WASOR: “SISTEMA DE CLASIFICACIÓN DE RESIDUOS SÓLIDOS USANDO TÉCNICAS DE INTELIGENCIA ARTIFICIAL”

Héctor Iván Tovar Jaimes

9015448

Javier Hernando Pinto Losada

9013898

Proyecto Aplicado para optar al título de
Magíster en Ciencia de Datos

Director
Hernán Darío Vargas Cardona

FACULTAD DE INGENIERÍA Y CIENCIAS
MAESTRÍA EN CIENCIA DE DATOS
SANTIAGO DE CALI, DICIEMBRE 01 DE 2025

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

INTRODUCCIÓN

Actualmente, la humanidad y el planeta enfrentan serios problemas ambientales, uno de los cuales es el incremento de los niveles de contaminación en ríos, mares, lagunas y otras fuentes hídricas. Esta contaminación, producida principalmente por desechos sólidos domésticos e industriales, está ocasionando la pérdida de fuentes de agua dulce potable, la desaparición de fauna y flora marítima, y enfermedades en humanos por el consumo de agua no potable. Para combatir estos problemas, los gobiernos han comenzado a crear políticas públicas como la reforestación de bosques y ríos, el tratamiento y manejo de desechos sólidos urbanos, incentivos tributarios para empresas con planes de acción ambiental y la creación de ministerios del medio ambiente, entre otras medidas. Sin embargo, según la ONU, en su informe Global Waste Management Outlook de 2024, en Sudamérica se producen 120 millones de toneladas de residuos sólidos por día, de los cuales solo el 6% se recicla, el 60% va a vertederos y el 34% son residuos sólidos no controlados [1].

El proyecto WASOR (Waste Sorting) tiene como objetivo desarrollar un sistema inteligente de clasificación de residuos sólidos mediante técnicas de inteligencia artificial, abordando la baja eficiencia en la gestión de residuos y el escaso porcentaje de reciclaje en la región, que según el informe de la ONU este representa un 6% en Sur América [1]. Este sistema aplica algoritmos de visión por computadora para identificar y clasificar diversos tipos de desechos, asignándolos automáticamente al contenedor adecuado según el código de colores de reciclaje. Además, se creó un dataset de imágenes de residuos, el entrenamiento y la comparación de modelos de clasificación de imágenes y el desarrollo de una interfaz de usuario que permite la clasificación de los residuos [2].

Los resultados obtenidos fueron un modelo que clasifica los residuos y una interfaz de usuario que facilita el proceso de reciclaje. Lo anterior, contribuirá a una gestión de residuos más eficiente y sostenible, fomentando prácticas de economía circular y ayudando a reducir la huella ambiental.

1 DEFINICIÓN DEL PROBLEMA

En la actualidad, los sistemas de gestión de residuos sólidos presentan importantes limitaciones, principalmente debido a la ineficiencia en la clasificación de materiales reciclables. Este proceso se ve dificultado por varios factores, entre ellos la frecuente mezcla de materiales incompatibles en un mismo desecho, la falta de educación ambiental que conduce a una disposición inadecuada de los residuos, la alta variabilidad en la apariencia visual de los materiales reciclables (como plásticos con etiquetas, residuos contaminados o deformados), y la ausencia de tecnologías adaptativas que permitan una identificación precisa en tiempo real. La dependencia de métodos manuales no solo ralentiza el proceso, sino que también incrementa la probabilidad de errores en la separación, lo que afecta negativamente la recuperación de materiales reutilizables y contribuye al agravamiento de la contaminación ambiental.

Esta situación pone en evidencia la necesidad urgente de desarrollar soluciones tecnológicas innovadoras que permitan automatizar y optimizar la clasificación de residuos, con el fin de mejorar la eficiencia del reciclaje y fomentar prácticas sostenibles. En este contexto, se hace indispensable implementar un sistema inteligente, basado en técnicas de inteligencia artificial, que sea capaz de identificar y clasificar con precisión los distintos tipos de residuos sólidos, facilitando su disposición adecuada y reduciendo el impacto ambiental derivado de un manejo ineficiente.

1.1 PLANTEAMIENTO DEL PROBLEMA

La contaminación derivada de los residuos sólidos representa un desafío ambiental crítico en el contexto global actual. Este problema se ha intensificado por el crecimiento demográfico, la urbanización acelerada y el aumento de los patrones de consumo, los cuales generan volúmenes crecientes de desechos [3]. Según el Global Waste Management Outlook 2024 de la ONU, se estima que la generación de residuos aumentará de 2.01 mil millones de toneladas en 2016 a 3.40 mil millones en 2050 si no se implementan cambios significativos en las prácticas de gestión [1].

Uno de los factores clave que limita la eficiencia del reciclaje es la clasificación inadecuada de los residuos desde su origen. En muchos sistemas de gestión, los residuos reciclables, orgánicos y no reciclables terminan mezclados, lo que genera lo que se conoce como contaminación cruzada: la presencia de materiales incompatibles o sucios dentro de una misma categoría, lo cual compromete la calidad del material recuperado e incluso puede volverlo inutilizable para los procesos de reciclaje. Esto lleva a que grandes cantidades de residuos potencialmente reciclables terminen en vertederos o incinerados [4].

La raíz de esta situación es compleja. Por un lado, los residuos suelen estar compuestos por múltiples materiales, como envases con partes plásticas, metálicas y de papel, lo que genera ambigüedad sobre su disposición correcta. Por otro lado, no existe una estandarización clara y universalmente comprendida en el etiquetado de los productos ni en los colores de las canecas, lo que incrementa la confusión en el momento de separar. A esto se suma una baja cultura de separación en la fuente: muchas personas no han recibido educación suficiente sobre clasificación o simplemente no comprenden las diferencias entre los tipos de residuos. Incluso quienes tienen disposición para reciclar enfrentan dificultades reales para identificar adecuadamente la categoría correcta para ciertos elementos, lo que se traduce en errores frecuentes de clasificación. [4]

A pesar de los avances tecnológicos disponibles, la clasificación manual sigue siendo el método predominante en muchas regiones del mundo. Este proceso es laborioso, propenso a errores humanos, costoso y difícil de escalar ante el volumen creciente de residuos. La complejidad de los flujos de desechos, sumada a la baja precisión en su manejo, no solo limita la recuperación de materiales aprovechables, sino que también contribuye al aumento de residuos acumulados en vertederos, cuerpos de agua y entornos naturales. Esto impacta negativamente la biodiversidad, degrada los suelos y representa un riesgo creciente para la salud pública.

En este contexto, resulta urgente abordar las causas estructurales y cognitivas que dificultan una clasificación adecuada, reconociendo que el problema no radica únicamente en la falta de

voluntad de los usuarios, sino en las dificultades reales del proceso: ambigüedad de los residuos, desconocimiento, falta de herramientas de apoyo y ausencia de sistemas automatizados que respalden las decisiones de clasificación. Comprender estas barreras es fundamental para diseñar soluciones más eficaces que permitan reducir la contaminación cruzada, mejorar la calidad del material reciclado y fortalecer el impacto ambiental positivo de los programas de gestión de residuos sólidos.

1.2 FORMULACIÓN DEL PROBLEMA

Los métodos convencionales de clasificación de residuos, basados principalmente en procesos manuales, presentan importantes limitaciones operativas. Estos métodos requieren una gran inversión de tiempo, esfuerzo humano y recursos, lo que repercute negativamente en la eficiencia del sistema de gestión de residuos. Además, la alta probabilidad de errores durante la separación afecta la correcta recuperación de materiales reciclables y dificulta la implementación de prácticas sostenibles a gran escala.

Ante este panorama, surge la necesidad de replantear el enfoque tradicional y avanzar hacia soluciones tecnológicas que permitan mejorar la precisión y velocidad en la clasificación de desechos. El desarrollo de herramientas basadas en inteligencia artificial ofrece una alternativa prometedora para automatizar esta tarea, optimizando el proceso y reduciendo su margen de error. Sin embargo, para que estos modelos puedan aprender a reconocer con precisión los distintos tipos de residuos, es fundamental contar con un conjunto de datos visuales robusto, diverso y representativo. La calidad y variedad del dataset en cuanto a tipos de residuos, condiciones de iluminación, ángulos de captura y escenarios de uso tiene un impacto directo en la capacidad del modelo para generalizar y funcionar adecuadamente en contextos reales. Por tanto, la construcción de un dataset de imágenes de residuos sólidos, etiquetado adecuadamente y con alta resolución, se convierte en un componente esencial para el éxito de cualquier sistema inteligente de clasificación. [5]

La principal motivación de este trabajo fue dar respuesta a la siguiente pregunta:

¿Cómo se puede desarrollar un sistema de clasificación de residuos sólidos utilizando técnicas de inteligencia artificial para optimizar la separación de diferentes tipos de desechos?

1.2.1 Sistematización

- ¿Cómo se puede garantizar que el dataset de imágenes de residuos sólidos sea de calidad y suficientemente representativo para optimizar el entrenamiento de los modelos de clasificación?
- ¿Qué técnicas y configuraciones de redes neuronales son más adecuadas para entrenar modelos de clasificación de imágenes de residuos sólidos?
- ¿Qué modelo de clasificación muestra un mejor rendimiento y capacidad de generalización según las métricas de evaluación seleccionadas?
- ¿Cómo diseñar una interfaz de usuario que facilite la clasificación en tiempo real de residuos sólidos, asegurando precisión en la asignación de recipientes y una experiencia intuitiva para el usuario?

2 JUSTIFICACIÓN

Los problemas ambientales, exacerbados por la alta contaminación de ríos y mares, la pérdida de fuentes hídricas, la reducción de la biodiversidad y el deterioro de los suelos, han generado un impacto negativo en la salud pública y en los ecosistemas globales. El calentamiento global, impulsado en gran parte por estas problemáticas, está provocando efectos devastadores como el derretimiento de los polos, el aumento del nivel del mar y la alteración irreversible de los ecosistemas. Ante este panorama, se hace urgente implementar soluciones innovadoras que contribuyan a mitigar estos efectos y restaurar el equilibrio ambiental.

Una de las causas principales de la contaminación es la gestión inadecuada de los residuos sólidos. En América Latina, la falta de educación ambiental y la ausencia de políticas públicas eficaces han incrementado el volumen de desechos mal gestionados, gran parte de los cuales terminan en vertederos o en cuerpos de agua. A pesar de los esfuerzos por mejorar el reciclaje, solo un pequeño porcentaje de los residuos sólidos es procesado de forma adecuada, lo que agrava la contaminación y su impacto en el medio ambiente.

Este proyecto se centra en el entrenamiento de un modelo basado en inteligencia artificial para clasificar residuos sólidos de manera automática, utilizando un prototipo funcional con un entorno visual que aprovecha la visión por computadora para realizar la segregación correcta y fomentar una disposición eficiente de los residuos.

El sistema propuesto permitirá a los usuarios identificar el tipo de residuo y la categoría de reciclaje correspondiente, mejorando las tasas de reciclaje y reduciendo los desechos mal gestionados. Además, al proporcionar retroalimentación inmediata sobre el tipo de residuo y el contenedor adecuado para su disposición, el sistema contribuirá a educar a la población en prácticas de reciclaje más efectivas y responsables.

Para garantizar la viabilidad del modelo, se utilizarán datasets de imágenes de residuos recopilados de repositorios públicos confiables, como GitHub y Kaggle. Estas bases de datos serán evaluadas para asegurar su calidad y relevancia, y se dividirán en un dataset para entrenamiento, pruebas y validación, asegurando una adecuada separación de los datos. Las imágenes serán

etiquetadas rigurosamente según las categorías de residuos, y se aplicarán técnicas de aumento de datos, como transformaciones geométricas, normalización, ajustes de color y aplicación de filtros, para garantizar la representatividad y suficiencia del dataset.

Este proyecto no solo responde a una necesidad urgente de mejorar la clasificación y disposición de residuos, sino que también se alinea con los esfuerzos globales para reducir la contaminación, conservar los recursos naturales y fomentar la sostenibilidad. A través de la adopción de tecnologías inteligentes, el proyecto puede hacer una contribución significativa a la conciencia ambiental, ayudando a la sociedad a comprender la importancia del reciclaje y la correcta disposición de residuos, y abriendo la puerta a un futuro más limpio y saludable para las generaciones venideras.

3 OBJETIVOS

3.1 OBJETIVO GENERAL

- Desarrollar un sistema de detección de residuos sólidos utilizando técnicas de inteligencia artificial para identificar y separar diferentes tipos de desechos.

3.2 OBJETIVOS ESPECÍFICOS

- Adquirir un dataset de imágenes de residuos sólidos, asegurando su calidad y representatividad para entrenar modelos de clasificación.
- Entrenar modelos de clasificación de imágenes aplicando técnicas basadas en redes neuronales profundas.
- Evaluar el rendimiento de los modelos mediante métricas de clasificación.
- Desarrollar un prototipo funcional que utilice visión por computadora para clasificar residuos sólidos, indicando en qué recipiente debe disponerse cada tipo de residuo.

4 MARCO TEÓRICO Y ANTECEDENTES

4.1 MARCO TEÓRICO

A continuación, se presentan los resultados de la revisión de la literatura para exponer los principales conceptos que sustentan el proyecto de investigación, enfocándose en la relación entre las problemáticas ambientales, como el bajo porcentaje de reciclaje y las soluciones tecnológicas basadas en inteligencia artificial. Esta sección destaca cómo la automatización y el aprendizaje automático pueden contribuir al entendimiento de los datos y la mejora en los procesos de reciclaje, promoviendo la sostenibilidad y la eficiencia.

4.1.1 Medio ambiente

El medio ambiente comprende el conjunto de elementos naturales, como el aire, el agua, el suelo, las plantas y los animales, que interactúan con los seres humanos y forman el entorno en el que vivimos. Este sistema dinámico regula procesos esenciales como el ciclo del agua y el clima, proporcionando recursos vitales para la supervivencia. Sin embargo, la presión humana, derivada de actividades como la producción masiva de residuos sólidos, está poniendo en peligro este equilibrio, agravando problemas como la contaminación y el cambio climático.

4.1.2 Crisis climática

La crisis climática es uno de los mayores desafíos actuales, impulsada por actividades humanas como la quema de combustibles fósiles y la gestión inadecuada de residuos sólidos. Según la ONU, el escaso reciclaje y la acumulación de residuos no controlados son factores que contribuyen a la emisión de gases de efecto invernadero, agravando fenómenos climáticos extremos y amenazando la biodiversidad. En este contexto, mejorar la gestión de residuos mediante tecnologías avanzadas como la automatización puede ser una herramienta crucial para mitigar estos impactos.

4.1.3 Reciclaje

El bajo porcentaje de reciclaje es un síntoma de sistemas de clasificación ineficientes, en su mayoría manuales, que presentan problemas como errores humanos, lentitud y altos costos

operativos. Esto no solo reduce la calidad del reciclaje, sino que también genera mayores volúmenes de residuos destinados a vertederos. La adopción de tecnologías como la visión artificial y el reconocimiento de imágenes ofrece una alternativa prometedora para mejorar la precisión y la velocidad de la clasificación, aumentando así las tasas de reciclaje.

4.1.4 Automatización de procesos

La automatización consiste en el uso de tecnologías avanzadas para realizar tareas de forma eficiente y precisa, reduciendo la intervención humana. En el reciclaje, la automatización permite implementar sistemas que clasifican residuos sólidos de manera autónoma, integrando hardware y software como cámaras y redes neuronales para optimizar la separación de materiales y reducir los errores.

4.1.5 Reconocimiento de imágenes

El reconocimiento de imágenes es una herramienta clave en los sistemas automatizados de reciclaje. Mediante algoritmos avanzados, esta tecnología identifica y clasifica materiales, como papel, plástico, vidrio y metal, con alta precisión. Esto permite una separación más efectiva de los residuos desde su origen, facilitando la reutilización y reduciendo el impacto ambiental.

4.1.6 Machine learning y deep learning

El machine learning y el deep learning son pilares de los sistemas modernos de automatización. Estas técnicas permiten que los modelos aprendan de grandes volúmenes de datos y mejoren su desempeño con el tiempo [6]. Aplicados al reciclaje, pueden identificar patrones complejos en los residuos, aumentando la precisión y adaptabilidad de los sistemas de clasificación automatizados.

4.1.7 Deep convolutional neural network (CNN)

Las redes neuronales convolucionales (CNN), como AlexNet y ResNet, han demostrado ser particularmente efectivas en el procesamiento de imágenes [7]. Estas redes son fundamentales para el desarrollo de sistemas que clasifiquen residuos sólidos mediante la interpretación de imágenes capturadas por cámaras, permitiendo un análisis detallado y una toma de decisiones precisa.

4.2 ANTECEDENTES

A continuación, en la ***Tabla 1.*** Estudios de clasificacion de imágenes usando ciencia de datos, se describen los trabajos de investigación que guardan relación con el objeto de estudio y presentan un marco de antecedentes en el uso de la ciencia de datos en la clasificación de imágenes.

Tabla 1. Estudios de clasificacion de imágenes usando ciencia de datos

Estudio	Contribución
Clasificación de imágenes usando redes neuronales convolucionales en Python Autor: Álvaro Artola Moreno Año: 2019	El trabajo de grado ofrece una guía integral para la implementación de redes neuronales convolucionales (CNN) enfocándose en la preparación de datos, selección de arquitecturas avanzadas como ResNet y VGG, y evaluación del rendimiento utilizando métricas estándar. Asimismo, incluye ejemplos prácticos con Python y bibliotecas como Keras y TensorFlow, proporcionando una base sólida para desarrollar proyectos de clasificación de imágenes y optimización de modelos [8]. Este trabajo aporta conocimientos fundamentales para el proyecto de clasificación de residuos al ofrecer estrategias para preparar datos, seleccionar arquitecturas adecuadas y evaluar modelos. Las herramientas y ejemplos prácticos en Python pueden ser directamente aplicables para entrenar el modelo de clasificación de residuos sólidos. Sin embargo, mientras el enfoque del trabajo de grado se centra en una guía general sobre CNN y su implementación técnica, el proyecto WASOR avanza un paso más al aplicar estas técnicas en un contexto específico de clasificación de residuos sólidos, incorporando un sistema funcional para guiar al usuario en el reciclaje.
A New Design Based-SVM of the CNN Classifier Architecturewith DropoutforOffline Arabic Handwritten Recognition Autores: Mohamed Elleuch, Rania Maalej and Monji Kherallah.	La investigación propone un enfoque híbrido que combina redes neuronales convolucionales (CNN) con máquinas de soporte vectorial (SVM) y el uso de dropout para reducir el sobreajuste, mejorando así la capacidad de clasificación y la robustez del modelo. Este enfoque puede optimizar la precisión y la generalización en problemas de clasificación de imágenes, como el reconocimiento de escritura a mano en árabe, y podría adaptarse para mejorar la identificación de residuos sólidos en contextos con alta diversidad de categorías y calidad de datos variable [9].

Año: 2016	El trabajo presenta un enfoque híbrido que mejora la precisión y robustez del modelo, lo que puede ser valioso para el proyecto WASOR, que enfrenta retos similares en la clasificación de residuos sólidos. A diferencia de este enfoque, el proyecto WASOR se enfoca en la clasificación de residuos mediante una solución automatizada en tiempo real para mejorar la eficiencia en la gestión de residuos, mientras que la investigación de Elleuch et al. se centra en el uso de SVM y CNN para la clasificación offline de escritura manuscrita.
Melanoma Thickness Prediction Based on Convolutional Neural Network with VGG-19 Model Transfer Learning Autores: Joanna Jaworek-Korjakowska, Pawel Kleczek, Marek Gorgon Año: 2019	En este artículo de investigación, se aborda la clasificación de imágenes mediante la implementación de técnicas avanzadas de aprendizaje por transferencia, utilizando redes neuronales convolucionales pre-entrenadas, como el modelo VGG-19. Además, se proponen estrategias para manejar datos desbalanceados, incluyendo el uso de funciones de pérdida especializadas como Focal Loss y la validación cruzada para mejorar la robustez del modelo. Estos enfoques permiten una clasificación más precisa incluso con conjuntos de datos limitados o diversos, destacando su utilidad en problemas relacionados con imágenes biomédicas [10]. Este artículo contribuye al proyecto actual al destacar el potencial de las técnicas de aprendizaje por transferencia y redes pre-entrenadas para resolver problemas de clasificación con datos complejos y desbalanceados, como ocurre en el reciclaje automatizado de residuos sólidos. Sin embargo, existen diferencias clave: mientras el estudio se centra en imágenes biomédicas (melanomas), el proyecto WASOR aplica estas técnicas a imágenes de residuos sólidos para clasificarlos según categorías reciclables.
Classification-assisted Deep Sparse Image Recognition Autores: Fuli Zhu, Wenhai Chen y Liang Chen Año: 2021	Este estudio propone un enfoque innovador para mejorar el reconocimiento de imágenes dispersas mediante el uso de redes neuronales profundas asistidas por clasificación. El modelo asistido por clasificación aborda desafíos típicos en datasets con poca información o datos dispersos, optimizando el proceso de identificación de patrones. Este método incrementa la precisión del reconocimiento en escenarios donde las imágenes contienen datos limitados o presentan estructuras no convencionales. La investigación subraya la importancia de integrar técnicas de aprendizaje profundo con estrategias de clasificación para resolver

	problemas complejos en el análisis de imágenes [11]. El enfoque de imágenes dispersas propuesto en este estudio puede ser útil en el proyecto WASOR al mejorar la clasificación de residuos sólidos cuando los datos son limitados o poco estructurados. Sin embargo, a diferencia de este enfoque, el proyecto WASOR se centra en la clasificación de residuos con un modelo CNN optimizado para identificar materiales específicos y asignarlos a contenedores, integrando además una interfaz que guía al usuario en el proceso de reciclaje.

5 COMPRENSIÓN DEL PROBLEMA, ADQUISICIÓN Y PREPARACIÓN DE DATASET DE IMÁGENES

Este capítulo aborda las etapas iniciales del desarrollo del sistema inteligente de clasificación de residuos sólidos, comenzando por la comprensión del problema ambiental que motiva esta investigación. Se describe el estado actual del reciclaje, los desafíos asociados y las limitaciones de las soluciones existentes, así como la propuesta de un sistema basado en visión por computador. Posteriormente, se detallan los procesos de adquisición, limpieza, etiquetado y normalización de datos, fundamentales para conformar un conjunto robusto de imágenes representativas. Estas acciones garantizaron la calidad y diversidad necesarias para entrenar modelos de clasificación eficaces y aplicables en contextos reales.

4.1 COMPRENSIÓN DEL PROBLEMA

El reciclaje y la gestión de residuos sólidos se han convertido en elementos clave dentro de las estrategias globales para mitigar el impacto ambiental del modelo económico lineal actual. Sin embargo, a pesar de los avances tecnológicos, normativos y educativos, la problemática persiste con una magnitud creciente. En este capítulo se describe el estado actual del reciclaje a nivel global, se identifican las soluciones existentes y sus limitaciones, y se presenta como una solución basada en inteligencia artificial puede contribuir a mejorar la eficiencia del proceso de clasificación de residuos.

5.1.1 Estado actual del reciclaje y desafíos asociados

Según el informe "Global Waste Management Outlook 2024" del Programa de las Naciones Unidas para el Medio Ambiente (UNEP), la generación mundial de residuos sólidos municipales (RSU) supera los 2.100 millones de toneladas anuales, y se proyecta que esta cifra alcance los 3.800 millones de toneladas para el año 2050 si no se toman medidas correctivas urgentes. A pesar de este volumen, solo el 19 % de estos residuos es reciclado de manera efectiva, lo que revela una brecha crítica entre la generación y la valorización de residuos [1].

En paralelo, el informe "The State of Recycling Today 2024" revela que la industria del reciclaje enfrenta una "crisis de confianza" debido a la contaminación cruzada de materiales, la baja calidad del material recuperado y la falta de transparencia en la cadena de valorización. En muchas ocasiones, los residuos clasificados como reciclables son rechazados o terminan en vertederos debido a una separación inadecuada desde la fuente o a prácticas de exportación con estándares insuficientes [2].

Entre los principales desafíos se destacan:

- La infraestructura insuficiente para la recolección y el tratamiento adecuado de los residuos.
- La baja participación ciudadana en la separación en la fuente.

- La falta de integración del sector informal, que desempeña un rol crucial en países en desarrollo.
- El limitado desarrollo de políticas públicas orientadas hacia modelos circulares.

Además, el aumento del consumo y la urbanización intensifican la generación de residuos sin que se logre una desaceleración proporcional mediante mecanismos preventivos.

5.1.1.1 Soluciones existentes y sus limitaciones

Diversas soluciones han sido implementadas a nivel global con el objetivo de mejorar la eficiencia de la gestión de residuos. Entre ellas se encuentran:

- **Sistemas mecánicos de separación y clasificación automatizada:** Utilizados en plantas industriales de gran escala, emplean sensores ópticos y separadores magnéticos. Aunque eficaces, presentan altos costos de implementación y requieren infraestructura robusta.
- **Esquemas de responsabilidad extendida del productor (REP)**: Obligan a los fabricantes a hacerse cargo del ciclo de vida completo de sus productos. Su implementación es todavía incipiente en muchos países del sur global.
- **Compostaje comunitario y valorización orgánica:** Representan una solución de bajo costo para residuos biodegradables, pero su efectividad requiere cambios culturales y soporte técnico sostenido.

Estas iniciativas, aunque prometedoras, presentan limitaciones comunes: fragmentación institucional, escasa adaptabilidad local, falta de sostenibilidad económica y débil inclusión del componente social. El modelo tradicional de reciclaje, centrado exclusivamente en la etapa postconsumo, ha demostrado ser insuficiente frente a la magnitud del problema actual.

5.1.1.2 Solución propuesta: clasificación inteligente mediante visión por computador

La propuesta de este proyecto consiste en el desarrollo de un sistema inteligente de clasificación automática de residuos sólidos mediante visión por computador e inteligencia artificial. Este sistema busca reconocer, clasificar y orientar la disposición correcta de los residuos en tiempo real a través de una interfaz amigable.

En la ***Figura 1*** presenta el flujo metodológico del sistema propuesto, que abarca desde la adquisición de imágenes de residuos hasta la implementación de un demo funcional. El proceso inicia con la recopilación de datos, seguida de su preparación mediante procesamiento y etiquetado, asegurando calidad y proporcionalidad entre clases. Posteriormente, se realiza el entrenamiento de los modelos preentrenados, y su evaluación mediante métricas de desempeño para seleccionar la configuración más adecuada. Finalmente, el modelo seleccionado se implementa en una interfaz HMI en tiempo real, validando su aplicabilidad en un entorno operativo.

Figura 1. Flujo del Sistema

Desde el punto de vista técnico, el uso de redes neuronales convolucionales (CNN) permite alcanzar alto desempeño en entornos controlados, reduciendo los errores humanos asociados a la clasificación visual. Además, mediante técnicas de aumento de datos (data augmentation) y entrenamiento robusto, el sistema puede adaptarse a condiciones reales de iluminación y contexto urbano.

Desde la perspectiva operativa, esta solución promueve:

- Automatización de bajo costo: Viable para su implementación en puntos ecológicos, instituciones educativas y municipios de tamaño medio.
- Prevención de contaminación cruzada: Al mejorar la clasificación desde el origen, se optimiza la calidad del material reciclable.
- Educación ambiental activa: Al interactuar con el usuario, se refuerza el aprendizaje sobre separación adecuada.
- Generación de datos estructurados: El sistema en etapas posteriores podrá integrarse con plataformas de gestión de residuos y aportar información sobre volúmenes y tipos de desechos clasificados.

Esta propuesta, al enfocarse en la automatización accesible, la interacción educativa y la adaptabilidad local, contribuye tanto a la mejora de la eficiencia operativa como al fortalecimiento de una cultura ambiental ciudadana.

5.2 ADQUISICIÓN DE DATOS

La fase de adquisición de datos es uno de los pilares fundamentales en el ciclo de vida de un proyecto de Ciencia de Datos, especialmente en aplicaciones de visión por computadora. La calidad, representatividad y diversidad del conjunto de datos inciden directamente en la capacidad del modelo para generalizar y desempeñarse correctamente en entornos reales. En

este trabajo, orientado a la clasificación automática de residuos sólidos, se desarrolló una estrategia rigurosa para construir un dataset sólido, confiable y éticamente adquirido.

5.2.1 Exploración de repositorios de imágenes

Se emprendió una búsqueda exhaustiva y sistemática de datasets disponibles en línea, con especial énfasis en repositorios que ofrecieran imágenes de dominio público o bajo licencias permisivas (Creative Commons, MIT, GPL, etc.). Las plataformas consultadas incluyeron:

- Kaggle: conocido por su amplia variedad de datasets etiquetados, algunos orientados específicamente a residuos sólidos urbanos.
- GitHub: se revisaron proyectos académicos y de código abierto que incluían imágenes de residuos clasificadas manualmente.
- Pexels y Unsplash: bancos de imágenes con políticas de uso libre que ofrecían recursos útiles para complementar clases poco representadas.
- OpenML y Google Dataset Search: motores de búsqueda especializados para bases de datos abiertas y científicas.

La exploración no solo se limitó a la descarga de imágenes, sino también a la validación de las fuentes. Se priorizaron datasets que incluyeran documentación clara, especificaciones técnicas, licencias explícitas de uso y una estructura de carpetas organizada.

5.2.2 Adquisición de imágenes

El proceso de adquisición culminó con la recopilación de 9.281 imágenes, obtenidas a partir de la plataforma abierta como GitHub. Estas imágenes están distribuidas en siete categorías de residuos sólidos, las cuales representan el sistema de clasificación comúnmente utilizado en procesos de reciclaje. Dichas categorías corresponden a:

- Papel: periódicos, cartón, hojas impresas, cuadernos.

Figura 2. Imágenes categoría papel

- Orgánico: frutas, vegetales, desechos de comida.

Figura 3. Imágenes categoría orgánico

- Plástico: botellas, envoltorios, envases plásticos de diversos tipos.

Figura 4. Imágenes categoría plástico

- Vidrio: frascos, botellas, fragmentos de vidrio.

Figura 5. Imágenes categoría vidrio

- Metal: latas, tapas metálicas, herramientas oxidadas.

Figura 6. Imágenes categoría metal

- Biológico: material médico, guantes, jeringas (representando residuos hospitalarios).

Figura 7. Imágenes categoría biológico

- Baterías: pilas y acumuladores, consideradas residuos peligrosos.

Figura 8. Imágenes categoría baterías

Cada imagen fue sometida a un proceso de validación manual, mediante inspección visual, para

verificar su pertinencia, claridad, enfoque y contenido. Se excluyeron imágenes borrosas, que no pertenecían a ninguna clase, duplicadas o con fondos que dificultaran el entrenamiento del modelo. Adicionalmente, se evitó el uso de imágenes que incluyeran marcas registradas o personas identificables, con el fin de cumplir con estándares éticos y de privacidad.

5.2.3 Consolidación del dataset

Una vez obtenidas y clasificadas las imágenes, se procedió a la consolidación del dataset siguiendo buenas prácticas para el desarrollo de modelos de clasificación de imágenes:

- Estandarización de Formato: todas las imágenes fueron convertidas al formato .jpg con compresión mínima para mantener la calidad visual.
- Estructura de Carpetas: se creó una carpeta para cada clase de residuos. (Orgánico, papel, Plástico, Vidrio, Metal, Biológico, Baterías)

Esta organización permite una carga eficiente de datos, que infieren las etiquetas directamente del nombre de las carpetas.

- Revisión del Balance de Clases: se identificaron ligeros desbalances en la cantidad de imágenes por categoría (por ejemplo, categorías como “plástico” y “papel” tendían a estar sobrerrepresentadas respecto a “biológico” o “baterías”).
- Licenciamiento y Ética: se garantizó que todas las imágenes utilizadas en el dataset provienen de fuentes legales y abiertas, en cumplimiento con las regulaciones sobre propiedad intelectual, incluyendo la Ley 1581 de 2012 en Colombia sobre protección de datos personales. Además, se documentaron las fuentes originales de cada conjunto descargado, asegurando trazabilidad.

5.3 PREPARACIÓN DE LOS DATOS

La preparación del conjunto de datos fue una etapa clave para asegurar la calidad, diversidad y representatividad de las imágenes utilizadas en el entrenamiento del modelo de clasificación de residuos. Este capítulo detalla las actividades desarrolladas durante el procesamiento, limpieza, etiquetado y normalización del dataset.

5.3.1 Limpieza de datos

El conjunto de datos original fue obtenido del repositorio open source de GitHub denominado Agamiko, el cual agrupa imágenes provenientes de múltiples fuentes, incluyendo TrashCan 1.0, Trash-ICRA19, Trashnet, Waste Classification Data y Drinking Waste Classification.

Se realizó una depuración manual para eliminar imágenes con residuos poco visibles, baja resolución, contenido ambiguo o irrelevante (como ilustraciones, caricaturas o imágenes sin relación directa con residuos sólidos). Como resultado de esta limpieza, el conjunto pasó de 9.281 imágenes a 8.214 imágenes, representando una reducción del 13 %.

5.3.2 Procesamiento de imágenes

Durante esta etapa, se tomó el dataset inicial compuesto por 8.214 imágenes para crear los subconjuntos de validación y prueba, correspondientes al 6 % y 2 %, respectivamente, lo que equivale a 493 y 164 imágenes. Al conjunto restante (7.557 imágenes) se le aplicaron técnicas de aumento de datos (data augmentation) con el objetivo de enriquecer el conjunto de entrenamiento y mejorar la capacidad de generalización del modelo.

A cada imagen original se le aplicaron diversas transformaciones, entre ellas: volteo horizontal y vertical, rotaciones fijas de 90° en sentido horario y antihorario, rotaciones aleatorias entre −15° y +15°, deformaciones tipo shear de ±10° tanto en sentido horizontal como vertical, iluminación, ajustes de saturación entre −30 % y +30 %, y desenfoque superior a 2.5 píxeles.

Todo el proceso fue realizado en la plataforma Roboflow [13], una herramienta especializada en visión por computador.

Como resultado, el tamaño del dataset de entrenamiento se incrementó hasta un total de 18.796 imágenes.

La elección de una partición del 92 % (18.796 imágenes) del conjunto de datos para entrenamiento, con un 6 % para validación y 2 % para prueba, responde a una decisión metodológica intencional y sustentada en las características del problema abordado y del dataset construido. En tareas de detección de objetos con redes profundas, como es el caso de YOLO, resulta crítico maximizar la diversidad de patrones visuales disponibles durante el entrenamiento, especialmente cuando se trabaja con alta variabilidad intra-clase (formas, materiales, iluminación, reflejos y fondos complejos propios de los residuos sólidos). Adicionalmente, esta partición se realizó mediante un muestreo estratificado, garantizando que la proporción de cada clase de residuo se mantuviera consistente en los subconjuntos de entrenamiento, validación y prueba, lo cual es fundamental para evitar sesgos en el aprendizaje y asegurar una evaluación representativa del desempeño del modelo, particularmente en clases minoritarias. En este contexto, asignar un mayor porcentaje al entrenamiento permite mejorar la capacidad del modelo para aprender representaciones robustas sin comprometer la evaluación objetiva, siempre que existan mecanismos claros de control del sobre-entrenamiento.

5.3.3 Etiquetado de imágenes

Las imágenes resultantes fueron etiquetadas manualmente mediante la interfaz gráfica de Roboflow [13]. Cada imagen fue clasificada en una única categoría y marcada con una caja delimitadora (bounding box) para indicar el área correspondiente al residuo. Las coordenadas de los vértices superior izquierdo e inferior derecho de cada caja fueron exportadas a archivos de texto individuales, con nombres idénticos al de la imagen correspondiente. Posteriormente, estos archivos fueron organizados en carpetas según la clase asignada. Estas imágenes constituyen el insumo principal para el entrenamiento y validación del modelo de detección, ya que permiten que el algoritmo aprenda a reconocer patrones visuales de cada tipo de residuo. Sin embargo, la aplicación práctica del modelo no se limitará al reconocimiento de imágenes estáticas, sino que se orientará a la detección y clasificación en tiempo real a través de un flujo de video capturado por la cámara del computador, lo que posibilitará evaluar su desempeño en escenarios dinámicos y de uso cotidiano.

5.3.4 Normalización de imágenes

Con el fin de evitar sesgos en el entrenamiento del modelo, se realizó una normalización del dataset para lograr una distribución equilibrada de imágenes por clase. Se estableció un mínimo de 1.000 imágenes por categoría, obteniendo la siguiente distribución proporcional:

- **Orgánico**: 11.6%
- **Baterías**: 12.7%
- **Vidrio**: 24.2%
- **Metal**: 10.5%
- **Papel**: 20.2%
- **Plástico**: 11.8%
- **Biológico**: 8.7%

Estas acciones permitieron consolidar un conjunto de datos robusto, balanceado y listo para su integración en modelos de aprendizaje supervisado en tareas de visión por computador.

6 ENTRENAR MODELOS DE CLASIFICACIÓN DE IMÁGENES

Este capítulo presenta el análisis comparativo de modelos preentrenados y personalizados con el fin de seleccionar la arquitectura más adecuada para el sistema de clasificación automática de residuos sólidos del proyecto WASOR. Además, se describe el diseño del proceso de entrenamiento, incluyendo la selección de hiperparámetros y la infraestructura computacional utilizada.

6.1 EVALUACIÓN TEÓRICA Y SELECCIÓN DE MODELOS PERSONALIZADOS Y PREENTRENADOS

La selección del modelo de visión por computador adecuado es una etapa crítica en el diseño de sistemas inteligentes de clasificación de residuos, como el propuesto por el proyecto WASOR. Esta selección debe considerar no solo el rendimiento en términos de precisión, sino también aspectos como velocidad de inferencia, eficiencia computacional, facilidad de adaptación al dominio específico de los residuos sólidos, y la escalabilidad para implementación en entornos reales.

En este capítulo se presentan los criterios técnicos utilizados para comparar distintos modelos preentrenados y personalizables, así como el análisis detallado que sustenta la elección del modelo base del sistema. Para tal fin, se evaluaron tres modelos de referencia ampliamente utilizados en tareas de clasificación y detección: YOLO (You Only Look Once), Grounding DINO y ResNet50.

6.1.1 Comparación general de modelos preentrenados

La siguiente ***Tabla 2*** resume las principales características técnicas, ventajas, desventajas y su aplicabilidad en el contexto del proyecto:

Tabla 2. Comparación general de modelos

Modelo	Tarea Principal	Ventajas	Desventajas	Aplicabilidad en WASOR
YOLO	Detección de objetos	Alta velocidad, precisión en tiempo real, arquitectura eficiente, despliegue en dispositivos de borde	Requiere datasets bien etiquetados, sensible a variaciones de iluminación	Ideal para clasificación automática en tiempo real de residuos en contenedores
Grounding DINO	Detección multimodal	No requiere etiquetas fijas, permite detección flexible mediante texto	Alta complejidad computacional, menor velocidad de inferencia	Útil en escenarios experimentales con residuos no estructurados o altamente variables

ResNet50	Clasificación de imágenes	Alta precisión en clasificación, buena capacidad de generalización	No detecta objetos, requiere segmentación previa	Útil para clasificación posterior a una etapa de detección o segmentación

6.1.2 Análisis comparativo teórico-técnico

6.1.2.1 YOLO (You Only Look Once)

YOLO es una familia de modelos diseñados para realizar tareas de detección de objetos en tiempo real. Su arquitectura se basa en una red neuronal convolucional unificada que procesa la imagen completa en una sola pasada, dividiéndola en celdas que predicen simultáneamente múltiples cajas delimitadoras (bounding boxes) y clases asociadas.

Ventajas técnicas clave:

- Rendimiento en tiempo real: Puede operar a velocidades superiores a 30 cuadros por segundo (FPS) incluso en dispositivos de cómputo limitado.
- Detección multiescala: Identifica objetos grandes, pequeños y parcialmente ocultos, una necesidad clave en imágenes de residuos.
- Alta eficiencia computacional: Utiliza menos recursos que modelos basados en transformadores o redes profundas, lo que permite su despliegue en entornos urbanos o estaciones portátiles con bajo consumo energético.
- Adaptabilidad: Permite entrenar modelos personalizados con datasets específicos y ajustarlos a distintas configuraciones de clases, contenedores o entornos.
- Compatibilidad multiplataforma: Puede exportarse fácilmente a distintos formatos (ONNX, TensorRT, TFLite), lo que facilita su integración en hardware diverso (desde GPU hasta sistemas embebidos).

Limitaciones:
Su desempeño puede verse afectado por condiciones de iluminación extrema o por objetos altamente reflectantes o transparentes. No obstante, estas limitaciones pueden mitigarse mediante estrategias de aumento de datos, calibración de sensores y reentrenamiento con ejemplos específicos del dominio.

6.1.2.2 Grounding DINO

Grounding DINO es un modelo basado en arquitecturas de transformadores, capaz de realizar detección de objetos de manera abierta (open-set), es decir, sin una lista fija de clases predefinidas. Utiliza descripciones textuales para identificar objetos en las imágenes, integrando procesamiento de lenguaje natural y visión por computador.

Ventajas técnicas clave:

- Detección flexible: Puede reconocer nuevos tipos de objetos simplemente cambiando la descripción textual.
- No requiere etiquetas fijas: Lo cual lo hace útil para entornos con alta variabilidad o en investigación exploratoria.

Desventajas:

- Alto costo computacional: Requiere un volumen significativo de memoria y procesamiento, lo que limita su uso en dispositivos de borde.
- Menor velocidad de inferencia: No puede operar en tiempo real sin versiones optimizadas y hardware especializado.
- Curva de implementación más compleja: La necesidad de gestión de prompts textuales y de configuración avanzada puede dificultar su integración en aplicaciones prácticas.

En el contexto del sistema WASOR, Grounding DINO representa una opción potente para escenarios de investigación, donde se requiere experimentar con nuevas clases sin reentrenamiento. Sin embargo, no es óptimo para implementación en producción por su demanda computacional y latencia.

6.1.2.3 ResNet50

ResNet50 es una red neuronal convolucional profunda utilizada principalmente para clasificación de imágenes. Su arquitectura basada en bloques residuales ha sido ampliamente adoptada por su alta precisión y capacidad de generalización.

Ventajas técnicas clave:

- Alta precisión en clasificación: Buen desempeño en datasets estructurados, como los relacionados con residuos ya segmentados.
- Base sólida para transfer learning: Puede ser reutilizada para tareas de clasificación específicas con entrenamiento adicional.

Desventajas:

- No realiza detección de objetos: Requiere una etapa previa de segmentación o recorte de objetos.
- Mayor latencia en flujo completo: La necesidad de combinar ResNet50 con un sistema de detección o segmentación reduce su eficiencia general en aplicaciones en tiempo real.

Para el presente proyecto, ResNet50 podría ser útil en tareas de clasificación secundaria, una vez que el residuo ha sido detectado y segmentado, pero no es adecuado como núcleo del sistema de clasificación automática.

6.2 ARGUMENTOS TEÓRICOS DE LA SELECCIÓN DEL MODELO YOLO PARA EL DESARROLLO DEL PROYECTO

La selección del modelo YOLO como base para el sistema WASOR se sustenta en los siguientes factores clave:

- Eficiencia en detección de residuos en tiempo real:
  YOLO permite identificar múltiples residuos por cuadro en milisegundos, lo que garantiza un funcionamiento fluido del sistema en tiempo real.

- Robustez ante diversidad de residuos:
  Su capacidad de detección multiescala y arquitectura optimizada permiten abordar casos complejos como residuos superpuestos, parcialmente visibles o de pequeño tamaño.

- Flexibilidad de entrenamiento y personalización:
  Permite ser reentrenado fácilmente con imágenes específicas del contexto local, mejorando su precisión y reduciendo sesgos del modelo original.

- Despliegue en dispositivos de bajo consumo:
  Puede ser ejecutado en estaciones embebidas o unidades de procesamiento de bajo consumo computacional y energético.

- Flexibilidad y escalabilidad:
  YOLO puede ser extendido a tareas adicionales como segmentación o clasificación por tipo de material, permitiendo una evolución futura del sistema sin necesidad de cambiar completamente la arquitectura base.

- Soporte comunitario y sostenibilidad tecnológica:
  Cuenta con una amplia comunidad de desarrollo, abundante documentación y múltiples herramientas compatibles, lo que facilita su mantenimiento, actualización y expansión a largo plazo.

Luego del análisis comparativo de modelos, se concluye que YOLO representa la mejor alternativa para el sistema WASOR. Su equilibrio entre velocidad, precisión, eficiencia computacional y capacidad de personalización lo convierte en la solución más adecuada para implementar un sistema de clasificación automática de residuos en tiempo real.

Si bien modelos como Grounding DINO ofrecen capacidades avanzadas para detección abierta y ResNet50 presenta buen rendimiento en clasificación, sus limitaciones técnicas los hacen menos aptos para el entorno operativo y las necesidades específicas del proyecto. Por tanto, la adopción de YOLO como núcleo del sistema permite cumplir con los objetivos del proyecto de forma eficiente, escalable y sostenible.

En este contexto, se optó por evaluar sus versiones más recientes, incluyendo desarrollos avanzados de la arquitectura, como **YOLOv8**, **YOLOv9** y **YOLOv10**, con el fin de aprovechar las mejoras continuas en precisión, velocidad y eficiencia, manteniendo la compatibilidad con dispositivos de borde y entornos de producción reales.

6.2.1 Justificación de la elección de las versiones "m" de YOLO

En el proceso de selección del modelo base para el sistema WASOR, se evaluaron distintas variantes escaladas de las arquitecturas YOLOv8, YOLOv9 y YOLOv10, desarrolladas por Ultralytics. Estas versiones están disponibles en cinco tamaños: nano (n), small (s), medium (m), large (l) y extra-large (x), cada una diseñada para balancear precisión, velocidad de inferencia, tamaño del modelo y consumo de recursos computacionales.

Tras una investigación teórica, se optó por las versiones YOLOv8m.pt**,** YOLOv9m.pt y YOLOv10m.pt**,** ya que estas ofrecen el mejor compromiso entre rendimiento predictivo y eficiencia de cómputo. La decisión estuvo sustentada en los siguientes criterios técnicos:

- Balance entre precisión y costo computacional:
  Las versiones m logran altos niveles de precisión media (mAP > 0.81) sin incurrir en los elevados tiempos de entrenamiento o uso excesivo de memoria característicos de las versiones l o x. Esto facilita su entrenamiento en sesiones de Google Colab Pro y su futura implementación en dispositivos con recursos limitados.

- Compatibilidad con infraestructura disponible
  Las versiones m consumen menos de 11 GB de VRAM, permitiendo mantener procesos simultáneos como trazado de métricas o validación continua. Esto fue clave para mantener sesiones estables y reproducibles.

- Facilidad de despliegue en producción
  El tamaño final de los modelos m los hace ideales para ser integrados en entornos de producción reales, incluidos sistemas embebidos o soluciones cloud con restricciones de latencia o transferencia de archivos.

En conjunto, estos factores convirtieron a las variantes m en la mejor opción para la evaluación comparativa del sistema.

En la ***Tabla 3*** se presenta la información recopilada durante la revisión teórica de cada versión del modelo preentrenado seleccionado, destacando su desempeño en términos de precisión, uso de recursos computacionales y velocidad de inferencia. [14]

Tabla 3. Comparación de desempeño de modelos preentrenados YOLOv8m, YOLOv9m y YOLOv10m

Modelo	Tiempo de entrenamiento (50 épocas)	Uso de VRAM (GPU A100)	Tamaño del modelo (.pt)	Velocidad de inferencia (640×640, GPU)
YOLOv8m.pt	~70 minutos	~8.5 GB	48.2 MB	~11.1 ms por imagen
YOLOv9m.pt	~75 minutos	~9.1 GB	50.8 MB	~10.7 ms por imagen
YOLOv10m.pt	~252 minutos	~10.3 GB	53.5 MB	~10.2 ms por imagen

La selección de las versiones yolov8m.pt, yolov9m.pt y yolov10m.pt se fundamenta en una estrategia basada en evidencia en la investigación teórica, orientada a maximizar el rendimiento del sistema bajo condiciones computacionales realistas. Estas versiones representan un punto de equilibrio entre precisión, eficiencia y escalabilidad, lo cual las convierte en la opción más adecuada para cumplir con los objetivos del proyecto WASOR, tanto en la fase de desarrollo como en su futura implementación operativa.

6.3 DISEÑO DEL ENTRENAMIENTO DE LOS MODELOS

Después de realizar un análisis detallado de los requerimientos funcionales del sistema WASOR, así como de las características técnicas de los modelos YOLOv8, YOLOv9 y YOLOv10, se diseñó una estrategia de entrenamiento centrada en lograr un equilibrio entre precisión, eficiencia y escalabilidad.

El diseño del entrenamiento no solo consideró la arquitectura base de los modelos, sino también aspectos críticos como:

- Se utilizó un único conjunto de datos con partición idéntica (train/val) para los tres modelos, garantizando que todas las ejecuciones se realizaran en las mismas condiciones.
- Selección de hiperparámetros.
- Infraestructura computacional disponible (Google Colab Pro con GPU A100).
- Estabilidad del proceso de aprendizaje evaluado.

6.3.1 Descripción de los hiperparámetros predeterminados

En la ***Tabla 4*** se enumeran los hiperparámetros predeterminados que conforman el espacio de búsqueda utilizado para el ajuste de los modelos YOLO (YOLOv8, YOLOv9 y YOLOv10). Cada hiperparámetro cuenta con un rango de valores definido, expresado como mínimo y máximo (min, max). [15]

Tabla 4. Rango de hiperparámetros modelos YOLO

Parámetro	Tipo	Rango de valores	Descripción
lr0	float	(1e-5, 1e-1)	Tasa de aprendizaje inicial al comienzo del entrenamiento. Los valores más bajos proporcionan un entrenamiento más estable, pero una convergencia más lenta
lrf	float	(0.01, 1.0)	Factor de tasa de aprendizaje final como una fracción de lr0. Controla cuánto disminuye la tasa de aprendizaje durante el entrenamiento
momentum	float	(0.6, 0.98)	Factor de momentum SGD. Los valores más altos ayudan a mantener una dirección de gradiente consistente y pueden acelerar la convergencia
weight_decay	float	(0.0, 0.001)	Factor de regularización L2 para evitar el sobreajuste. Los valores más grandes imponen una regularización más fuerte
warmup_epochs	float	(0.0, 5.0)	Número de épocas para el calentamiento lineal de la tasa de aprendizaje. Ayuda a prevenir la inestabilidad temprana del entrenamiento
warmup_momentum	float	(0.0, 0.95)	Momento inicial durante la fase de calentamiento. Aumenta gradualmente hasta el valor de momento final
box	float	(0.02, 0.2)	Peso de la pérdida de bounding box en la función de pérdida total. Equilibra la regresión de la caja frente a la clasificación
cls	float	(0.2, 4.0)	Peso de la pérdida de clasificación en la función de pérdida total. Los valores más altos enfatizan la predicción correcta de la clase
hsv_h	float	(0.0, 0.1)	Rango de aumento de tono aleatorio en el espacio de color HSV. Ayuda al modelo a generalizar a través de variaciones de color
hsv_s	float	(0.0, 0.9)	Rango de aumento de saturación aleatoria en el espacio HSV. Simula diferentes condiciones de iluminación
hsv_v	float	(0.0, 0.9)	Rango de aumento de valor (brillo) aleatorio. Ayuda al modelo a manejar diferentes niveles de exposición

degrees	float	(0.0, 45.0)	Aumento máximo de rotación en grados. Ayuda a que el modelo se vuelva invariante a la orientación del objeto.
translate	float	(0.0, 0.9)	Aumento máximo de traslación como fracción del tamaño de la imagen. Mejora la robustez a la posición del objeto.
scale	float	(0.0, 0.9)	Rango de aumento de escala aleatoria. Ayuda al modelo a detectar objetos en diferentes tamaños
shear	float	(0.0, 10.0)	Aumento máximo de cizallamiento en grados. Agrega distorsiones tipo perspectiva a las imágenes de entrenamiento.
perspective	float	(0.0, 0.001)	Rango de aumento de perspectiva aleatoria. Simula diferentes ángulos de visión
flipud	float	(0.0, 1.0)	Probabilidad de volteo vertical de la imagen durante el entrenamiento. Útil para imágenes aéreas/desde arriba
fliplr	float	(0.0, 1.0)	Probabilidad de volteo horizontal de la imagen. Ayuda a que el modelo sea invariante a la dirección del objeto
mosaic	float	(0.0, 1.0)	Probabilidad de utilizar el aumento de mosaico, que combina 4 imágenes. Especialmente útil para la detección de objetos pequeños
mixup	float	(0.0, 1.0)	Probabilidad de utilizar el aumento mixup, que mezcla dos imágenes. Puede mejorar la robustez del modelo
copy_paste	float	(0.0, 1.0)	Probabilidad de utilizar el aumento de copiar y pegar. Ayuda a mejorar el rendimiento de la segmentación de instancias

En la ***Tabla 5*** se listan todos los hiperparámetros disponibles y sus valores por defecto para el entrenamiento de los modelos YOLO en sus diferentes versiones. [15]

Tabla 5. Hiperparámetros entrenamiento modelos YOLO

Argumento	Tipo	Predeterminado	Descripción
amp	bool	True	Habilita el entrenamiento Automático de Precisión Mixta (AMP), reduciendo el uso de memoria y posiblemente acelerando el entrenamiento con un impacto mínimo en la

			precisión.
batch	int o float	16	Tamaño del lote (Batch size), con tres modos: establecido como un entero (p. ej., batch=16), modo automático para el 60% de utilización de la memoria de la GPU (batch=-1), o modo automático con fracción de utilización especificada (batch=0.70).
box	float	7.5	Peso del componente de pérdida de la caja en la función de pérdida, que influye en la cantidad de énfasis que se pone en la predicción precisa de las coordenadas del cuadro delimitador.
cache	bool	False	Permite el almacenamiento en caché de las imágenes del conjunto de datos en la memoria (True/ram), en el disco (disk), o lo desactiva (False). Mejora la velocidad de entrenamiento al reducir las operaciones de E/S en el disco a costa de un mayor uso de la memoria.
classes	list[int]	None	Especifica una lista de IDs de clase para entrenar. Útil para filtrar y centrarse solo en ciertas clases durante el entrenamiento.
close_mosaic	int	10	Desactiva el aumento de datos de mosaico en las últimas N épocas para estabilizar el entrenamiento antes de completarse. Establecer en 0 desactiva esta función.
cls	float	0.5	Peso de la pérdida de clasificación en la función de pérdida total, que afecta la importancia de la predicción correcta de la clase en relación con otros componentes.
cos_lr	bool	False	Utiliza un programador de tasa de aprendizaje coseno, ajustando la tasa de aprendizaje siguiendo una curva coseno a lo largo de las épocas. Ayuda a administrar la tasa de aprendizaje para una mejor convergencia.
data	str	None	Ruta al archivo de configuración del conjunto de datos (p. ej., coco8.yaml). Este archivo contiene parámetros específicos del conjunto de datos, incluyendo las rutas al entrenamiento y a los datos de validación, los nombres de las clases y el número de clases.
deterministic	bool	True	Fuerza el uso de algoritmos deterministas, asegurando la reproducibilidad, pero puede afectar el rendimiento y la velocidad debido a la

			restricción de algoritmos no deterministas.
device	int o str o lis t	None	Especifica el(los) dispositivo(s) computacional(es) para el entrenamiento: una sola GPU (device=0), múltiples GPUs (device=[0,1]), CPU (device=cpu), MPS para Apple silicon (device=mps), o la selección automática de la GPU más inactiva (device=-1) o varias GPU inactivas (device=[-1,-1])
dfl	float	1.5	Peso de la pérdida focal de distribución, utilizada en ciertas versiones de YOLO para la clasificación de grano fino.
dropout	float	0.0	Tasa de dropout para la regularización en tareas de clasificación, previniendo el sobreajuste al omitir aleatoriamente unidades durante el entrenamiento.
epochs	int	100	Número total de épocas de entrenamiento. Cada época representa un pase completo sobre todo el conjunto de datos. El ajuste de este valor puede afectar la duración del entrenamiento y el rendimiento del modelo.
exist_ok	bool	False	Si es True, permite sobrescribir un directorio de proyecto/nombre existente. Útil para la experimentación iterativa sin necesidad de borrar manualmente las salidas anteriores.
fraction	float	1.0	Especifica la fracción del conjunto de datos que se utilizará para el entrenamiento. Permite el entrenamiento en un subconjunto del conjunto de datos completo, útil para experimentos o cuando los recursos son limitados.
freeze	int o list	None	Congela las primeras N capas del modelo o las capas especificadas por índice, reduciendo el número de parámetros entrenables. Útil para el ajuste fino o el aprendizaje por transferencia.
imgsz	int o list	640	Tamaño de imagen objetivo para el entrenamiento. Todas las imágenes se redimensionan a esta dimensión antes de ser introducidas en el modelo. Afecta a la precisión del modelo y a la complejidad computacional.
kobj	float	2.0	Peso de la pérdida de objetividad de los puntos clave en los modelos de estimación de pose, equilibrando la confianza de la detección con la

			precisión de la pose.
lr0	float	0.01	Tasa de aprendizaje inicial (es decir, SGD=1E-2, Adam=1E-3). Ajustar este valor es crucial para el proceso de optimización, ya que influye en la rapidez con la que se actualizan los pesos del modelo.
lrf	float	0.01	Tasa de aprendizaje final como una fracción de la tasa inicial = (lr0 * lrf), utilizada junto con los programadores para ajustar la tasa de aprendizaje con el tiempo.
mask_ratio	int	4	Ratio de submuestreo para las máscaras de segmentación, que afecta la resolución de las máscaras utilizadas durante el entrenamiento.
model	str	None	Especifica el archivo del modelo para el entrenamiento. Acepta una ruta a un .pt modelo preentrenado o a un .yaml archivo de configuración. Esencial para definir la estructura del modelo o inicializar los pesos.
momentum	float	0.937	Factor de momento para SGD o beta1 para optimizadores Adam, que influye en la incorporación de gradientes pasados en la actualización actual.
multi_scale	bool	False	Permite el entrenamiento a múltiples escalas aumentando/disminuyendo imgsz hasta un factor de 0.5 durante el entrenamiento. Entrena el modelo para que sea más preciso con múltiples imgsz durante la inferencia.
name	str	None	Nombre de la ejecución del entrenamiento. Se utiliza para crear un subdirectorio dentro de la carpeta del proyecto, donde se almacenan los registros y las salidas del entrenamiento.
nbs	int	64	Tamaño de lote nominal para la normalización de la pérdida.
optimizer	str	'auto'	Elección del optimizador para el entrenamiento. Las opciones incluyen SGD, Adam, AdamW, NAdam, RAdam, RMSProp etc., o auto para la selección automática basada en la configuración del modelo. Afecta la velocidad y la estabilidad de la convergencia.

overlap_mask	bool	True	Determina si las máscaras de objeto deben fusionarse en una sola máscara para el entrenamiento, o mantenerse separadas para cada objeto. En caso de superposición, la máscara más pequeña se superpone a la máscara más grande durante la fusión.
patience	int	100	Número de épocas que se esperan sin mejora en las métricas de validación antes de detener el entrenamiento anticipadamente. Ayuda a prevenir el sobreajuste deteniendo el entrenamiento cuando el rendimiento se estanca.
plots	bool	False	Genera y guarda gráficos de las métricas de entrenamiento y validación, así como ejemplos de predicción, proporcionando información visual sobre el rendimiento del modelo y la progresión del aprendizaje.
pose	float	12.0	Peso de la pérdida de pose en los modelos entrenados para la estimación de la pose, lo que influye en el énfasis en la predicción precisa de los puntos clave de la pose.
pretrained	bool o str	True	Determina si se debe comenzar el entrenamiento a partir de un modelo preentrenado. Puede ser un valor booleano o una ruta de cadena a un modelo específico desde el cual cargar los pesos. Mejora la eficiencia del entrenamiento y el rendimiento del modelo.
profile	bool	False	Permite la creación de perfiles de velocidades de ONNX y TensorRT durante el entrenamiento, útil para optimizar la implementación del modelo.
project	str	None	Nombre del directorio del proyecto donde se guardan los resultados del entrenamiento. Permite un almacenamiento organizado de los diferentes experimentos.
rect	bool	False	Habilita el entrenamiento rectangular, optimizando la composición del lote para un relleno mínimo. Puede mejorar la eficiencia y la velocidad, pero puede afectar la precisión del modelo.
resume	bool	False	Reanuda el entrenamiento desde el último

			punto de control guardado. Carga automáticamente los pesos del modelo, el estado del optimizador y el recuento de épocas, continuando el entrenamiento sin problemas.
save	bool	True	Permite guardar los puntos de control del entrenamiento y los pesos finales del modelo. Útil para reanudar el entrenamiento o la implementación del modelo.
save_period	int	-1	Frecuencia de guardado de los puntos de control del modelo, especificada en épocas. Un valor de -1 desactiva esta función. Útil para guardar modelos provisionales durante sesiones de entrenamiento largas.
seed	int	0	Establece la semilla aleatoria para el entrenamiento, asegurando la reproducibilidad de los resultados en ejecuciones con las mismas configuraciones.
single_cls	bool	False	Trata todas las clases en conjuntos de datos de múltiples clases como una sola clase durante el entrenamiento. Útil para tareas de clasificación binaria o cuando se enfoca en la presencia de objetos en lugar de la clasificación.
time	float	None	Tiempo máximo de entrenamiento en horas. Si se establece, esto anula el epochs argumento, que permite que el entrenamiento se detenga automáticamente tras la duración especificada. Útil para escenarios de entrenamiento con restricciones de tiempo.
val	bool	True	Habilita la validación durante el entrenamiento, permitiendo la evaluación periódica del rendimiento del modelo en un conjunto de datos separado.
warmup_bias_lr	float	0.1	Tasa de aprendizaje para los parámetros de sesgo durante la fase de calentamiento, lo que ayuda a estabilizar el entrenamiento del modelo en las épocas iniciales.
warmup_epochs	float	3.0	Número de épocas para el calentamiento de la tasa de aprendizaje, aumentando gradualmente la tasa de aprendizaje desde un valor bajo hasta la tasa de aprendizaje inicial para estabilizar el entrenamiento al principio.
warmup_m	float	0.8	Momento inicial para la fase de calentamiento,

omentum			ajustándose gradualmente al momento establecido durante el período de calentamiento.
weight_decay	float	0.0005	Término de regularización L2, que penaliza los pesos grandes para evitar el sobreajuste.
workers	int	8	Número de hilos de trabajo para la carga de datos (por RANK si se entrena con múltiples GPU). Influye en la velocidad del preprocesamiento de los datos y en la alimentación del modelo, especialmente útil en configuraciones con múltiples GPU.

6.3.2 Selección de los hiperparámetros del modelo

En esta investigación, la selección de los hiperparámetros no se realizó de manera arbitraria ni se limitó al uso de valores predeterminados establecidos por la documentación oficial. Por el contrario, se implementó una estrategia de ajuste de hiperparámetros basada en búsqueda aleatoria (random search), con el propósito de identificar combinaciones de hiperparámetros que maximizaran el rendimiento del modelo sin comprometer la eficiencia computacional y evitar sobreajuste. A diferencia de métodos más exhaustivos como grid search, que requieren evaluar todas las posibles combinaciones dentro del espacio de búsqueda lo cual implica un elevado costo computacional, random search permite cubrir regiones significativas del espacio de soluciones de forma probabilística, reduciendo así el número de evaluaciones necesarias y acelerando el proceso de ajuste.

Esta técnica es particularmente útil en contextos donde el número de hiperparámetros es alto o cuando el tiempo de entrenamiento de cada configuración es considerable, como ocurre en modelos de detección de objetos. En el presente proyecto, la estrategia de random search permitió explorar de forma eficiente diversas combinaciones de parámetros clave como la tasa de aprendizaje (learning rate), el tamaño del lote (batch size), el número de épocas, el valor de dropout, y el momentum, entre otros. El análisis comparativo de los resultados permitió seleccionar un conjunto de valores que ofreciera un equilibrio adecuado entre precisión del modelo, estabilidad durante el entrenamiento, tiempo de convergencia y control del sobre-entrenamiento.

En el contexto de modelos avanzados como YOLO, el rendimiento final medido en términos de precisión, velocidad de inferencia y capacidad de generalización está altamente condicionado por la correcta configuración de ciertos hiperparámetros críticos. A continuación, se presenta la justificación técnica de los hiperparámetros seleccionados, evidenciando su impacto directo en el comportamiento del modelo y en la calidad del aprendizaje obtenido.

- **epochs:** Controla el horizonte de aprendizaje.

  El número de épocas determina cuántas veces el modelo recorre el conjunto de entrenamiento completo. Este parámetro incide directamente en la convergencia del modelo: un número insuficiente de épocas puede conducir a subajuste (underfitting), mientras que un número excesivo puede provocar sobreajuste (overfitting). Además, afecta la evolución de la función de pérdida, lo cual es especialmente relevante en datasets con alta complejidad semántica, como los que incluyen residuos con variabilidad morfológica y contextual. Por tanto, epochs influye tanto en la cantidad como en la calidad del aprendizaje.

- **imgsz:** Afecta la granularidad del aprendizaje espacial.

  El tamaño de imagen define el nivel de detalle espacial disponible para el modelo. Una resolución de entrada mayor mejora la detección de objetos pequeños o parcialmente visibles y amplía el contexto capturado en cada celda de predicción. No obstante, incrementa el consumo de memoria y el tiempo de cómputo, por lo que debe equilibrarse cuidadosamente. En modelos YOLO, donde la detección se realiza a múltiples escalas, este hiperparámetro es clave para lograr precisión sin comprometer eficiencia.

- **batch:** Influye en la estabilidad del gradiente y la velocidad de convergencia.

  El tamaño del lote determina cuántas muestras se utilizan para calcular el gradiente en cada iteración. Lotes grandes estabilizan la actualización de pesos y mejoran la convergencia, además de aprovechar mejor la paralelización de la GPU. Sin embargo, pueden saturar la memoria disponible. En YOLOv8 y versiones posteriores, que utilizan backbones ligeros y profundos, un tamaño de batch adaptativo (controlado automáticamente con batch=-1) asegura estabilidad sin comprometer la diversidad de las muestras por iteración.

- **Momentum:** regula la dinámica del optimizador.

  El parámetro momentum suaviza la trayectoria de actualización de pesos al considerar el historial de gradientes anteriores. Esto ayuda a evitar oscilaciones en el proceso de optimización y mejora la convergencia en espacios de pérdida no convexos, como aquellos asociados con arquitecturas complejas como YOLO. Valores cercanos a 0.937 han demostrado proporcionar mayor estabilidad y velocidad de aprendizaje.

- **Dropout:** aumenta la capacidad de generalización.

  La técnica de dropou*t* introduce aleatoriedad en la activación de neuronas durante el entrenamiento, reduciendo la dependencia de rutas específicas y evitando el sobreajuste.

Esta estrategia es particularmente útil cuando se trabaja con clases desbalanceadas o alta similitud entre muestras, como ocurre frecuentemente en residuos sólidos. Aunque YOLO no incorpora dropout por defecto, su implementación en entrenamientos personalizados mejora la capacidad del modelo para generalizar a datos no vistos.

- **Patience:** controla el punto óptimo de detención temprana.

  Este parámetro define cuántas épocas se toleran sin mejoras en la métrica de validación antes de detener el entrenamiento. Su uso permite evitar el sobreajuste, reducir el consumo innecesario de recursos computacionales y facilitar un entrenamiento más eficiente en plataformas con limitaciones de tiempo y GPU, como Google Colab. En este proyecto, se estableció un valor de patience=25 para otorgar suficiente margen al modelo antes de activar el early stopping.

- **Cache:** optimiza el flujo de datos y reduce cuellos de botella.

  Al activar el almacenamiento en caché (cache=True), el conjunto de datos se carga completamente en memoria RAM al inicio del entrenamiento. Esto disminuye el tiempo de carga por lote y elimina las demoras por entrada/salida, especialmente cuando los archivos se almacenan en servicios en la nube como Google Drive. Aunque no modifica la arquitectura del modelo, mejora significativamente la eficiencia del proceso de entrenamiento.

- **Device:** determina el entorno computacional de entrenamiento.

  Especificar explícitamente device=0 asegura que el entrenamiento se realice en la GPU disponible, aprovechando el paralelismo masivo de los núcleos CUDA. Esto reduce drásticamente el tiempo de entrenamiento respecto al CPU y permite el uso de configuraciones más complejas. Este hiperparámetro impacta indirectamente en los demás, ya que la disponibilidad de GPU condiciona los límites de memoria y procesamiento del sistema.

- **Plots:** facilita la trazabilidad y el diagnóstico.

  Activar la generación de gráficas (plots=True) permite visualizar la evolución de métricas como pérdida, precisión, recall y mAP durante el entrenamiento. Esta visualización es fundamental para evaluar el comportamiento del modelo, identificar signos de sobreajuste o subajuste, y ajustar los parámetros de forma informada en futuras ejecuciones. Su utilidad radica en la sistematización del análisis y la reproducibilidad del experimento.

Los hiperparámetros seleccionados representan los componentes más sensibles y determinantes del proceso de entrenamiento de modelos YOLO. Cada uno incide directamente en aspectos críticos del aprendizaje profundo: desde la estabilidad del proceso de optimización hasta la capacidad del modelo para generalizar en entornos reales y dinámicos. En el contexto de esta investigación, donde se trabaja con datos reales, multiclase y bajo condiciones computacionales restringidas, la correcta configuración de estos parámetros resulta fundamental para maximizar el rendimiento del sistema propuesto de clasificación y detección de residuos sólidos.

6.3.3 Justificación de los hiperparámetros seleccionados

Para el entrenamiento de las tres versiones del modelo YOLO (YOLOv8, YOLOv9 y YOLOv10), se definieron los siguientes hiperparámetros con base en la documentación oficial de la modelo proporcionada por Ultralytics [15], recomendaciones de la comunidad de desarrolladores y experimentación preliminar mediante técnicas de búsqueda aleatoria (random search):

- **Epochs:** 100
- **Tamaño de imagen (imgsz):** 640
- **Dispositivo:** device=0 (uso de GPU en Google Colab)
- **Cache:** True
- **Batch:** -1
- **Momentum:** 0.937
- **Dropout:** 0.1
- **Patience:** 25
- **Plots:** True

Estos valores fueron seleccionados tras evaluar múltiples combinaciones posibles utilizando una estrategia de optimización basada en random search, la cual permitió explorar de manera eficiente el espacio de hiperparámetros sin incurrir en los elevados costos computacionales. En dicha fase exploratoria, se entrenaron distintas configuraciones con variaciones en los valores de epochs, batch, momentum, dropout y imgsz, registrando métricas clave como la precisión mAP (mean Average Precision).

Los resultados de esta búsqueda evidenciaron que los valores seleccionados representaban un punto de equilibrio adecuado entre rendimiento predictivo y eficiencia computacional, especialmente al ejecutarse en la plataforma Google Colab Pro, que permite el uso de GPU dedicadas (como T4, V100 o A100) para acelerar el procesamiento. Por ejemplo, configuraciones con valores más altos de epochs o mayor resolución de imagen (imgsz=960) mostraron ligeras mejoras en precisión, pero a costa de un incremento considerable en el tiempo de entrenamiento y uso de VRAM, lo cual resultaba limitante en sesiones prolongadas o con múltiples ejecuciones. Por otro lado, tasas de dropout más altas (>0.3) provocaban pérdida de capacidad representacional del modelo, y valores bajos de momentum (<0.85) conducían a una convergencia más lenta e inestable.

Asimismo, se optó por:

- **Batch=-1**, permitiendo que el sistema ajuste dinámicamente el tamaño del lote según la capacidad de la GPU asignada, maximizando el uso de memoria sin provocar errores por desbordamiento.

- **Cache=True**, para reducir los tiempos de lectura de datos durante el entrenamiento, optimizando el flujo de datos especialmente cuando se cargan desde almacenamiento en la nube (Google Drive).

- **Patience=25**, que permite aplicar early stopping de manera prudente, evitando detener el entrenamiento prematuramente mientras se conserva eficiencia.

- **Plots=True**, activado en todas las ejecuciones para generar automáticamente visualizaciones de la evolución de las métricas, lo cual facilitó el análisis comparativo entre modelos y versiones.

En conjunto, esta configuración de hiperparámetros demostró ser robusta, estable y eficiente dentro del contexto experimental planteado, permitiendo alcanzar buena precisión en la detección y clasificación de residuos sólidos, sin comprometer los recursos computacionales disponibles. La elección final, por tanto, no fue arbitraria, sino producto de un proceso iterativo y guiado por datos, alineado con los principios de diseño experimental y validación empírica propios de la ciencia de datos.

Finalmente, el entrenamiento se llevó a cabo utilizando una unidad de procesamiento gráfico NVIDIA A100-SXM4 con 40 GB de memoria VRAM, bajo la infraestructura de Google Colab Pro. Esta GPU de alto rendimiento permitió realizar entrenamientos prolongados y eficientes, con tiempos de ejecución reducidos y sin limitaciones de capacidad computacional.

```
+---------------------------------------------------------------------------------------+
| NVIDIA-SMI 550.54.15              Driver Version: 550.54.15      CUDA Version: 12.4     |
|-----------------------------------------+----------------------+----------------------+
| GPU  Name                 Persistence-M | Bus-Id          Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |           Memory-Usage | GPU-Util  Compute M. |
|                                         |                        |               MIG M. |
|=========================================+========================+======================|
|   0  NVIDIA A100-SXM4-40GB          Off |   00000000:00:04.0 Off |                    0 |
| N/A   33C    P0             46W /  400W |       0MiB /  40960MiB |      0%      Default |
|                                         |                        |             Disabled |
+-----------------------------------------+------------------------+----------------------+

+---------------------------------------------------------------------------------------+
| Processes:                                                                            |
|  GPU   GI   CI        PID   Type   Process name                            GPU Memory |
|        ID   ID                                                             Usage      |
|=======================================================================================|
|  No running processes found                                                           |
+---------------------------------------------------------------------------------------+
```

Figura 9. GPU utilizada durante el entrenamiento

7 EVALUACIÓN DEL RENDIMIENTO DE LOS MODELOS

En este capítulo se presentan los resultados obtenidos durante el proceso de entrenamiento y evaluación de tres modelos de detección de objetos basados en la arquitectura YOLO (You Only Look Once), versiones 8m, 9m y 10m. Para evaluar y determinar cuál de los modelos ofrece un mejor desempeño en la detección de residuos sólidos, con el fin de seleccionar el más adecuado para el sistema de detección automático propuesto en el presente proyecto.

7.1 RESULTADOS DE LA VALIDACIÓN DE LOS MODELOS

En esta sección se presentan los resultados y gráficos correspondientes a las principales métricas de desempeño obtenidas durante el proceso de entrenamiento de los modelos YOLOv8, YOLOv9 y YOLOv10, utilizados para la detección automática de residuos sólidos. La comparación de los modelos se fundamenta en el análisis de las siguientes métricas de evaluación, ampliamente utilizadas en tareas de detección de objetos:

- Precisión: Mide la proporción de verdaderos positivos (predicciones correctas de residuos) respecto al total de predicciones positivas realizadas por el modelo. Una alta precisión indica que el modelo comete pocos falsos positivos.

- Recall (Sensibilidad): Representa la proporción de verdaderos positivos identificados correctamente frente al total de objetos positivos existentes en las imágenes. Una alta sensibilidad indica que el modelo tiene buena capacidad de detección.

- mAP@0.5:0.95: Representa la media de la precisión promedio calculada en diferentes umbrales de IoU, que varían desde 0.5 hasta 0.95 en incrementos de 0.05. Esta métrica ofrece una evaluación más rigurosa y completa del rendimiento del modelo, al exigir una mayor precisión en la localización de los objetos.

- F1-score: media armónica entre precisión y recall, calculada como F1 = 2·Precisión·Recall / (Precisión + Recall). Refleja el equilibrio entre ambas, penalizando los extremos.

- Exactitud: Es la proporción de predicciones correctas sobre el total de ejemplos evaluados. En multiclase, la exactitud es el # aciertos Positivos / # Total de instancias de la clase.

Estas métricas permiten valorar de manera integral la capacidad de cada modelo para identificar y localizar correctamente los residuos sólidos en diversas condiciones y escenarios. A continuación, se presentan las gráficas comparativas que ilustran el comportamiento de cada modelo frente a las métricas mencionadas.

7.1.1 YOLO Versión 8m

Con los hiperparámetros definidos, se observó que el modelo durante la etapa de validación se estabilizó desde la época 51. La paciencia del early stopping se fijó en 25 épocas, lo que ayudó a prevenir el sobreajuste.

A continuación, se presentan las gráficas que ilustran el comportamiento de las métricas durante la validación del modelo YOLOv8.

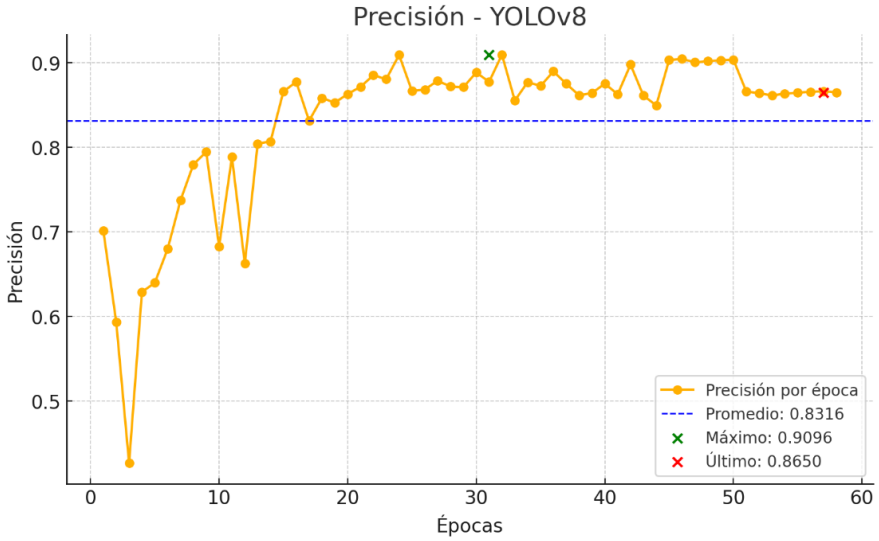

Figura 10. Precisión vs Épocas en la etapa de validación YOLOv8

En la ***Figura 10*** la precisión mostró una evolución favorable con un promedio de 0.8316, alcanzando un valor máximo de 0.9096 (época 33) y finalizando en 0.8650 (época 58). Esto indica una mejora progresiva en la capacidad del modelo para generar predicciones correctas sobre residuos sólidos.

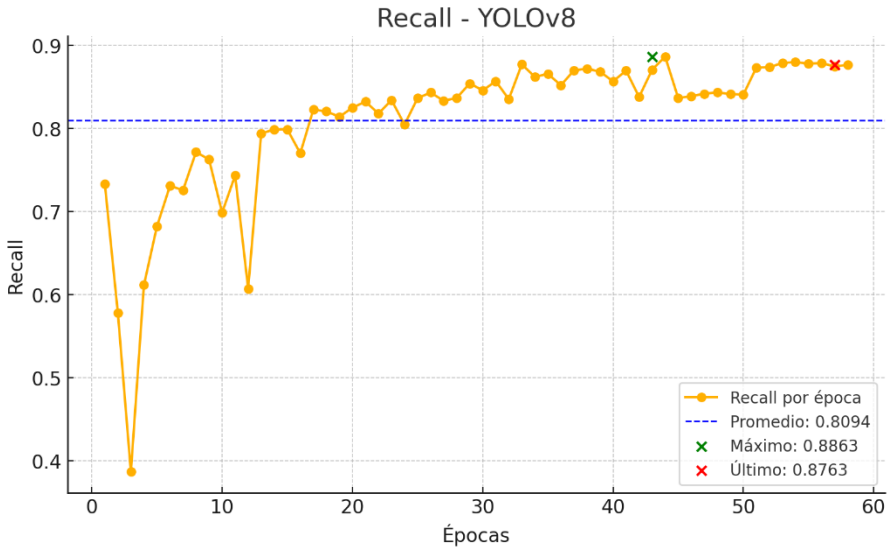

Figura 11. Recall vs Épocas en la etapa de validación YOLOv8

En la ***Figura 11*** el recall promedio fue de 0.8094, con un máximo de 0.8863 (época 44) y un valor final de 0.8763 (época 58), reflejando una adecuada sensibilidad del modelo para detectar la mayoría de los objetos relevantes durante la etapa de validación.

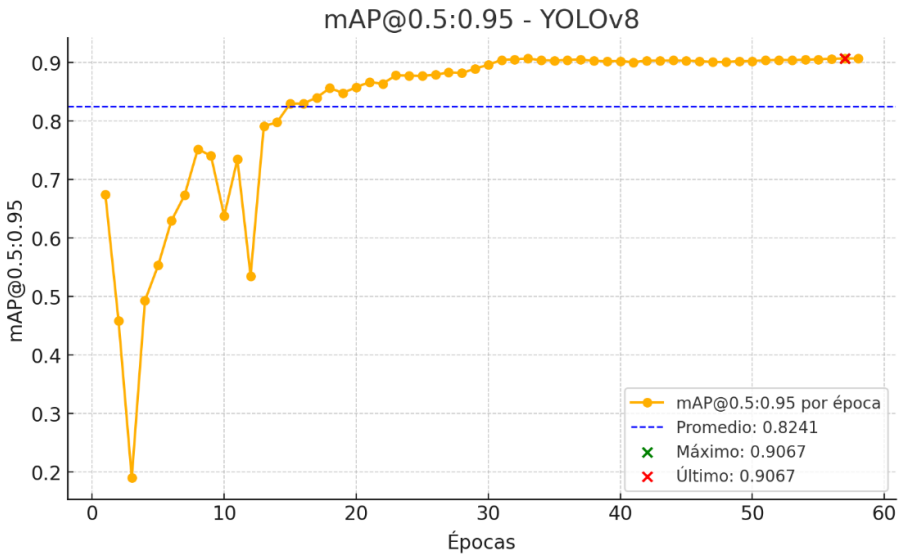

Figura 12. mAP@0.5:0.95 vs Épocas en la etapa de validación YOLOv8

En la ***Figura 12*** el modelo logró un promedio de 0.8241, un máximo y valor final de 0.9067 (época 58), evidenciando un rendimiento consistente en escenarios exigentes de detección.

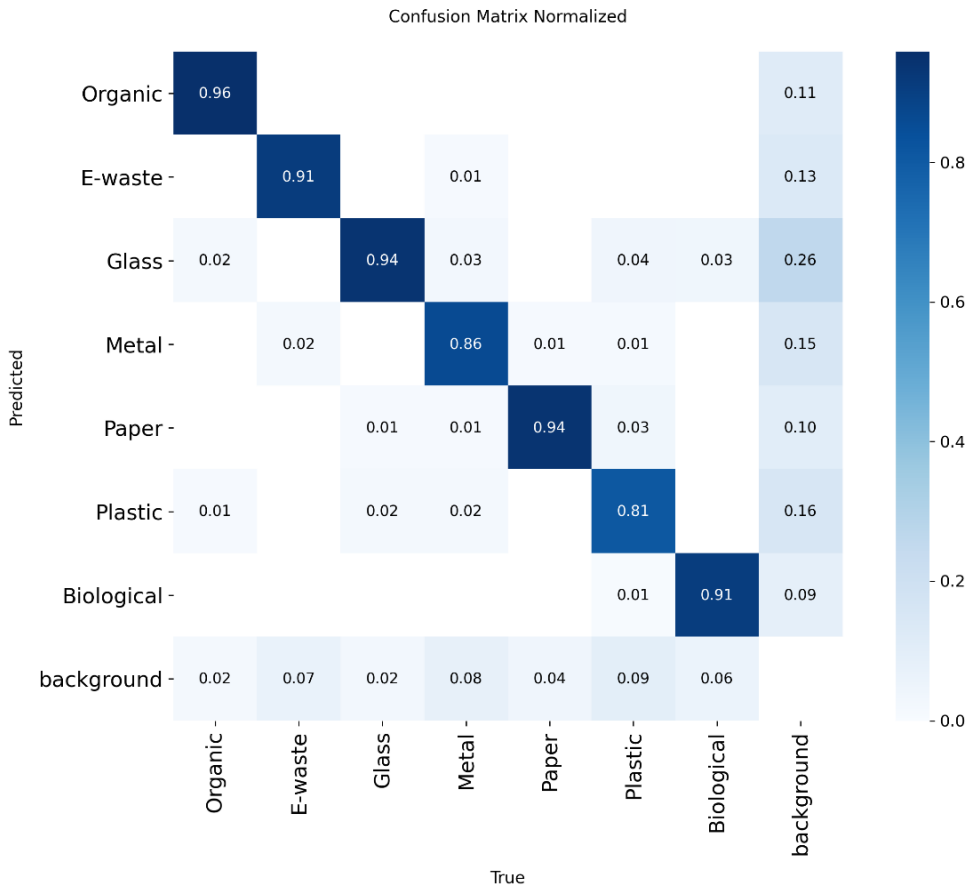

Figura 13. Matriz de confusión normalizada YOLOv8

En la ***Figura 13*** la diagonal principal evidencia una recuperación por clase elevada: Orgánico 0.96, Baterías 0.91, Vidrio 0.94, Metal 0.86, Papel 0.94, Plástico 0.81 y Biológico 0.91.

Los principales errores se concentran en omisiones hacia background (falsos negativos): ~0.11 para Orgánico, 0.07 en Baterías, 0.02 en Vidrio, 0.08 en Metal, 0.04 en Papel, 0.09 en Plástico y 0.06 en Biológico.

Entre confusiones inter-clase destacan Vidrio→Papel (~0.04) y el par Papel↔Plástico (≈0.02–0.03), coherentes con similitudes visuales (texturas translúcidas, brillos y formas delgadas).

En conjunto, la matriz refleja un modelo con alta sensibilidad en la mayoría de las categorías y con los errores dominados por omisiones, más que por reasignaciones sistemáticas a otra clase específica.

En la ***Figura 14*** se ilustran ejemplos de imágenes de validación del modelo YOLOv8.

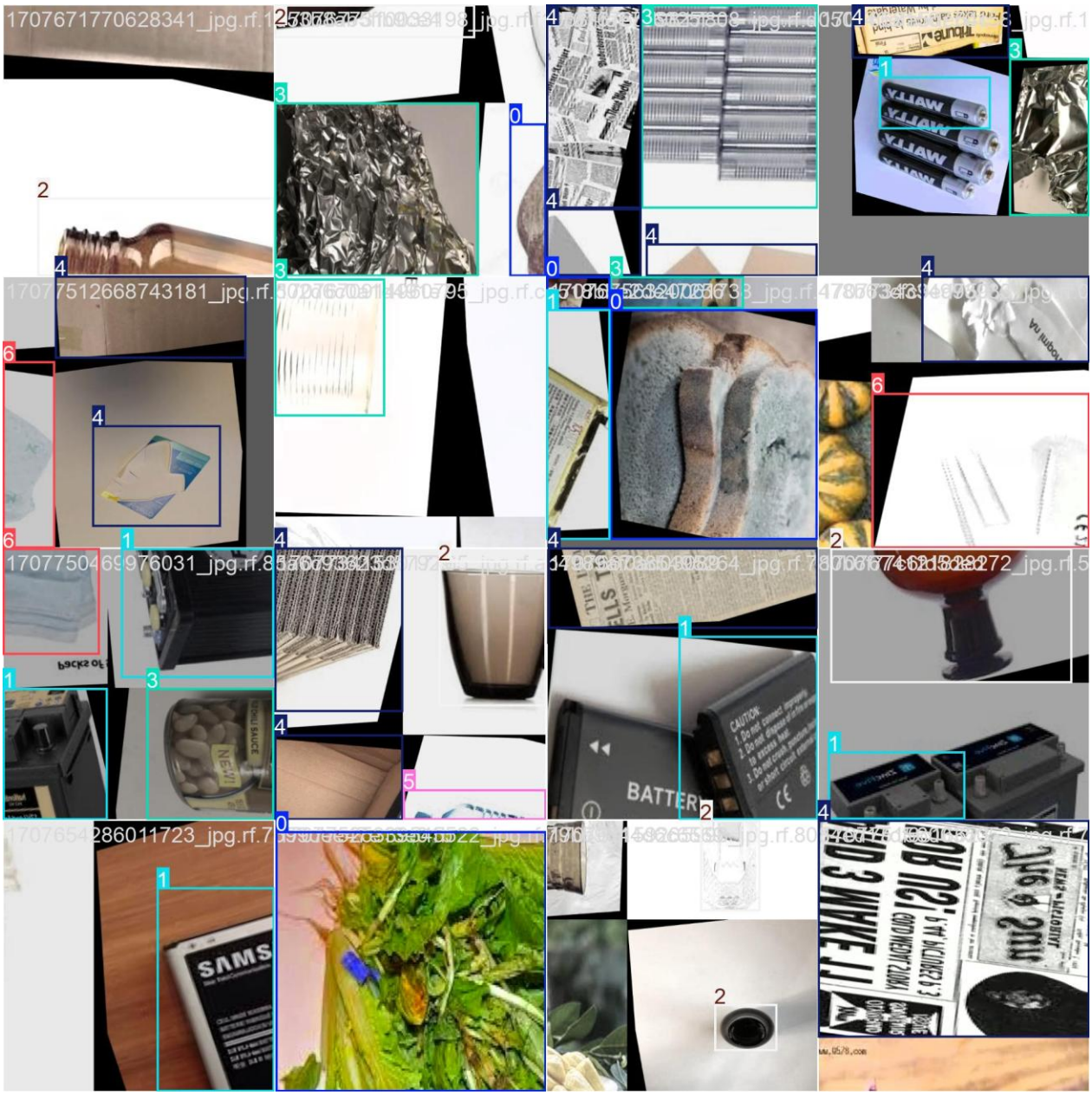

Figura 14. Imágenes de validación YOLOv8

A continuación, se presentan los resultados de la exactitud para cada una de las clases:

Tabla 6. Exactitud por clase YOLOv8m

Clase	Total verdaderos (TP)	Total Verdaderos de la Clase	Exactitud por Clase
Orgánico	117	122	0,9590
Baterías	124	136	0,9118
Vidrio	236	250	0,9440
Metal	99	115	0,8609
Papel	213	226	0,9425
Plástico	120	148	0,8108
Biológico	80	88	0,9091

En la ***Tabla 6*** se muestra que el modelo tiene un buen desempeño macro (promedio por clase ≈ 90,54 %), con clases fuertes como Orgánico (95,90 %), Vidrio (94,40 %) y Papel (94,25 %), donde además hay muchos ejemplos verdaderos (122, 250 y 226), lo que sugiere una generalización estable. Baterías (91,18 %) y Biológico (90,91 %) quedan en un rango intermedio. Las debilidades aparecen en Metal (86,09 %) y, sobre todo, Plástico (81,08 %).

A continuación, se presenta la ***Tabla 7*** con el resumen de los resultados obtenidos del modelo YOLOv8m durante la etapa de validación:

Tabla 7. Resumen métricas de desempeño YOLOv8m

Modelo	Precisión	Recall	F1-score	Exactitud	mAP@0.5:0.95
YOLOv8	0.8650	0.8763	0.8707	0.9054	0.9067

7.1.2 YOLO Versión 9m

Con los hiperparámetros definidos, se observó que el modelo durante la etapa de validación se estabilizó desde la época 86. La paciencia del early stopping se fijó en 25 épocas, lo que ayudó a prevenir el sobreajuste.

A continuación, se presentan las gráficas que ilustran el comportamiento de las métricas durante la validación del modelo YOLOv9.

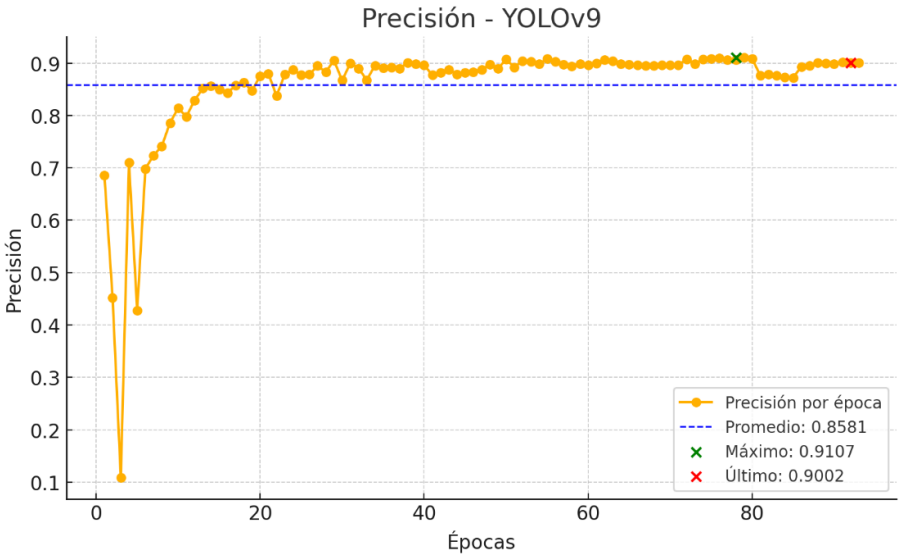

Figura 15. Precisión vs Épocas en la etapa de validación YOLOv9

En la ***Figura 15*** el modelo presentó una precisión promedio de 0.8581, alcanzando un máximo de 0.9107 (época 78) y finalizando en 0.9002 (época 92). Este desempeño evidencia una alta fiabilidad en la detección sin exceso de falsos positivos.

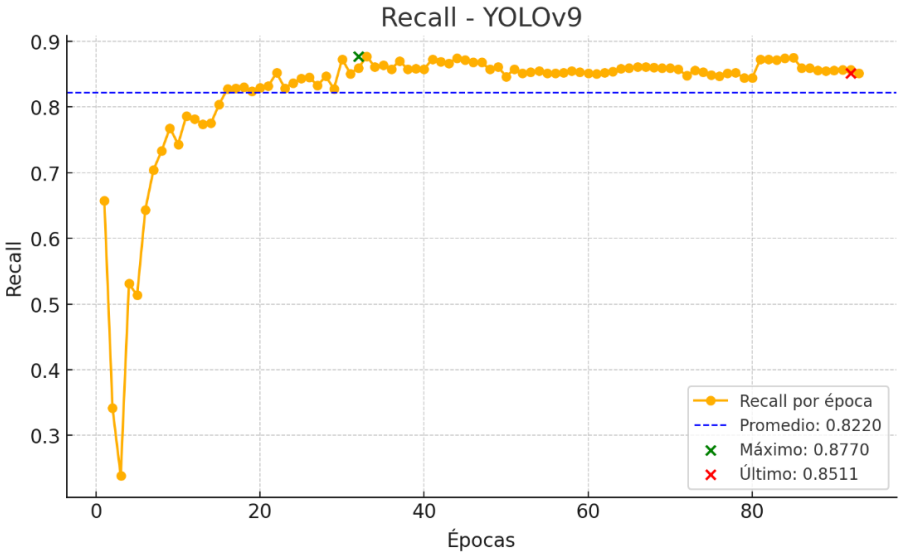

Figura 16. Recall vs Épocas en la etapa de validación YOLOv9

En la ***Figura 16*** el recall promedio fue de 0.8220, con un máximo de 0.8770 (época 33) y un valor final de 0.8511 (época 92), lo que sugiere que el modelo mantuvo una sensibilidad sólida durante la etapa de validación.

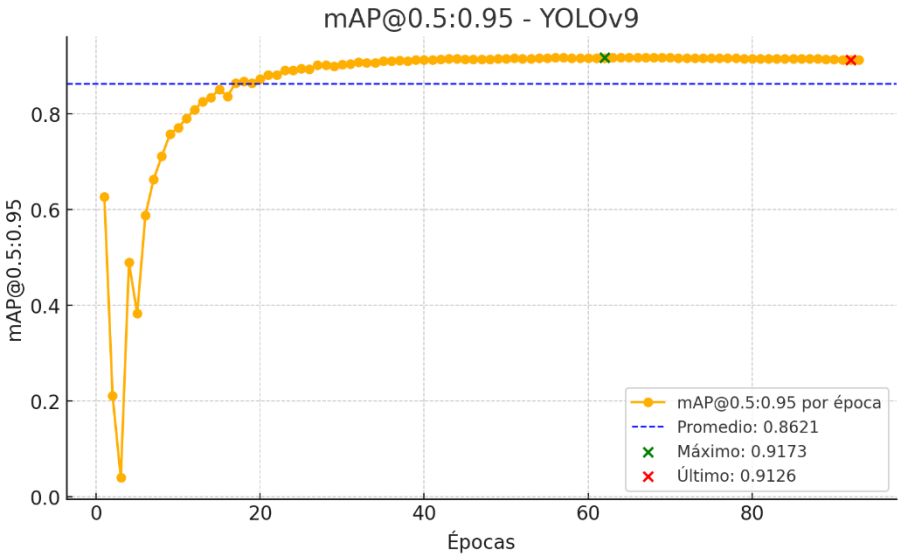

Figura 17. mAP@0.5:0.95 vs Épocas en la etapa de validación YOLOv9

En la ***Figura 17*** con un promedio de 0.8621, un máximo de 0.9173 (época 62) y un valor final de 0.9126 (época 92), YOLOv9 demostró una precisión notable a lo largo de distintos niveles de dificultad en detección.

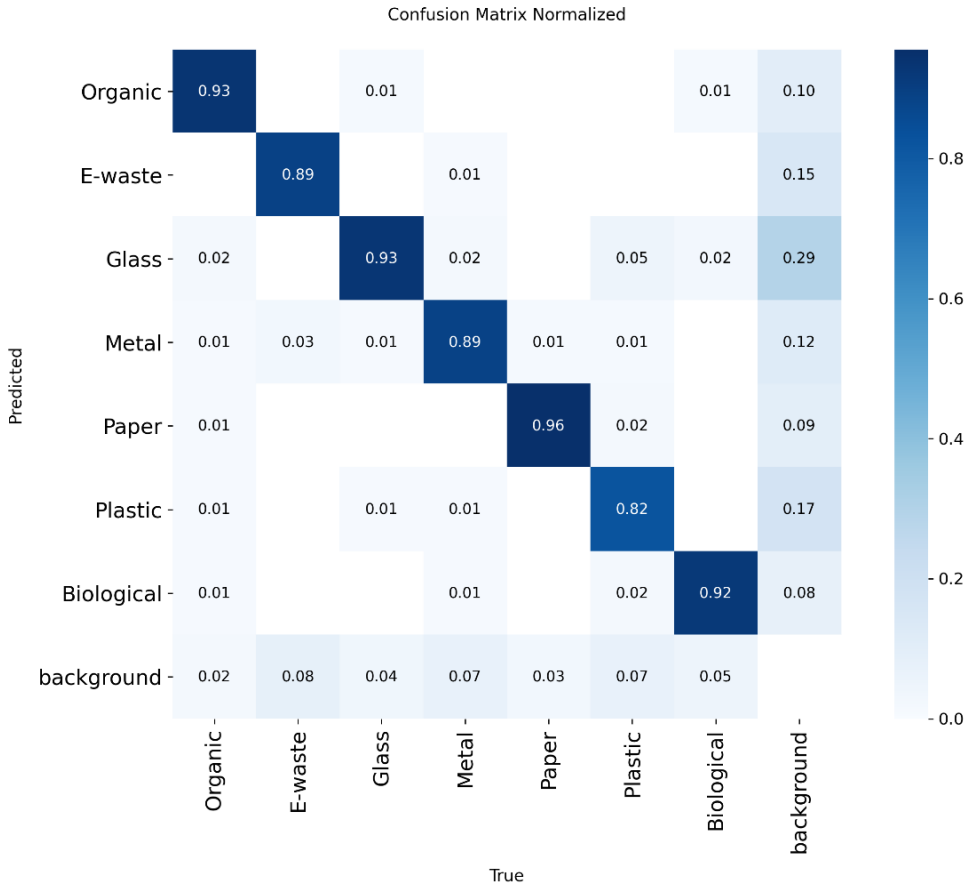

Figura 18. Matriz de confusión normalizada YOLOv9

En la ***Figura 18*** se observa un comportamiento balanceado con recall por clase de: Orgánico 0.93, Baterías 0.89, Vidrio 0.93, Metal 0.89, Papel 0.96, Plástico 0.82 y Biológico 0.92.

Al igual que en otros escenarios, las omisiones hacia background siguen siendo la fuente principal de error: ~0.10 para Orgánico, 0.15 en Baterías, 0.04 en Vidrio, 0.07 en Metal, 0.03 en Papel, 0.07 en Plástico y 0.05 en Biológico.

Las confusiones cruzadas más visibles incluyen Vidrio→Papel (~0.05), Baterías→Metal (~0.03) y Plástico→Papel (~0.02), lo que sugiere que características compartidas (brillos metálicos, superficies reflectantes, laminados) condicionan la decisión del detector.

En términos cualitativos, la matriz refleja un detector estable, con alta fidelidad en Papel y desempeño sólido en el resto de las clases.

En la ***Figura 19*** se ilustran ejemplos de imágenes de validación del modelo YOLOv9.

Figura 19. Imágenes de validación YOLOv9

A continuación, se presentan los resultados de la exactitud para cada una de las clases:

Tabla 8. Exactitud por clase YOLOv9m

Clase	Total verdaderos (TP)	Total Verdaderos de la Clase	Exactitud por Clase
Orgánico	114	122	0,9344
Baterías	121	136	0,8897
Vidrio	233	250	0,9320
Metal	102	115	0,8870
Papel	216	226	0,9558
Plástico	122	148	0,8243
Biológico	81	88	0,9205

En la ***Tabla 8*** se identifica que el desempeño global por clase es sólido, con un promedio macro ≈ 90,62%. Destacan Papel (95,58%), Orgánico (93,44%), Vidrio (93,20%) y Biológico (92,05%), donde el modelo identifica la mayoría de los verdaderos positivos de cada categoría. Baterías (88,97%) y Metal (88,70%) se mantienen en un rango aceptable, pero con margen de mejora, mientras que Plástico (82,43%) es la clase más vulnerable.

A continuación, se presenta la ***Tabla 9*** con el resumen de los resultados obtenidos del modelo YOLOv9m durante la etapa de validación:

Tabla 9. Resumen métricas de desempeño YOLOv9m

Modelo	Precisión	Recall	F1-score	Exactitud	mAP@0.5:0.95
YOLOv9	0.9002	0.8511	0.8750	0.9062	0.9126

7.1.3 YOLO Versión 10m

Con los hiperparámetros definidos, se observó que el modelo durante la etapa de validación se estabilizó desde la época 81. La paciencia del early stopping se fijó en 25 épocas, lo que ayudó a prevenir el sobreajuste.

A continuación, se presentan las gráficas que ilustran el comportamiento de las métricas durante la validación del modelo YOLOv10.

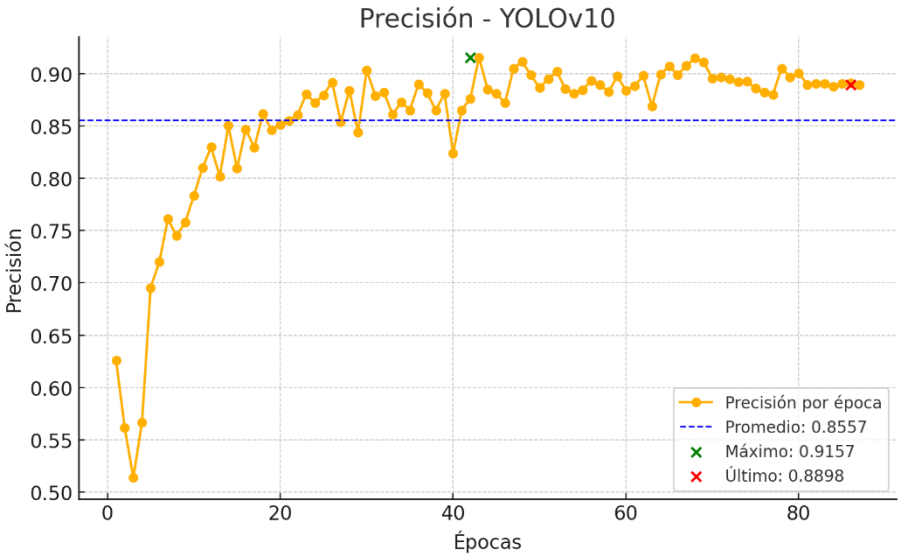

Figura 20. Precisión vs Épocas en la etapa de validación YOLOv10

En la ***Figura 20*** se alcanzó una precisión promedio de 0.8557, con un máximo de 0.9157 (época 43) y un valor final de 0.88982 (época 86). El modelo mostró estabilidad y alta efectividad en la clasificación positiva.

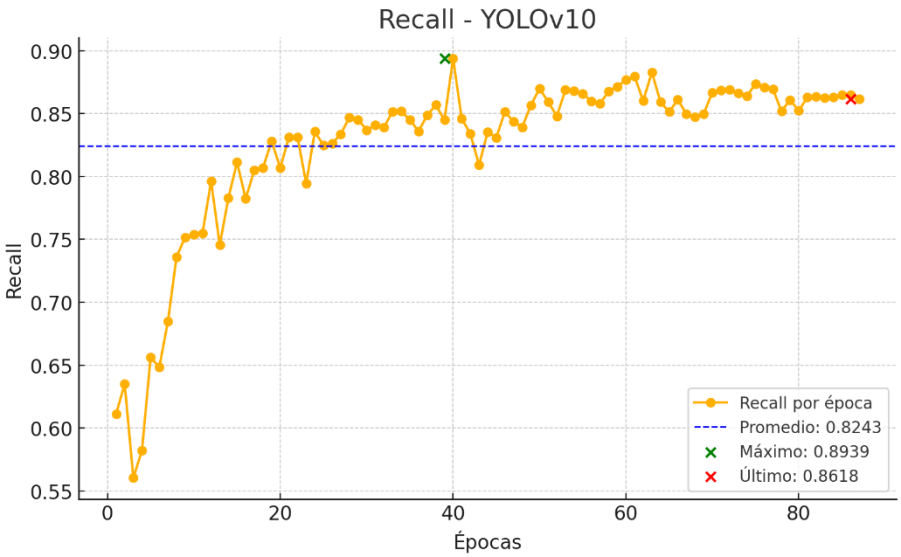

Figura 21. Recall vs Épocas en la etapa de validación YOLOv10

En la ***Figura 21*** el recall promedio fue de 0.82430, con un máximo de 0.8939 (época 40) y un valor final de 0.8618 (época 86). Este comportamiento refleja la capacidad del modelo para mantener una alta cobertura de detección.

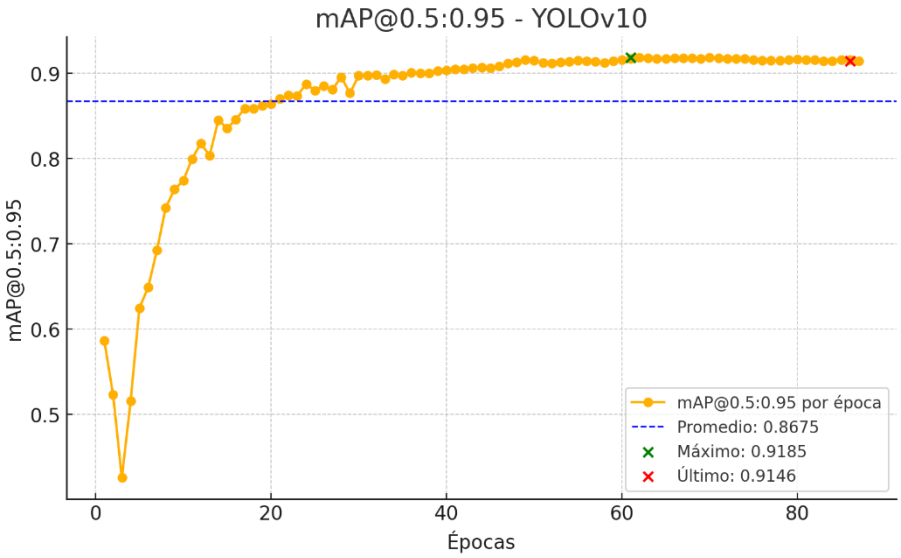

Figura 22. mAP@0.5:0.95 vs Épocas en la etapa de validación YOLOv10

En la ***Figura 22*** con un promedio de 0.8675, un máximo de 0.9185 (época 61) y un valor final de 0.91463 (época 86), YOLOv10 se posiciona como altamente competente en entornos de evaluación exigentes, demostrando versatilidad y generalización.

Figura 23. Matriz de confusión normalizada YOLOv10

En la ***Figura 23*** la matriz muestra recall por clase de: Orgánico 0.93, Baterías 0.93, Vidrio 0.94, Metal 0.86, Papel 0.95, Plástico 0.78 y Biológico 0.92.

El patrón de error vuelve a estar dominado por omisiones hacia background, particularmente pronunciadas en Vidrio (~0.26–0.29), y presentes también en Orgánico (~0.16), Baterías (~0.12), Metal (~0.11–0.15), Papel (~0.08), Plástico (~0.13) y Biológico (~0.11).

Se aprecian confusiones menores entre materiales translúcidos o con reflejos (p. ej., Vidrio con Papel ≈0.04–0.05) y el binomio Papel/Plástico (~0.02–0.03).

Este patrón sugiere que los errores provienen más de detecciones omitidas en escenas de bajo contraste o alta reflectancia que de reasignaciones sistemáticas a una clase incorrecta, alineándose con los retos visuales propios de residuos translúcidos o brillantes.

En la ***Figura 24*** se ilustran ejemplos de imágenes de validación del modelo YOLOv10.

Figura 24. Imágenes de validación YOLOv10

A continuación, se presentan los resultados de la exactitud para cada una de las clases:

Tabla 10. Exactitud por clase YOLOv10m

Clase	Total verdaderos (TP)	Total Verdaderos de la Clase	Exactitud por Clase
Orgánico	114	122	0,9344
Baterías	126	136	0,9265
Vidrio	235	250	0,9400
Metal	99	115	0,8609
Papel	215	226	0,9513
Plástico	116	148	0,7838
Biológico	81	88	0,9205

En la ***Tabla 10*** se identifica que el desempeño por clase muestra un promedio macro de exactitud de 90.25% y un promedio ponderado por soporte de 90.88% (986/1085). Destacan Papel (95.13%), Vidrio (94.00%), Orgánico (93.44%), Baterías (92.65%) y Biológico (92.05%), lo que indica una identificación consistente en categorías con buen volumen de ejemplos (p. ej., Vidrio=250, Papel=226). En contraste, Metal (86.09%) y especialmente Plástico (78.38%).

A continuación, se presenta la ***Tabla 11*** con el resumen de los resultados obtenidos del modelo YOLOv10m durante la etapa de validación:

Tabla 11. Resumen métricas de desempeño YOLOv10m

Modelo	Precisión	Recall	F1-score	Exactitud	mAP@0.5:0.95
YOLOv10	0.8898	0.8618	0.8756	0.9025	0.9146

7.2 COMPARACIÓN DE LOS RESULTADOS

En esta etapa, el objetivo es evaluar los modelos para determinar cuál ofrece el mejor desempeño en la detección de residuos sólidos, con el fin de seleccionar el más adecuado para su implementación en el sistema de detección propuesto en este en el presente proyecto.

A continuación, se muestra la ***Tabla 12*** con el resumen de los resultados obtenidos tras la validación de cada modelo:

Tabla 12. Resumen métricas de desempeño modelo YOLO v8m, v9m, v10m

Modelo	Precisión	Recall	F1-score	Exactitud	mAP@0.5:0.95
YOLOv8	0.8650	0.8763	0.8707	0.9054	0.9067
YOLOv9	0.9002	0.8511	0.8750	0.9062	0.9126
YOLOv10	0.8898	0.8618	0.8756	0.9025	0.9146

A continuación, se presentan los gráficos que muestran el comportamiento por época de cada una de las métricas evaluadas en las tres versiones de YOLO:

7.2.1 Precisión por época

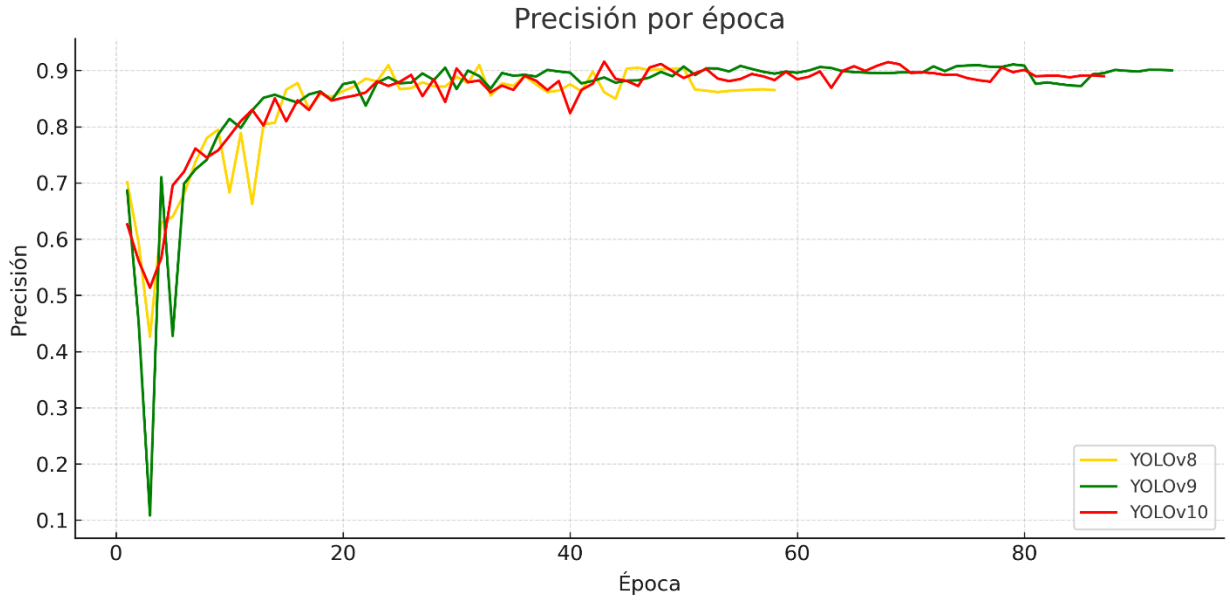

Figura 25. Precisión vs Épocas versiones de YOLO

En la ***Figura 25*** se identifica que:

- Arranque inestable (0–5 épocas): hay oscilaciones fuertes; uno de los modelos cae transitoriamente (YOLO 9), típico de la fase de "calentamiento" del optimizador.
- Mejora rápida (≈5–15): los tres modelos suben velozmente y superan ~0.80 de precisión.
- Meseta estable (≈20 en adelante): todos convergen y se mantienen entre ~0.88 y ~0.91, con pequeñas ondulaciones. No se observan caídas sostenidas que sugieran sobreajuste.

- Comparativa: desde ≈25 épocas, YOLOv9 y YOLOv10 suelen estar levemente por encima de YOLOv8 que muestra pequeños baches alrededor de 50–60. Al final, YOLOv9 y YOLOv10 convergen cerca de 0.90, obteniendo un mejor comportamiento en la métrica de precisión.

7.2.2 Recall por época

Figura 26. Recall vs Épocas versiones de YOLO

En la **Figura 26** se identifica que:

- Arranque (0–5 épocas): alta inestabilidad; caídas puntuales normales en calentamiento, sobre todo en v9.
- Subida rápida (≈5–20): los tres superan 0.80 de recall; YOLO9 y YOLO10 recuperan antes que YOLO8.
- Meseta estable (≈20–90): se mantienen entre 0.84–0.88 con pequeñas ondulaciones, sin descensos prolongados que sugieran sobreajuste.
- Comparativa:
    - YOLO9 suele liderar entre ~20–40 épocas.
    - YOLO10 toma una ligera ventaja entre ~45–70.
    - Final (~80–90): YOLO10 ≈ YOLO8 ≥ YOLO9 por una diferencia mínima (~0.01–0.02).

7.2.3 mAP@0.5:0.95 por época

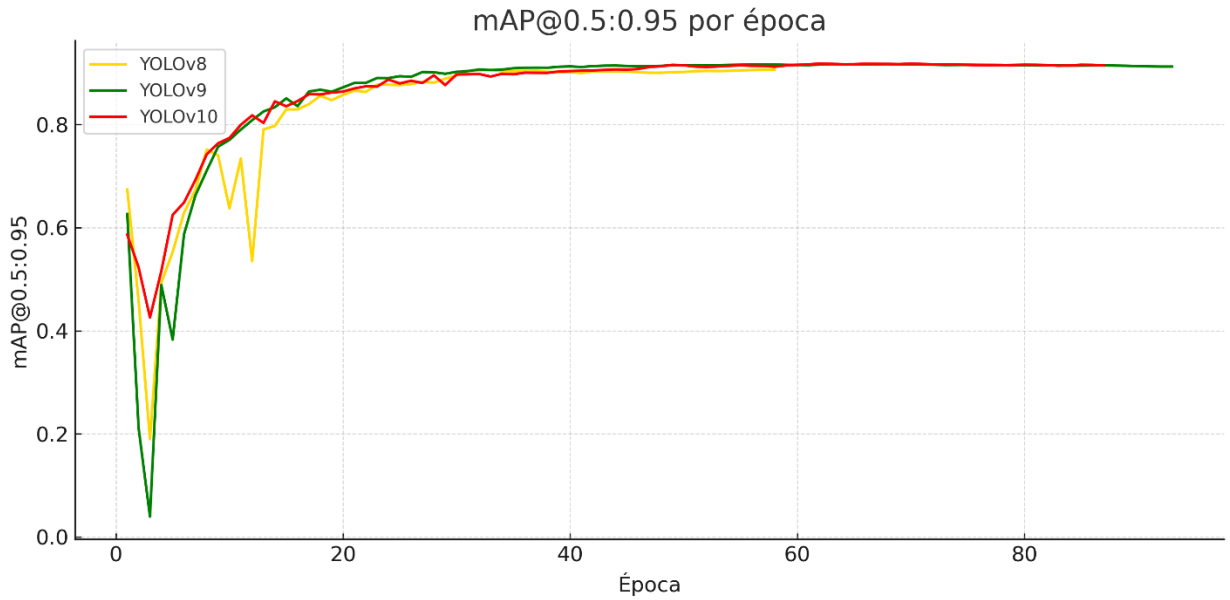

Figura 27. Map50-95 vs Épocas versiones de YOLO

En la ***Figura 27*** se identifica que:

- Inicio (0–5): alta volatilidad, con caídas puntuales (especialmente YOLO9).
- Subida rápida (≈5–20): los tres modelos escalan hasta ~0.80–0.85.
- Convergencia (≈20–40): estabilizan alrededor de ~0.86–0.88; YOLO9/YOLO10 llegan antes y YOLO8 queda levemente atrás.
- Meseta (≈40–90): mejora lenta y sostenida hasta ~0.90–0.92, sin señales de sobreajuste.
- Comparativa: YOLO9 y YOLO10 muestran una ligera ventaja temprana, pero al final las diferencias son mínimas (<0.01–0.02)

7.3 SELECCIÓN DEL MODELO

Para la selección del modelo más adecuado se consideraron como criterios principales las métricas de precisión, recall y mAP@0.5:0.95. A partir de estos indicadores se formularon cuatro argumentos que justifican la elección final.

7.3.1 Calidad integral de detección

El modelo YOLOv10 alcanza el mayor valor de mAP@0.5:0.95 (0.9146), métrica estándar que permite evaluar de manera simultánea la capacidad de clasificación y localización en un rango

amplio de umbrales IoU. Asimismo, en las curvas de desempeño se observa que YOLOv9 y YOLOv10 logran estabilizarse con mayor rapidez que YOLOv8 y conservan una ventaja sostenida durante la validación. Después de las 40–50 épocas, las mejoras son mínimas, lo que sugiere un proceso de convergencia adecuado.

7.3.2 Equilibrio precisión–recall

En cuanto al equilibrio entre precisión y recall, YOLOv10 obtiene el mejor F1-score (0.8756), lo que evidencia un balance más consistente entre falsos positivos (precisión) y falsos negativos (recall) en el umbral seleccionado. De manera numérica, YOLOv8 alcanza Precisión 0.8650 y Recall 0.8763; YOLOv9, Precisión 0.9002 y Recall 0.8511; mientras que YOLOv10 registra Precisión 0.8898 y Recall 0.8618. Aunque YOLOv9 sobresale en precisión y YOLOv8 en recall, dichas ventajas son unidireccionales. Por consiguiente, el F1-score superior de YOLOv10 refleja que este modelo tiene un mejor equilibrio entre precisión y recall.

7.3.3 Dinámica de entrenamiento

En relación con la dinámica de entrenamiento, las curvas de aprendizaje de los tres modelos muestran mesetas estables. No obstante, YOLOv10 presenta oscilaciones de menor amplitud y ausencia de sobreajuste evidente, en particular en recall, donde el comportamiento final se distribuye como YOLOv8 ≥ YOLOv10 ≥ YOLOv9. En términos de precisión, tanto YOLOv9 como YOLOv10 se sitúan ligeramente por encima de YOLOv8 a partir de la época 25. Este comportamiento respalda la estabilidad del proceso de entrenamiento y la capacidad de generalización de YOLOv10.

7.3.4 Interpretación para despliegue

Desde una perspectiva aplicada, lo deseable es maximizar el mAP sin deteriorar el recall. En este sentido, YOLOv10 ofrece el balance más adecuado (mAP 0.9146 y F1 0.8756), lo que lo convierte en la opción más consistente para su implementación práctica. Cabe destacar que la métrica de exactitud, aunque similar entre los tres modelos, resulta menos informativa en tareas de detección debido al desbalance inherente de negativos; por tanto, no constituye un criterio determinante en la selección final.

7.3.5 Modelo Seleccionado

En términos generales, los tres modelos evaluados presentan un desempeño satisfactorio. Sin embargo, el análisis conjunto de las métricas cuantitativas y de las curvas de validación evidencia que YOLOv10 ofrece la mejor combinación entre calidad integral de detección, equilibrio entre precisión - recall y estabilidad frente a oscilaciones y sobreajuste. Estos elementos, en conjunto, lo posicionan como el modelo más consistente y confiable dentro del conjunto evaluado.

De acuerdo con los resultados obtenidos y considerando la calidad de detección multi-IoU, el equilibrio precisión–recall y la estabilidad de entrenamiento, el modelo YOLOv10 es el más adecuado para el alcance del presente proyecto, al situarse como el más equilibrado para la tarea de detección de residuos sólidos. Su alto mAP@0.5:0.95 refleja su capacidad de generalización en escenarios con variabilidad en la calidad de las detecciones, convirtiéndolo en la opción más recomendada para su implementación final en el sistema propuesto. Este análisis justifica técnicamente la selección del modelo y sienta las bases para la validación en pruebas reales y el desarrollo de aplicaciones futuras en sistemas inteligentes de gestión ambiental.

La ***Figura 28*** se evidencia el desempeño del modelo de detección y clasificación de residuos orgánicos en un conjunto representativo de imágenes, donde se observan distintos tipos de residuos alimentarios en diferentes estados de conservación y descomposición en este caso en la categoría orgánico. El sistema logra identificar correctamente la clase Orgánico en escenarios con variaciones significativas de textura, color, forma e iluminación, asignando valores de confianza elevados en la mayoría de los casos. Asimismo, se aprecia que el modelo mantiene capacidad de detección incluso cuando los residuos presentan moho, deterioro visible o fondos no homogéneos, lo cual demuestra su robustez frente a condiciones más cercanas a entornos reales. No obstante, en algunos ejemplos se registran valores de confianza menores, asociados principalmente a imágenes con mayor complejidad visual o degradación avanzada del material orgánico, lo que resalta la influencia de las condiciones del entorno sobre la precisión del modelo. En conjunto, los resultados confirman la efectividad del sistema para la clasificación de residuos y validan su aplicabilidad como base para sistemas de apoyo en procesos de gestión y separación de residuos.

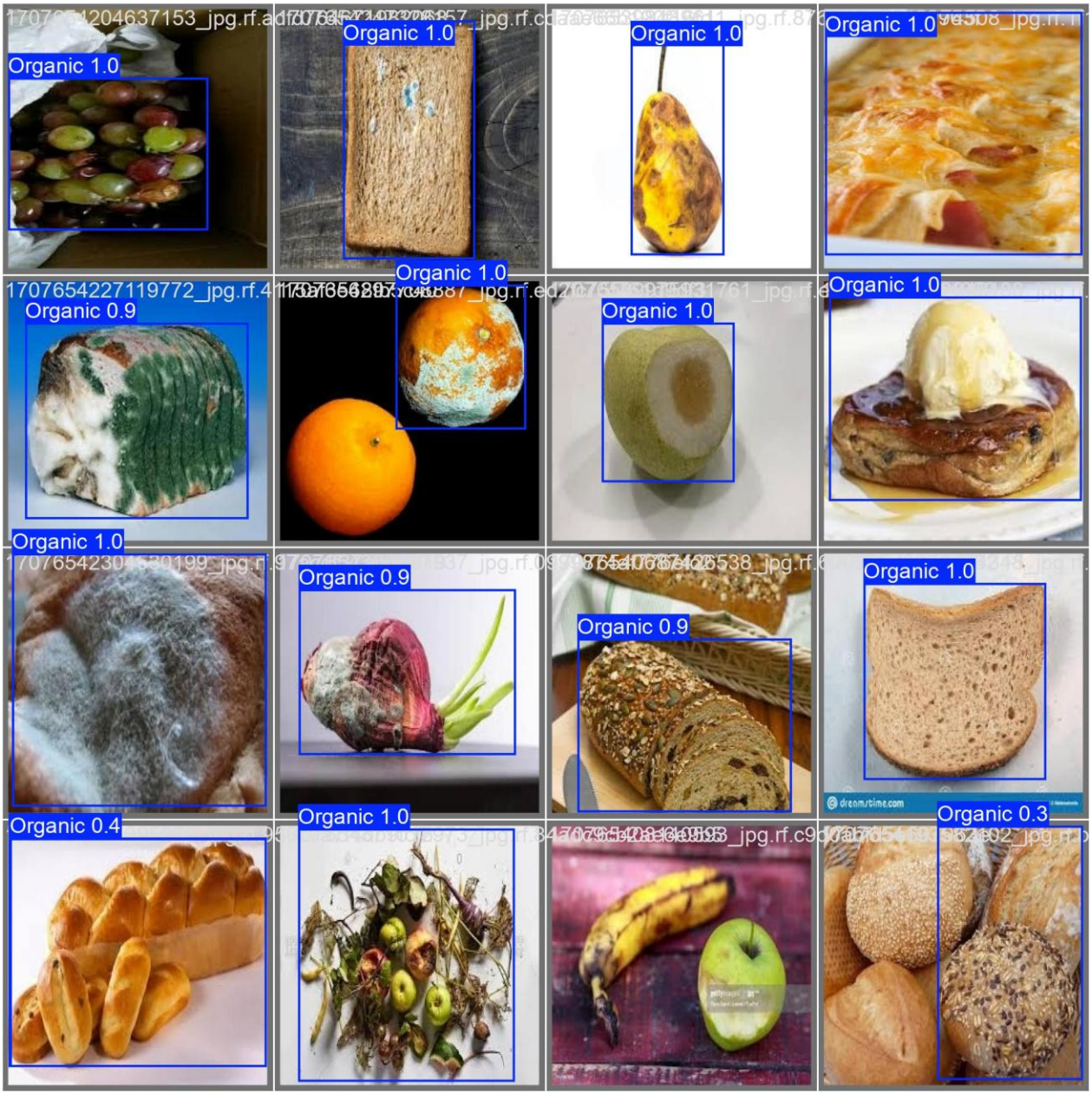

Figura 28. Prueba de detección YOLO10m

8 INTERFAZ DE USUARIO

La interfaz de usuario (Human Machine Interface – HMI) constituye el componente visual e interactivo del sistema WASOR. Su función principal es permitir la interacción entre el usuario y el modelo de inteligencia artificial encargado de la clasificación de residuos sólidos, garantizando una experiencia sencilla, intuitiva y educativa. Este módulo integra el modelo entrenado con un entorno gráfico que muestra, en tiempo real, el resultado de la detección, el tipo de residuo identificado y el contenedor recomendado según el código de colores del reciclaje.

La implementación de la HMI representa la fase final del desarrollo del sistema, donde se materializa la comunicación entre los procesos de inferencia del modelo YOLOv10m y la visualización de los resultados al usuario. Este apartado describe el diseño conceptual, la arquitectura de software, los componentes funcionales y las pruebas de desempeño de la interfaz desarrollada.

8.1 DISEÑO CONCEPTUAL DE LA INTERFAZ

El diseño de la interfaz se fundamentó en principios de usabilidad, minimalismo y retroalimentación visual inmediata. Se buscó que el sistema no solo clasificara residuos, sino que además educara al usuario sobre la correcta disposición de estos.

Los objetivos específicos del diseño HMI fueron:

- Mostrar en tiempo real la imagen capturada y su predicción.
- Visualizar el nombre del residuo y el color del contenedor correspondiente.
- Permitir una interacción fluida, sin requerir conocimientos técnicos.
- Mantener un diseño limpio y moderno.

La interfaz se desarrolló en Python utilizando las librerías Tkinter, OpenCV, imutils, NumPy, Pillow y Ultralytics para la integración del modelo YOLOv10m exportado en formato .pt.

8.2 COMPONENTES DE LA INTERFAZ HMI

La Interfaz Humano–Máquina (HMI) del sistema WASOR fue desarrollada con un diseño compacto que integra en un solo script la lógica de inferencia, la presentación visual y la gestión de eventos.

Esta integración permite la comunicación directa entre el modelo de detección de residuos y el usuario final, garantizando una ejecución eficiente y una experiencia de uso fluida.

A continuación, se describen los componentes principales que conforman la HMI:

8.2.1 Captura de Video (Cámara)

Este componente es responsable de la adquisición de imágenes en tiempo real, utilizando una cámara USB o integrada. Para ello, se implementa la función cv2.VideoCapture() de la librería

OpenCV, que permite obtener los fotogramas de video que serán posteriormente procesados por el modelo. Además, se emplea la librería imutils para redimensionar las imágenes y mantener la proporción visual en el lienzo de la interfaz, optimizando así la fluidez y la estabilidad del sistema.

8.2.2 Inferencia

En esta etapa se carga el modelo YOLOv10m, previamente entrenado con el dataset de residuos sólidos, para realizar la predicción del tipo y la ubicación del residuo en cada cuadro de video. El modelo identifica las clases definidas: plástico, papel, vidrio, metal, biológico, baterías y orgánico, asignando a cada una un nivel de confianza y una posición espacial mediante coordenadas de bounding box.
El procesamiento numérico de estas predicciones se gestiona con la librería NumPy, garantizando eficiencia y precisión en el cálculo.

8.2.3 Visualización

Este módulo tiene como objetivo presentar los resultados del análisis de manera clara e interactiva.

La visualización se realiza mediante el componente Canvas de Tkinter, donde se muestran los fotogramas procesados junto con las cajas delimitadoras y etiquetas generadas por el modelo.
Asimismo, la interfaz incluye un panel lateral derecho que se actualiza dinámicamente para mostrar mensaje informativo, en el panel lateral izquierdo se visualiza el tipo de residuo y color del contenedor correspondiente (verde, azul, gris, rojo, entre otros). En el panel central se visualiza el video de la cámara donde se delimita el residuo detectado e indica el tipo de residuo y el porcentaje de inferencia.

Para asegurar la compatibilidad entre los formatos de imagen, se emplea la librería Pillow (PIL), que convierte los fotogramas de NumPy a objetos ImageTk.PhotoImage compatibles con la visualización en Tkinter.

8.2.4 Mensaje Informativo

Este componente tiene una función educativa y de retroalimentación visual. Se encarga de mostrar mensajes informativos o consejos sobre reciclaje, reforzando la conciencia ambiental del usuario. Además, gestiona la respuesta visual del sistema mediante íconos o variaciones cromáticas en función del tipo de residuo detectado, favoreciendo una experiencia más intuitiva y pedagógica.

En conjunto, estos componentes conforman un diseño ligero y funcional, que permite a la HMI ejecutar tareas de detección, visualización y comunicación de resultados en tiempo real, fortaleciendo la interacción entre la inteligencia artificial del sistema y el usuario final.

8.3 DISEÑO GRÁFICO Y EXPERIENCIA DE USUARIO

La interfaz fue desarrollada bajo una paleta de colores inspirada en los principios de sostenibilidad y ecología, predominando tonos verdes y blancos. Se implementó un diseño moderno para pantallas de 1280x720 píxeles.

Elementos visuales principales:

- Panel superior: muestra el título *"WASOR – Clasificador de Residuos"*.
- Panel principal: despliega el flujo de video procesado con las cajas delimitadoras.
- Panel lateral izquierda: indica el nombre del residuo, su categoría y el color del contenedor.
- Panel lateral derecha: muestra mensajes de concientización ambiental como *"Desde la tierra hasta la mesa, el reciclaje de residuos orgánicos nutre nuestro ciclo natural"*.

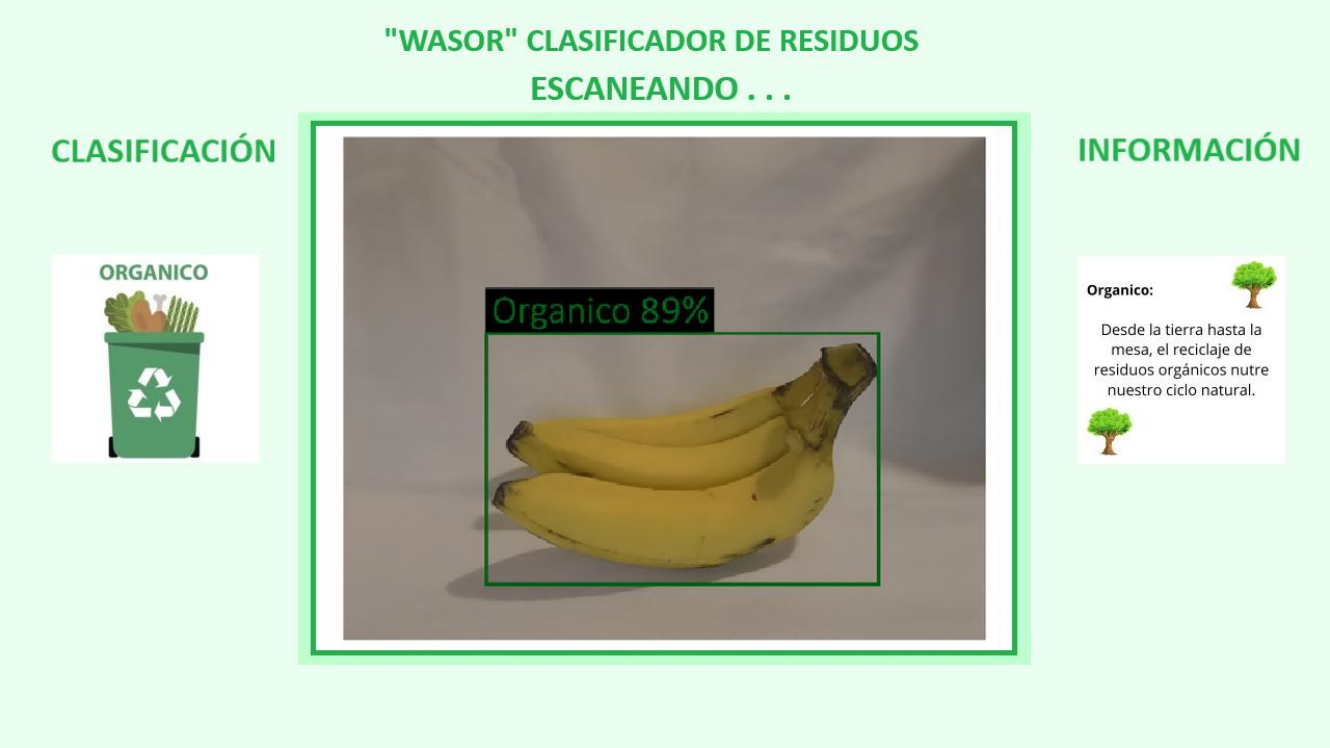

Figura 29. Interfaz de Usuario

Se priorizó la legibilidad y la simplicidad para facilitar la interacción incluso en contextos educativos o demostrativos, donde el usuario no posee formación técnica.

8.4 INTEGRACIÓN CON EL MODELO YOLOv10m

El modelo YOLOv10m fue integrado mediante la librería Ultralytics (versión 8.1.0). La inferencia se ejecuta sobre cada cuadro capturado, garantizando un proceso de detección de objetos en tiempo real.

Los resultados del modelo son interpretados y enviados a la interfaz, donde se dibuja un rectángulo alrededor del residuo detectado y se muestra la clase correspondiente. Cada clase está asociada a un color de contenedor según lo definido a continuación:

A continuación, se muestra la ***Tabla 13*** con la asociación de colores según el tipo de residuo.

Tabla 13. Color del contenedor por tipo de residuo

Clase	Color contenedor
Plástico	Gris
Papel	Amarillo
Vidrio	Azul
Metal	Naranja
Biológico	Rojo
Baterías	Café
Orgánico	Verde

La predicción incluye una etiqueta textual con el porcentaje de confianza, lo cual refuerza la transparencia del sistema y facilita su validación en pruebas de campo.

8.5 PRUEBAS DE DESEMPEÑO DE LA HMI

Se realizaron pruebas experimentales de la interfaz bajo condiciones reales utilizando una cámara HD de 30 FPS. Los resultados demostraron que el sistema mantiene una buena tasa de procesamiento con una latencia visual baja la cual es imperceptible para el usuario.

Resultados observados:

- Detección correcta del residuo en más del 90% de las pruebas en tiempo real.
- Cambio instantáneo de etiqueta y color de contenedor tras la detección.
- Interfaz estable durante sesiones prolongadas (>30 minutos).
- El sistema presenta un desempeño en **tiempo real**, al mantener un tiempo de cómputo que permite una detección continua sin latencia perceptible.
- Consumo promedio de CPU del 20% y de memoria RAM del 3% en equipos de gama media.

Estos resultados validan la funcionalidad operativa de la HMI y su potencial para ser implementada a futuro en puntos ecológicos automatizados o kioscos educativos.

8.6 IMPACTO Y POTENCIAL DE LA HMI

Más allá de su utilidad tecnológica, la interfaz WASOR cumple una función pedagógica y social. Al brindar retroalimentación inmediata y visual sobre el tipo de residuo, fomenta la educación ambiental y promueve la participación ciudadana en la separación en la fuente.

8.7 LIBRERÍAS UTILIZADAS

A continuación, se muestra la ***Tabla 14*** con las librerías utilizadas en el desarrollo del proyecto.

Tabla 14. Librerías

Librería / Módulo	Rol dentro de la HMI
tkinter (from tkinter import *)	Creación de la ventana principal y paneles.
Pillow (from PIL import Image, ImageTk)	Conversión de frames de NumPy a formatos compatibles con Tkinter (ImageTk.PhotoImage).
OpenCV (import cv2)	Captura de video, dibujo de cajas delimitadoras y preprocesamiento.
imutils (import imutils)	Redimensionamiento eficiente y mantenimiento del aspecto de video.
NumPy (import numpy as np)	Manejo de arreglos para las imágenes y resultados de detección.
Ultralytics (from ultralytics import YOLO)	Carga e inferencia del modelo YOLOv10m entrenado.
math (import math)	Cálculos geométricos y auxiliares.

La interfaz HMI del sistema WASOR representa la materialización tangible del proceso de clasificación automatizada. Mediante la integración del modelo YOLOv10m con un entorno visual interactivo, se logró un sistema funcional y de fácil uso.

Este módulo no solo mejora la eficiencia del reciclaje, sino que también constituye una herramienta educativa poderosa para promover la cultura ambiental. En conjunto con el modelo de inteligencia artificial, la interfaz HMI consolida el sistema WASOR como una solución tecnológica sostenible orientada a la gestión inteligente de residuos sólidos.

9 CONCLUSIONES Y TRABAJOS FUTUROS

9.1 CONCLUSIONES

El desarrollo del proyecto WASOR permitió demostrar la viabilidad técnica y tecnológica de aplicar técnicas de inteligencia artificial y visión por computador en la clasificación automática de residuos sólidos.

A partir del diseño, entrenamiento, evaluación e implementación del modelo YOLOv10m, así como la integración de una interfaz de usuario (HMI) funcional, se logró cumplir con los objetivos planteados y validar la utilidad del sistema en contextos reales de gestión ambiental.

Las principales conclusiones son las siguientes:

1. **Relevancia del procesamiento de datos y la calidad del dataset:**

   La construcción del dataset de imágenes fue un proceso fundamental para garantizar el desempeño del modelo de clasificación. La recopilación, depuración y normalización de las imágenes permitió conformar un conjunto de datos variado, balanceado y representativo de las diferentes categorías de residuos sólidos. Este trabajo aseguró la disponibilidad de datos confiables y de calidad, lo cual se reflejó en un mejor aprendizaje del modelo y en una mayor capacidad de generalización frente a escenarios reales.

2. **Entrenamiento Basado en Redes Neuronales Profundas**

   El entrenamiento de modelos de clasificación de imágenes mediante redes neuronales profundas permitió desarrollar un sistema capaz de identificar residuos sólidos con buen desempeño. El uso de arquitecturas como YOLOv8m, YOLOv9m y YOLOv10m, junto con técnicas de data augmentation y ajuste de hiperparámetros, aseguró una convergencia estable y un buen rendimiento. Estos resultados confirman que las redes neuronales profundas son una solución efectiva y adecuada para la clasificación automática de residuos.

3. **Importancia del enfoque experimental y comparativo:**

   La evaluación del rendimiento de los modelos utilizando métricas especializadas como precisión, recall y mAP permitió comparar objetivamente el comportamiento de las distintas versiones de YOLO. Este análisis evidenció diferencias en la capacidad de detección y generalización de cada modelo, permitiendo identificar a YOLOv10m como la arquitectura con mejor equilibrio entre exactitud, eficiencia computacional y estabilidad durante el entrenamiento. Los resultados obtenidos confirman la importancia de las

métricas de clasificación como herramienta fundamental para seleccionar modelos confiables en tareas de visión por computador.

4. **Eficiencia en la clasificación automática:**

   El modelo YOLOv10m alcanzó altos niveles de precisión y estabilidad, superando el 91% de mAP@0.5:0.95 en el dataset propuesto. Esto evidencia su capacidad para identificar residuos de distintas categorías con un margen de error bajo y un rendimiento adecuado para aplicaciones en tiempo real.

5. **Interfaz HMI como puente entre la tecnología y el usuario:**

   La implementación del prototipo funcional permitió integrar de manera coherente el modelo de detección entrenado con la interfaz de usuario, demostrando la viabilidad operativa del sistema WASOR en escenarios reales. La HMI desarrollada facilitó la interacción entre el usuario y el modelo, permitiendo visualizar las detecciones en tiempo real y presentar la clasificación del residuo de forma clara e intuitiva. Este resultado evidencia que el prototipo cumple con los requisitos funcionales planteados y constituye un paso decisivo hacia la aplicación práctica del sistema, validando su capacidad para apoyar procesos de separación y gestión inteligente de residuos sólidos.

6. **Contribución al reciclaje inteligente y la sostenibilidad:**

   El sistema WASOR constituye un aporte a la automatización de procesos de reciclaje, ofreciendo una alternativa tecnológica accesible que promueve la separación en la fuente y la reducción de residuos no aprovechables. Su aplicación en entornos urbanos, institucionales o industriales puede optimizar la eficiencia del reciclaje y fomentar la economía circular.

7. **Pertinencia social y académica del proyecto:**

   WASOR demuestra cómo la ciencia de datos puede ser aplicada a problemas ambientales concretos, contribuyendo a los Objetivos de Desarrollo Sostenible (ODS), particularmente al ODS 12: “Producción y consumo responsables”. Asimismo, consolida un modelo replicable para investigaciones futuras en el ámbito de la inteligencia ambiental.

En síntesis, WASOR evidencia que la combinación de aprendizaje profundo, visión por computador e interfaces inteligentes puede transformar los procesos de gestión de residuos en soluciones sostenibles, eficientes y pedagógicas, aportando tanto al conocimiento científico como al desarrollo tecnológico del país.

9.2 TRABAJOS FUTUROS

A partir de los resultados obtenidos y las oportunidades de mejora identificadas, se proponen las siguientes líneas de trabajo para el fortalecimiento del sistema WASOR y su aplicación práctica:

1. **Ampliación del dataset:**

   Incluir nuevas clases de residuos (como textiles, electrónicos y empaques compuestos) y capturas en condiciones de iluminación variable, exteriores y ambientes industriales, para incrementar la capacidad de generalización del modelo. Incluir también imágenes con múltiples clases de residuos en una misma escena para fortalecer la detección multiclase.

2. **Optimización del modelo para dispositivos embebidos:**

   Implementar versiones livianas del modelo YOLO (como YOLOv10-nano o YOLOv8s) compatibles con Raspberry Pi, Jetson Nano o ESP32-CAM, que permitan la operación autónoma sin necesidad de GPU.

3. **Integración con sistemas IoT y bases de datos:**

   Conectar el sistema HMI a una base de datos en la nube que registre el historial de clasificaciones y volúmenes de residuos. Esto facilitaría la analítica ambiental y la toma de decisiones en políticas de reciclaje urbano.

4. **Incorporación de control físico automatizado:**

   Integrar actuadores y servomotores que permitan mover los residuos hacia el contenedor adecuado, generando un sistema robótico de clasificación automática en tiempo real.

5. **Desarrollo de una versión móvil o web:**

   Crear una aplicación multiplataforma (Android/iOS/Web) que utilice la cámara del dispositivo para identificar residuos, acercando la tecnología a la ciudadanía y promoviendo educación ambiental interactiva.

6. **Entrenamiento continuo y autoaprendizaje:**

   Implementar técnicas de active learning que permitan que el modelo aprenda de nuevos datos de los usuarios, actualizando su conocimiento de manera iterativa sin requerir un reentrenamiento completo.

7. **Validación en entornos reales:**

Realizar pruebas piloto en instituciones educativas, empresas o municipios para medir el impacto del sistema en la reducción de residuos mal clasificados y en el aumento de materiales reciclados.

El proyecto WASOR marca un precedente en la integración de inteligencia artificial con sostenibilidad ambiental. Su desarrollo demuestra que la innovación tecnológica, aplicada de forma responsable, puede contribuir significativamente a la gestión eficiente de residuos, a la educación ambiental y al avance hacia ciudades inteligentes y sostenibles.

10 REFERENCIAS BIBLIOGRÁFICAS

[1] U. N. Environment, «Global Waste Management Outlook 2024,» 2024. [En línea]. Available: https://wedocs.unep.org/bitstream/handle/20.500.11822/44939/global_waste_management_outlook_2024.pdf?sequence=3.. [Último acceso: 07 Diciembre 2024].

[2] A. J. M. a. J. Browne, «Waste management models and their application to sustainable waste management,» vol. 24, nº 3, pp. 297-308, 2004.

[3] B. Mundial, «Los desechos: un análisis actualizado del futuro de la gestión de los desechos sólidos,» 20 09 2018. [En línea]. Available: https://www.bancomundial.org/es/news/immersive-story/2018/09/20/what-a-waste-an-updated-look-into-the-future-of-solid-waste-management. [Último acceso: 06 04 2025].

[4] S. R. S. G. F. a. C. V. S. Kumar, «Challenges and opportunities associated with waste management in India,» *Royal Society Open Science,* vol. 4, nº 2, p. 100764, 2017.

[5] I. M. I. Y. Ihar Yeuseyenka, «Detection and Selection of Moving Objects in Video Images Based on Impulse and Recurrent Neural Networks,» *Journal of Data Analysis and Information Processing,* vol. 10, nº 2, pp. 127-141, 2024.

[6] A. Burkov, The Hundred-Page Machine Learning Book, eBook, 2019.

[7] Y. S. a. S. W. Q. Guo, «Research on deep learning image recognition technology in garbage classification,» de *ACCTCS*, 2021.

[8] Á. A. Moreno, *Clasificación de imágenes usando redes neuronales convolucionales en Python,* Sevilla, 2019.

[9] R. M. a. M. K. M. Elleuch, «A new design based-SVM of the CNN classifier architecture with dropout for offline Arabic handwritten recognition,» *Procedia Computer Science,* vol. 70, nº 1, pp. 1712-1723, 2016.

[10] P. K. a. M. G. J. Jaworek-Korjakowska, «Melanoma thickness prediction based on convolutional neural network with VGG-19 model transfer learning,» *Computer Vision Fundation,* vol. 1, p. 9, 2019.

[11] W. C. a. L. C. F. Zhu, «Classification-assisted Deep Sparse Image Recognition," 2021 IEEE 20th International Conference on Cognitive Informatics & Cognitive Computing,» *IEEE,* pp. 189-193, 31 10 2021.

[12] A. F. A. P. A. T. a. S. M. Marjory Appel, «The State of Recycling Today 2024,» *Recycling Today Media Group,* 2024.

[13] Roboflow, «Give your computer vision model superpowers,» [En línea]. Available: https://roboflow.com. [Último acceso: 24 05 25].

[14] Ultralytics, «https://docs.ultralytics.com/,» Ultralytics Inc, 2025. [En línea]. Available: https://docs.ultralytics.com/es/tasks/detect/#models. [Último acceso: 29 10 2025].

[15] ultralytics, «https://docs.ultralytics.com/,» ultralytics Inc, 2025. [En línea]. Available:

https://docs.ultralytics.com/modes/train/#resuming-interrupted-trainings. [Último acceso: 29 10 2025].

[16] S. H. y. A. T. S. P. Gundupalli, «A review on automated sorting of source-separated municipal solid waste for recycling,» *Waste Management,* vol. 60, nº 1, pp. 56-74, 2017.