales Reglamentarias (SR) y Señales Informativas (SI), todas diseñadas para su visualización dinámica en un monitor HDMI. En la Tabla 4.2.2 se especifica la correspondencia exacta entre cada tipo de detección y los archivos de señales que están programados para ser mostrados por el sistema.

Tipo de Detección	Archivos de Imágenes Asociadas
NADA	SI-16A.png, SP-47.png, INTERES 2K.png, SP-46.png, SP-23.png, SP-39.png, SR-56.png
CAR	SR-01.png, SP-46A.png, sR-47.png, SR-40.png
MOTORBIKE	SR-M.png
TRUCK	SP-79.png, SR-43.png
BUS	SI-08.png, SR-41.png
TODAS_VARIAS	SI-35A.png, SR-28A.png, SR-26.png

Tabla 4.2.2: Asociación de clases detectadas con señales de tránsito proyectadas

La Figura 4.2.2 ofrece una representación visual que muestra la clasificación de las señales según el tipo de vehículo o las condiciones del tráfico, con una muestra representativa de las imágenes en formato .png que se proyectarían para cada norma. Asimismo, para ejemplificar cómo se aplica esta lógica en el ámbito del código, la ilustración 4.2.3 incluye un fragmento del diccionario en Python que se utiliza para vincular cada clase detectada con su respectiva lista de rutas de archivos de imagen.

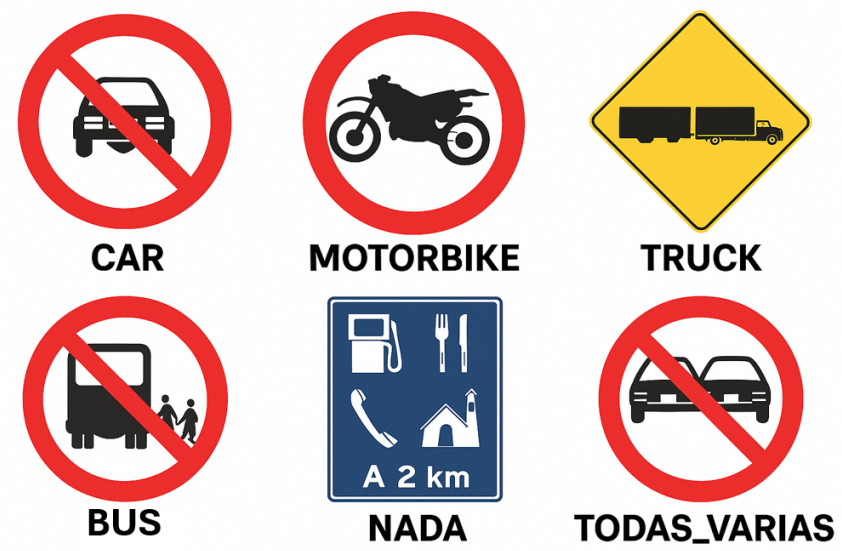

Figura 4.2.2: *Visualización Señales por Reglas*

```python
rules = {
    "NADA": ["SI-16A.png", "SP-47.png", "INTERES 2K.png",
             "SP-46.png", "SP-23.png", "SP-39.png", "SR-56.png"],

    "CAR": ["SR-01.png", "SP-46A.png", "sR-47.png", "SR-40.png"],

    "MOTORBIKE": ["SR-M.png"],

    "TRUCK": ["SP-79.png", "SR-43.png"],

    "BUS": ["SI-08.png", "SR-41.png"],

    "TODAS_VARIAS": ["SI-35A.png", "SR-28A.png", "SR-26.png"]
}
```

Figura 4.2.3: *fragmento del Diccionario Reglas Python*

4.2.4. Ejecución en Tiempo Real y Comportamiento Observado

Durante las etapas de evaluación del sistema de proyección lógica, se llevó a cabo un análisis detallado de diversos escenarios, tanto en el modo real, que se conecta al flujo de detecciones del modelo YOLOv11n, como en el modo manual, que implica simulaciones. Estas evaluaciones demostraron que el sistema responde de forma precisa y adecuada a los diferentes patrones de tráfico y a las variables ambientales, proyectando señales de manera efectiva en la pantalla externa.

La Tabla 4.2.3 presenta un resumen de algunos escenarios clave de prueba junto con el comportamiento que se esperaba y el que se observó del sistema, lo que valida la solidez y la capacidad de respuesta de las reglas lógicas.

Escenario	**Entrada Detectada**	**Señales Proyectadas**	**Comportamiento Observado**
Detección solo de CAR	`Clase CAR` vehiculos_detectados.json	SR-01, SP-46A, SR-47, SR-40	El sistema transicionó rápidamente para mostrar la secuencia de señales específicas para automóviles en el monitor HDMI, con proyección clara y fluida.
Detección CAR + TRUCK	Ambas clases presentes en vehiculos_detectados.json	SI-35A, SR-28A, SR-26	Al detectar múltiples clases simultáneamente, el sistema activó la regla `TODAS_VARIAS`, proyectando señales de tráfico general y advertencia en el monitor HDMI.
Sin Detección (por más de 3s)	Archivo vehiculos_detectados.json vacío por 3 segundos	SI-16A, SP-47, SP-46, etc.	El sistema entró en un modo pasivo, proyectando señales neutras o informativas generales en el monitor HDMI, lo que indica un estado de baja actividad vehicular.
Entrada de Clase Inválida	Clase BICYCLE (o cualquier otra clase no definida en las reglas)	Ninguna señal asociada a esa clase se proyectó.	El sistema ignoró la entrada, confirmando su capacidad para manejar clases no reconocidas sin generar errores o proyecciones inesperadas.

Tabla 4.2.3: Escenarios de prueba y comportamiento del sistema de proyección

Las pruebas llevaron a cabo la validación de la habilidad del sistema para comprender las detecciones y activar la regla lógica apropiada, lo cual a su vez provoca la visualización correcta en el monitor. La Figura 4.2.4 representa una imagen del sistema de mensajeria variable HDMI durante la proyección continua de señales en bucle, evidenciando la fluidez con la que se muestran las imágenes. Por otro lado, la Figura 4.2.5 muestra un extracto del archivo vehiculos_detectados.json que contiene datos de detección en tiempo real, ilustrando el formato de la información que nutre la lógica de decisión.

Figura 4.2.4: *SMV HDMI Durante la Proyección Dinámica*

4.2.5. Arquitectura Técnica y Control de Proyección

La solidez y la fluidez del sistema de proyección de señales en un sistema de mensajería variable HDMI se respaldan en una arquitectura técnica que utiliza programación multihilo. Esta estrategia permite que las tareas relacionadas con la evaluación de la lógica de detección y la visualización de las señales se realicen de manera simultánea y asíncrona, lo cual evita que la lectura y el procesamiento de las entradas interrumpan la presentación en pantalla. Este diseño establece un fundamento para una futura expansión hacia pantallas más avanzadas, ya que la distinción entre la lógica de decisión y la interfaz visual es un aspecto crucial en sistemas complejos.

Los mecanismos clave de control de proyección implementados son:

Control por Timestamp: Cada detección y la última proyección activa tienen asignada una marca de tiempo. Este sistema emplea esos timestamps para garantizar que únicamente se tomen en cuenta las detecciones más recientes, lo que permite evitar la proyección de señales obsoletas y la repetición innecesaria de una misma señal, a menos que haya habido un cambio significativo en el entorno vehicular. Esta funcionalidad es vital para la eficiencia y reactividad del sistema.

Proyección por Hilo Separado: La responsabilidad de presentar las imágenes de las señales en el sistema de mensajería variable HDMI se asigna a un hilo de ejecución separado (threading.Thread). Este hilo es el encargado de manejar el ciclo de proyección para una categoría específica (por ejemplo, CAR, TRUCK). Al operar en un hilo distinto, la visualización de las imágenes se realiza de manera fluida y constante, sin perturbar el ciclo principal dedicado a la lectura y evaluación de las detecciones.

```
{
    "clase": "car",
    "confianza": 0.52,
    "coordenadas": {
        "x1": 187,
        "y1": 255,
        "x2": 235,
        "y2": 327
    },
    "timestamp": "2025-07-16T16:59:30.618182"
},
```


Figura 4.2.5: *fragmento del archivo vehiculos_ detectados.json*

Detención Controlada (stop_event): Para garantizar transiciones fluidas y prevenir conflictos entre proyecciones, se utiliza un mecanismo conocido como .evento de detención"(threading.Event). Cuando el sistema identifica un cambio en el tipo de vehículo dominante, es decir, cuando se activa una nueva clave, se envía una señal al stop_event del hilo de proyección anterior. Esto interrumpe la secuencia de señales que estaba en curso y permite que un nuevo hilo comience a proyectar las señales asociadas con la nueva clave, asegurando que en cada momento solo haya una secuencia de señales activa.

Temporización de Proyección: Cada señal en una secuencia de visualiza en el monitor durante un lapso constante de 5 segundos. Este proceso es gestionado a través de la función cv2.imshow() junto con una pausa (cv2.waitKey()) en el hilo de proyección, lo que garantiza que cada señal sea claramente visible antes de que la siguiente se muestre en el ciclo o se inicie una nueva secuencia.

La Figura 4.2.6 muestra un segmento de código de la función mostrar_imagen(), la cual se encarga de la proyección individual de cada señal. Por otro lado, la Figura 4.2.7 presenta la implementación de la lógica de decisión de señal; en este contexto, la función obtener_clave_a_proyectar actúa como el elemento central de la lógica del sistema de proyección de señales viales. Su función es establecer qué grupo de señales de tránsito debe mostrarse, basándose en el análisis de los vehículos que son identificados por un sistema de visión artificial.

4.2.6. Visualización Estadística del Sistema de Reglas

Para ofrecer una perspectiva cuantitativa sobre cómo se comporta el sistema de reglas lógicas y la frecuencia de activación de cada tipo de señal, se incorporó una herramienta de visualización estadística interna. Esta herramienta, creada con bibliotecas de Python y Matplotlib, facilita la creación de gráficos de barras que muestran con qué frecuencia se activaron las distintas reglas lógicas y, como resultado, cuántas veces las señales relacionadas con cada categoría de vehículo fueron proyectadas en el monitor.

```python
# Mostrar imagen en su tamaño original y registrar métrica
def mostrar_imagen(nombre_archivo, clave):
    ruta_imagen = os.path.join(IMG_FOLDER, nombre_archivo)
    img = cv2.imread(ruta_imagen)
    if img is None:
        print(f"Imagen no encontrada: {nombre_archivo}")
        return

    monitor = obtener_monitor_hdmi()
    screen_x, screen_y = monitor.x, monitor.y

    print(f" Mostrando imagen en su tamaño original: {nombre_archivo}")
```

Figura 4.2.6: *Fragmento de Código Python de la Función mostrar_ imagen()*

```python
#  Lógica de decisión de señal
def obtener_clave_a_proyectar(clases_detectadas):
    tiempo_desde_ultimo = time.time() - ultimo_timestamp_detectado

    if len(clases_detectadas) >= 2:
        return "TODAS_VARIAS"
    elif len(clases_detectadas) == 1:
        return clases_detectadas[0]
    elif tiempo_desde_ultimo > 2:
        return "NADA"
    else:
        return None
```

Figura 4.2.7: *fragmento del archivo logica de decisión*

Entrada: La herramienta consume los datos de un archivo metricas.json, donde se registra cada activación de una regla lógica con su respectivo timestamp.

Salida: El resultado es un gráfico de barras que representa visualmente la frecuencia de activación de cada regla (ej, CAR, TRUCK, BUS, MOTORBIKE, NADA, TODAS_VARIAS), permitiendo identificar qué tipos de tráfico o condiciones de vía fueron predominantes durante el período de monitoreo.

Esta funcionalidad estadística es valida para:

Monitorear el rendimiento: Comprender la distribución de los escenarios detectados por el sistema.

Análisis de patrones de tráfico: Obtener información sobre la composición vehicular o la ocurrencia de situaciones específicas (ej., alta frecuencia de "NADA"si el tráfico es bajo).

Optimización futura: Identificar qué reglas se activan más o menos, lo que podría influir en futuras decisiones de diseño o en la asignación de recursos.

La Figura 4.2.8 muestra un ejemplo de este gráfico de barras, ilustrando la frecuencia de proyección de señales por clase, lo cual proporciona una perspectiva agregada del comportamiento del sistema a lo largo del tiempo de operación.

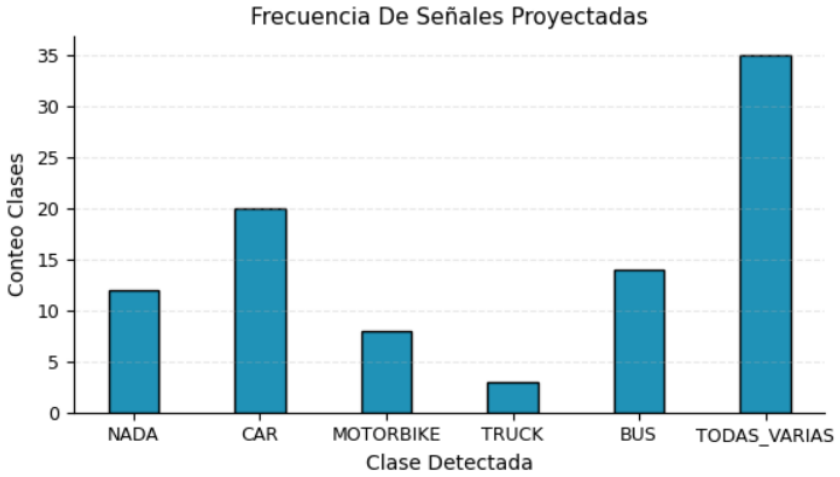

Figura 4.2.8: *Diagrama de Barras Frecuencia de Proyección*

4.3. Interfaz Gráfica De Usuario (GUI) Monitoreo

La fase final del proyecto culmina en la creación de una interfaz gráfica de usuario (GUI) web que es interactiva, concebida como el núcleo para la gestión y supervisión de todo el sistema inteligente encargado de proyectar señales viales. Esta GUI no solo permite una visualización en tiempo real del funcionamiento del sistema, sino que también incluye capacidades para controlar, configurar y analizar métricas recopiladas, promoviendo así la integración de los módulos de detección de vehículos y de reglas lógicas. Desarrollada con el framework Flask para la lógica del backend y utilizando HTML para la presentación del frontend, esta interfaz constituye un elemento esencial que confirma tanto la operatividad como la aplicabilidad práctica del sistema propuesto. Su diseño se orienta a facilitar un control intuitivo y monitorización constante en un entorno de uso real, sirviendo como la entrada principal para usuarios autorizados.

4.3.1. Descripción General y Arquitectura de la Interfaz

La interfaz gráfica de usuario fue desarrollada como un centro de control integral, permitiendo a los usuarios interactuar de manera fluida con todos los componentes del sistema de mensajería variable inteligente. Su estructura basada en la web permite el acceso desde ubicaciones remotas y ofrece la posibilidad de escalar, lo que hace factible llevar a cabo el monitoreo y la gestión desde cualquier dispositivo que tenga un navegador compatible.

Las características esenciales y la disposición de la interfaz están pensadas para mostrar de manera precisa el estado actual del sistema así como sus métricas

acumuladas. La disposición en diferentes vistas y módulos funcionales asegura una navegación intuitiva y una experiencia de usuario optimizada. La interacción entre el frontend y el backend se realiza a través de rutas de Flask, las cuales se comunican con archivos Python (app.py, Modelo.py, Reglas.py) para actualizar la información visualizada de manera dinámica, utilizando formatos como JSON, CSV y PNG.

Se aclara que la implementación del sistema de visualización se realizó en un entorno local de desarrollo. La URL de IP interna (192.168.xxx.xxx) utilizada corresponde a una conexión local entre los componentes principales del proyecto, específicamente el servidor de procesamiento y la interfaz de visualización. Este tipo de configuración fue seleccionada intencionalmente, ya que permite una comunicación de baja latencia y un rendimiento óptimo durante la fase de pruebas y depuración.

La finalidad de esta metodología fue aislar el sistema de variables externas y asegurar que las pruebas reflejaran con precisión el comportamiento del prototipo sin las complejidades inherentes a un despliegue en red. Es fundamental destacar que el alcance de este trabajo de grado se limita al desarrollo, prueba e implementación de un prototipo funcional en un ambiente controlado. El objetivo del proyecto no incluye la puesta en producción a gran escala con un dominio público o una salida a Internet. La metodología de implementación se centró en demostrar la viabilidad técnica y el funcionamiento del sistema en un entorno real, como se evidencia en el grabación del Anexo D (**Anexo Video SMV IA**).

Este enfoque nos permitió validar la integración de los componentes de hardware y software y confirmar que el concepto del proyecto es funcional. Por lo tanto, el uso de una IP interna no es una limitación, sino una elección metodológica que se alinea con el propósito de la investigación, el cual es crear y validar un prototipo, no abordar los aspectos de infraestructura de red para un despliegue comercial o a nivel de producción.

Proceso de Acceso y Autenticación del Sistema

La seguridad y el acceso controlado son aspectos fundamentales en cualquier sistema de monitoreo. Para garantizar que solo el personal autorizado pueda interactuar con el sistema de detección y proyección de señales viales, se implementó un robusto proceso de autenticación de usuarios.

4.3.2. Vista de Inicio

La interfaz comienza con una página de inicio (inicio.html) diseñada para presentar la identidad institucional del proyecto. Esta vista actúa como la primera interacción del usuario con el sistema, proporcionando información esencial y puntos de acceso principales.

Identidad Institucional: La página exhibe de manera prominente la imagen institucional de IT VIAL SAS, reforzando la marca y el contexto empresarial del proyecto.

Eslogan y Descripción: Un eslogan pertinente y una breve descripción de la empresa se incluyen para contextualizar al usuario sobre la misión y capacidades de IT

VIAL S.A.S en el ámbito de la ingeniería vial inteligente.

Opciones de Navegación Inicial: Se ofrecen dos botones principales que dirigen al usuario hacia destinos clave.

Web Vial: Este botón redirige al usuario al sitio web oficial de IT VIAL SAS, permitiendo el acceso a información corporativa y otros proyectos de la empresa.

Iniciar Sesión: Este es el punto de entrada para acceder a las funcionalidades de monitoreo. Al hacer clic, el usuario es dirigido a la página de autenticación, la cual es la puerta de acceso al panel de control principal.

La Figura 4.3.1 muestra una captura de pantalla de la página de inicio, ilustrando su diseño y los elementos interactivos disponibles para el usuario.

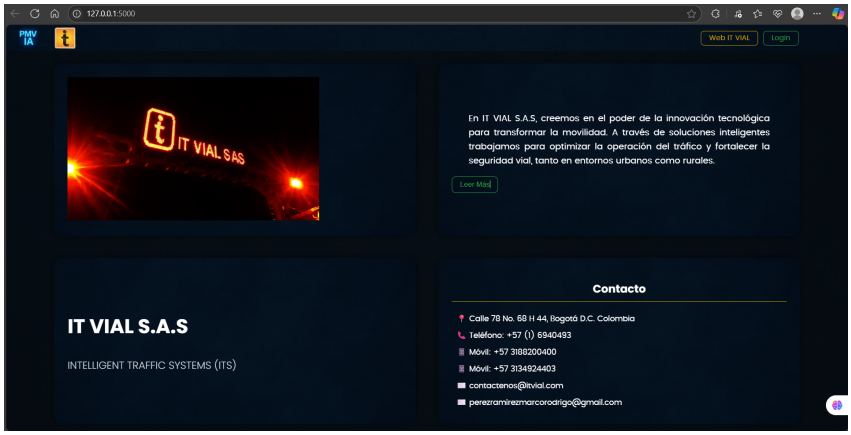

Figura 4.3.1: *Página Inicio GUI Monitoreo*

4.3.3. Módulo de Autenticación

La seguridad del sistema se fundamenta en un módulo de autenticación implementado en la vista login.html. Este módulo es el filtro de acceso al panel de control, garantizando que solo el personal autorizado con credenciales válidas pueda proceder.

Credenciales Requeridas: Para acceder, el usuario debe introducir un nombre de usuario y una contraseña predefinidos. Para esta implementación se generó únicamente un usuario y clave válidas para el acceso a la interfaz propia de monitoreo del sistema.

Implementación Segura con Flask Sessions: La gestión de la autenticación se realiza de manera segura a través de sesiones de Flask. Esto significa que, al ingresar correctamente sus credenciales, se establece una sesión en el servidor que permanece activa durante el tiempo de navegación, lo que elimina la necesidad de autenticarse repetidamente. Las sesiones no solo aumentan la seguridad, sino que también mejoran la experiencia del usuario.

Sistema de Mensajes Flash: Para proporcionar retroalimentación al usuario, se utiliza el sistema de flash messages de Flask. Esto permite mostrar alertas temporales en la interfaz, como mensajes de Çredenciales inválidas.[en] caso de un intento fallido de inicio de sesión o confirmaciones de éxito.

La Figura 4.3.2 presenta una captura de pantalla de la página de inicio de sesión, mostrando el formulario de entrada de credenciales y un posible mensaje de alerta.

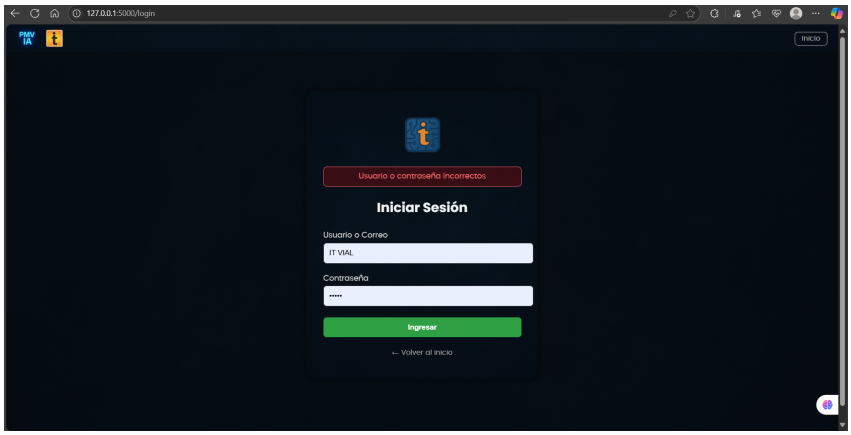

Figura 4.3.2: *Página Login GUI Monitoreo*

4.3.4. Panel Principal de Control

Una vez que el usuario ha completado el proceso de autenticación con éxito, es llevado al panel principal (index.html), que actúa como el centro de control y supervisión del sistema. Esta interfaz ha sido creada para ofrecer un acceso ágil y bien organizado a todas las funciones operativas del proyecto, las cuales están divididas en cuatro módulos principales.

Cada uno de estos módulos está claramente designado y vinculado a su correspondiente plantilla HTML y a funcionalidades específicas del backend, lo que facilita una navegación fluida para el usuario que ha sido autorizado. Los módulos son:

Ver Tránsito: Dedicado al monitoreo en tiempo real de las detecciones vehiculares y la proyección dinámica de señales.

Ver Métricas: Proporciona acceso a las métricas de rendimiento tanto del modelo de detección como del sistema de proyección de señales.

Historial: Permite consultar un registro detallado de todas las señales proyectadas a lo largo del tiempo.

Aprendizaje del Sistema: Ofrece herramientas para la gestión y prueba de modelos personalizados, facilitando la mejora continua del sistema de detección.

La Figura 4.3.3 muestra una captura de pantalla del panel principal, destacando la organización de los módulos y los puntos de acceso a cada funcionalidad.

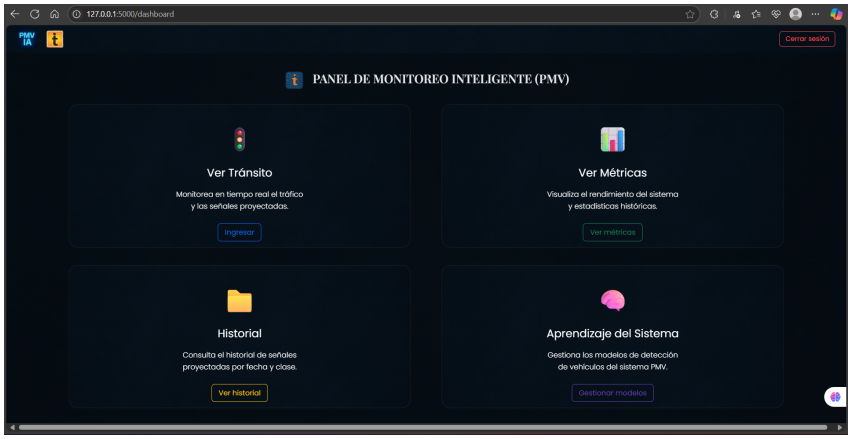

Figura 4.3.3: *Página Dashboard GUI Monitoreo*

4.3.5. Módulo de Monitoreo de Tránsito

El componente Ver Tránsito (ver_transito.html) constituye el núcleo operativo de la interfaz gráfica de usuario, ofreciendo una representación en tiempo real del tránsito de vehículos detectado y la reacción del sistema de proyección de señales. Este componente es esencial para el seguimiento en directo de las condiciones del tráfico y para la verificación instantánea del funcionamiento del sistema. Sus funciones principales son las siguientes:

Monitoreo en Tiempo Real: La interfaz permite observar de manera continua las detecciones de vehículos realizadas por el modelo de detección de vehículos y las señales de tránsito que están siendo proyectadas en el sistema de mensajería variable HDMI en respuesta a dichas detecciones.

Control de Operación: Se proporcionan dos botones interactivos que permiten al usuario gestionar el flujo de datos y la proyección:

Iniciar detección: Este botón activa el módulo de detección de objetos (implementado en Modelo.py). Al ser presionado, se inicia el proceso de captura y detección de vehículos a partir de la fuente de video configurada.

Iniciar proyección: Este botón se encarga de poner en marcha el sistema de reglas lógicas y representación visual, conforme a lo que se encuentra en Reglas.py. Su función principal es comenzar la visualización activa de las señales de tráfico asociadas a las detecciones. Para lograrlo, utiliza programación en hilos y la biblioteca OpenCV, lo que permite la edición y presentación de imágenes en el panel de control.

Flujo de Video en Tiempo Real: La GUI integra un feed de video en tiempo real, generado por la función generar_frames() dentro de Modelo.py. Este stream de video muestra las imágenes con los bounding boxes y etiquetas de los vehículos detectados, proporcionando una confirmación visual instantánea de la capacidad de detección del modelo.

Visualización de Señales Proyectadas en Vivo: Complementando el feed de detección, se integra una visualización en vivo de las señales que el sistema está proyectando en el monitor. Esto se logra a través de la función generar_frames_reglas() en Reglas.py, permitiendo al usuario observar directamente qué señal está activa en el SMV virtual.

La Figura 4.3.4 muestra una captura de pantalla del módulo Ver Tránsito, ilustrando la interfaz con el feed de video de detección y la previsualización de las señales proyectadas.

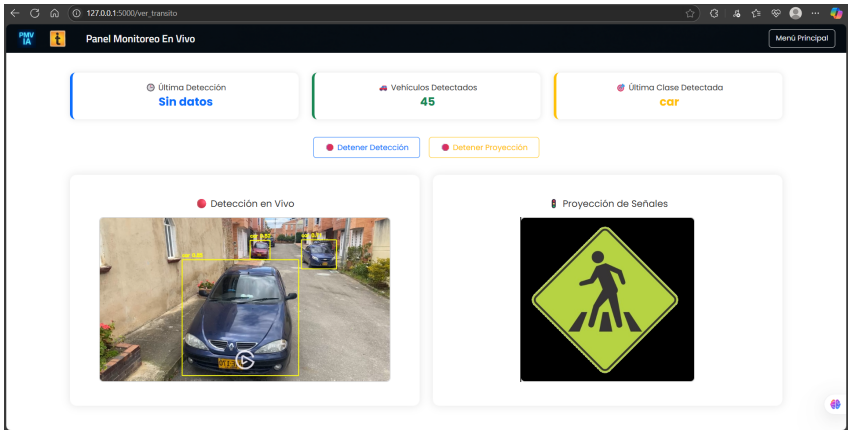

Figura 4.3.4: *Página Ver Tránsito GUI Monitoreo*

4.3.6. Módulo de Visualización de Métricas

El componente Ver Métricas (metricas.html) es esencial para la valoración del rendimiento a largo plazo del sistema de monitoreo. Ofrece una perspectiva integrada de la información producida por el modelo de detección de objetos y el subsistema de reglas lógicas, convirtiendo datos en crudo en información útil mediante gráficos y Indicadores Clave de Desempeño (KPIs).

Este módulo carga y presenta dos tipos principales de información:

4.3.7. Métricas del Modelo de Detección YOLOv11n

La plataforma permite la carga y visualización de las métricas de rendimiento del modelo, cruciales para evaluar la precisión y confiabilidad en la detección de vehículos. Esta información se obtiene directamente del archivo results.csv, que se crea durante las fases de entrenamiento y evaluación del modelo. Las visualizaciones automáticas producidas por el sistema, ya sea a través de matplotlib o directamente desde los registros de Ultralytics, se presentan en esta sección e incluyen:

confusion_matrix.png: Matriz de confusión (normalizada) que detalla los aciertos y errores de clasificación para cada clase vehicular.

F1_curve.png: Curva F1, que muestra el equilibrio entre precisión y recall a diferentes umbrales de confianza.
P_curve.png: Curva de Precisión, ilustrando la proporción de detecciones correctas.

PR_curve.png: Curva de Precisión-Recall, un indicador clave del rendimiento del detector.

R_curve.png: Curva de Recall, mostrando la proporción de objetos reales detectados.

results.png: Un gráfico consolidado que resume las métricas de entrenamiento y validación a lo largo de las épocas (pérdidas, mAP, etc.).

La Figura 4.3.5 y la Figura 4.3.6 presentan una selección de estas gráficas generadas automáticamente, ofreciendo una visión completa del rendimiento del modelo YOLOv11n.

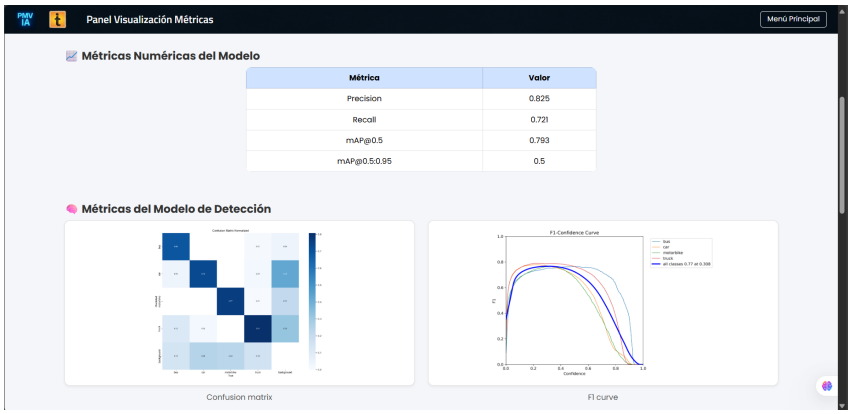

Figura 4.3.5: *Metricas Modelo Detección de Vehículo*

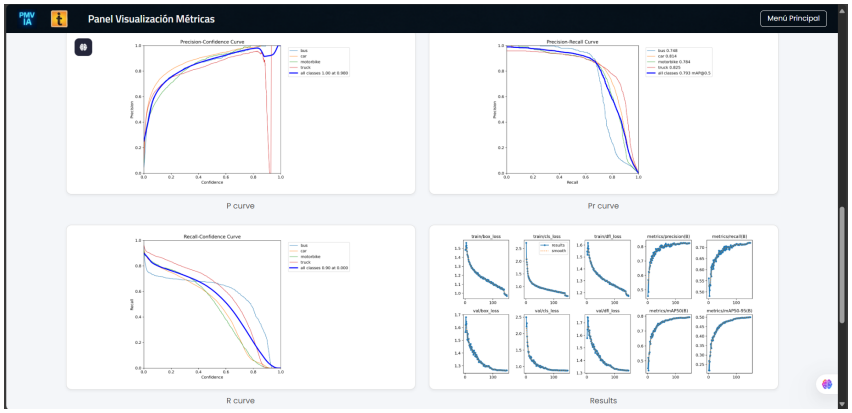

Figura 4.3.6: *Metricas Modelo Detección de Vehículos*

4.3.8. Resultados Métricas del Modelo de Detección

Métrica	Valor	Descripción
Precisión	0.825	El 82.5 % de las predicciones positivas del modelo son correctas.
Recall	0.721	El modelo detecta aproximadamente el 72.1 % de los casos positivos reales.
mAP @0.5	0.793	Media de precisión en diferentes clases con un umbral de confianza de 0.5.
mAP @0.5:0.95	0.500	Media de precisión considerando múltiples umbrales de confianza, indicando el rendimiento general promedio del modelo.

Tabla 4.3.1: Métricas de Evaluación Numéricas del Modelo de Detección

Matriz de confusión

Visualiza la cantidad de verdaderos positivos, falsos positivos, verdaderos negativos y falsos negativos para facilitar la comprensión del rendimiento del modelo en la diferenciación entre las diversas clases. La mayoría de los datos reflejados en la gráfica *Confusion matrix*, que se aprecia en la Figura 4.3.5, muestra valores que oscilan de medios a altos en la diagonal, lo cual indica que se han logrado clasificaciones correctas. No obstante, se observan ciertas confusiones que podrían ser mejoradas. Esta matriz indica que el modelo se desempeña bastante bien al identificar las clases principales, aunque algunas confusiones todavía se manifiestan entre clases semejantes. Su desempeño es fiable.

Curva F1-Confidence:

Muestra cómo varía la puntuación F1 (balance entre precisión y recall) cuando se cambia el umbral de confianza. Valores más altos indican mejor equilibrio y desempeño del modelo. La curva *F1 curve* observada en la Figura 4.3.5 muestra cómo cambia el F1 en función de diferentes niveles de confianza. Los valores están en torno a 0.177 y 0.308 en diferentes clases, lo cual indica que la precisión y recall combinados proporcionan datos aceptables, pero aún pueden mejorar en algunos niveles de confianza. La curva muestra que la capacidad de equilibrio entre precisión y recall todavía tiene margen para ajustarse y mejorar; podría beneficiarse de un ajuste en el umbral de confianza o en los hiperparámetros del modelo.

Precision-Confidence Curve (Curva de Precisión-Confianza)

Esta gráfica ilustra la conexión entre la precisión, que se refiere a la exactitud de las predicciones correctas, y la confianza del modelo en estas predicciones. Una curva que se acerque al área superior izquierda indica una mejor capacidad del modelo para realizar predicciones precisas con un alto grado de confianza. En la gráfica conocida como *P curve* que se presenta en la Figura 4.3.6, diferentes líneas representan variados modelos o categorías. La curva correspondiente a las clases que van de 1.00 a 0.980 se sitúa a un nivel bastante elevado, lo cual sugiere que se trata de un rango de confianza con una precisión notable.

Precision-Recall Curve (Curva de Precisión-Recall)

Esta gráfica analiza la conexión entre la precisión, que se define como la proporción de verdaderos positivos en comparación con todos los positivos que el modelo predice, y el recall, que mide la proporción de verdaderos positivos identificados frente al total real de positivos. Un valor mayor en el área bajo la curva sugiere un rendimiento superior del modelo en sus capacidades de clasificación. En el gráfico denominado *Pr curve* que aparece en la Figura 4.3.6, los modelos que se aproximan más a la esquina superior derecha evidencian un mejor balance entre precisión y recall, lo que resulta ideal en tareas de clasificación.

Recall-Confidence Curve (Curva de Recall-Confianza)

La gráfica *R curve* visualizada en la Figura 4.3.6 muestra cómo varía el recall del modelo con diferentes niveles de confianza en las predicciones. A medida que aumenta la confianza, el recall generalmente disminuye, ya que el modelo se vuelve más selectivo. La línea que desciende indica que, con mayor confianza, el porcentaje de detecciones correctas disminuye. Esto ayuda a entender el trade-off entre confianza y sensibilidad del modelo.

4.3.9. Métricas del Sistema de Proyección de Señales

Además de las métricas del modelo de detección, este módulo ofrece una visión estadística del comportamiento del sistema de reglas lógicas. Estos datos se obtienen del archivo metricas.json, que registra la frecuencia y el tipo de señales proyectadas; las visualizaciones específicas para el sistema de proyección son:

frecuencia proyecciones.: Gráfico de barras que muestra la frecuencia absoluta o relativa con la que cada regla lógica (y, por ende, cada tipo de señal) fue activada.

tendencia proyecciones: Gráfico que ilustra la tendencia de activación de las reglas a lo largo del tiempo, permitiendo identificar patrones temporales en el tráfico o en el uso de señales.

La Figura 4.3.7 muestra un ejemplo de estas gráficas de métricas del sistema de proyección, ofreciendo un panorama del uso dinámico del SMV.

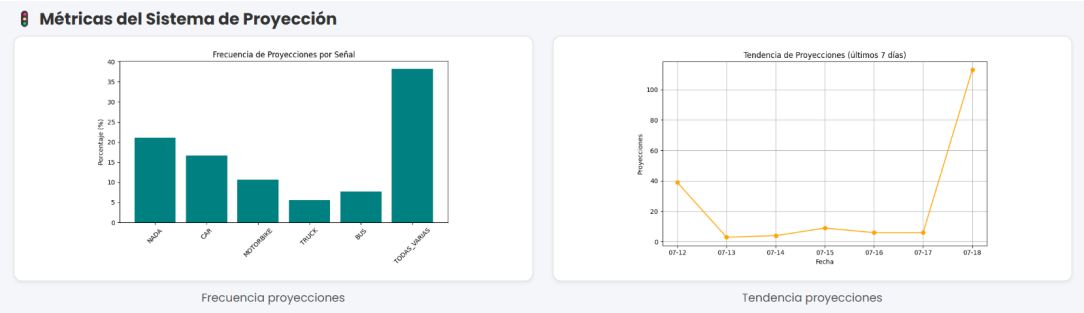

Figura 4.3.7: *Metricas Modelo Detección de Vehículos*

4.3.10. Indicadores Clave de Rendimiento (KPIs)

Para una síntesis rápida del estado operativo del sistema, el módulo de métricas presenta una serie de Indicadores Clave de Rendimiento (KPIs) actualizados dinámicamente:

Total de Proyecciones: El número acumulado de veces que el sistema ha activado una secuencia de proyección de señales.

Vehículos Detectados: El conteo total de vehículos identificados por el modelo de detección.

Clase Más Frecuente: La categoría vehicular (o regla lógica) que ha sido activada con mayor recurrencia.

Última Actualización: El timestamp de la última vez que los datos de las métricas fueron actualizados.

La generación de todas estas gráficas y KPIs se realiza automáticamente mediante funciones Python integradas (generar_graficas_metricas()) que utilizan la librería matplotlib, asegurando que la visualización de los datos sea siempre coherente y esté al día. La Figura 4.3.8 ilustra la presentación de estos KPIs en la interfaz.

Figura 4.3.8: *Presencia de KPIs en la Interfaz*

4.3.11. Módulo de Historial de Señales Proyectadas

El componente Historial (historial.html) ofrece capacidades de análisis y auditoría del pasado, lo que permite a los usuarios acceder a un registro secuencial de todas las señales emitidas por el sistema. Esta funcionalidad es crucial para la capacidad de seguimiento, la validación del comportamiento a través del tiempo y la identificación potencial de patrones de tráfico o eventos extraordinarios.

Visualización Tabular/Listado: Los datos del historial se organizan en un formato definido, generalmente en forma de tabla o lista, lo que simplifica la interpretación y el estudio de grandes cantidades de información. Cada registro contiene información como la señal emitida, la categoría de detección que la desencadenó y la marca de tiempo precisa de la emisión.

Funcionalidades de Filtrado Avanzado: Para mejorar la capacidad de análisis y la eficiencia en la búsqueda, el módulo incorpora herramientas de filtrado que permiten al usuario refinar los resultados del historial. Se implementan filtros por:

Clase de Detección: Permite al usuario seleccionar una o varias clases específicas (ej., CAR, TRUCK, BUS, MOTORBIKE) para visualizar solo las proyecciones relacionadas con esas categorías vehiculares.

Rango de Fechas: Habilita la selección de un período de tiempo específico, mostrando únicamente las proyecciones que ocurrieron dentro de las fechas de inicio y fin definidas. Esto es crucial para analizar el comportamiento del sistema en intervalos específicos, como durante picos de tráfico o después de actualizaciones.

Fuente de Datos (log_senales.json): Toda la información del historial se recupera del archivo log_senales.json. Este archivo actúa como una base de datos plana, donde cada evento de proyección de señal se registra de forma secuencial con todos los metadatos necesarios para el filtrado (clase, timestamp).

Acceso mediante API RESTful: La obtención y el filtrado de datos del historial se llevan a cabo de manera eficaz y modular mediante una ruta API en Flask (/api/historial). Esta estructura permite que la interfaz gráfica de usuario solicite datos y aplique filtros de manera asíncrona, evitando la necesidad de recargar la página completa, lo que también simplifica la posibilidad de integración con otras herramientas o sistemas en caso de ser requerido. El backend en Python procesa el JSON y lo presenta a la interfaz conforme a los criterios de filtrado establecidos. En la Figura 4.3.9 se puede observar una imagen del módulo del historial, donde se muestra cómo se exponen los registros de señales proyectadas junto con las opciones de filtrado disponibles.

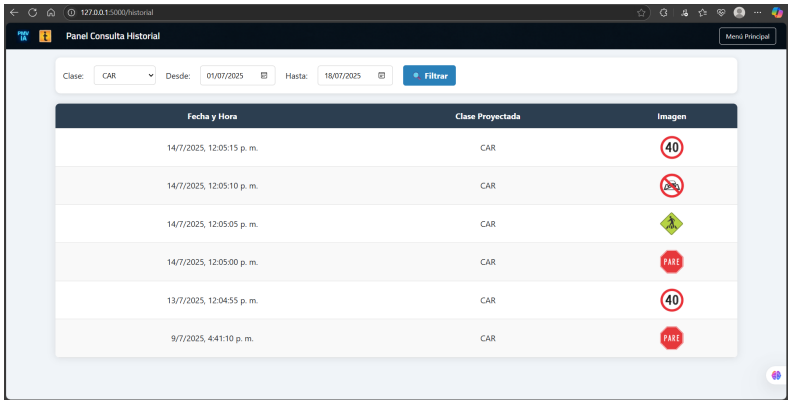

Figura 4.3.9: *Página Historial GUI Monitoreo*

4.3.12. Módulo de Aprendizaje y Gestión del Sistema

El módulo Aprendizaje del Sistema (modelo.html) es una funcionalidad avanzada que dota al sistema de la capacidad de evolucionar y adaptarse, permitiendo la gestión

directa de los modelos de detección que el sistema utiliza. Esta sección es crucial para la mejora continua del rendimiento del sistema sin necesidad de una intervención profunda en el código base; las funcionalidades clave que este módulo habilita son:

Carga de Modelos Personalizados (/upload_model): Los usuarios con permisos pueden cargar nuevos modelos personalizados en formato .pt a través de la interfaz web. Esto posibilita la actualización o sustitución del modelo de detección que está en uso, introduciendo versiones mejoradas o modelos que han sido entrenados con conjuntos de datos recientes, todo sin interrumpir la función principal del sistema. La interfaz se encarga de garantizar una carga segura de los archivos, almacenándolos en la carpeta /models.

Prueba de Modelos (/probar_modelo): Una vez subido un modelo, el usuario tiene la opción de elegir un modelo determinado para realizar pruebas. Esta característica activa una sesión de inferencia de prueba, que puede llevarse a cabo en un entorno controlado o con un conjunto de datos de pruebas específico, permitiendo evaluar el rendimiento del nuevo modelo antes de su implementación completa.

Detención de Modelos (/detener_modelo): Proporciona la capacidad de detener el modelo de detección actualmente activo. Esto es útil para tareas de mantenimiento, actualización o para cambiar a un modelo diferente de forma segura.

Eliminación de Modelos (/eliminar_modelo/<nombre>): Permite la eliminación de modelos antiguos o no deseados de la carpeta /models, ayudando a mantener el sistema organizado y liberando espacio de almacenamiento. Esta función incluye un manejo seguro para evitar eliminaciones accidentales.

La administración de la carpeta /models se realiza directamente en esta sección de la interfaz gráfica de usuario, facilitando la gestión del ciclo de vida del modelo. La interacción con las rutas claves de Flask (/upload_model, /probar_modelo, /detener_modelo, /eliminar_modelo/) garantiza que las operaciones se lleven a cabo de manera controlada y segura. La Figura 4.3.10 presenta una captura de pantalla del módulo de Aprendizaje del Sistema, destacando las funcionalidades para cargar, probar y gestionar los modelos.

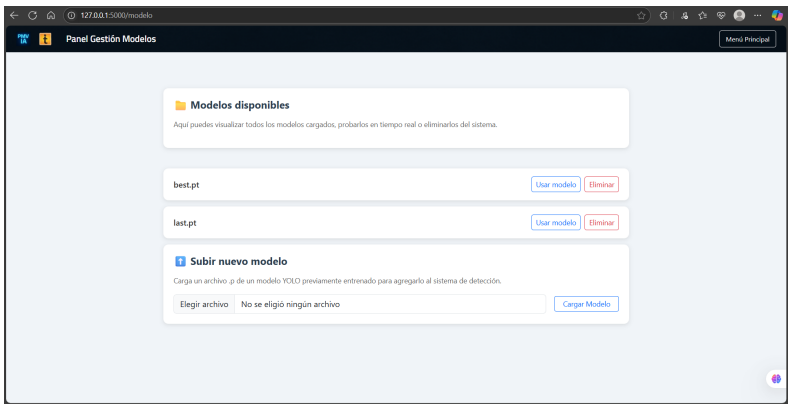

Figura 4.3.10: *Página Aprendizaje del Sistema*

4.3.13. Observaciones Generales de la Implementación de la GUI

La implementación de la GUI de monitoreo ha demostrado ser un componente vital para la operatividad y gestión del sistema, generando observaciones clave que resaltan su robustez y diseño:

Conectividad Integral de Datos: Toda la información mostrada en la interfaz, que incluye las detecciones en tiempo real y las métricas del pasado, está vinculada a través de rutas de Flask y funciones escritas en Python. Estas funciones desempeñan un papel clave en la lectura, el procesamiento y la actualización de los datos almacenados en archivos JSON y CSV (vehiculos_detectados.json, metricas.json, log_senales.json, results.csv), asegurando que la información presentada sea coherente y esté al día.

Modos de Operación Versátiles: El sistema es capaz de manejar fluidamente tanto el modo de detección automática (alimentado por la cámara y el modelo YOLO) como el modo de simulación (entrada manual), manteniendo la integridad y la lógica de las proyecciones y el registro de datos en ambos escenarios. Esta dualidad es fundamental para la flexibilidad en entornos de prueba y despliegue.

Diseño Interactivo y Extensible: El diseño de la interfaz, aunque funcional, es objetivamente personalizable. La base HTML, combinada con la capacidad de integrar CSS y frameworks como Bootstrap, permite una mejora continua de la experiencia de usuario, adaptándola a estándares de diseño más modernos y amigables.

Estructura de Proyecto Modular: La organización del proyecto en archivos Python separados (app.py, Modelo.py, Reglas.py) y una estructura de carpetas bien definida (templates/, static/, models/) facilita la depuración, el mantenimiento y la escalabilidad futura. Esta modularidad es un principio clave de la ingeniería de software que contribuye a la robustez del sistema.

La Figura 4.3.11 ilustra la estructura de directorios del proyecto, resaltando la organización modular de los componentes.

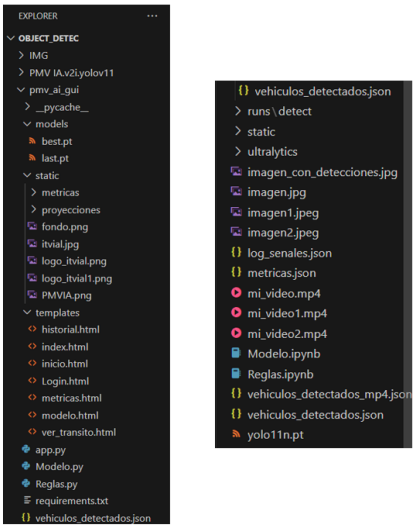

Figura 4.3.11: *Estructura de Directorios*

4.4. Resultados Generales e Integración del Sistema

La minuciosa evaluación de cada elemento del Sistema de Mensajería Variable Inteligente (SMV AI) que abarca desde la fase de percepción mediante el modelo de detección de objetos, pasando por la capa cognitiva que utiliza reglas lógicas para la señalización, hasta la funcionalidad completa de la interfaz gráfica del usuario para su control y supervisión ha llevado a la validación integral y satisfactoria del prototipo realizado. Los resultados que se presentan en este capítulo no solo validan la operatividad de cada módulo de manera independiente, sino que, aún más importante, evidencian la exitosa integración y la sinergia funcional de todo el sistema.

El modelo encargado de detectar (vehículos) ha demostrado ser una solución eficiente y sólida para la clasificación en tiempo real de los vehículos, sirviendo como la base de datos esencial para la toma de decisiones. Su desempeño, evaluado mediante métricas como Precisión, Recall y mAP, respalda su utilidad en un entorno vial cambiante. La Figura 4.4.1 muestra la proyección dinámica de las señales de tránsito en una pantalla HDMI, así como la visualización concurrente de la detección de vehículos en tiempo real, lo que pone de relieve la capacidad de respuesta del sistema y la coherencia existente entre la percepción y la acción.

Figura 4.4.1: *Proyección Dinámica de Señales de Tránsito*

Basándose en esta perspectiva, el módulo de reglas lógicas ha demostrado su eficacia en la interpretación de las detecciones y en la activación de respuestas de señalización que son coherentes y relevantes según el contexto. La utilización de una arquitectura de procesamiento multihilo ha permitido que estas decisiones se reflejen de forma dinámica y fluida en el sistema de mensajería variable HDMI. La latencia controlada, junto con la habilidad de cambiar entre diversas secuencias de señales en respuesta a las variaciones en el tráfico, son evidencias contundentes de la reactividad del sistema.

Por otro lado, la Interfaz Gráfica de Usuario (GUI) ha logrado centralizar de manera efectiva el control y la supervisión, optimizando así la funcionalidad del proyecto. Esta plataforma web interactiva no solo permite visualizar en tiempo real las detecciones y proyecciones, sino que también ofrece acceso a métricas de rendimiento consolidadas, un historial de eventos y manejo avanzado de modelos de detección. Su diseño intuitivo promueve una interacción sencilla por parte del usuario y pone de manifiesto la capacidad del sistema para ser administrado y monitoreado de forma eficiente. La Figura 4.4.2 ilustra una vista integral de la interfaz de usuario, abarcando varios

aspectos del monitoreo y control del sistema integrado.

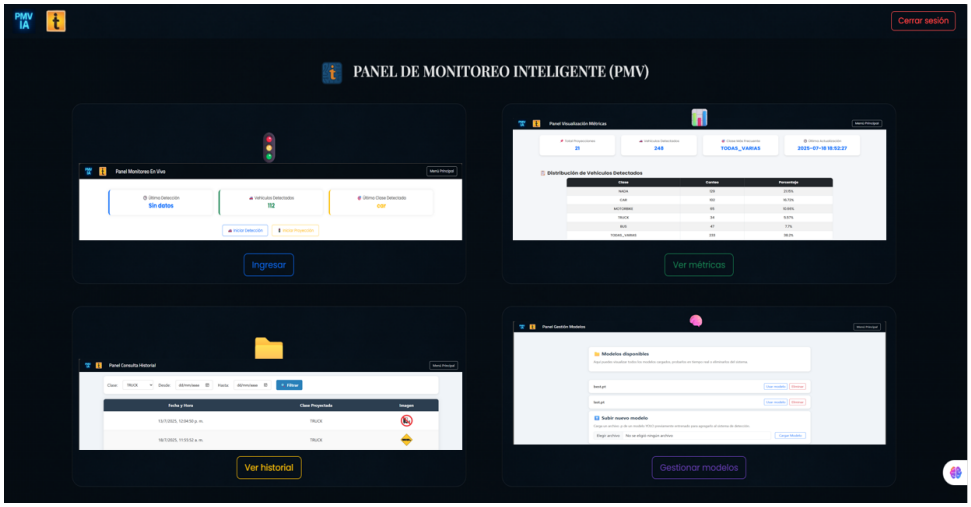

Figura 4.4.2: *Proyección Dinámica de Señales de Tránsito*

En esta sección se presenta una descripción detallada de la implementación y los resultados del sistema de visualización, un componente crítico para la validación y el funcionamiento integral del proyecto. La elección de una arquitectura de visualización dual, compuesta por una interfaz gráfica de usuario y una pantalla física HDMI, se justifica por la necesidad de abordar dos objetivos distintos pero complementarios: el análisis profundo del sistema en un entorno de desarrollo y la demostración de la viabilidad de la solución en un contexto real.

El primer pilar de la visualización es la Interfaz Gráfica de Monitoreo, descrita en detalle en la sección 4.3. Esta herramienta fue diseñada como el centro de control y diagnóstico para el investigador. Su implementación responde a la necesidad de no solo mostrar el resultado final, sino también de interpretar en tiempo real el comportamiento del modelo de detección de objetos.

El segundo pilar es la implementación del Sistema de Mensajería Variable (SMV) físico, que utiliza un monitor HDMI como pantalla de visualización. Este componente representa el producto final del proyecto y es la manifestación tangible de la solución propuesta. La elección de un monitor HDMI, una tecnología estándar y accesible, se justifica por su facilidad de integración y su capacidad para simular un sistema de mensajería variable real. Este monitor no es solo un duplicado de la interfaz gráfica, sino que su propósito principal es demostrar la viabilidad práctica y la aplicabilidad de la solución en la infraestructura vial.

Mediante este monitor, el sistema demuestra su funcionalidad de una manera clara y directa, la detección de un vehículo por el modelo de Machine Learning se correlaciona con las reglas lógicas programadas para generar una proyección dinámica de una señal de tránsito relevante. Esta parte de la implementación evidencia que el sistema no es solo una herramienta de software, sino una solución completa que puede integrarse en un contexto físico. El Anexo D (Anexo Video SMV IA) sirve como prueba de esta

funcionalidad, mostrando cómo el sistema reacciona en un entorno real.

La combinación de la interfaz gráfica de monitoreo y la pantalla física HDMI proporciona una solución de visualización completa. La primera se enfoca en el análisis y control del rendimiento, mientras que la segunda valida la aplicabilidad y el potencial práctico de la solución en un entorno similar al que se enfrentaría en un despliegue real.

En resumen, los hallazgos generales respaldan la factibilidad y el éxito del Sistema de Mensajería Variable Inteligente. Se ha diseñado y desarrollado un prototipo operativo que combina de manera efectiva subsistemas complejos de visión artificial y lógica de control, resultando en una solución innovadora para la señalización dinámica de las vías. Este proyecto no solo respalda la hipótesis propuesta, sino que también establece una base tecnológica robusta y ampliable para investigaciones futuras y posibles implementaciones a gran escala dentro del contexto de los sistemas de transporte inteligentes, representando un progreso significativo en la adaptación de la infraestructura vial a las condiciones de tráfico en Colombia.

Capítulo 5

Conclusiones y Recomendaciones

Este capítulo resume los hallazgos más relevantes del proyecto titulado APLICACIÓN DE MACHINE LEARNING EN SISTEMAS DE MENSAJERÍA VARIABLE (SMV) CON VISUALIZACIÓN DINÁMICA DE SEÑALES DE TRÁNSITO BASADAS EN DETECCIÓN DE VEHÍCULO. Se valida el logro de los objetivos establecidos y se ofrece una visión completa sobre las consecuencias de los resultados adquiridos. Además, se discuten las limitaciones encontradas a lo largo del desarrollo y se presentan recomendaciones estratégicas para futuras investigaciones y desarrollo, con el objetivo de mejorar y ampliar las capacidades del sistema propuesto.

5.1 Conclusiones

La investigación y el desarrollo del sistema han logrado cumplir de manera exitosa con los objetivos establecidos, evidenciando la viabilidad y el potencial de una solución de señalización vial dinámica que utiliza inteligencia artificial. El primer aspecto fundamental para su aprobación se basa en la solidez y eficacia del módulo de detección de vehículos, que se implementa empleando el modelo YOLOv11n. Las evaluaciones y métricas cuantitativas respaldan su habilidad para reconocer y clasificar diversos tipos de vehículos en tiempo real, ofreciendo una base perceptiva confiable, clave para el funcionamiento del sistema. Este rendimiento se valida en su capacidad de distinguir entre diferentes clases de vehículos, lo cual es esencial para tomar decisiones informadas en el futuro.

Con esta base establecida, el diseño e implementación del módulo de reglas lógicas ha mostrado una efectividad notable. Este componente no solo entiende las detecciones de vehículos, sino que también las convierte en acciones de señalización que son coherentes y pertinentes al contexto. La integración de hilos de ejecución ha sido fundamental para asegurar una representación fluida y dinámica de las señales en el monitor HDMI, simulando de manera exitosa un sistema de mensajes variable reactivo. Esto valida la capacidad del sistema para ajustar los mensajes viales según las condiciones del tráfico observadas, gestionando las transiciones visuales con una latencia controlada.

Finalmente, la interfaz gráfica de usuario ha sido el elemento esencial que ha logrado unir y consolidar todos los subsistemas del proyecto. Esta plataforma centralizada y fácil de usar no solo facilita el control, la configuración y la supervisión del sistema en

tiempo real, sino que también proporciona acceso a métricas de rendimiento detalladas, un historial de proyecciones y la capacidad de manejar los modelos de detección. La interfaz comprueba la usabilidad y el funcionamiento integral del sistema para los operadores, evidenciando la integración cohesiva de todos sus componentes y ofreciendo una visión completa de su funcionamiento. En resumen, este proyecto ha conseguido crear un prototipo innovador que, mediante la combinación de tecnologías de última generación, brinda una solución flexible y eficiente para la gestión dinámica del tráfico, estableciendo las bases para su futura escalabilidad.

5.2 Limitaciones y Direcciones Futuras

A pesar del considerable éxito alcanzado en la exhibición del prototipo, es crucial señalar las limitaciones que se encontraron durante su proceso de desarrollo y las pruebas realizadas, ya que estas limitaciones marcan el camino hacia futuras investigaciones y mejoras. En primer lugar, el ambiente de laboratorio donde se realizaron las pruebas, aunque controlado y adecuado para la validación de conceptos, no puede reproducir completamente la complejidad y diversidad de un entorno real de tráfico. Esto abarca aspectos como condiciones meteorológicas desfavorables, cambios bruscos en la luz y la presencia de obstrucciones vehiculares. En segundo lugar, la actual utilización de un monitor HDMI para proyectar señales, aunque funcional en el contexto del prototipo, no aborda las consideraciones específicas de hardware necesarias para los Sistemas de Mensajería Variable de grado industrial, tales como controladores para matrices LED, requisitos de luminosidad para visibilidad durante el día y la noche, así como la resistencia a condiciones climáticas adversas; estos aspectos no fueron contemplados en esta etapa del proyecto.

En lo que respecta al desempeño y la eficacia del modelo de detección de vehículos, se observa una limitación significativa relacionada con el proceso de optimización para situaciones muy concretas, debido a la carencia de recursos computacionales significativos que no se manejan adecuadamente desde las condiciones actuales del prototipo. Por último, y sumamente relevante para una implementación práctica, aunque el modelo de detección muestra una notable precisión, su ejecución en tiempo real presenta una alta exigencia computacional. Esta alta demanda de recursos de hardware es una limitación importante que debe tenerse en cuenta para la implementación a gran escala o en dispositivos embebidos que cuenten con restricciones de costo y tamaño.

A pesar de que estas limitaciones son naturales en la etapa de prototipado y validación de conceptos, no disminuyen la funcionalidad central del proyecto. Por el contrario, actúan como elementos fundamentales para la formulación de futuras líneas de investigación y desarrollo. Abordar estos desafíos permitirá avanzar hacia la creación de un sistema más sólido, escalable y adecuado para su operación en complejas situaciones viales en Colombia.

Referencias Bibliográficas

[1] Congreso de la República de Colombia, "Ley 769 de 2002: Por la cual se expide el Código Nacional de Tránsito Terrestre y se dictan otras disposiciones," agosto 2002. [En línea].

[2] Ministerio de Transporte de Colombia, "Portal Oficial." Ministerio de Transporte de Colombia, 2025. [En línea].

[3] Organización Mundial de la Salud (OMS), *Global status report on road safety 2023*. Ginebra: OMS, 2023. [En línea].

[4] Federal Highway Administration (FHWA), "Use of Machine Learning in Traffic Management," tech. rep., U.S. Department of Transportation, 2021.

[5] International Organization for Standardization (ISO), "ISO 39001:2012 - Road Traffic Safety Management Systems (RTSMS)," 2012.

[6] T. Yu, "MDDFNet: Mamba-based Dynamic Dual Fusion Network for Traffic Sign Detection," *arXiv preprint arXiv:2505.05491*, 2025. [En línea].

[7] M. Mounesh, "Real-Time Traffic Control Using YOLOv5 Vehicle Detection and Dynamic Signal Timing Adjustment." GitHub, 2025. [En línea].

[8] Institute of Electrical and Electronics Engineers (IEEE), "IEEE 802.11p: Wireless Access in Vehicular Environments (WAVE)," 2010.

[9] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (Las Vegas, NV, USA), pp. 779–788, 2016.

[10] FLIR Systems, "Sistemas de Transporte Inteligente (ITS)." FLIR. [En línea].

[11] Aleatica S.A., "Sistemas de transporte inteligente para autopistas." Aleatica. [En línea].

[12] G. Jocher *et al.*, "YOLOv5 – Real-Time Object Detection," *arXiv preprint arXiv:2104.06856*, 2021. [en línea].

[13] Instituto Tecnológico de Aragón (ITA), "Detección de obstáculos en tiempo real para una movilidad inteligente y segura." ITA. [en línea].

[14] Instituto Tecnológico de Aragón (ITA), "Visión artificial para vehículos conectados." ITA. [en línea].

[15] Cadena SER Málaga, "Científicos malagueños diseñan un sistema inteligente de videovigilancia en tiempo real para aeropuertos," *SER Málaga*, May 2025. [en línea].

[16] J. Hernández and et al., "Heimdall: Urban Traffic Monitoring with AI-based Anomaly Detection," *arXiv preprint arXiv:2103.01506*, 2021. [en línea].

[17] Transporte Evolución, "El desafío de la ciberseguridad en los sistemas de transporte inteligente." Transporte Evolución. [en línea].

Datos del Autor

Marco Rodrigo Pérez Ramírez

- **Código de estudiante:** 2297174
- **Programa Académico:** Pregrado
- **Facultad:** Ingeniería Electrónica
- **Universidad:** Universidad Santo Tomás, Seccional Tunja
- **Correo electrónico:** marco.perezr@usantoto.edu.co

- **Resumen biográfico:** Estudiante de Ingeniería Electrónica de la Universidad Santo Tomás, Seccional Tunja. Su formación se centra en el diseño y la implementación de sistemas electrónicos complejos. Sus intereses y experiencia radican en el desarrollo de soluciones basadas en Machine Vision, abarcando la detección, clasificación, segmentación y reconocimiento de patrones en entornos dinámicos. Ha participado en proyectos que implican la instrumentación, la adquisición de datos y el desarrollo de sistemas inteligentes, incluyendo la aplicación de Machine Learning y Deep Learning para el análisis de información visual y la automatización de procesos.

Anexos

Anexo A: Código Fuente Principal

Este anexo presenta los códigos fuente de los módulos clave que forman la base del desarrollo lógico del sistema. En particular, se examinan los archivos Modelo.ipynb y Reglas.ipynb. Estos documentos, que han sido exportados de sus respectivos entornos de Jupyter Notebook, no solo permiten una comprensión exhaustiva de la implementación técnica de la detección de objetos y la lógica decisional, sino que son necesarios para replicar, auditar o ampliar las capacidades del sistema. Debido a su extensión y para conservar la fluidez del documento actual, se incluyen enlaces directos a estos archivos, presuponiendo que el lector tiene acceso a la estructura de carpetas del proyecto.

Código de Modelo.ipynb

El archivo Modelo.ipynb constituye la esencia fundamental de nuestro sistema de visión por computador. Este documento abarca la implementación integral del modelo de detección de objetos, que se basa en la innovadora arquitectura YOLOv11n. Su propósito es procesar de manera inmediata los marcos de vídeo obtenidos, permitiendo la identificación y clasificación eficaz de los vehículos que circulan en la zona de interés. Además, contiene funciones para cargar el modelo previamente entrenado, así como para el preprocesamiento de imágenes, que incluye escalado y normalización; también se encarga de interpretar las salidas generadas por la red neuronal, lo que permite la creación de cuadros delimitadores precisos en torno a cada vehículo detectado, junto con sus correspondientes etiquetas de clase y el nivel de confianza en la detección. Este cuaderno es vital para el módulo de percepción del sistema.

`anexo Modelo.ipynb`

Código de Reglas.ipynb

El documento Reglas.ipynb actúa como el cerebro decisional del proyecto. Su labor principal es dirigir la lógica para la proyección dinámica de señales en el Sistema de Mensajería Variable, que se conecta a través de un monitor HDMI, funcionando como el vínculo entre los datos crudos provenientes de la detección de objetos y las acciones de señalización dirigidas al conductor. Este módulo avanzado procesa los resultados iniciales de la detección de vehículos, que son suministrados por el `Modelo.ipynb`, y aplica un conjunto sofisticado reglas definidas previamente y ajustables. Estas reglas tienen en cuenta múltiples variables en tiempo real, tales como la densidad del tráfico, la velocidad promedio de los vehículos, la ocurrencia de eventos particulares y otras condiciones operativas, para identificar la señal más adecuada y oportuna que debería mostrarse. La adaptabilidad de este módulo permite modificar la reacción del SMV AI ante diferentes circunstancias en la vía y variadas estrategias de gestión del tráfico, garantizando así una respuesta inteligente y contextual.

`anexo Reglas.ip`

Anexo B: Código de la Interfaz Gráfica de Usuario (GUI)

Este anexo presenta un análisis exhaustivo de la arquitectura y el código fuente de la Interfaz Gráfica de Usuario (GUI). Esta GUI ha sido creada utilizando el microframework Flask, que, aunque ligero, es también muy eficaz, sirviendo como el principal punto de interacción para los operadores del sistema. El propósito de este anexo es descomponer la organización modular del proyecto, mostrando la estructura de directorios que da forma a su lógica y recursos. Además, se incluyen descripciones de archivos Python importantes como app.py, modelo.py y reglas.py, y se ilustra el desarrollo del frontend con ejemplos de plantillas HTML, lo que facilita una comprensión clara de cómo los usuarios interactúan con el sistema de monitoreo.

Estructura de Carpetas GUI

La correcta organización de los archivos es fundamental para la mantenibilidad y escalabilidad de cualquier aplicación. La siguiente estructura de directorios ha sido diseñada meticulosamente para ilustrar la organización lógica y física de los componentes de la GUI. Esta disposición facilita una gestión eficiente de la lógica de la aplicación, los recursos estáticos (CSS, imágenes) y las plantillas de visualización, permitiendo un desarrollo colaborativo y una navegación intuitiva dentro del proyecto.

```
/GUI/
 app.py
 Modelo.py
 Reglas.py
 templates/
    inicio.html
    login.html
    index.html
    ver_transito.html
    metricas.html
    historial.html
    modelo.html
 static/
    css/
    js/
    img/
        logo.png
    metricas/
        frecuencia_proyecciones.png
 models/
    best.pt
 logs/
    log_senales.json
 data/
    vehiculos_detectados.json
```

Código de app.py (Aplicación Flask)

El archivo app.py es el núcleo central de la aplicación web Flask, actuando como el punto de entrada principal para todas las interacciones del usuario. Este módulo es responsable de definir todas las rutas URL que los usuarios pueden visitar (ej. /ver _transito), gestionar el proceso de autenticación de usuarios para asegurar el acceso seguro, y coordinar de manera fluida la comunicación entre los componentes del frontend (lo que el usuario ve) y la lógica de backend (la detección de objetos y la proyección de señales). Es aquí donde se establece la conexión inicial con los módulos de visión y reglas, orquestando el flujo de datos y comandos para el funcionamiento en tiempo real.

`anexo app.py`

Código de modelo.py (Módulo de la GUI)

Aunque el modelo principal de detección se encuentra en el archivo general del sistema llamado Modelo.py, este archivo específico de la interfaz gráfica de usuario está diseñado para funcionar como una capa intermedia que simplifica la comunicación y el manejo de datos vinculados al modelo de detección desde la plataforma web. Situado de forma estratégica en el diseño de la GUI, alberga funciones y clases particulares que permiten al frontend realizar peticiones al backend para recibir resultados de detección, enviar comandos para iniciar o detener el procesamiento de video y presentar de manera efectiva la información sobre los objetos detectados a los operadores. Esta pieza es fundamental para la integración de la inteligencia artificial con la experiencia del usuario.

`anexo Modelo.py`

Código de reglas.py (Módulo de la GUI)

De manera similar al módulo modelo.py de la interfaz gráfica, el archivo reglas.py, que forma parte fundamental de la misma, lleva a cabo la lógica requerida para que la interfaz se comunique eficazmente con el sistema de reglas del backend. Su función es esencial para activar y mostrar las señales en el Sistema de Mensajería Variable. Facilita que la interfaz gráfica obtenga las recomendaciones de señal del módulo principal de reglas, envíe instrucciones para su visualización y administre su estado actual, ofreciendo a los operadores un control total sobre las advertencias y mensajes dirigidos al tránsito.

`anexo Reglas.py`

Ejemplo de Plantilla HTML: ver_transito.html

La interfaz de usuario del sistema se desarrolla mediante un conjunto de plantillas HTML que, en conjunción con Flask, presentan la información de forma dinámica y permiten la interacción. Un claro ejemplo de estas plantillas es ver_transito.html, que ha sido elaborada específicamente para el seguimiento en tiempo real del tráfico. Su diseño en HTML permite mostrar un flujo de video en vivo (usualmente a través de un video incrustado), superponer los resultados de detección (marcos y etiquetas) y ofrecer controles fundamentales para que los operadores puedan iniciar, pausar o detener los

procesos relacionados con la detección y proyección de señales. Esta plantilla actúa como un recurso visual clave que permite a los usuarios observar la actividad del sistema de manera rápida y eficaz.

`anexo ver_transito.html`

Anexo C: Manual de Usuario del Sistema PMV AI

Este manual ofrece una orientación completa sobre la instalación, ajuste y funcionamiento del Sistema de Mensajería Variable Inteligente. Su propósito es convertirse en un recurso esencial para los usuarios finales, los operadores y el personal técnico que necesiten interactuar con el sistema en su uso diario, así como en actividades de mantenimiento básico. La meta es garantizar una implementación y un uso óptimo del sistema, maximizando su eficacia en la gestión del tráfico.

1. Requisitos del Sistema

Para garantizar el funcionamiento óptimo y estable del sistema, es imprescindible que el entorno cumpla con los siguientes requisitos mínimos de hardware y software. La adherencia a estas especificaciones es crucial para evitar cuellos de botella en el rendimiento y asegurar una experiencia de usuario fluida.

- **Hardware:** Los componentes físicos esenciales para el procesamiento y la interacción del sistema incluyen:

  Unidad de Procesamiento (CPU/GPU): Se recomienda una unidad con capacidad de cómputo para Machine Learning, PC con una GPU compatible con CUDA (para aprovechar la aceleración por hardware de los modelos), con la recomendacion para un desarrollo escalable implementando las plataformas NVIDIA Jetson Nano/AGX Xavier para despliegues embebidos.

  Memoria RAM: Un mínimo de 8 GB es necesario para manejar eficientemente la carga de modelos, el procesamiento de video y la ejecución de la interfaz gráfica.

  Almacenamiento: Una unidad de estado sólido (SSD) de al menos 64 GB es fundamental para el sistema operativo, las bibliotecas de software y, especialmente, para el almacenamiento rápido de los modelos de detección.

  Cámara de Video: Cualquier cámara compatible con la biblioteca OpenCV es adecuada (ej., cámaras USB, IP o CSI), utilizada para la captura del flujo de tránsito.

  Monitor/Pantalla: Un dispositivo de visualización con entrada HDMI es indispensable para la proyección dinámica de las señales del SMV.

- **Software:** Los elementos lógicos necesarios para la ejecución del sistema comprenden:

  Sistema Operativo: Windows 10/11 o Linux (Ubuntu 18.04 LTS o versiones superiores son altamente recomendadas).

  Python: Se requiere la versión **3.8 o superior** para garantizar la compatibilidad con todas las bibliotecas utilizadas.

  Frameworks de Machine Learning: Dependiendo de la configuración del modelo, se necesitará PyTorch o TensorFlow.

Librerías Principales: Un conjunto de bibliotecas Python esenciales, incluyendo OpenCV (para procesamiento de imágenes y video), Flask (para la GUI), NumPy (para operaciones numéricas eficientes), Pandas (para manejo de datos estructurados) y Matplotlib (para visualización de métricas), entre otras.

Navegador Web: Cualquier navegador moderno y actualizado como Google Chrome, Mozilla Firefox o Microsoft Edge es compatible para acceder y operar la interfaz gráfica de usuario.

2. Instalación y Configuración

Para un despliegue exitoso del sistema, siga cuidadosamente los pasos descritos a continuación. Es crucial adherirse al orden especificado para gestionar correctamente las dependencias y evitar posibles conflictos o errores durante la instalación.

Crear Entorno Virtual: La creación de un entorno virtual es una práctica recomendada para aislar y gestionar de forma limpia las dependencias de su proyecto, evitando conflictos con otras instalaciones de Python en el sistema.

```
python3 -m venv venv_pmvai
source venv_pmvai/bin/activate   # En Linux/macOS
# venv_pmvai\Scripts\activate.bat   # En Windows
```

Instalar Dependencias: Con el entorno virtual activado, proceda a instalar todas las bibliotecas de Python requeridas, las cuales están listadas en el archivo ‘requirements.txt‘ ubicado en la raíz del proyecto.

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

Nota: Si aún no posee el archivo ‘requirements.txt‘, puede generarlo automáticamente una vez que haya instalado todas las dependencias del proyecto utilizando el comando: ‘pip freeze >requirements.txt‘.

Configuración Inicial del Sistema: Antes de la primera ejecución, realice los siguientes ajustes esenciales:

- **Modelos:** Coloque su archivo de modelo entrenado (por ejemplo, ‘best.pt‘) dentro de la carpeta designada para modelos, usualmente ubicada en ‘models/‘ dentro de la estructura de su proyecto.
- **Rutas de Cámara:** Si el sistema requiere la integración con una cámara específica, configure la ruta de acceso o el índice de la cámara en el archivo de configuración del proyecto (generalmente ‘config.py‘ o un archivo similar).
- **Credenciales de Acceso:** Para modificar las credenciales predeterminadas de inicio de sesión de la interfaz gráfica de usuario (GUI), edite la sección de autenticación en el archivo ‘app.py‘.

3. Operación de la Interfaz Gráfica de Usuario (GUI)

Para poner en marcha la interfaz gráfica de usuario, navegue hasta la carpeta principal de su proyecto (donde reside el archivo ‘app.py‘) en su terminal o línea de comandos. Asegúrese de que su entorno virtual está activado y, a continuación, ejecute el siguiente comando:

```
Python app.py
```

Una vez que la aplicación se inicie, se le indicará una dirección URL en la consola (habitualmente ‘http://127.0.0.1:5000‘ o ‘http://localhost:5000‘). Abra esta dirección en su navegador web preferido (Google Chrome, Mozilla Firefox, Microsoft Edge, etc.) para acceder a la GUI del sistema PMV AI.

3.1. Inicio de Sesión

Al acceder a la URL del sistema, será redirigido automáticamente a la página de inicio de sesión. Aquí deberá introducir las credenciales predeterminadas:

Usuario: ‘IT VIAL‘
Contraseña: ‘PMVIA‘

Después de ingresar los datos, haga clic en el botón Ïngresar"para acceder al panel principal del sistema.

3.2. Panel Principal (Dashboard)

Tras un inicio de sesión exitoso, será dirigido al panel principal del sistema, también conocido como Dashboard. Esta es la vista central desde la cual podrá navegar y acceder a los diversos módulos que le permiten interactuar con el sistema y monitorear su funcionamiento:

- **Ver Tránsito:** Acceso al módulo de monitoreo en tiempo real del tránsito y control de operaciones de detección y proyección.
- **Ver Métricas:** Página para la visualización de gráficos y estadísticas de rendimiento del modelo y proyección de señales.
- **Historial:** Permite consultar registros detallados de eventos pasados y señales proyectadas.
- **Aprendizaje del Sistema:** Módulo dedicado a la gestión y actualización de los modelos de detección de objetos.

3.3. Módulo Ver Tránsito

Este módulo constituye el centro de operación en tiempo real del sistema. Proporciona la visualización en vivo del flujo de video capturado y ofrece controles directos para los procesos de detección de vehículos y proyección de señales.

- **Botón Iniciar Detección:** Al hacer clic en este botón, el sistema comenzará a procesar el flujo de video de la cámara y ejecutará el modelo YOLO para la detección de vehículos. Observará cómo el flujo de video en la interfaz gráfica se actualiza, mostrando los cuadros delimitadores (*bounding boxes*) alrededor de los vehículos detectados.

- **Botón Iniciar Proyección:** Este botón activa la función principal del sistema. Al presionarlo, el sistema activará la proyección dinámica de la señal recomendada en el monitor HDMI conectado. La señal mostrada cambiará automáticamente y en tiempo real, basándose en las detecciones de vehículos y la aplicación de las reglas lógicas definidas.

- **Botones de Detener:** Para pausar temporalmente los procesos, utilice los botones "Detener Detecciónz "Detener Proyección". Esto le permitirá detener el análisis de video o la transmisión de señales al panel sin cerrar la aplicación.

3.4. Módulo Ver Métricas

Acceda a este módulo para obtener una visión clara y cuantitativa del rendimiento del modelo de detección de objetos y las estadísticas de proyección de señales. Este análisis es fundamental para evaluar la eficacia y la eficiencia del sistema.

- **Métricas del Modelo:** Podrá observar gráficos y valores numéricos clave como la Precisión, el Recall y el mAP (*mean Average Precision*). Estas métricas son indicativas de la exactitud y fiabilidad del modelo de detección en la identificación de vehículos.

- **Métricas de Proyección:** Visualice gráficos que detallan la frecuencia de cada tipo de señal proyectada y las tendencias de proyección a lo largo del tiempo. Estos datos le permitirán analizar patrones de tránsito, evaluar la respuesta del sistema a diferentes condiciones y tomar decisiones informadas sobre la configuración de reglas.

3.5. Módulo "Historial"

El apartado "Historialçonstituye un recurso fundamental para llevar a cabo auditorías y análisis retrospectivos. Ofrece la posibilidad de acceder a un registro minucioso y en orden cronológico de todas las señales emitidas por el sistema. Es posible emplear las opciones de filtrado proporcionadas para realizar búsquedas concretas según la clase de señal, el período de fechas, la duración de la proyección y otros criterios pertinentes, lo que permite una evaluación detallada de eventos anteriores y la detección de patrones.

3.6. Módulo Aprendizaje del Sistema

Este módulo es fundamental para la gestión, mantenimiento y actualización continua de los modelos de detección que impulsan el sistema inteloigente. Desde aquí, los usuarios avanzados o el personal técnico pueden:

- **Cargar nuevos modelos YOLO entrenados:** Permite integrar versiones más recientes o especializadas de los modelos de detección en el sistema.

- **Probar el rendimiento de los modelos cargados:** Ofrece funcionalidades para evaluar la precisión y eficiencia de los modelos antes de su despliegue en producción.

  Activar o desactivar modelos específicos: Brinda la flexibilidad de seleccionar qué modelo está activo en tiempo real, ideal para pruebas A/B o despliegues por fases.

- **Eliminar modelos obsoletos o no deseados:** Ayuda a mantener el sistema limpio y optimizado, retirando modelos que ya no son necesarios.

5. Contacto y Soporte

Para cualquier consulta, reporte de errores o solicitud de soporte técnico relacionado con el sistema, por favor contacte a Marco Rodrigo Pérez Ramírez a través del correo electrónico `marco.perezr@usantoto.edu.co`.

Anexo D: Video Demostrativo del Funcionamiento del Proyecto

Este anexo proporciona acceso a un video explicativo que ilustra el funcionamiento integral del sistema. El video presenta una demostración práctica de la solución desarrollada, mostrando implementaciones en condiciones reales, la detección de vehículos mediante Machine Learning, la interpretación de las reglas lógicas y la visualización dinámica de las señales de tránsito en el sistema de mensajería variable (HDMI). Se recomienda visualizar este recurso audiovisual para obtener una comprensión clara de las capacidades del sistema y su interacción con el entorno.

`Anexo Video SMV IA`

Con el propósito de reforzar el análisis técnico y facilitar una comprensión integral del comportamiento del sistema desarrollado, se ha dispuesto un repositorio digital que contiene el material complementario correspondiente al capítulo de anexos. Este conjunto de archivos incluye evidencia audiovisual, registros de pruebas, códigos y demás recursos que sustentan y amplían los contenidos presentados en el cuerpo del documento.

El acceso a este material se realiza mediante una carpeta alojada en la plataforma OneDrive, diseñada para permitir la consulta remota de los anexos por parte de los evaluadores, lectores académicos y demás interesados en la validación y reproducción del proyecto.

Para garantizar la seguridad de la información y restringir su acceso únicamente a personal autorizado, se ha establecido un mecanismo de autenticación. La carpeta requiere el ingreso de una contraseña, la cual corresponde al identificador del sistema: *SMV*

Anexo E: Repositorio Conjunto Datos

Este anexo brinda una validación completa y transparente del proyecto, se ha dispuesto un repositorio digital que contiene el banco de imágenes utilizado para el entrenamiento, validación y prueba del modelo. El acceso a este material se gestiona a través de la plataforma OneDrive.

El dataset utilizado para este proyecto es un conjunto de datos robusto y diverso, compuesto por un total de 12,000 archivos de imágenes. Su composición fue diseñada estratégicamente para incluir una amplia gama de escenarios de conducción, lo que permite que el algoritmo de detección de objetos sea más fiable y generalizable. Este enfoque minimiza el riesgo de que el modelo se sobreajuste a un entorno específico y garantiza que pueda funcionar en una variedad de condiciones reales.

Para el desarrollo del modelo, el dataset se dividió rigurosamente en tres conjuntos, siguiendo una metodología estándar en el campo del Machine Learning:

Conjunto de Entrenamiento (70 %): Este es el núcleo del proceso de aprendizaje. Con 8,400 imágenes, este conjunto se utilizó para que el modelo aprendiera a identificar los patrones, características y variaciones de los vehículos. Las imágenes abarcan una variedad de entornos, horarios y condiciones de iluminación, lo cual es crucial para el desempeño del sistema en situaciones reales, ya que el modelo asimila cómo se ven los vehículos bajo diferentes luces, sombras y ángulos.

Conjunto de Validación (15 %): Con 1,800 imágenes, este conjunto fue esencial para ajustar los hiperparámetros del modelo durante el proceso de entrenamiento. La validación constante en este subconjunto de datos no vistos previamente ayuda a detectar y prevenir el sobreajuste, un fenómeno en el que el modelo memoriza los datos de entrenamiento en lugar de aprender a generalizar.

Conjunto de Pruebas (15 %): Las 1,800 imágenes restantes se emplearon para evaluar el rendimiento final del modelo. Este conjunto, que el modelo nunca vio durante el entrenamiento o la validación, proporciona una evaluación imparcial de su desempeño, demostrando su capacidad real para detectar vehículos en un contexto no conocido.

`Anexo Repositorio Dataset`