Implementación de Algoritmos de Detección e Identificación de Vehículos con tecnologías de Machine Learning y Deep Learning.

Otto Arturo Andrade y Brayan Fonseca Gonzalez

Trabajo de grado para optar al título de Ingenieros Electrónicos

Director

Jaime Guillermo Barrero

Msc. en potencia eléctrica

Universidad Industrial de Santander

Facultad de Ingenierías Fisicomecánicas

Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones

Ingeniería Electrónica

Bucaramanga

2025

Dedicatoria

Quiero agradecer a la Universidad por permitirme conocer una pequeña parte de esta ilustre profesión. Soy la suma de todos los esfuerzos y la confianza que mis padres, amigos y familiares han depositado en mí. Decir que estoy simplemente agradecido es no corresponder a las ideas, sentimientos y acciones que me han hecho ser la persona que soy. Llevo presente en mi ser la significancia de todos los hechos que han desembocado en este trabajo de grado.

Otto Arturo Andrade Camelo

Este trabajo de grado refleja una parte del conocimiento adquirido durante mi formación universitaria. Por ello, quiero dedicarlo a mi familia, especialmente a mi mamá, quien, con mucho esfuerzo y sacrificio, me apoyó a lo largo de este camino. Asimismo, deseo expresar mi más sincero agradecimiento a mis familiares y amigos, quienes siempre estuvieron a mi lado en los momentos difíciles, permitiéndome alcanzar la meta de culminar mis estudios.

Brayan Steven Fonseca Gonzalez

Agradecimientos

Toda mi vida he estado rodeado de personas maravillosas que de una u otra manera cimentaron que esté a puertas de graduarme. Agradezco primeramente a mis padres porque incondicionalmente siempre creyeron en mí y en lo que podía ser, a mis hermanos por tantos momentos de felicidad y apoyo; respecto a mis amigos y familiares, tenemos a María, Rosalba, Gloria, Andrés, Miguel, David, Angie, Rubi y demás nombres que llegan a mi mente al pensar en lo agradecido que estoy con la vida por tan insigne fortuna. Gracias a Dios por hacer posibles tantos momentos y experiencias que me hicieron vivir feliz y construir mi persona.

A la Universidad Industrial de Santander por otorgarme una oportunidad de superarme, a tantos profesores maravillosos que me inculcaron la belleza de este arte y el amor por aprender, de una u otra forma en esta tesis dejo algo de todo lo bonito que viví durante mi estancia en la universidad.

Otto Arturo Andrade Camelo

En primer lugar, quiero expresar mi más profundo agradecimiento a mi familia: a mi mamá y a mis hermanos, quienes siempre me apoyaron y se sacrificaron para que pudiera estudiar. También a mis padrinos, que estuvieron presentes en todo momento y me brindaron su ayuda incondicional. A mis primos y tíos, gracias por preocuparse por mí y ofrecerme siempre su apoyo.

A todos los amigos que conocí durante mi paso por la universidad como Sergio, Maria, Camilo, Luzdey, Tatiana, Vanessa, Isabella y muchísimos más les agradezco de corazón. No solo

me ayudaron académicamente, sino también emocionalmente; siempre fueron un gran apoyo. Me considero muy afortunado de haber conocido a tantas personas maravillosas y especiales. Gracias a todos ustedes por contribuir a hacerme una mejor persona.

Quiero agradecer también a la Universidad Industrial de Santander (UIS), mi alma mater, por brindarme la oportunidad de explorar un mundo nuevo lleno de excelentes profesores y profesionales. Durante mi tiempo en la universidad, tuve la oportunidad de trabajar con la institución, competir deportivamente y participar en presentaciones culturales junto al Teatro UIS. Estas experiencias han sido fundamentales para moldear la persona que soy hoy, y no puedo estar más agradecido con la UIS por haberme abierto sus puertas.

Brayan Steven Fonseca Gonzalez

Tabla de Contenido

Lista de Figuras

Lista de Tablas

Lista de Apéndices

pág.

Resumen

Titulo: Implementación de algoritmos de detección e identificación de vehículos con tecnologías de machine learning y deep learning. [*]

Autores: Otto Arturo Andrade Camelo, Brayan Steven Fonseca Gonzalez. [**]

Director: Jaime Guillermo Barrero Pérez. Msc. en potencia eléctrica.

Palabras Clave: Machine Learning, Visión por computadora, Detección, Procesamiento de imágenes.

Descripción: Los métodos tradicionales de detección vehicular presentan limitaciones en precisión y adaptación a distintos entornos, lo que ha impulsado el uso de Machine Learning y Deep Learning para optimizar la identificación y el seguimiento vehicular en aplicaciones de seguridad vial y control del tráfico.

Este proyecto desarrolla un sistema de detección de vehículos basado en aprendizaje automático, mejorando el reconocimiento de matrículas y la gestión del tráfico. Implementado en Windows 11, procesa imágenes a aproximadamente 25 FPS en CPU y 43 FPS en GPU (según el hardware, ver D), garantizando un desempeño en tiempo real.

Se empleó un conjunto de datos de matrículas colombianas procesado en Roboflow y, tras evaluar distintos modelos, se seleccionó YOLOv8 por su precisión y velocidad (mAP@0.5 >90 %). Técnicas como ecualización de histograma y transformaciones de homografía mejoraron su rendimiento en condiciones adversas. Finalmente, la interfaz gráfica con PySide6 permitió una gestión eficiente e integración en aplicaciones reales.

---

[*] Trabajo de grado

[**] Facultad de Ingenierías Fisicomecánicas, Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones. Director Jaime Guillermo Barrero Pérez.

Abstract

Title: Implementation of Vehicle Detection and Identification Algorithms Using Machine Learning and Deep Learning Technologies. [*]

Authors: Otto Arturo Andrade Camelo, Brayan Fonseca Gonzales. [**]

Director: Jaime Guillermo Barrero Pérez, Msc. in electrical power.

Keywords: Machine Learning, Computer Vision, Image Processing, Detection.

Description: Traditional vehicle detection methods have limitations in accuracy and adaptability to different environments, driving the adoption of Machine Learning and Deep Learning to enhance vehicle identification and tracking in traffic control and road safety applications.

This project develops a vehicle detection system based on machine learning, improving license plate recognition and traffic management. Implemented on Windows 11, it processes images at approximately 25 FPS on CPU and 43 FPS on GPU (depending on the hardware, see D), ensuring real-time performance.

A dataset of Colombian license plates processed in Roboflow was used, and after evaluating multiple models, YOLOv8 was selected for its balance between accuracy and speed (mAP@0.5 >90%). Preprocessing techniques such as histogram equalization and homography transformations improved performance under adverse conditions. Finally, the graphical interface developed with PySide6 enabled efficient system management and seamless integration into real-world applications.

---

[*] Bachelor Thesis

[**] Faculty of Physical and Mechanical Engineering, School of Electrical, Electronic and Telecommunications Engineering. Director Jaime Guillermo Barrero Pérez.

1. Objetivos

1.1. Objetivo General

Implementar un algoritmo de detección e identificación de vehículos y extracción para placas en sistemas de video en tiempo real utilizando tecnologías de inteligencia artificial.

1.2. Objetivos Específicos

- Adquirir y organizar una base de datos con videos de cámaras de vigilancia ya existentes a partir de trabajos previos para que estos puedan ser analizados en tareas de investigación.
- Implementar un algoritmo que permita identificar placas en vehículos, evaluando su rendimiento en términos de precisión, velocidad de procesamiento y tolerancia.
- Desarrollar un aplicativo que permita reportar a una base de datos privada el número de la matrícula del vehículo e imágenes del mismo en el momento del registro.
- Identificar posible mejoras o ajustes necesarios al realizar pruebas de campo para evaluar la efectividad y la aplicabilidad de la solución ante la variabilidad de las situaciones diarias.

2. Introducción

2.1. Antecedentes locales

La seguridad ciudadana abarca medidas para proteger a los ciudadanos, mientras que la seguridad privada se enfoca en la vigilancia y resguardo de bienes. Este sector enfrenta desafíos como la falta de coordinación con entidades públicas, la alta rotación de personal y la escasa actualización tecnológica. No obstante, busca mantenerse eficiente mediante la implementación de sistemas tecnológicos avanzados, entre los cuales destacan los sistemas autónomos de reconocimiento y detección, que han demostrado mejorar la seguridad y optimizar las operaciones de estas empresas.

En 2024, Bucaramanga registró 18 hurtos a vehículos Ministerio de Defensa (2025), evidenciando la necesidad de fortalecer la seguridad. La videograbación permite monitorear en tiempo real, disuadir delitos y gestionar emergencias, pero su eficacia puede mejorarse con inteligencia artificial y análisis de datos. En este contexto, las empresas de vigilancia privada requieren identificar vehículos específicos en sus grabaciones, optimizando así la respuesta ante incidentes y facilitando investigaciones en parqueaderos y zonas restringidas.

2.2. Causas del problema

La identificación de vehículos en áreas de acceso restringido, como conjuntos residenciales, enfrenta múltiples dificultades. Los trabajadores de seguridad tienen limitaciones para memorizar detalles específicos debido a la gran cantidad de vehículos y la variabilidad en sus características. Además, la sobrecarga de información y las condiciones adversas de operación de las cámaras de

vigilancia afectan la precisión del proceso. Aunque existen soluciones tecnológicas, muchas no son intuitivas ni permiten una personalización eficiente, lo que genera una mayor dependencia del factor humano.

Es necesario desarrollar una herramienta que permita detectar e identificar vehículos de manera rápida y eficiente, extrayendo además su placa para optimizar la gestión de datos. Este sistema debe facilitar la visualización y mejorar la experiencia del usuario, permitiendo segmentar el video en fragmentos más cortos para agilizar la verificación de registros por parte de los operarios o solicitantes.

2.3. Investigación de soluciones

Respecto a las soluciones actuales, existen algunas basadas en detección de matrículas y seguimiento de objetos; además de ello, también está el análisis de patrones de comportamiento, pero no es usada debido a su costo y dificultad de implementación. Los sistemas de grabación y vigilancia actuales están formados por cámaras en su mayoría digitales IP con resoluciones que van desde los 360p hasta los 1080p, dotadas de gran flexibilidad y capacidad de transmisión de datos a través de redes IP. Existen también variaciones como la infrarroja, nocturna, panorámica, etc. Estas grabaciones pueden terminar en sistemas de almacenamiento y procesamiento locales como los DVR's (Grabadoras de video digital) o pueden alojar la información en nube de almacenamiento en línea como los NVR (Network Video Recorder). Asociadas a estas existen plataformas de gestión de videos (VMS), las cuales ofrecen funciones como la visualización en tiempo real, búsqueda de grabaciones, gestión de usuarios y control remoto de cámaras. Existen algunos otros accesorios como sensores, aplicaciones de monitoreo móvil y dispositivos biométricos.

El principio de funcionamiento de los algoritmos de detección y seguimiento se basa en modelos estadísticos de los cambios en el fondo. Algoritmos como el Adaptive Kernel Density Estimation Algorithm ofrecen buenos resultados en la detección, pero a costa de un mayor uso de memoria y una menor eficiencia en tiempo real Vargas and Moncada (2021). Por otro lado, MOG (Mixture of Gaussians) se adapta mejor a los cambios en el fondo, aunque es menos efectivo cuando no hay un contraste perceptible entre el fondo y el objeto. Existen muchos otros algoritmos que se actualizan con relativa rapidez, pero en la práctica, es más eficiente emplear arquitecturas que ya los incorporen y optimicen. OpenCV, por ejemplo, cuenta con una comunidad con gran soporte y facilita la programación en Python, permitiendo el uso de múltiples métodos asociados a clases para algoritmos específicos. Para ello, se puede planificar un flujo de trabajo en el que el primer paso del algoritmo es capturar cada cuadro del video de vigilancia de entrada. Luego, una vez obtenido el cuadro del video, se aplican procesos de preprocesamiento para reducir el ruido de fondo y cualquier interferencia que pueda ocurrir durante las etapas de procesamiento de imágenes Ariza and Merchan (2009).

Continuando con los estudios previos que sirvan como elementos de partida, se observa que localmente ya hay presencia de implementaciones de este estilo pero dichos algoritmos fueron creados hace mucho tiempo y muy pocos han sido implementados con las últimas tecnologías existentes. Existen investigaciones que utilizan la clasificación de color en el sector agrícola, para determinar el estado de maduración de las frutas, la presencia de enfermedades o el estado de salud en las plantas. En el ámbito internacional se puede encontrar suficientes referencias a cómo usando YOLO y OpenCV es posible hacer seguimiento e identificación en videograbaciones de

seguridad Vargas and Archila (2023). Las pruebas revisadas abarcan una cantidad considerable de dispositivos y diferentes capacidades de procesamiento. En todo caso, esta es una tecnología relativamente novedosa, por lo que continúan publicando documentos que profundizan en su uso semanalmente, por lo que es importante permanecer vigente a las nuevas aplicaciones y formas de utilizar la inteligencia artificial.

2.4. Definiciones y conceptos previos

El propósito de esta sección es presentar una serie de definiciones, en su mayoría asociadas a las tecnologías y elementos presentes en este proyecto, con la finalidad de que ayude a contextualizar y entender a cabalidad el diseño planteado.

2.4.1. Los sistemas de Reconocimiento Automático de Placas Vehiculares (ANPR).

Los sistemas de Reconocimiento Automático de Placas Vehiculares ANPR, son aplicaciones de visión por computador que están compuestos de hardware y software adecuados que permiten la lectura de la placa vehicular Orellana Preciado and Ortega Castro (2020). Dada su versatilidad y precisión, los sistemas ANPR han evolucionado considerablemente en los últimos años, incorporando inteligencia artificial y mejorando su capacidad de adaptación a condiciones desafiantes, como placas deterioradas, iluminación variable y entornos urbanos complejos.

2.4.2. Roboflow. Roboflow es una plataforma que permite a los desarrolladores crear sus propias aplicaciones de visión artificial, independientemente de sus habilidades o experiencia. Agiliza el proceso entre el etiquetado de los datos y el entrenamiento del modelo Roboflow (2025). Roboflow desempeñó un papel clave en la gestión y optimización de los conjuntos de datos utilizados para el entrenamiento del modelo de reconocimiento de placas vehiculares.

2.4.3. Inteligencia Artificial y Redes Neuronales Convolucionales. La inteligencia artificial (IA) es un conjunto de técnicas computacionales que permiten a los sistemas emular funciones del cerebro humano Steels and de Mantaras (2018). En este proyecto, la IA se aplicó al reconocimiento automático de placas vehiculares, facilitando su identificación en imágenes y

videos en tiempo real.

Dentro de la IA, las redes neuronales convolucionales (CNN) constituyen actualmente el estado del arte de varios problemas de visión computacional, dado su buen desempeño problemas de reconocimiento e interpretación en imágenes y video Massiris et al. (2018). Para la detección de placas y caracteres, se empleó YOLO (You Only Look Once), un modelo de alto rendimiento en tareas de detección de objetos en tiempo real Redmon et al. (2016), optimizando el procesamiento y la precisión del sistema.

2.4.4. Tecnologías para Procesamiento y Visualización de Imágenes. Python es un lenguaje de programación de propósito general creado por Guido van Rossum en los años 90, ampliamente utilizado en desarrollo web, inteligencia artificial y análisis de datos Nolasco Valenzuela (2018). Su versatilidad lo convierte en una herramienta clave para la implementación de sistemas de visión por computadora.

Para la construcción de la interfaz gráfica del sistema (GUI), se empleó PySide6, un conjunto de herramientas que facilita el desarrollo de aplicaciones GUI en Python. Su integración permitió la visualización en tiempo real de imágenes y video, así como la implementación de controles interactivos Wu et al. (2025).

Por otro lado, OpenCV (Open Source Computer Vision Library) es una biblioteca de código abierto diseñada para visión artificial y aprendizaje automático. En este proyecto, OpenCV fue esencial en el preprocesamiento de imágenes, optimizando la calidad de entrada para la detección precisa de placas vehiculares y caracteres OpenCV (2019).

3. Metodología

El desarrollo de este proyecto se llevó a cabo siguiendo un enfoque ágil, con reuniones semanales para la asignación de tareas, seguimiento de avances y evaluación de resultados. Como paso previo, se realizó una exhaustiva revisión de la literatura y análisis de implementaciones similares, lo que permitió establecer condiciones de diseño, identificar limitaciones y definir la viabilidad de la solución propuesta.

El proceso metodológico se estructuró en cinco etapas principales. La primera consistió en la recolección y tratamiento de los datos, utilizando conjuntos de datos públicos alojados en Roboflow, los cuales fueron etiquetados nuevamente con mejores cajas delimitadoras y sometidos a un proceso de aumento de datos adaptado a sus características y distribución; este mejoramiento fue necesario debido a que el área de las cajas no era el adecuado, en unas partes introduciendo ruido e información innecesaria en las etiquetas y por otro lado omitía información relevante de la imagen. La segunda etapa abarcó la selección y entrenamiento del modelo, donde se probaron diversas versiones de YOLO para evaluar su rendimiento en la segmentación de placas y la detección de caracteres, determinando el modelo más adecuado para cada tarea en función de métricas de desempeño. Posteriormente, en la tercera fase, se desarrolló un módulo de preprocesamiento de imágenes para optimizar la calidad de entrada del modelo, implementando técnicas como ecualización de histograma y transformaciones de homografía, con el objetivo de mejorar la precisión del reconocimiento en distintos escenarios. En la cuarta etapa, se construyó la interfaz gráfica utilizando PySide6, integrando completamente los modelos de predicción con una base de datos local

para la gestión de registros. Esta fase incluyó la implementación de funcionalidades clave, como almacenamiento y consulta de usuarios, personalización de la herramienta y una interfaz intuitiva que facilita su uso en entornos reales. Finalmente, la última etapa consistió en la validación y prueba del sistema, realizando evaluaciones de rendimiento tanto a nivel de los modelos individuales como de la aplicación en su conjunto. Se diseñaron casos de uso basados en escenarios reales de la industria y se llevaron a cabo pruebas en distintos entornos para medir la eficacia del sistema en condiciones variadas.

El desarrollo de código se realizó mayormente utilizando Visual Studio Code como entorno de programación y GIT como software para el control de versiones, de este modo las distintas herramientas se pudieron implementar sin realizar interferencias en cambios previos, minimizando los tiempos invertidos en solucionar problemas en el código. Este proyecto cuenta con dos bases de datos, una de videos y la otra de imágenes de placas de vehículos, las cuales pueden ser consultadas en el apéndice A.

A lo largo del proceso, se identificaron y abordaron diversos desafíos técnicos, como la necesidad de grandes cantidades de memoria para el entrenamiento de los modelos en entornos como Google Colab, el uso de GPU para mejorar el rendimiento en la fase de inferencia, y la optimización de técnicas de preprocesamiento de imágenes para lograr un equilibrio entre generalización y especificidad. Además, se consideraron las implicaciones del uso de información personal y comercial de los vehículos, asegurando el cumplimiento de las normativas nacionales sobre protección de datos.

El enfoque metodológico adoptado permitió una implementación estructurada y eficiente

del sistema, garantizando que cada etapa contribuyera a la precisión, escalabilidad y aplicabilidad de la solución propuesta. Respecto al funcionamiento general del programa, este se puede visualizar por medio del diagrama de bloques mostrado en la figura 1.

El completo desarrollo del programa y producto final se elaboró utilizando Python y está alojado en un repositorio público en Github (Ver Apéndice B del presente documento).

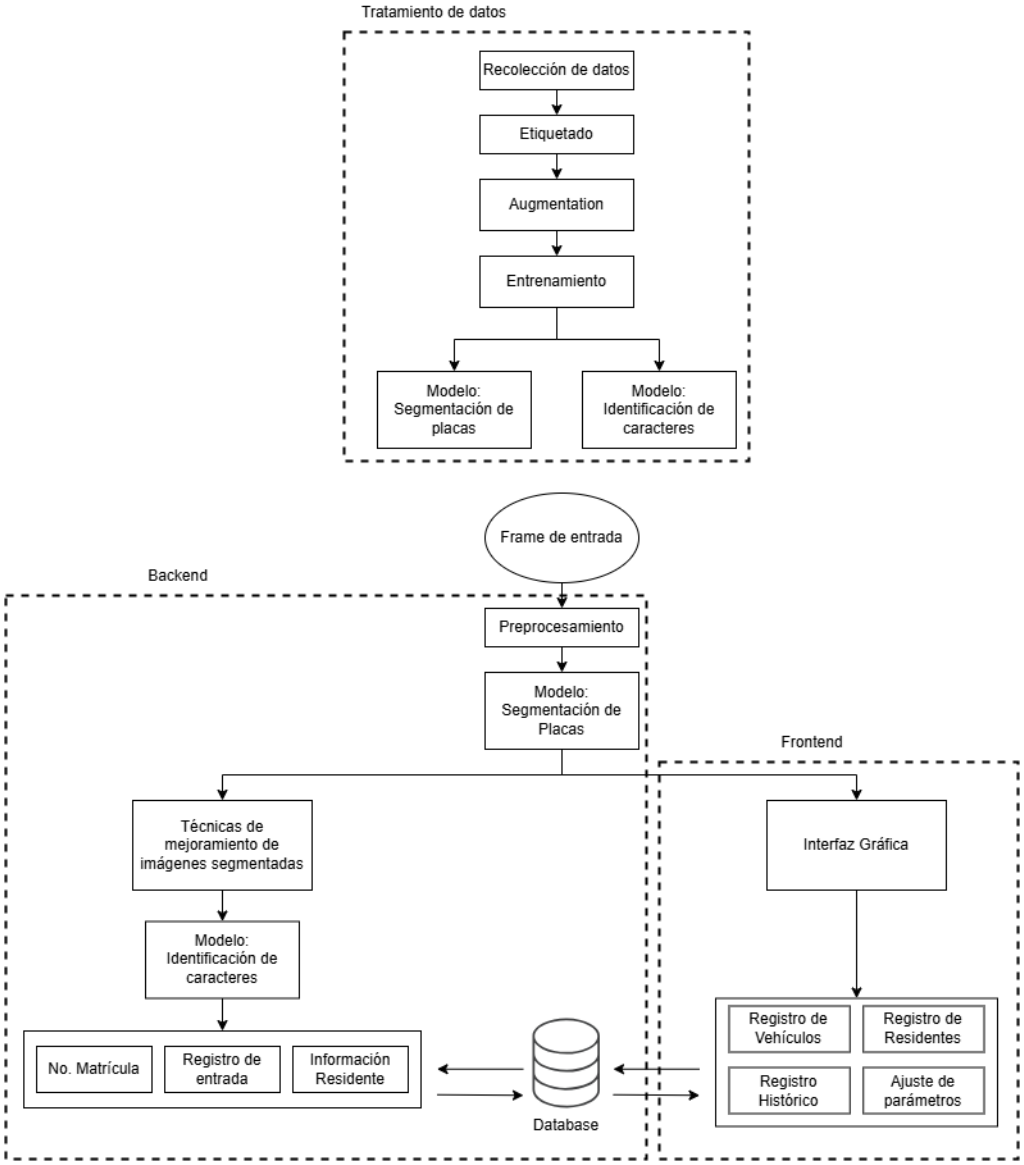
3.1. Preprocesamiento de los datos

El preprocesamiento de los datos se basó en la utilización de información pública disponible en la plataforma Roboflow. Para el modelo de detección de placas, se empleó un conjunto de datos de 1,399 imágenes de matrículas colombianas, previamente aumentadas mediante las herramientas de la plataforma. En este caso, se mantuvieron las etiquetas originales y solo se revisaron los procesos de aumentación aplicados, los cuales incluían los siguientes parámetros: tres imágenes generadas por cada imagen original, inversión horizontal, rotaciones de 90° en sentido horario y antihorario, rotaciones aleatorias entre -33° y +33°, así como deformaciones por cizalladura de hasta ±15° en direcciones horizontal y vertical.

Para la detección de caracteres, debido a la naturaleza multiclase del problema, se recopiló un conjunto de datos de 1,262 imágenes únicas con aproximadamente 7,500 caracteres, seleccionados de distintos directorios dentro de la misma plataforma y se procedió a realizar el etiquetado manual para asegurar una alta fiabilidad en los datos y en las cajas para cada etiqueta. En este caso, se evaluaron dos enfoques para el aumento de datos. Inicialmente, se aplicaron transformaciones manuales mediante librerías externas; sin embargo, los resultados mostraron un balance deficiente entre etiquetas, generando diferencias abruptas en la distribución de clases que afectaban el en-

Figura 1

Diagrama de bloques descriptivo del proceso



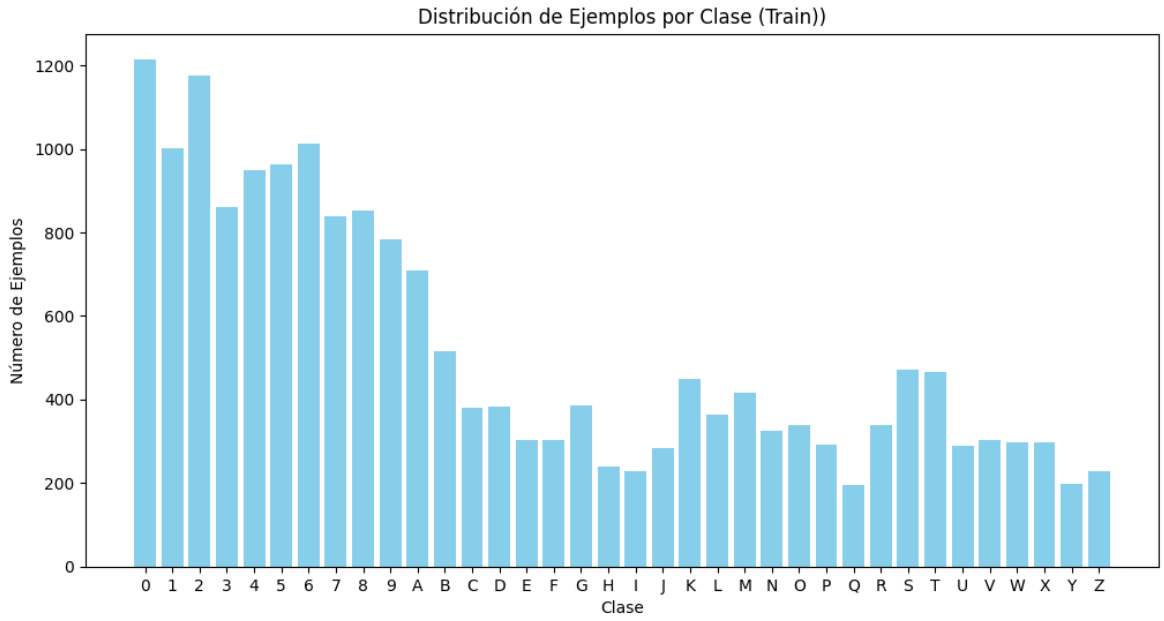
trenamiento del modelo. Ante esto, se optó por emplear el aumento de datos proporcionado por Roboflow, con los siguientes parámetros: tres imágenes generadas por cada imagen original, recortes con un zoom mínimo del 0 % y máximo del 15 %, rotaciones entre -15° y +15°, cizalladura de hasta ±10° en direcciones horizontal y vertical, variaciones de brillo entre -20 % y +20 %, y adición

de ruido en hasta el 1.01 % de los píxeles. Esta estrategia permitió obtener un balance relativamente estable entre números y letras. Si bien se consideró la aplicación de técnicas de submuestreo para reducir el predominio de ciertas clases, los resultados de entrenamiento indicaron que las métricas asociadas a etiquetas minoritarias eran satisfactorias, por lo que no fue necesario modificar el conjunto de datos adicionalmente.

Dado que Roboflow permite exportar los datos en diferentes formatos, estos se adaptaron específicamente para su uso con modelos YOLO, asegurando la compatibilidad en la estructura de etiquetado y almacenamiento de los datos. En general, el preprocesamiento se enfocó en optimizar la calidad y representatividad de los datos para mejorar el desempeño de los modelos de detección e identificación de vehículos. La figura 2 muestra en un diagrama de barras la distribución final de los datos usados para el entrenamiento.

Figura 2

Diagrama con la distribución de caracteres.



3.2. Selección de modelos

Para la detección e identificación de placas vehiculares y caracteres, se optó por el uso de modelos de la familia YOLO, desarrollados por Ultralytics, debido a su amplia aceptación en la comunidad de visión por computadora y su eficiencia en tareas de detección de objetos en tiempo real. Esta arquitectura permite una velocidad de inferencia significativamente mayor sin comprometer en exceso la precisión, lo que la hace ideal para aplicaciones en entornos urbanos donde el procesamiento en tiempo real es crucial Ultralytics Inc. (2024b).

La librería Ultralytics YOLO proporciona una implementación optimizada y de fácil uso, que permite el acondicionamiento, entrenamiento y validación de modelos en pasos simplificados. Además, cuenta con un selector automático de hiperparámetros basado en optimización bayesiana, lo que mejora la convergencia del modelo sin necesidad de ajustes manuales extensivos. También ofrece compatibilidad con diferentes frameworks de hardware, como CUDA para GPU NVIDIA, OpenVINO para Intel y CoreML para dispositivos Apple, facilitando su implementación en distintos entornos de producción. Otra ventaja clave de la librería, es su integración con múltiples formatos de anotación y datasets, como COCO, VOC y YOLO, lo que permite una mayor flexibilidad en el preprocesamiento de los datos.

Dada la variedad de versiones de YOLO, fue necesario evaluar cuál ofrecía el mejor balance entre precisión y velocidad de inferencia. Una de las versiones más utilizadas en la literatura es YOLOv5, que se destacó por su rapidez de predicción y eficiencia en dispositivos de bajo consumo. Sin embargo, dado el avance en la capacidad computacional de los equipos modernos y la nece-

sidad de asegurar la mejor precisión posible en condiciones reales, se optó por evaluar versiones más recientes de YOLO.

Se evaluaron las versiones YOLOv8, YOLOv9, YOLOv10, YOLOv11 y YOLOv12, destacándose YOLOv8 por sus mejoras en la estructura del backbone y detección sin depender del uso de cajas anclas, logrando alta precisión con bajo consumo computacional. YOLOv9 no presentó avances significativos, mientras que YOLOv10 mostró una leve mejora en mAP, pero con un aumento considerable en los tiempos de entrenamiento y procesamiento. Finalmente, YOLOv12 incorporó optimizaciones en el pipeline de datos y compatibilidad con nuevos entornos de despliegue.

Todos los modelos evaluados fueron preentrenados con el conjunto de datos COCO, lo que permitió acelerar el entrenamiento mediante transferencia de aprendizaje. Tras analizar las métricas de segmentación de placas y detección de caracteres, se seleccionó YOLOv8 con configuración predeterminada y selección automática de hiperparámetros, logrando un equilibrio óptimo entre precisión y velocidad para su implementación en entornos urbanos y de seguridad vehicular.

3.2.1. Métricas de evaluación. En general, se escogieron la mayor cantidad de métricas para evaluar el rendimiento de los modelos predictivos de visión por computadora, las cuales se pueden encontrar definidas a continuación Ultralytics Inc. (2024a).

La métrica *IoU* (Intersection over Union) se utiliza para evaluar la superposición entre la predicción del modelo y la anotación real. Se define como el cociente entre el área de la intersección y el área de la unión de ambas regiones:

$$IoU = \frac{\text{Área de la Intersección}}{\text{Área de la Unión}} \tag{1}$$

La precisión (*Precision*) mide la proporción de verdaderos positivos (TP) respecto a todos los elementos que el modelo ha clasificado como positivos, incluyendo los falsos positivos (FP):

$$\text{Precisión} = \frac{TP}{TP+FP} \tag{2}$$

La sensibilidad o *Recall* indica la proporción de verdaderos positivos detectados respecto al total de elementos positivos reales, incluyendo los falsos negativos (FN):

$$\text{Recall} = \frac{TP}{TP+FN} \tag{3}$$

La Precisión Promedio (*Average Precision*, AP) se calcula como el área bajo la curva de precisión vs. recall. Se expresa como:

$$AP = \int_0^1 p(r)\,dr \tag{4}$$

El mAP (*mean Average Precision*) representa el promedio de las precisiones promedio calculadas para todas las clases del conjunto de datos:

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{5}$$

El mAP@50 es una métrica que corresponde al mAP calculado con un umbral de IoU igual a 0.5:

$$mAP@50 = \frac{1}{N}\sum_{i=1}^{N} AP_i^{IoU=0,5} \tag{6}$$

El mAP@50-95 representa el promedio del mAP en distintos umbrales de IoU, desde 0.5 hasta 0.95 con incrementos de 0.05, lo cual proporciona una evaluación más completa del rendimiento del modelo:

$$mAP@50\text{-}95 = \frac{1}{10}\sum_{t=0,5}^{0,95} mAP_t \tag{7}$$

Finalmente, la métrica *Fitness* utilizada en algunos modelos como YOLO combina en un único valor la precisión, el recall, el mAP@50 y el mAP@50-95 para facilitar la comparación entre diferentes modelos o configuraciones:

$$\text{Fitness} = \frac{\text{Precisión} + \text{Recall} + \text{mAP@50} + \text{mAP@50-95}}{4} \tag{8}$$

En general, el objetivo de estas métricas es evaluar la calidad del modelo en la detección precisa de objetos, específicamente en este caso, de las cajas delimitadoras que contienen caracteres o placas esperadas. Estas medidas permiten cuantificar qué tan bien el modelo identifica correctamente las regiones relevantes dentro de una imagen. Para ello, se consideran distintos factores, tales como los verdaderos positivos (predicciones correctas de objetos presentes), los falsos

Tabla 1
Comparación de métricas de rendimiento para diferentes modelos de segmentación de placas

Comparación de métricas de rendimiento para diferentes modelos de YOLO						
Modelo	**Métricas**					
	Precision	Recall	mAP50	mAP50-90	Fitness	Parameters(M)
YOLOv8n	0.9988	1.0	0.9951	0.8493	0.8638	3.2
YOLOv9c	0.9913	0.9739	0.9941	0.9289	0.9354	25.5
YOLOv10n	0.9915	0.9885	0.9948	0.9226	0.9298	2.3
YOLOv10l	0.9913	0.9744	0.9943	0.9332	0.9393	24.4
YOLOv11n	0.9971	0.9830	0.9943	0.9307	0.9371	2.6
YOLOv11l	0.9820	0.9915	0.9943	0.9391	0.9445	25.3
YOLOv12n	0.9831	0.9909	0.9944	0.9227	0.9299	2.6

positivos (predicciones incorrectas de objetos inexistentes), los falsos negativos (objetos reales que el modelo no detectó) y, en algunas métricas, el área de superposición entre la predicción y la anotación real. Estas variables se combinan para obtener indicadores como precisión, recall, IoU o mAP, los cuales proporcionan una visión integral del rendimiento del sistema de detección en distintos escenarios.

Los resultados de los distintos métodos con sus métricas de rendimiento se adjuntan en las tablas 1 y 2 .

Respecto al comportamiento de los modelos durante el entrenamiento, todos se entrenaron con ajustes similares, basado en recomendaciones de trabajos relacionados. Usando un tamaño de lote de 32 y un tiempo de 100 épocas, buscando la convergencia completa a resultados estables. En general, la ejecución en todos los casos mostró una generalización adecuada alrededor de las 60 épocas, con oscilaciones despreciables hasta completar el entrenamiento, como se muestra en la figura 5.

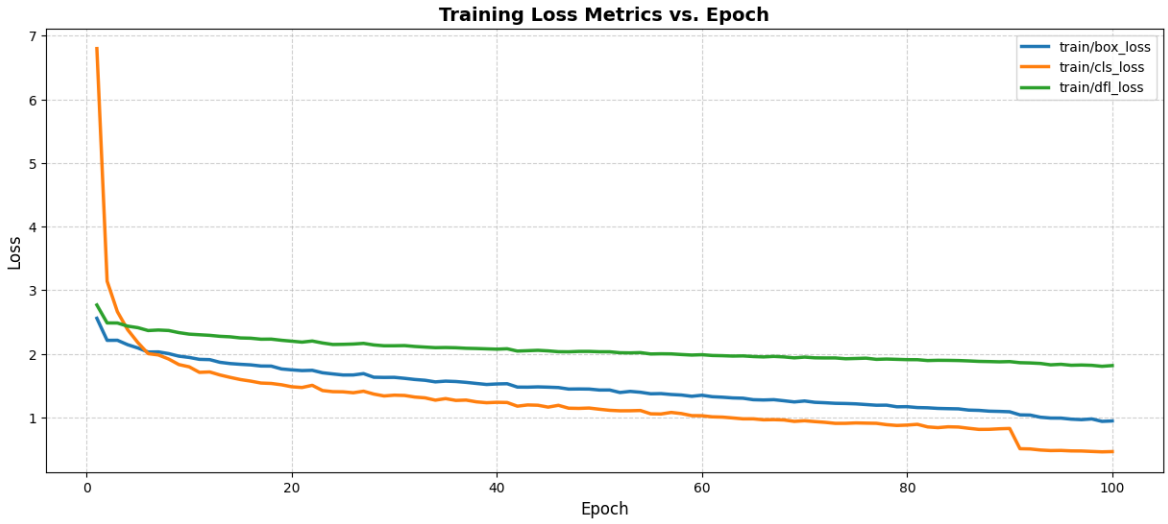
Tabla 2

Comparación de métricas de rendimiento para diferentes modelos de detección de caracteres

Comparación de métricas en el reconocimiento de caracteres					
Modelo	**Métricas**				
	Precision	Recall	mAP50	mAP50-90	Fitness
YOLOv8n	0.9094	0.8971	0.9064	0.7830	0.7953
YOLOv9t	0.9130	0.9048	0.9053	0.7815	0.7939
YOLOv10m	0.9487	0.9534	0.9809	0.8641	0.8758
YOLOv11n	0.9176	0.8911	0.8942	0.7689	0.7815
YOLOv12n	0.9257	0.8969	0.9077	0.7810	0.7936

Figura 3

Entrenamiento de modelos en el tiempo: Pérdidas en el grupo de entrenamiento



3.3. Procesamiento de imágenes

El procesamiento de imágenes es una etapa fundamental en el sistema de detección e identificación de placas vehiculares, ya que optimiza la calidad visual del frame de entrada antes de su análisis por los modelos de segmentación y reconocimiento de caracteres. Este procedimiento

Figura 4

Entrenamiento de modelos en el tiempo: Pérdidas en el grupo de validación

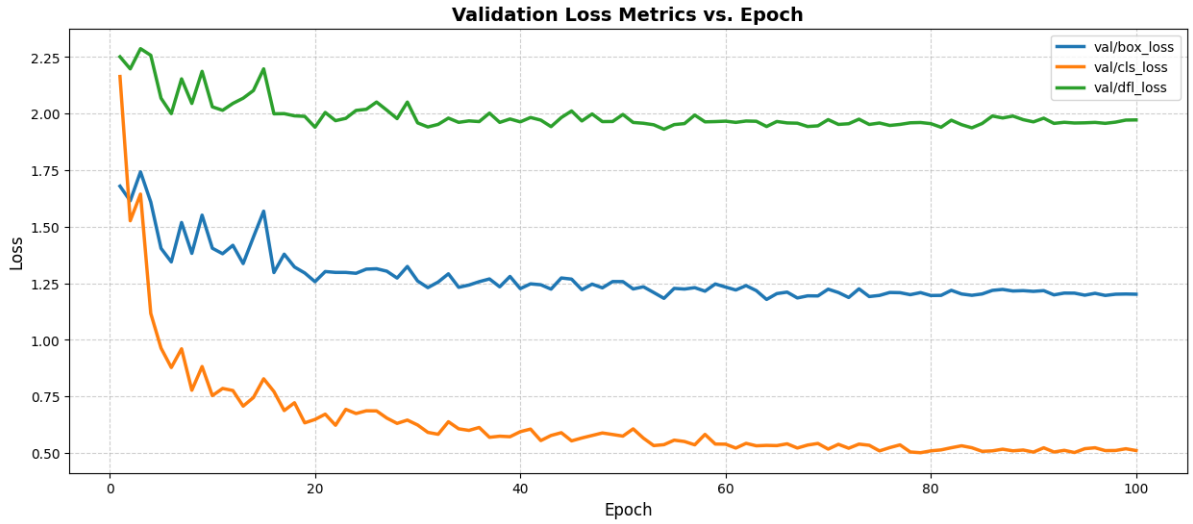
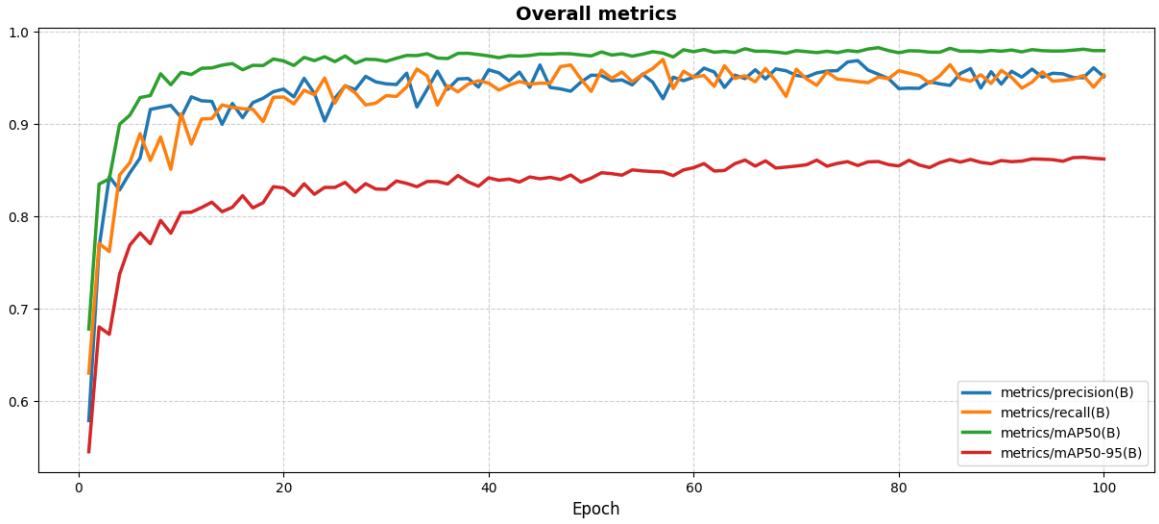


Figura 5

Métricas de comportamiento general



consta de múltiples pasos secuenciales que buscan mejorar la claridad, el contraste y la perspectiva de la imagen para facilitar la extracción de información relevante.

3.3.1. Preprocesamiento del Frame de Entrada. El primer paso en el flujo de trabajo consiste en normalizar el contraste del cuadro capturado, este paso es crucial para mejorar la visibilidad de la placa en diferentes condiciones de iluminación y asegurar una segmentación más robusta. Para ello, se emplea la función AutoContrast de la biblioteca Pillow, la cual ajusta automáticamente el histograma de la imagen, maximizando el rango dinámico sin alterar su contenido.

3.3.2. Segmentación de Placas. Tras la normalización, la imagen es procesada por el modelo de segmentación basado en YOLOv8, que identifica regiones candidatas a contener una placa vehicular. Si la confianza en la predicción del modelo sobrepasa el umbral de aceptación, se extrae el área correspondiente y se le añade un margen proporcional considerando la resolución del dispositivo de captura; esta confianza se refiere a la probabilidad de correspondencia que el modelo atribuye a cada posible clase respecto a la entrada. Esta región delimitada es posteriormente sometida a un conjunto de filtros para mejorar su legibilidad antes de ser analizada por el modelo de reconocimiento de caracteres.

3.3.3. Aplicación de Filtros en la Imagen de la Placa. Los filtros aplicados están diseñados para optimizar la imagen en escala de grises, ya que este formato es más adecuado para la detección de bordes y caracteres. Las técnicas empleadas en esta etapa incluyen:

- Filtrado Bilateral: Suaviza la imagen preservando los bordes, reduciendo el ruido sin perder detalles estructurales.
- Filtro de Afilado: Mejora los contornos mediante convolución con kernels de realce, lo que

facilita la detección de líneas y caracteres.

- CLAHE (Contrast Limited Adaptive Histogram Equalization): Mejora el contraste localmente evitando la sobre-exposición en regiones de alto brillo, permitiendo una mejor diferenciación entre caracteres y fondo Sharma and Kamra (2023).

Cada uno de estos filtros tiene parámetros ajustables dependiendo de factores como la altura de la cámara, distancia del vehículo y las condiciones de iluminación, lo que requiere calibración específica para diferentes escenarios de instalación.

3.3.4. Detección de Esquinas y Corrección de Perspectiva. Una vez mejorada la calidad visual de la imagen, se realizó la detección de bordes mediante un algoritmo modificado de Canny Gonzalez and Woods (2008). La modificación consiste en la aplicación de operaciones morfológicas para reducir artefactos y mejorar la consistencia de los contornos detectados. Posteriormente, se aplicó la Transformada de Hough, que permite identificar líneas predominantes en la imagen Illingworth and Kittler (1987). A partir de la intersección de estas líneas, se obtuvieron puntos candidatos a ser esquinas de la placa. Para seleccionar los puntos más relevantes, se siguió el siguiente procedimiento:

1. Se divide la imagen en una malla de 3×3 y solo se consideran los puntos ubicados en los cuadrantes de las esquinas.

2. Se aplica K-Means Clustering para agrupar los puntos restantes en cuatro grupos representativos.

3. Los cuatro centroides obtenidos se consideran como las esquinas reales de la placa.

Con estos puntos identificados, se emplea una transformación homográfica utilizando OpenCV, la cual permite corregir la perspectiva de la imagen y generar una vista rectificada de la placa. Este paso es clave para mejorar la precisión del reconocimiento de caracteres en placas capturadas desde ángulos oblicuos Collins (2007).

En instalaciones específicas, los puntos de corrección de perspectiva pueden ser configurados manualmente, lo que reduce los tiempos de procesamiento y mejora la estabilidad del sistema en entornos controlados.

3.3.5. Validación y Manejo de Casos Especiales. Se implementan diversas comprobaciones intermedias para determinar la validez de los puntos detectados. En caso de que no se identifiquen esquinas confiables, el sistema omite la corrección de perspectiva y envía directamente la imagen al modelo de reconocimiento de caracteres; esta opción; por supuesto, también puede ser configurada por defecto. Además, cuando se detecta una placa, se delimita visualmente en la interfaz del sistema, mostrando su posición en la imagen original. Paralelamente, se almacena la región recortada sin aplicar homografía, lo que permite registrar evidencia visual en su estado original.

4. Análisis de resultados

En esta sección, se presentan los hallazgos obtenidos tras la implementación y prueba del sistema, relacionándolos con los objetivos específicos planteados inicialmente. Se busca demostrar el cumplimiento de los requisitos mínimos establecidos y analizar el impacto de los resultados en un contexto técnico y no técnico. El costo del producto se puede ver en el apéndice C

Más allá de los aspectos puramente técnicos, el análisis también considera factores como la usabilidad del sistema, su adaptabilidad a diferentes dispositivos de captura y su escalabilidad para futuras implementaciones. Se discuten comparaciones con estudios previos y se identifican posibles áreas de mejora, proporcionando una visión integral sobre las fortalezas y limitaciones del sistema. El hardware y software utilizado se presenta en el apéndice D. Para ver una guía de usuario se puede ver el siguiente video: `https://youtu.be/LQO-XPebqDO?si=vlTcTZSvMANF9u_e` y para una mirada más técinca y detallada se puede visitar la siguiente página web: `https://sites.google.com/e3t.uis.edu.co/iadivtmldl-241`. El proyecto está orientado a un público diverso que incluye entidades de seguridad pública y privada, administradores de estacionamientos y zonas de acceso restringido, desarrolladores de software y empresas tecnológicas, así como compañías interesadas en soluciones de tráfico inteligente y gestión de movilidad urbana.

4.1. Implementación de modelos YOLO

Los resultados obtenidos tras la implementación y evaluación de los modelos de detección y reconocimiento de caracteres confirman la eficacia del enfoque basado en redes neuronales convolucionales, específicamente mediante el uso de modelos de la familia YOLO (You Only Look

Once). A continuación, se presentan los hallazgos más relevantes, teniendo en cuenta las métricas clave de desempeño y el impacto de las mejoras introducidas en el preprocesamiento de imágenes.

4.1.1. Desempeño en la detección de placas. En el caso de los modelos destinados a la segmentación de placas, se observó que las versiones más recientes de YOLO, desde la v8 hasta la v12, presentan resultados muy similares en términos de precisión (precision), recall y mAP50. Esto sugiere que, en general, todos los modelos pueden detectar de manera efectiva la ubicación de la matrícula en la imagen. Sin embargo, la principal diferencia se evidenció en el mAP50-95 y el fitness, lo que indica que los modelos con mayor número de parámetros logran un mejor ajuste en la predicción de los márgenes de la placa, reduciendo errores de segmentación.

A pesar de esta mejora en la delimitación, el uso de modelos más complejos conlleva un aumento significativo en el tiempo de inferencia y en la demanda computacional, lo que puede afectar la fluidez del sistema en entornos de tiempo real. Dado que la diferencia en rendimiento es marginal en relación con la precisión global del sistema, se justificó el uso de modelos más ligeros con menor tiempo de respuesta, seleccionando YOLOv8n (nano) como la versión óptima por defecto. Este modelo logra un equilibrio adecuado entre velocidad y precisión, permitiendo su implementación incluso en dispositivos con hardware de consumo moderado.

4.1.2. Reconocimiento de caracteres. Para la tarea de reconocimiento de caracteres, los resultados muestran que los modelos más ligeros presentan un rendimiento similar entre diferentes generaciones, lo que permitió establecer YOLOv8n como la opción por defecto para esta tarea. Sin embargo, en este caso, el uso de modelos más complejos sí mostró mejoras significativas en la precisión de clasificación, especialmente en caracteres con formas similares en vocales, letras

y números que tienden a ser confundidos. Esto sugiere que, si se dispone de hardware más potente o aceleración por GPU, el uso de modelos más grandes puede ser una opción recomendable.

Para ofrecer flexibilidad en diferentes entornos de implementación, la aplicación permite seleccionar entre modelos de diferentes versiones y complejidades, adaptándose a las capacidades computacionales del usuario y las necesidades específicas del sistema.

4.1.3. Impacto del Preprocesamiento y Corrección de Imágenes. Una mejora clave en el proceso de reconocimiento de caracteres fue la implementación de técnicas de preprocesamiento de imágenes, incluyendo normalización de contraste, filtrado bilateral, afilado, CLAHE y corrección de perspectiva mediante transformaciones homográficas. Estas optimizaciones permitieron mejorar la calidad visual de los caracteres antes de la inferencia, logrando una mejora de aproximadamente un 2 % en la precisión de detección de ciertos caracteres.

No obstante, se identificó un costo computacional asociado a estas mejoras. En escenarios donde se aplicó la corrección automática de perspectiva, se observó una reducción de hasta 1 fotograma por segundo en el rendimiento del sistema. Para minimizar este impacto, se permite la configuración manual de los puntos de corrección de perspectiva en los archivos de configuración de la aplicación, lo que prácticamente elimina el costo computacional adicional, favoreciendo su uso para configuraciones de escenarios específicos.

4.2. Interfaz gráfica de usuario (GUI)

La evaluación de la interfaz gráfica (GUI) se basa en cinco criterios clave: **usabilidad**, asegurando una interacción intuitiva para funciones como búsqueda y visualización de matrículas; **rendimiento**, midiendo la rapidez en el procesamiento de consultas; **accesibilidad**, garantizando

compatibilidad con distintos dispositivos; **estabilidad**, verificando la ausencia de errores críticos; y **experiencia del usuario (UX)**, optimizando el diseño y la navegación. Estos aspectos aseguran una interfaz eficiente, funcional y adaptable a los requerimientos del proyecto. La versión final de la interfaz gráfica se puede observar en la siguiente figura:

Figura 6
Interfaz gráfica final del proyecto

Se realizaron pruebas para evaluar la **usabilidad**, midiendo la facilidad con la que los usuarios ejecutaban tareas como registro y búsqueda de matrículas, obteniendo resultados positivos sin necesidad de capacitación. En cuanto al **rendimiento**, se analizó el tiempo de respuesta en distintas configuraciones de hardware, logrando un procesamiento eficiente con variaciones según la calidad del video y el equipo utilizado. La **accesibilidad** fue validada en diferentes dispositivos y resoluciones, asegurando compatibilidad general. Finalmente, se evaluó la **experiencia del usuario (UX)**, destacando el diseño organizado y la facilidad de navegación, con mejoras aplicadas según la retroalimentación para optimizar la usabilidad. Cabe aclarar que dichas pruebas fueron realizadas a personas de interés que probaron el producto y llenaron una encuesta.

4.2.1. Comparación con alternativas. Para evaluar la efectividad del sistema desarrollado, se comparó con alternativas disponibles en el mercado, tanto comerciales como de código abierto, enfocadas en el reconocimiento de matrículas. Se seleccionaron herramientas representativas como OpenALPR y PlateRecognizer, las cuales fueron sometidas a pruebas en condiciones similares a las del sistema propuesto.

Los resultados mostraron que, aunque estas soluciones ofrecen un rendimiento adecuado en identificación, presentan limitaciones en personalización y adaptación a entornos específicos. En contraste, el sistema desarrollado demostró mayor flexibilidad, permitiendo ajustes en los parámetros de reconocimiento y optimización del procesamiento de imágenes para mejorar la precisión en condiciones adversas, como baja iluminación y ángulos complejos.

Mientras que muchas opciones comerciales requieren licencias costosas, la propuesta presentada se posiciona como una alternativa accesible y adaptable sin comprometer precisión ni velocidad. Por otro lado, las soluciones de código abierto disponibles son limitadas y, en muchos casos, carecen de una interfaz gráfica intuitiva, lo que dificulta su adopción por usuarios sin experiencia técnica. Frente a esto, nuestro sistema ofrece una experiencia más accesible y fácil de usar, destacándose como una opción superior dentro de este ámbito.

5. Conclusiones y recomendaciones a futuro

El presente trabajo ha desarrollado un sistema de detección e identificación de vehículos basado en Machine Learning y Deep Learning, utilizando modelos de la familia YOLO para la segmentación de placas y el reconocimiento de caracteres. La solución propuesta prioriza un balance entre precisión, eficiencia computacional y flexibilidad, permitiendo su implementación en distintos entornos con capacidades de hardware variables.

Uno de los principales hallazgos del estudio fue la confirmación de que los modelos más recientes y pesados de YOLO no siempre ofrecen una mejora significativa en la segmentación de matrículas. A través del análisis de métricas como mAP50, mAP50-95 y fitness, se determinó que el modelo YOLOv8n (nano) ofrece un rendimiento equiparable a versiones más grandes en términos de precisión y recall. Dado que el objetivo principal del sistema es operar en tiempo real o con una latencia mínima, YOLOv8n fue seleccionado como modelo por defecto para la segmentación de placas, evitando la necesidad de hardware especializado para su ejecución en entornos estándar.

En el caso del reconocimiento de caracteres, se encontró un comportamiento similar, donde YOLOv8n mantiene un desempeño competitivo frente a modelos más recientes y ligeros de otras generaciones. Su consolidación dentro de la comunidad, junto con una documentación más extensa y mayor compatibilidad con herramientas de desarrollo, lo convierten en una opción estable y confiable para esta tarea. Sin embargo, a diferencia del proceso de segmentación, el reconocimiento de caracteres sí experimenta mejoras notables al utilizar modelos más complejos, especialmente

en la identificación de caracteres con morfologías similares como ‘I’, ‘l’, ‘O’ y ‘0’. Para brindar flexibilidad en distintas condiciones operativas, la interfaz gráfica del sistema permite seleccionar modelos alternativos de mayor capacidad, siempre que los recursos computacionales lo permitan.

Se integró una herramienta de corrección homográfica para mejorar la perspectiva y claridad de las matrículas antes del procesamiento. Aunque esta función incrementa la precisión hasta en un 2 % en ciertos caracteres, también reduce la velocidad de operación, afectando la tasa de inferencia en hasta 1 fotograma. Por ello, se recomienda activarla según las condiciones del entorno. Actualmente, la modificación de sus parámetros requiere acceso a los archivos de configuración y debe ser realizada por personal técnico con conocimientos en ajuste de imágenes.

Desde una perspectiva práctica, uno de los desafíos más importantes fue la computación requerida para entrenar los modelos. A pesar de utilizar conjuntos de datos optimizados y tamaños de imagen relativamente pequeños, el proceso de entrenamiento de los modelos YOLO demandó una cantidad significativa de horas de cómputo, incluso en entornos con aceleración por GPU. En algunos casos, se hizo necesario recurrir a configuraciones avanzadas de gestión de memoria y procesamiento en plataformas como Google Colab y Kaggle, debido a que la combinación de tamaño de lote, complejidad de los modelos y otros parámetros superó los límites de hardware disponibles.

La evaluación de la interfaz gráfica mostró alta usabilidad y eficiencia, permitiendo que incluso operarios sin experiencia en software avanzado la utilizaran sin dificultades. Las pruebas destacaron la claridad en la disposición de los elementos, la facilidad de uso y la rapidez en la visualización de resultados. En comparación con otras soluciones, la interfaz desarrollada resultó

más amigable y adaptable al entorno de vigilancia. Una interfaz bien diseñada no solo facilita la adopción del sistema, sino que también optimiza el flujo de trabajo y mejora la experiencia del usuario.

En general, los resultados obtenidos evidencian que la implementación de modelos de detección y reconocimiento de matrículas en tiempo real es viable con un uso eficiente de los recursos. El sistema desarrollado ofrece una solución escalable y adaptable, con opciones configurables que permiten ajustar su desempeño según los requerimientos específicos de cada entorno.

La implementación de este sistema de detección vehicular mejora la seguridad y eficiencia en la gestión del tráfico, optimizando el control de matrículas y reduciendo errores. Además, moderniza un sector con poca actualización tecnológica y alta rotación de personal, automatizando tareas clave para reducir la dependencia de operarios y garantizar una gestión más estable y confiable. Su impacto fortalece la seguridad y organización vehicular, promoviendo la transformación digital y futuras innovaciones.

Respecto a las sugerencias para posibles futuros trabajos, se recomienda ampliar y diversificar el conjunto de datos de entrenamiento incorporando imágenes capturadas en distintas condiciones climáticas, horarios y tipos de cámaras, además de aplicar técnicas avanzadas de data augmentation para mejorar la robustez del modelo, como la simulación de desgaste en las placas, suciedad, reflejos y oclusiones parciales. También sería beneficioso optimizar el rendimiento del sistema para hardware de bajo costo mediante técnicas como quantization y pruning. En términos de escalabilidad, se sugiere desarrollar una API adaptable que permita la integración del sistema con plataformas de terceros y facilite su implementación como servicio en la nube o en una red de

servidores, garantizando así un procesamiento con mayor disponibilidad y de mayor capacidad de acceso. Finalmente, se propone la posibilidad de expansión del sistema en otros países ajustando la detección y reconocimiento de matrículas según normativas locales, además de explorar su aplicación en gestión de tráfico inteligente, control de peajes y análisis de movilidad urbana. Además, Se recomienda implementar el sistema de detección vehicular en plataformas más portables y de bajo consumo, como dispositivos embebidos, para garantizar su funcionamiento en entornos con recursos limitados. Tecnologías como NVIDIA Jetson, Raspberry Pi con aceleradores TPU, o FPGA pueden ser utilizadas para ejecutar el modelo de manera eficiente sin comprometer el rendimiento.

Referencias Bibliográficas

Ariza, L. and Merchan, S. (2009). Reconocimiento de placas de automóvil usando la plataforma ecb_at91. Universidad Industrial de Santander.

Collins, R. (2007). Introduction to computer vision. *CSE Department, Penn State University*, 31:lecture 16. Véase también el texto original en: `https://www.cse.psu.edu/~rtc12/CSE486/`.

Gonzalez, R. C. and Woods, R. E. (2008). *Digital Image Processing*. Prentice Hall, Upper Saddle River, NJ, 3rd edition.

Illingworth, J. and Kittler, J. (1987). The adaptive hough transform. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(5):690–698.

Massiris, M., Delrieux, C., and Fernández, J. (2018). Detección de equipos de protección personal mediante red neuronal convolucional yolo. In *Actas de las XXXIX Jornadas de Automática, Badajoz, 5-7 de Septiembre de 2018*, pages 1022–1029. DOI del libro: `https://doi.org/10.17979/spudc.9788497497565`.

Ministerio de Defensa (2025). Información estadística de defensa y seguridad. Gov.Co. Recuperado el 13 de febrero de 2025.

Nolasco Valenzuela, J. S. (2018). *Python*. RA-MA Editorial.

OpenCV (2019). Open-source software library designed for computer vision and machine learning applications.

Orellana Preciado, J. P. and Ortega Castro, J. C. (2020). Tiempo de viaje, sistema de reconocimiento automático de placas vehiculares (anpr), para detección de infractores en ruta machala, santa rosa. *Polo del Conocimiento: Revista científico - profesional*, 5(1):400–412.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788.

Roboflow (2025). About - our company.

Sharma, R. and Kamra, A. (2023). A review on clahe based enhancement techniques. In *2023 6th International Conference on Contemporary Computing and Informatics (IC3I)*, volume 6, pages 321–325.

Steels, L. and de Mantaras, R. L. (2018). The barcelona declaration for the proper development and usage of artificial intelligence in europe. *AI Communications*, 31:485–494. Véase también el texto original en: `https://www.iiia.csic.es/barcelonadeclaration/`.

Ultralytics Inc. (2024a). Oficial documentation for yolo ultralytics library.

Ultralytics Inc. (2024b). Ultralytics is a company dedicated to developing cutting-edge artificial intelligence models, particularly in computer vision.

Vargas, A. and Archila, C. (2023). Implementación de un prototipo de bajo costo para la detección de personas desde vehículos en movimiento, mediante el uso de redes neuronales profundas. Universidad Industrial de Santander.

Vargas, J. and Moncada, W. (2021). Prototipo de reconocimiento automático de placas vehiculares usando redes neuronales profundas sobre un microcontrolador. Universidad Industrial de Santander. Editor: Universidad Industrial de Santander.

Wu, Q., Liang, T., Fang, H., Cao, J., Wang, M., and He, D. (2025). Multiple object characterization of recyclable domestic waste using binocular stereo vision. *Instrumentation Science & Technology*, 53(2):194–223.

Apéndice A. Repositorio Github tesis-colombian-plate-recognition

El código desarrollado se publicó como software libre bajo la licencia GPL-3.0 en un repositorio público en Github que se puede consultar a través del enlace adyacente. `https://github.com/oaacUis/tesis-colombian-plate-recognition` En la página inicial del repositorio hay una pequeña descripción sobre su desarrollo, funcionamiento y utilización, entre otras características y condiciones a tener en cuenta (*Readme*).

Apéndice B. Resultados de métricas por caracter para el modelo YOLOv8n.

Clase	No. imágenes	Instancias	Box(P	R	mAP50	mAP50-90)
0	39	46	0.956	0.913	0.936	0.803
1	37	46	0.925	0.935	0.908	0.725
2	38	40	0.949	0.923	0.925	0.799
3	42	52	0.907	0.885	0.883	0.747
4	25	30	0.773	0.680	0.693	0.581
5	34	41	0.900	0.876	0.846	0.723
6	46	52	0.848	0.788	0.790	0.674
7	37	38	0.913	0.895	0.907	0.778
8	33	33	0.925	0.909	0.930	0.803
9	25	29	0.912	0.931	0.928	0.827
A	22	24	0.999	0.958	0.959	0.792
B	28	31	0.965	0.968	0.992	0.847
C	16	17	0.976	1.000	0.995	0.899
D	13	13	0.915	0.829	0.869	0.732
E	13	13	0.882	0.846	0.831	0.726
F	13	14	0.975	1.000	0.995	0.861
G	23	23	0.866	0.844	0.857	0.688
H	10	10	0.950	0.900	0.957	0.854
I	14	14	0.823	0.786	0.777	0.670
J	9	9	0.954	1.000	0.995	0.880
K	16	16	0.737	0.700	0.669	0.609
L	11	11	0.966	0.909	0.971	0.852
M	12	14	0.888	0.857	0.860	0.758
N	13	13	0.836	0.923	0.924	0.796
O	8	8	0.885	0.962	0.954	0.863
P	13	13	0.954	0.923	0.938	0.834
Q	9	10	0.970	1.000	0.995	0.829

Clase	No. imágenes	Instancias	Box(P	R	mAP50	mAP50-90)
R	14	15	1.000	0.961	0.995	0.845
S	27	31	0.867	0.903	0.863	0.739
T	17	17	0.978	1.000	0.995	0.812
U	8	9	0.844	0.889	0.828	0.742
V	12	12	0.906	0.801	0.908	0.760
W	4	5	1.000	0.916	0.995	0.916
X	9	9	0.850	0.889	0.878	0.789
Y	2	2	0.829	1.000	0.995	0.846
Z	14	14	0.918	0.800	0.893	0.791

Apéndice C. Bases de datos

Como parte de este proyecto, se llevó a cabo una exhaustiva investigación de videos utilizados en proyectos similares, además de la recopilación de muestras propias para el entrenamiento y evaluación del sistema. Estas muestras pueden ser consultadas en el siguiente enlace:

`https://1drv.ms/f/c/7140141e7ee85278/EjjWUBRfrpJNpqRccRRgC38BO4FLrSo0sJM0rqZvlMSSAg`
`e=hKbKBT`

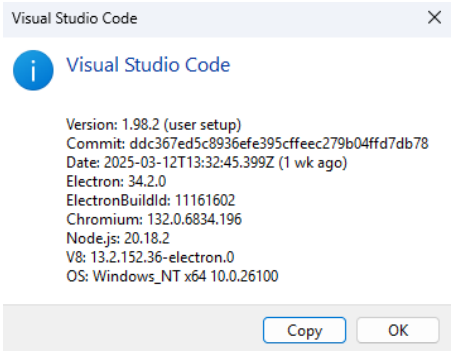
Para la detección de placas se utilizó el siguiente dataset de Roboflow: `https://universe.roboflow.com/kappsai/plate_detection_colombia/dataset/1`

Para la detección de caracteres se utilizó el siguiente dataset de Roboflow: `https://app.roboflow.com/tesis-vyo8x/characters-heyxh/3`

Apéndice D. Hardware y software

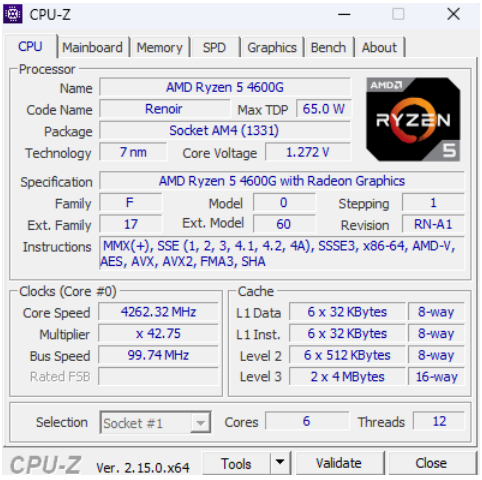
El principal software para programar el proyecto fue Visual Studio Code con la siguiente información relevante:

Figura 7
Versión de Visual Studio Code utilizado



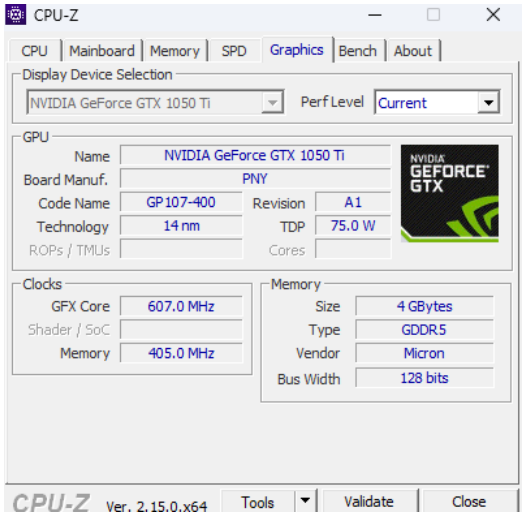
La CPU utilizada tiene las siguientes caracteristicas:

Figura 8
Características CPU utilizada



La tarjeta gráfica utilizada fue la siguiente:

Figura 9
Características tarjeta gráfica utilizada



Apéndice E. Costos

ESTRUCTURA DE LOS COSTOS

El modelo de negocio propuesto para la comercialización del sistema de detección vehicular se basa en una suscripción mensual con un costo entre $10 y $25 dólares, dependiendo de la cantidad de dispositivos conectados y el nivel de servicio requerido. Esta tarifa ha sido establecida teniendo en cuenta factores clave como el costo operativo, la competitividad en el mercado y el valor agregado que se ofrece a los clientes.

Justificación del precio

El precio de suscripción mensual es altamente competitivo en comparación con otras soluciones similares en el mercado, que pueden alcanzar costos significativamente más altos. Se han considerado los siguientes aspectos para establecer esta tarifa:

- **Menor costo en el mercado:** Al analizar las tarifas de servicios similares, se evidencia que

la mayoría de soluciones disponibles superan los $50 dólares mensuales, lo que posiciona a nuestra plataforma como una alternativa más accesible.

- **Actualizaciones mensuales:** La suscripción incluye mejoras continuas en el software, optimización del rendimiento y corrección de errores, asegurando que el sistema se mantenga actualizado con las últimas tecnologías de reconocimiento de matrículas.

- **Soporte técnico incluido:** Los usuarios suscritos recibirán asistencia técnica para resolver dudas, configurar el sistema y garantizar su correcto funcionamiento sin costos adicionales.

Comparación de costos con el mercado

Para demostrar la competitividad de nuestra propuesta, en la Tabla 3 se presenta una comparación con otras soluciones comerciales de reconocimiento de matrículas:

Tabla 3
Comparación de costos con soluciones del mercado

Proveedor	Costo Mensual	Incluye Soporte y Actualizaciones
OpenALPR	$65	Sí
Plate Recognizer	$50	Sí
Propuesta (Smart Resident Guard)	**$10 - $25**	**Sí**

Como se observa, nuestra solución ofrece una opción asequible sin comprometer la calidad del servicio. Además, el costo de suscripción incluye beneficios adicionales que otras opciones más costosas no ofrecen, como actualizaciones periódicas y soporte técnico sin cargos extras.

Estrategia de precios escalonados

Para adaptarnos a distintos tipos de clientes, se plantea un esquema de precios escalonado basado en el nivel de servicio y el número de dispositivos conectados:

Tabla 4
Planes de suscripción

Plan	Costo Mensual	Características
Básico	$10	Hasta 3 dispositivos, soporte estándar
Avanzado	$15	Hasta 5 dispositivos, actualizaciones premium
Empresarial	$25	Ilimitados dispositivos, soporte prioritario

De esta manera, la propuesta permite atender tanto a pequeños negocios como a grandes empresas, asegurando accesibilidad y calidad de servicio. Con este modelo, se logra una máxima rentabilidad sin generar una barrera económica para la adopción del sistema.