



Universidad Internacional de La Rioja
Escuela Superior de Ingeniería y Tecnología

Grado en Ingeniería Informática

Identificación de imágenes de perros mediante extracción de características con MobileNetV2

Trabajo fin de estudio presentado por:	Daniela Alejandra Díaz Ruíz
Director/a:	Luís Pedraza
Fecha:	16/07/2025
Repositorio del código fuente:	https://github.com/dasniela/TFG-DogFinderV2

Resumen

El presente trabajo de fin de grado propone un sistema de identificación de imágenes de perros perdidos mediante el uso de técnicas de inteligencia artificial, específicamente a través de la arquitectura MobileNetV2. El objetivo principal es proporcionar una herramienta que agilice la reunificación de los perros perdidos con sus dueños.

Uno de los principales factores que influye en el reencuentro es el tiempo de reacción, ya que estudios demuestran que mientras más tiempo pasa, disminuyen las probabilidades de localizar al perro perdido. Por ello, resulta crucial contar con herramientas automatizadas que agilicen este proceso desde las primeras horas.

Tras analizar el estado del arte, se seleccionó MobileNetV2 como el modelo utilizado como extractor de características por su eficiencia en entornos con recursos limitados. Se conformó un conjunto diverso de imágenes y se diseñó una arquitectura modular capaz de procesarlas y compararlas mediante la extracción de vectores de características y la búsqueda de similitud utilizando FAISS. El sistema incorpora además mecanismos de detección de duplicados, almacenamiento en base de datos y evaluación de desempeño a través de métricas como F1-score, latencia y precisión.

Los resultados evidencian que el uso de técnicas de aprendizaje profundo puede mejorar significativamente la velocidad y precisión frente a métodos manuales. Se concluye que esta solución representa un avance real hacia la automatización de procesos de búsqueda de perros perdidos, siendo viable su futura implementación en aplicaciones móviles.

Palabras clave: "Reconocimiento de imágenes", "Búsqueda de similitud", "Extracción de características", "MobileNetV2", "Perros perdidos", "FAISS"

Abstract

This final degree project presents an image identification system for lost dogs using artificial intelligence techniques, specifically through the MobileNetV2 architecture. The main objective is to provide a tool that facilitates the reunification of lost dogs with their owners.

One of the main factors influencing reunification is response time, as several studies show that the longer it takes, the lower the chances of locating the lost dog. Therefore, it is essential to have automated tools that speed up the process from the very first hours.

After analyzing the state of the art, MobileNetV2 was selected as the feature extractor model due to its efficiency in resource-constrained environments. A diverse image dataset was compiled, and a modular architecture was designed to process and compare these images by extracting feature vectors and performing similarity searches using FAISS. The system also incorporates duplicate detection, database storage, and performance evaluation through metrics such as F1-score, latency, and accuracy.

The results show that deep learning techniques can significantly improve speed and accuracy compared to manual methods. It is concluded that this solution is a real step forward in automating the search for lost pets and is viable for future implementation in mobile applications.

Keywords: "Image recognition", "Similarity search", "Feature extraction", "MobileNetV2", "Lost dogs", "FAISS"

Índice de contenidos

1.	Introducción.....	1
1.1.	Motivación	1
1.2.	Planteamiento del trabajo.....	1
1.3.	Estructura del trabajo.....	2
2.	Contexto y Estado del Arte	3
2.1.	Introducción al contexto del problema.	3
2.1.1.	La importancia del problema de mascotas perdidas a nivel global.....	4
2.1.2.	Datos relevantes sobre perros extraviados y su localización	4
2.1.3.	Métodos y herramientas para la búsqueda de mascotas perdidas.....	5
2.1.4.	Situación actual en México.....	7
2.1.5.	Plataformas existentes para la búsqueda de perros perdidos	10
2.2.	Uso de inteligencia artificial para la identificación visual de animales	14
2.2.1.	Redes neuronales convolucionales	14
2.2.2.	Arquitecturas CNN tradicionales para la identificación de imágenes	16
2.2.3.	MobileNetV2	17
2.2.4.	Comparativa entre VGG, ResNet y MobileNetV2	19
2.2.5.	Análisis de proyectos de código abierto en GitHub	19
2.2.6.	Extracción de características con modelos preentrenados	21
2.2.7.	Optimización de la búsqueda de similitud	22
2.2.8.	Principales desafíos en el reconocimiento de imágenes de perros.....	22
3.	Objetivos y metodología de trabajo	24
3.1.	Objetivo general	24
3.2.	Objetivos específicos.....	24

3.3.	Metodología de trabajo.....	25
3.3.1.	Cronograma.....	26
4.	Requisitos del sistema	27
4.1.	Requisitos Funcionales	27
4.2.	Requisitos no funcionales.....	28
5.	Implementación del sistema de similitud de imágenes.....	30
5.1.	Evaluación y elección del modelo de reconocimiento de imágenes.....	30
5.1.1.	Análisis de DogFaceNet	30
5.1.2.	Justificación de la elección de MobileNetV2	30
5.2.	Recolección y preparación del dataset de imágenes	31
5.2.1.	Control y depuración del conjunto de imágenes	32
5.3.	Estructura modular del desarrollo y organización del código	33
5.4.	Implementación del modelo de reconocimiento	34
5.4.1.	Preparación del entorno y bibliotecas.....	34
5.4.2.	Uso de MobileNetV2 como extractor de características.....	34
5.4.3.	Preprocesamiento de imágenes.....	35
5.4.4.	Extracción de características	36
5.5.	Implementación de la base de datos	36
5.5.1.	Elección de tecnología.....	36
5.5.2.	Estructura de la Base de Datos en SQLite.....	36
5.5.3.	Creación y configuración de la Base de Datos.....	37
5.6.	Registro automático de imágenes.....	37
5.6.1.	Desafíos encontrados	37
5.7.	Detección y prevención de duplicados.....	38

5.7.1.	Optimización con FAISS para la detección de duplicados	38
5.7.2.	Proceso de registro y detección de duplicados	38
5.8.	Implementación de la búsqueda de similitud	39
5.8.1.	Fundamentos de la Similitud por Vectores de Características	39
5.8.2.	Optimización de la Búsqueda con FAISS.....	40
5.8.3.	Proceso de búsqueda de similitud.....	41
6.	Pruebas y evaluación	43
6.1.	Preparación inicial y modelo	44
6.2.	Pruebas funcionales y carga de datos	44
6.2.1.	Carga Inicial de imágenes y persistencia	44
6.2.2.	Detección de Duplicados	45
6.3.	Pruebas de búsqueda de similitud	47
6.3.1.	Procesamiento de la imagen de consulta.....	47
6.3.2.	Recuperación de resultados y aplicación del umbral	48
6.3.3.	Ordenamiento y Visualización de Resultados	48
6.3.4.	Ejemplos de resultados de la búsqueda de similitud	48
6.3.5.	Pruebas con imágenes desafiantes	51
6.4.	Evaluación de rendimiento y precisión	53
6.4.1.	Latencia de búsqueda.....	53
6.4.2.	Escalabilidad del registro.....	54
6.4.3.	Evaluación de Precisión	54
6.4.4.	Optimización para Entornos de Recursos Limitados	59
6.4.5.	Limitaciones del sistema	61
7.	Conclusiones y trabajo futuro	62

7.1.	Conclusiones del trabajo	62
7.2.	Líneas de trabajo futuro	65
	Referencias bibliográficas	66
Anexo A.	Resultados de búsqueda de similitud en imágenes	73
Anexo B.	Resultados pruebas con imágenes desafiantes.....	75
Anexo C.	Matriz de pruebas y requisitos relacionados	77
Anexo D.	Matriz de relación: requisitos, pruebas y resultados.	78

Índice de figuras

Figura 1. Métodos y distancias de recuperación de perros perdidos en Montgomery, Ohio...	4
Figura 2. Ejemplo de publicación en grupo de Facebook.....	6
Figura 3. Percepción hacia los perros callejeros	8
Figura 4. Características para considerar a un perro callejero	8
Figura 5. Petco Love Lost.....	11
Figura 6. Imagen de prueba: golden retriever	12
Figura 7. Flujo de búsqueda de un perro perdido en Pet Love Lost.....	13
Figura 8. Resultados de búsqueda Pet Love Lost.	13
Figura 9. Descripción del funcionamiento de una red neuronal convolucional.....	15
Figura 10. Ejemplo de red VGG 16	16
Figura 11. Arquitectura ResNet.....	17
Figura 12. Arquitectura de MobileNetV2.	18
Figura 13. Cronograma.....	26
Figura 14. Ejemplos de imágenes de diferentes datasets utilizados.....	31
Figura 15. Expansión de dimensiones para el lote	35
Figura 16. Salida del sistema durante carga inicial de imágenes y persistencia.	45
Figura 17. Imágenes con similitud por características	46
Figura 18. Salida del sistema durante carga y sincronización del índice FAISS	47
Figura 19. Salida del sistema durante procesamiento de imagen de consulta	48
Figura 20. Resultados de búsqueda de un perro blanco con manchas café	49
Figura 21. . Resultados de búsqueda de un perro de raza husky siberiano.	50
Figura 22. Prueba con imagen de un perro mestizo.	51
Figura 23. Prueba con imagen de perro con mala iluminación.....	52

Figura 24. Distribución de latencia de búsqueda	53
Figura 25. Salida del sistema durante prueba de detección de duplicados	55
Figura 26. Grupo de perros con pelaje dorado.	56
Figura 27. Grupo de perros con pelaje blanco con manchas.	56
Figura 28. Grupo de perros de la raza dachshund.	56
Figura 29. F1-Score Promedio vs Umbral de Similitud	58
Figura 30. Prueba con imagen de entrada de un gato.	59
Figura 31. Medición de huella de memoria del modelo	60

Índice de tablas

Tabla 1. Grupos de Facebook de perros perdidos.....	9
Tabla 2. Comparativa entre VGG, ResNet, MobileNetV2	19
Tabla 3. Descripción de los datasets utilizados para la búsqueda de similitud de imágenes..	32
Tabla 4. Resultados simplificados por imagen y umbral.	57
Tabla 5. Resultados de análisis de umbral de similitud.....	57

1. Introducción

En este primer capítulo se presenta el marco teórico que motiva el desarrollo del proyecto, así como el planteamiento del problema que se busca resolver. Además, se describe la organización del documento para ofrecer una visión clara y ordenada de los temas que se abordarán en los capítulos siguientes.

1.1. Motivación

Durante mi experiencia como voluntaria en un refugio de perros en México, pude ver de cerca la magnitud del problema de los animales en situación de calle y lo difícil que es lograr que un perro extraviado se reencuentre con su familia. Esta vivencia me llevó a adoptar a Lummi y Maya, dos perritas que hoy forman parte fundamental de mi vida. No puedo imaginar lo que sería perderlas, y es justamente ese vínculo emocional lo que me inspira a desarrollar este trabajo.

En un contexto donde las búsquedas de perros perdidos siguen siendo manuales y poco eficaces, las herramientas basadas en inteligencia artificial, como el reconocimiento de imágenes, pueden ofrecer una solución real. Automatizar este proceso no solo incrementa las posibilidades de reunir a más perros con sus familias, sino que también contribuye a reducir la población de perros en situación de calle y da esperanza a quienes enfrentan la angustia de haber perdido a su compañero.

1.2. Planteamiento del trabajo

En la actualidad, millones de perros se extravían y muchos de ellos no logran reencontrarse con sus familias. Los métodos tradicionales de búsqueda, como carteles físicos, redes sociales y visitas a albergues, resultan lentos, desorganizados y poco eficaces, especialmente cuando el tiempo es un factor determinante.

En un escenario donde la inteligencia artificial ha demostrado su capacidad para resolver problemas complejos de forma automática y eficiente, este proyecto plantea una propuesta concreta, viable y con alto potencial de impacto: el desarrollo de un sistema de identificación de perros mediante reconocimiento de imágenes y aprendizaje profundo.

La arquitectura MobileNetV2 se selecciona como base del modelo por su equilibrio entre precisión y eficiencia, lo que permite su uso en dispositivos con recursos limitados, como teléfonos móviles. A través de la extracción de características visuales y el uso de FAISS como motor de búsqueda por similitud, el sistema puede analizar y comparar imágenes en cuestión de milisegundos, mejorando radicalmente la velocidad y precisión del proceso de búsqueda. Este trabajo se enfoca en el diseño, implementación y validación de dicho sistema, con una arquitectura modular y escalable que permite su futura integración en aplicaciones móviles y plataformas de ayuda animal. El objetivo no es solo técnico, sino también social: automatizar un proceso que hoy depende del azar y convertirlo en una oportunidad real de reencuentro para miles de perros y sus familias.

1.3. Estructura del trabajo

El documento se organiza en siete capítulos principales. En el Capítulo 1 se introduce la motivación que impulsa el desarrollo del proyecto, se plantea el problema a resolver y se presenta la estructura general del trabajo. A continuación, el Capítulo 2 analiza el contexto en el que se enmarca la investigación, proporcionando datos relevantes sobre perros perdidos, revisando los métodos actuales de búsqueda y haciendo un recorrido por el estado del arte en inteligencia artificial aplicada al reconocimiento de imágenes.

El Capítulo 3 está dedicado a exponer los objetivos generales y específicos que guían el desarrollo del sistema, así como la metodología empleada para su construcción. En el Capítulo 4 se definen los requisitos funcionales y no funcionales que el sistema debe cumplir para garantizar su correcto funcionamiento y calidad. Posteriormente, el Capítulo 5 describe el proceso de implementación técnica, desde la selección del modelo MobileNetV2 hasta la integración de los módulos encargados de la búsqueda de similitud y la detección de duplicados. El Capítulo 6 detalla el proceso de pruebas funcionales realizadas, las métricas obtenidas y la evaluación del rendimiento del sistema, enfatizando en cómo se validó la eficacia y precisión del proyecto. Finalmente, el Capítulo 7 presenta las conclusiones alcanzadas y plantea posibles líneas de trabajo futuro para mejorar o ampliar el sistema desarrollado. Además, se incluyen anexos que contienen los resultados detallados de las pruebas, matrices de requisitos y otras evidencias que complementan la evaluación del sistema.

2. Contexto y Estado del Arte

El lugar que ocupa el perro en el ámbito familiar y personal del ser humano es innegable. Kurdek (2008) estudió a los perros como figuras de apego y comprobó que los dueños generan lazos emocionales con ellos, les dan un trato como si fueran un miembro de la familia y los consideran apoyo emocional. Algunos autores hablan del efecto mascota que puede explicarse como la capacidad de los animales de compañía para proporcionar apoyo emocional, de forma similar a como lo hacen los seres humanos (Kanat-Maymon, Wolfson, Cohen, & Roth, 2021). Pero ¿cuál es la razón de este lazo tan fuerte? Bericat (2025) señala que, a diferencia de las relaciones humanas, el vínculo entre una persona y su perro se distingue por su constancia y ausencia de condiciones o expectativas. Por ello, su pérdida o desaparición suele ser psicológica y emocionalmente difícil para el dueño.

El presente estudio busca responder las siguientes preguntas: ¿Qué herramientas utilizan actualmente los dueños para encontrar a sus perros perdidos? ¿Cómo puede la inteligencia artificial, y en particular el reconocimiento automatizado de imágenes, agilizar de forma significativa los esfuerzos de búsqueda en comparación con los métodos tradicionales? ¿De qué manera el uso de MobileNetV2 como extractor de características permite desarrollar un modelo capaz de identificar eficazmente perros similares, optimizando la precisión, la velocidad y el uso de recursos? Para responder a estas preguntas, se iniciará con un análisis del panorama actual y las limitaciones de las herramientas disponibles en México.

2.1. Introducción al contexto del problema.

El extravío de mascotas, en particular de perros, representa un problema significativo a nivel global que afecta tanto al bienestar animal como a las familias que los consideran parte fundamental de su hogar. Esta situación ha dado lugar a la creación de diversas estrategias y herramientas para facilitar su localización. En este apartado se analizará la magnitud del problema, los datos más relevantes, los métodos actualmente utilizados y el panorama específico en México, incluyendo una revisión de las plataformas existentes dedicadas a la búsqueda de perros extraviados.

2.1.1. La importancia del problema de mascotas perdidas a nivel global

Lord et al. (2007) señalan que los perros extraviados que permanecen en la calle pueden enfrentar serios peligros como heridas, hambre e incluso la muerte. Según datos de American Humane Society (AHS, 2022), aproximadamente el 33% de las mascotas se pierden al menos una vez en su vida, y muchas de ellas no logran reunirse nuevamente con sus dueños. Ante este panorama, resulta crucial actuar con rapidez y utilizar las herramientas tecnológicas disponibles. La inteligencia artificial puede desempeñar un papel clave, especialmente a través de sistemas de reconocimiento de imágenes que permiten identificar perros similares que ya existen en una base de datos. Esta tecnología se puede integrar en páginas web o aplicaciones especializadas para ampliar el alcance de la búsqueda y favorece la colaboración entre personas, incrementando de manera significativa las probabilidades de lograr un reencuentro exitoso.

2.1.2. Datos relevantes sobre perros extraviados y su localización

Se han realizado diversos estudios, principalmente en Estados Unidos, para obtener datos y estadísticas sobre el extravío de perros. Weiss et al. (2012), describen que se encuestaron a 1015 dueños de mascotas y encontraron que el 14% había perdido a su perro en los últimos cinco años. Por otro lado, en el estudio de Lord et al. (2007) se realizó una encuesta telefónica a los dueños de 187 perros perdidos en Montgomery, Ohio, con el objetivo de identificar los métodos utilizados para localizarlos y los factores que influyen en el reencuentro. Como se muestra en la Figura 1, el 8% de los perros regresó por sí solo a casa, el 71% fue encontrado a menos de una milla del hogar y el 21% a distancias mayores.

Figura 1. Métodos y distancias de recuperación de perros perdidos en Montgomery, Ohio.



Fuente: Elaboración propia a partir de Lord et al. (2007)

Estos datos indican que mientras más tiempo pasa y más lejos está el perro de su hogar, menor es la probabilidad de encontrarlo. Por eso, una respuesta rápida y eficaz aumenta las posibilidades de éxito. El estudio también muestra las limitaciones de los métodos tradicionales: solo el 38% de quienes encontraron una mascota y buscaron al dueño vía refugios o anuncios lograron reunirlos. Esto evidencia la necesidad de soluciones más eficientes. La inteligencia artificial y el reconocimiento de imágenes permiten analizar y comparar automáticamente las características físicas de los perros, agilizando la identificación y el emparejamiento. Esta tecnología es especialmente útil en las primeras horas críticas, superando las limitaciones de la búsqueda manual y aumenta las probabilidades de éxito.

2.1.3. Métodos y herramientas para la búsqueda de mascotas perdidas

Es importante identificar los métodos que emplean los dueños cuando sus perros se extravían y evaluar su efectividad.

2.1.3.1. Métodos tradicionales

Los métodos tradicionales son efectivos cuando el perro recién desapareció. Algunos de ellos son:

- Realizar una búsqueda física inmediata en las zonas cercanas al último lugar donde fue visto el perro, incluyendo posibles escondites y sitios familiares para el animal.
- Crear y colocar carteles llamativos con una imagen reciente del perro y datos de contacto. En un artículo con recomendaciones para encontrar mascotas perdidas, Cook (2010) sugiere ofrecer una recompensa, aunque advierte sobre el riesgo de fraudes por parte de personas que aseguran tener al animal sin ser cierto, solo para obtener dinero.
- Visitar albergues caninos para verificar si el perro ha sido rescatado y llevado a alguno de estos centros.

2.1.3.2. Identificación mediante microchips y placas de identificación

Los microchips son pequeños dispositivos, aproximadamente del tamaño de un grano de arroz, que se implantan bajo la piel de la mascota (Humane World for Animals). En caso de que el perro se extravíe, estos dispositivos pueden ser escaneados para obtener la información de su dueño. Por su parte, las placas de identificación se colocan en el collar del perro con su nombre y con información de contacto del dueño.

2.1.3.3. Uso de redes sociales

Las redes sociales han transformado la manera en la que la sociedad se comunica. Zahnow et al. (2024) explican que plataformas como Foursquare, Weibo y los grupos de Facebook fortalecen la cohesión social al facilitar el intercambio de información y promover la colaboración comunitaria. En el contexto de la búsqueda de perros perdidos, estas plataformas se han convertido en unas herramientas clave. Según Faster Capital (2025) redes sociales como Facebook, Twitter e Instagram permiten difundir información más allá del círculo cercano de contactos, alcanzando a audiencias mucho más amplias. En particular, Facebook cuenta con numerosos grupos dedicados a la publicación de casos de perros extraviados o encontrados. En la Figura 2 se muestra un ejemplo de una publicación en un grupo de perros perdidos en Facebook. En ella se indica la ubicación donde se encontró al perro y una estimación de su edad. Si los dueños están buscándolo, deben revisar manualmente cada publicación para identificar si alguna imagen corresponde a su mascota, lo que convierte el proceso en algo lento y poco eficiente.

Figura 2. Ejemplo de publicación en grupo de Facebook.



Fuente: (Perros perdidos, 2025)

Un estudio de Arathoon et al. (2024) resalta el papel de estas plataformas como espacios comunitarios que forman redes de apoyo para facilitar la localización de animales perdidos. Los usuarios valoran la rapidez y el amplio alcance de las publicaciones. Sin embargo, también identifican obstáculos que afectan su eficacia:

- La falta de clasificación entre perros perdidos y encontrados obliga a revisar manualmente las publicaciones.
- La abundancia de grupos y mensajes genera un ruido informativo que entorpece la búsqueda.
- Comentarios negativos o poco empáticos dentro de los grupos pueden provocar frustración o desmotivación.
- Las políticas de las plataformas, como el bloqueo temporal por publicaciones repetidas, limitan la visibilidad de los casos.

Ante estas barreras, las herramientas basadas en inteligencia artificial, como el reconocimiento automatizado de imágenes, surgen como una alternativa prometedora. Un modelo capaz de analizar y comparar imágenes de perros perdidos y encontrados podría acelerar el proceso de búsqueda y superar muchas de las limitaciones de los métodos tradicionales.

2.1.3.4. Contratación de empresas especializadas en la búsqueda de mascotas extraviadas.

Existen empresas dedicadas a la localización de perros extraviados que usan métodos tradicionales y digitales. Ofrecen búsqueda física, colocación de carteles, reparto de volantes, uso de megafonía y drones. También realizan campañas en redes sociales.

2.1.4. Situación actual en México

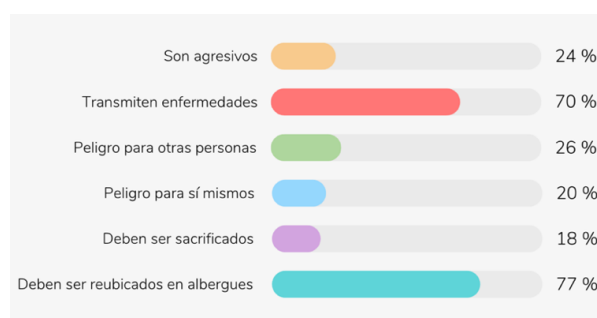
A pesar de la estrecha historia que une al ser humano con el perro, una relación que se remonta a unos 30,000 años (Bericat, 2025), en muchos países la presencia de perros en situación de calle representa un problema alarmante.

2.1.4.1. Estadísticas sobre la población de mascotas y la frecuencia de extravíos.

Según la diputada mexicana Ana Villagrán (2021) cada año se abandonan en México cerca de 500,000 perros y gatos. De los aproximadamente 23 millones de perros existentes en el país,

el 70% vive en la calle, principalmente debido a negligencia y abandono. En la Ciudad de México se sacrifican alrededor de 10 mil perros al mes. Mars Petcare (2020) analizó la percepción de la población mexicana sobre los perros callejeros. Como muestra la Figura 3, entre las principales preocupaciones destacan la agresividad, los riesgos para la salud y la seguridad. A pesar de que la mayoría prefiere soluciones como la reubicación en albergues, estos lugares están saturados. Actualmente, cerca de 96 mil perros viven en albergues (NMAS, 2024), lo que refleja una alta entrada de animales y baja tasa de adopción. Además, los perros alojados en estos centros rara vez son considerados una opción para la adopción: solo el 27% de los encuestados adoptaría en un albergue y el 12% considera que esos perros no son aptos como mascotas.

Figura 3. Percepción hacia los perros callejeros



Fuente: Elaboración propia a partir de (Mars Petcare, 2020)

La Figura 4 muestra los criterios más comunes para identificar a un perro como callejero, entre ellos su forma de buscar comida, el estado físico y la falta de identificación.

Figura 4. Características para considerar a un perro callejero



Fuente: (Mars Petcare, 2020)

Por lo tanto, un perro extraviado puede confundirse fácilmente con un perro callejero. Debido a la alta población de perros en situación de calle en México, es probable que un perro perdido permanezca en la vía pública, ya sea por la falta de interés de la comunidad o por la gran cantidad de animales en condiciones similares.

2.1.4.2. Herramientas tradicionales y digitales para la búsqueda de perros perdidos en México

En México, los dueños de mascotas perdidas recurren tanto a métodos tradicionales como a digitales. Entre los primeros destacan el aviso de boca en boca entre vecinos y la colocación de carteles en la vía pública. La búsqueda en albergues suele ser poco efectiva debido a la saturación de estos espacios, lo que dificulta la recepción de nuevos animales. A esto se suma que el uso de microchips no es obligatorio y que no todos los centros veterinarios cuentan con lectores compatibles, lo que reduce aún más su utilidad práctica. Las placas de identificación tampoco son obligatorias ni se tiene un registro oficial sobre su uso, por lo que es común que los perros extraviados carezcan de cualquier forma visible de identificación.

En el ámbito digital, uno de los recursos más usados son los grupos de Facebook para reportar perros perdidos y encontrados. Estos grupos suelen organizarse por región, facilitando la búsqueda local. La Tabla 1 presenta algunos ejemplos representativos con datos actualizados hasta abril de 2025.

Tabla 1. Grupos de Facebook de perros perdidos

Grupo de Facebook	Fecha de creación	Publicaciones nuevas (último mes)	Número de miembros
Perros perdidos Mexico	23 de abril de 2010	10, 000	64, 700
Perros perdidos C.D.M.X(D.F) y Edo. Mex.	No disponible	No disponible	159,800
Perros perdidos, encontrados y en adopción de Querétaro	5 de diciembre de 2010	2,283	82, 300

Fuente: Elaboración propia a partir de datos consultados en los respectivos grupos de Facebook en abril de 2025.

Además, empresas como Ubicapet¹ y Petloc², ofrecen servicios especializados que combinan métodos tradicionales y digitales para mejorar las probabilidades de encontrar a la mascota.

¹ Página web de Ubicapet: <https://ubicapet.com.mx/>

² Página web de Petloc: <https://petloc.com.mx/>

2.1.5. Plataformas existentes para la búsqueda de perros perdidos

En este apartado se analizan las plataformas digitales usadas para buscar perros perdidos, tanto a nivel internacional como en México. La mayoría de estas plataformas emplean métodos tradicionales como anuncios en redes sociales, grupos comunitarios, volantes y publicidad local. Dado que este estudio se enfoca en el uso de inteligencia artificial y reconocimiento de imágenes, tras una breve descripción de las plataformas convencionales se abordarán las que incorporan tecnologías avanzadas de reconocimiento visual.

A nivel internacional, se identificaron las siguientes plataformas:

- [PawBoost](#): permite publicar avisos en redes sociales y enviar notificaciones a grupos de rescate, imprimir volantes con código QR y añadir al perro perdido en una galería.
- [Pet FBI](#): organización sin fines de lucro que cuenta con una base de datos de mascotas extraviadas y alertas en Estados Unidos y Canadá.

En México, existen las siguientes plataformas:

- [Ubicapet](#): se especializa en búsquedas usando redes sociales y métodos convencionales
- [Petloc](#): se enfoca en la búsqueda de campo mediante herramientas tradicionales como volantes, perifoneo y lonas.

Plataformas que utilizan reconocimiento de imágenes:

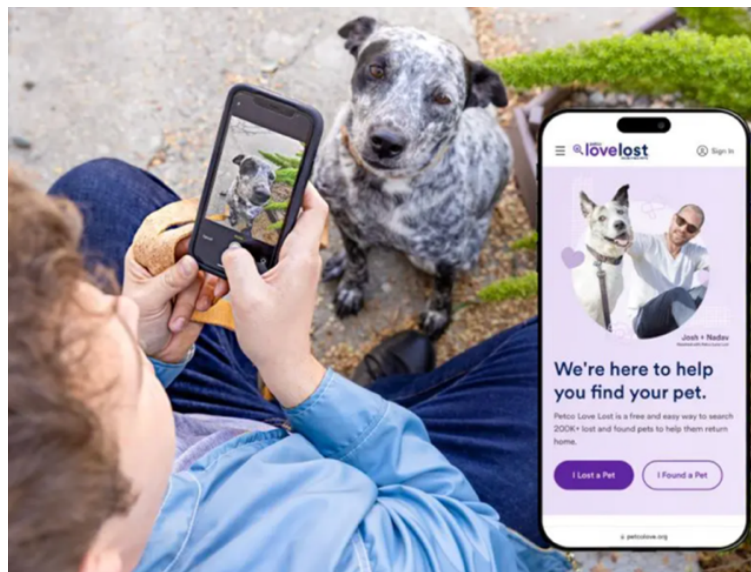
- [Petnow](#): emplea identificación biométrica basada en el análisis de la huella nasal canina, la cual es única para cada perro y comparable a las huellas dactilares humanas. Sin embargo, su eficacia depende de la calidad y ángulo de las imágenes.
- [Petco Love Lost](#): plataforma gratuita desarrollada por la organización sin fines de lucro Petco Love que usa reconocimiento facial para hacer emparejamiento entre imágenes de perros perdidos y encontrados en Estados Unidos. Está patrocinada por empresas como Petco y Hills. A continuación, se presenta un análisis de su funcionamiento.

2.1.5.1. Análisis de la plataforma Petco Love Lost

La plataforma Petco Love Lost es un referente importante para este trabajo, ya que se centra en hacer emparejamiento de imágenes de perros en una base de datos. Por ello, se realiza un análisis detallado de su funcionamiento, ventajas y limitaciones.

La historia de Petco Love Lost comienza con la aplicación Finding Rover, basada en un algoritmo de reconocimiento facial canino llamado Pet Match que fue desarrollado en colaboración con el Software Development Center de la Universidad de Utah (Das, 2013). Posteriormente, Petco Love adquirió Finding Rover e integró esta tecnología en su plataforma. La Figura 5 ilustra el uso imágenes para localizar perros perdidos.

Figura 5. Petco Love Lost



Fuente: (Petco Love, 2025)

Características principales

Es una plataforma gratuita para Estados Unidos, que atiende a tres tipos de usuarios: los propietarios que han perdido a su mascota y pueden reportar la pérdida además de consultar los reportes de animales encontrados, las personas que han hallado un perro y desean notificar el hallazgo para facilitar la localización de su dueño y los refugios y organizaciones de rescate que pueden registrar a los animales bajo su cuidado con el fin de aumentar las posibilidades de coincidencia con mascotas reportadas como perdidas. Entre sus funcionalidades destacan la carga y análisis automático de fotografías para detectar similitudes entre perros, la generación de alertas que pueden compartirse en redes sociales para ampliar el alcance, y el envío de notificaciones en tiempo real sobre posibles coincidencias entre mascotas reportadas y registradas.

Análisis de algoritmo Pet Match

Estudios del 2025 (Quraishi) señalan que el sistema puede capturar hasta 512 características, incluyendo la forma de los ojos, la longitud de los bigotes, marcas distintivas y la curvatura de la cola. Esta tecnología está patentada y ha recibido tres premios Anthem Awards (Petco Love, 2025).

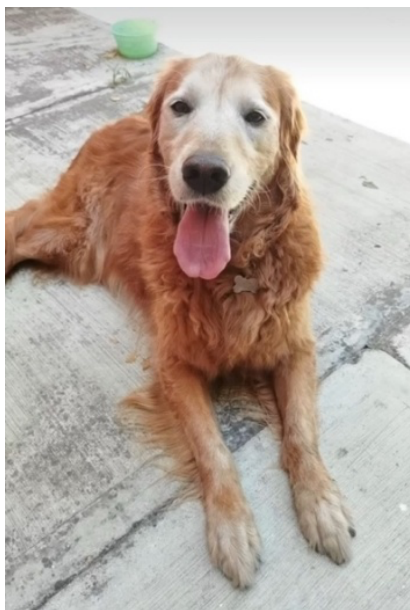
Fuente de datos

La base de datos de Pet Love Losts se nutre de reportes de usuarios, registros de refugios y publicaciones en redes sociales. Según Petco Love (2025) cuenta con más de 300,000 mascotas registradas e información de 3,000 refugios en Estados Unidos, mientras que Quraishi (2025), destaca su capacidad para rastrear informes en redes sociales para ampliar su alcance.

Evaluación práctica del desempeño

Para valorar el desempeño de la plataforma, se realizó una prueba de búsqueda de un perro perdido en las inmediaciones de Central Park, en Nueva York. La imagen utilizada, mostrada en la Figura 6, corresponde a un perro golden retriever, una raza muy común, lo que incrementa las probabilidades de encontrar coincidencias en la base de datos.

Figura 6. Imagen de prueba: golden retriever



Fuente: Dataset Dog_Mx_Dataset

La Figura 7 muestra los pasos que debe seguir un usuario para buscar un perro perdido. Primero, se debe ingresar la ubicación donde el perro fue visto por última vez. Luego, se indica la fecha del extravío. Finalmente, el usuario sube una imagen del perro y ajusta el área que corresponde al rostro. La imagen debe ser de buena calidad y mostrar al perro de frente.

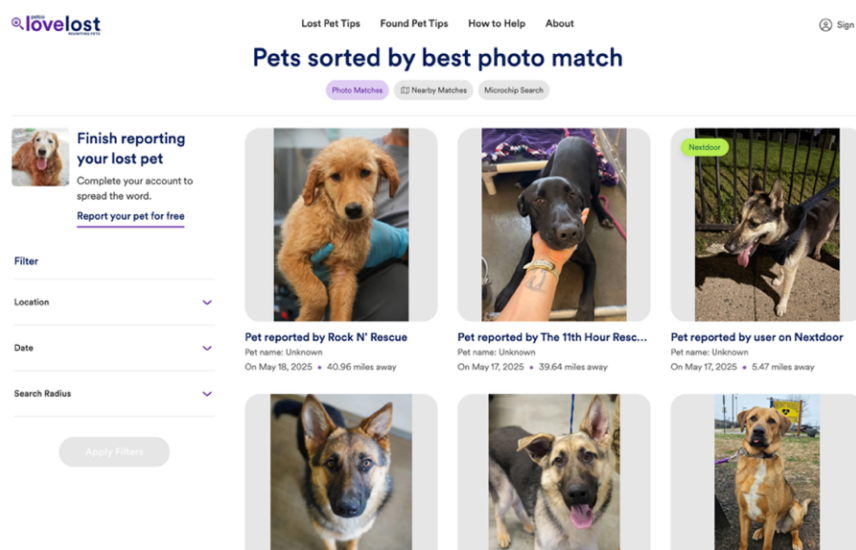
Figura 7. Flujo de búsqueda de un perro perdido en Pet Love Lost.



Fuente: Elaboración propia a partir de (Petco Lovelost, s.f.)

En la Figura 8 se muestran los resultados poco satisfactorios de la prueba de búsqueda, donde solo se detectó una coincidencia parcial y el resto no presentó similitudes físicas evidentes.

Figura 8. Resultados de búsqueda Pet Love Lost.



Fuente: (Petco Lovelost, s.f.)

Si no hay coincidencias claras, Petco Love Lost permite buscar por ubicación o número de microchip.

Ventajas y desventajas: Entre las principales ventajas de Petco Love Lost destacan que es una plataforma gratuita, con una interfaz intuitiva y una base de datos nacional centralizada que facilita la búsqueda sin necesidad de consultar múltiples fuentes. Utiliza tecnología patentada de reconocimiento facial para identificar coincidencias rápidamente y cuenta con el respaldo de una red de refugios y comunidades en Estados Unidos que amplía su alcance.

Entre sus limitaciones se encuentra la inclusión de perros no reportados como perdidos, ya que los refugios registran a todos los animales bajo su cuidado. Además, requiere imágenes frontales de buena calidad. Enfrenta desafíos por la diversidad de razas y los cambios físicos que los perros pueden presentar después de pasar tiempo perdidos, afectando la precisión del algoritmo.

2.2. Uso de inteligencia artificial para la identificación visual de animales

Google (s.f.) define a la inteligencia artificial como un conjunto de tecnologías que permiten a las computadoras realizar tareas complejas que normalmente requieren inteligencia humana, como interpretar y traducir lenguaje, analizar datos, reconocer imágenes y tomar decisiones. Estas tecnologías habilitan a las máquinas para aprender, razonar y actuar de forma autónoma, especialmente al procesar grandes volúmenes de información que resultan difíciles de manejar para una persona. El reconocimiento de imágenes es una de las ramas principales de la visión artificial, se refiere a la capacidad de un sistema para identificar y clasificar objetos dentro de una imagen. Estas tecnologías permiten crear sistemas que comparan imágenes de animales y detectan similitudes, lo que resulta especialmente útil para localizar perros perdidos.

2.2.1. Redes neuronales convolucionales

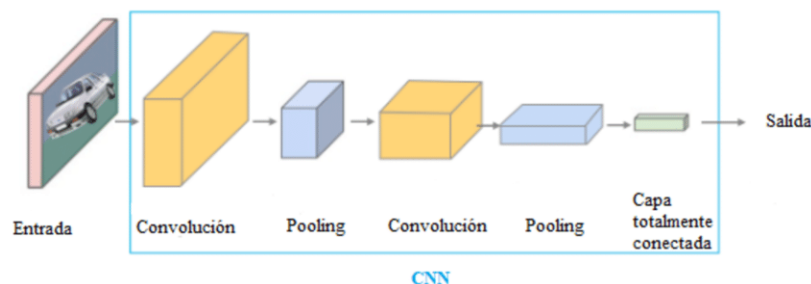
Las redes neuronales convolucionales (CNN) son una variante de las redes neuronales artificiales, son especialmente eficaces en el análisis y procesamiento de información visual, como imágenes y videos. Gracias a su estructura jerárquica, pueden aprender de forma automática patrones relevantes, lo que ha transformado tareas como el reconocimiento y clasificación de imágenes, incluyendo aquellas que contienen animales. Según Taye (2023), las

CNN están compuestas por múltiples capas que permiten detectar, segmentar y clasificar elementos dentro de una imagen sin intervención humana. Durante el entrenamiento, ajustan sus parámetros con base en los datos de entrada, lo que les permite extraer características visuales sin necesidad de definirlas previamente.

Como se muestra en la Figura 9, los componentes principales de una CNN incluyen:

- **Capa de entrada:** Una imagen digital, que sirve como entrada del modelo, está compuesta por píxeles organizados en una matriz. En imágenes en escala de grises, cada píxel tiene un valor entre 0 y 255 que indica su brillo, mientras que en imágenes a color RGB cada píxel contiene tres valores que representan la intensidad de rojo, verde y azul. Estos valores combinados determinan el color y la intensidad de cada punto de la imagen.
- **Capa de convolución:** aplica filtros (*kernels*) que se ajustan en el entrenamiento, para extraer características visuales como bordes o formas.
- **Capa de pooling:** reduce la dimensionalidad de los mapas de características, conservando la información más relevante mediante técnicas como max pooling, min pooling o average pooling dentro de cada subregión.
- **Función de activación:** determina si se debe o no activar una neurona en respuesta a una entrada determinada generando la salida correspondiente.
- **Capa totalmente conectada:** conecta cada neurona con todas las neuronas de las capas anterior y posterior. Recibe como entrada un vector aplanado y se encarga de realizar la clasificación.
- **Capa de salida:** Produce el resultado final, como puede ser la categoría de la imagen.

Figura 9. Descripción del funcionamiento de una red neuronal convolucional



Fuente: (Andrade, 2021)

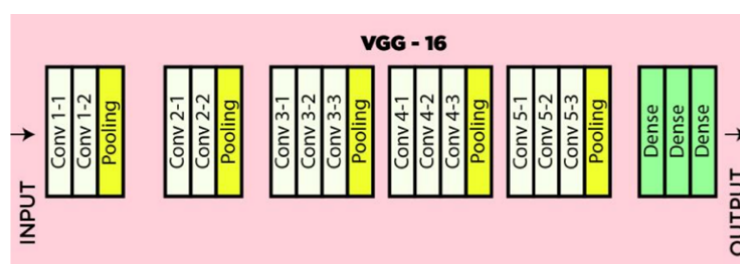
2.2.2. Arquitecturas CNN tradicionales para la identificación de imágenes

Esta sección analiza las características principales de las arquitecturas CNN tradicionales.

2.2.2.1. Visual Geometry Group (VGG).

La arquitectura VGG es una de las más reconocidas por su diseño simple y eficaz. Desarrollada por el Visual Geometry Group de la Universidad de Oxford, su estructura modular está basada en la repetición de bloques de tres capas convolucionales seguidas por capas de max pooling. Según Lorient (2022), esta organización facilita la escalabilidad del modelo, permitiendo variantes como VGG-11, VGG-13, VGG-16 y VGG-19, según el número de capas utilizadas. A pesar de su efectividad, VGG presenta algunas limitaciones: su alta demanda computacional, el gran número de parámetros que la hacen poco adecuada para entornos con recursos limitados. Además, puede presentar problemas de sobreajuste, lo que afecta la capacidad de generalización. La Figura 10 ilustra la arquitectura de VGG-16, mostrando las capas convolucionales, de agrupamiento y totalmente conectadas, lo que ayuda a comprender su funcionamiento.

Figura 10. Ejemplo de red VGG 16



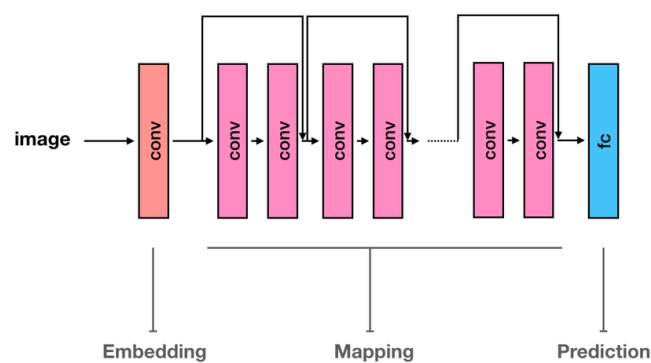
Fuente: (Lorient, 2022)

2.2.2.2. Residual Networks (ResNet)

ResNet es una familia de arquitecturas de redes neuronales convolucionales que se destaca por introducir conexiones residuales, lo que permite entrenar redes muy profundas, incluso con cientos de capas, sin que se presente el problema del desvanecimiento del gradiente. Como señalan Binta Islam et al. (2023), estas conexiones permiten que la información se salte determinadas capas y se transmita directamente desde etapas iniciales hasta capas más profundas, manteniendo el flujo de datos y mejorando el aprendizaje.

Según He et al. (2015), el número de parámetros de la arquitectura varía según su versión: ResNet-50 tiene 25.6 millones, ResNet-101 alcanza los 44.5 millones y ResNet-152, los 60.2 millones. La Figura 11 muestra la arquitectura de ResNet, donde se observan los principales componentes y la disposición de las capas, incluida la integración de las conexiones residuales que favorecen el entrenamiento eficiente de redes profundas.

Figura 11. Arquitectura ResNet.



Fuente: (Rousseau, 2020)

2.2.3. MobileNetV2

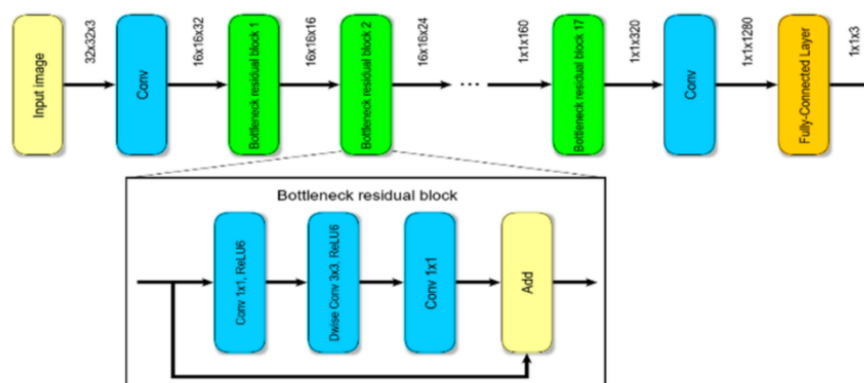
En aplicaciones que requieren procesar imágenes en dispositivos con recursos limitados como teléfonos móviles o sistemas embebidos es crucial emplear arquitecturas de redes neuronales que equilibren precisión y eficiencia computacional. Según Lorient (2022), arquitecturas como VGG y ResNet han demostrado ser altamente precisas en la clasificación de perros con tasas de acierto de hasta un 90%. Sin embargo, su elevado consumo de memoria y procesamiento dificulta su implementación en dispositivos móviles.

Sandler et al. (2018) explican que, aunque las redes neuronales han alcanzado niveles de precisión superiores al desempeño humano en tareas complejas, esa exactitud suele exigir grandes cantidades de recursos computacionales, lo que limita su uso en sistemas con capacidades restringidas. Es así como surgen las arquitecturas MobileNet, una familia de CNNs eficientes desarrolladas por Google, que incluye varias versiones como MobileNetV1, MobileNetV2 y MobileNetV3. Su rendimiento ha sido comparado con el de arquitecturas más complejas. Según Bouraya y Belangour (2024), MobileNetV2 alcanza una precisión del 94.68%, mientras que MobileNetV3Small y MobileNetV3Large logran 96.85% y 97.53% respectivamente.

Como se observa en la Figura 12, los principales componentes de MobileNetV2 son los siguientes:

- **Convoluciones separables en profundidad:** dividen la operación de convolución estándar en dos pasos. Primero, una convolución en profundidad aplica un filtro distinto a cada canal de la imagen. Luego, una convolución 1x1 combina la información entre canales. Esta técnica reduce el costo computacional hasta nueve veces, con una pérdida mínima de precisión.
- **Cuellos de botella lineales:** son las capas estrechas, tanto a la entrada como a la salida de los bloques residuales invertidos, lo que disminuye el costo computacional.
- **Bloques residuales invertidos:** inspirados en ResNet, conectan los cuellos de botella lineales para reducir el uso de memoria durante la inferencia. Según Awofeso (2024), los bloques constan de tres capas secuenciales: la primera es una capa de expansión que aumenta el número de canales mediante una convolución 1x1, la segunda es una convolución separable en profundidad que extrae características y la tercera es una capa de proyección que reduce la dimensión usando una activación lineal que conserva la información relevante.

Figura 12. Arquitectura de MobileNetV2.



Fuente: (Seidaliyeva, Akhmetov, Ilipbayeva, & Matson, 2020)

De acuerdo con los análisis realizados por Howard et al. (2019) sobre MobileNetV1 y por Sandler (2018) sobre MobileNetV2, se menciona que MobileNetV1 cuenta con 4.2 millones de parámetros, mientras que MobileNetV2 reduce esta cifra a 3.4 millones. Esta disminución representa una mejora importante en la eficiencia estructural del modelo, ya que mantiene

un buen rendimiento con un menor costo computacional. Ambas versiones están diseñadas para funcionar en dispositivos con recursos limitados, como teléfonos móviles, y la optimización lograda en MobileNetV2 ha favorecido su uso en aplicaciones que requieren un equilibrio entre precisión, velocidad y eficiencia.

2.2.4. Comparativa entre VGG, ResNet y MobileNetV2

Con base en el análisis previo de las arquitecturas VGG, ResNet y MobileNetV2, en la Tabla 2 presenta una comparación detallada entre estas redes neuronales convolucionales.

Tabla 2. Comparativa entre VGG, ResNet, MobileNetV2

Características	VGG	ResNet	MobileNetV2
Arquitectura	Capas convolucionales 3×3 combinadas con capas de max pooling	Bloques residuales con conexiones de atajo (skip connections)	Bloques residuales invertidos y convoluciones separables en profundidad
Uso de memoria	Alto consumo	Consumo moderado	Bajo consumo
Aplicaciones	Clasificación de imágenes en sistemas con recursos amplios	Clasificación, detección y segmentación de alta precisión	Aplicaciones móviles y dispositivos embebidos con recursos limitados
Ventajas	Arquitectura simple y escalable (VGG-11, VGG-13, VGG-16 y VGG-19)	Facilita el entrenamiento de redes profundas sin desvanecimiento del gradiente	Alta eficiencia en entornos limitados, buen balance entre precisión y consumo de recursos
Desventajas	Requiere alto poder de cómputo, riesgo de sobreajuste	Estructura más compleja	Menor precisión que modelos más grandes, menor capacidad para ajuste fino
Optimizado para móviles	No adecuada	Poco adecuada	Altamente adecuada
Parámetros	VGG-16: 138 millones VGG-19: 144 millones	ResNet-50: 25.6 millones ResNet-101: 44.5 millones ResNet-152: 60.2 millones	MobileNet-v1: 4.2 millones MobileNet-v2: 3.4 millones

2.2.5. Análisis de proyectos de código abierto en GitHub

Para comprender el estado actual y las tendencias en el reconocimiento de perros mediante imágenes, se realizó una revisión de proyectos disponibles en GitHub. Este análisis permitió identificar diferentes enfoques, herramientas y soluciones implementadas en el ámbito del reconocimiento de perros.

2.2.5.1. Proyectos de clasificación de razas de perros.

Se identificaron proyectos enfocados en clasificar las imágenes de perros según su similitud con distintas razas, es decir, su objetivo es predecir a qué raza se asemeja más el perro.

- [Dog-Breed-Identifier](#): proyecto que utiliza un modelo preentrenado ResNet50V2 para clasificación de razas.
- [Dog-Breed-Image-Classification](#): proyecto que implementa un modelo ResNet-50 para clasificar las imágenes considerando 70 razas de perros.

2.2.5.2. Proyectos de aplicaciones web y móviles para la búsqueda de perros perdidos.

Esta categoría abarca proyectos que ayudan a localizar perros perdidos mediante alertas, fotografías y geolocalización, pero no emplean algoritmos de reconocimiento de imágenes.

- [PetFindr](#): aplicación web diseñada para facilitar el reporte de perros perdidos.
- [Lost-Hound](#): aplicación móvil diseñada para reunir perros perdidos con sus dueños

2.2.5.3. Proyecto de reconocimiento de imágenes de perros.

Durante la búsqueda de proyectos diseñados específicamente para el reconocimiento de imágenes de perros, se identificaron los siguientes ejemplos relevantes:

- [Pets-Face-Recognition](#): proyecto que implementa técnicas de reconocimiento facial para identificar tanto perros como gatos. Sin embargo, dado que el enfoque de este trabajo está centrado exclusivamente en perros, su utilidad para este estudio fue limitada.
- [DogFaceNet](#): proyecto que resultó interesante y útil. Es un modelo no preentrenado basado en la función de pérdida por tripletas (triplet loss), la misma que emplea FaceNet, desarrollado por Google para el reconocimiento, verificación y agrupamiento de rostros humanos. Este método entrena al modelo con tres imágenes: una imagen ancla que sirve como referencia principal, una imagen positiva que representa la misma entidad que la ancla, y una imagen negativa que corresponde a una entidad diferente. Según Sukri y Ruhaiyem (2019), el objetivo es reducir la distancia entre el ancla y la imagen positiva y aumentar la distancia respecto a la imagen negativa. Esto implica que cada imagen facial se transforma en un vector o *embedding*, y la similitud se determinan comparando las distancias entre estos vectores, siendo una distancia más corta entre el ancla y el positiva una mayor probabilidad de coincidencia. Según Mougeot et al. (2019), creadores de DogFaceNet, su modelo logra un 92% de precisión en verificación y un 88% en reconocimiento con una sola muestra, usando un conjunto abierto de 48 perros. Además, puede agrupar imágenes de perros desconocidos. Aunque DogFaceNet fue considerado

para este TFG, las razones por las que no se eligió se detallan en la sección Justificación de la elección de MobileNetV2.

2.2.6. Extracción de características con modelos preentrenados

Según Pietikäinen y Silven (2022) en el pasado, la extracción de características se realizaba mediante algoritmos estadísticos o la aplicación manual de filtros sobre los datos de entrada, que luego se usaban en etapas posteriores del procesamiento. Sin embargo, con el avance del aprendizaje automático y profundo, el uso de redes neuronales ha revolucionado este proceso, aumentando notablemente la cantidad y calidad de datos procesables.

De acuerdo con Sundeep et al. (2023), la extracción de características consiste en transformar datos crudos, como los píxeles de una imagen, en vectores de características o embeddings. Estos vectores son representaciones de menor dimensión que capturan los patrones y las relaciones presentes en los datos.

Entre las técnicas más comunes para obtener estos vectores de características se encuentran:

- El uso de redes neuronales convolucionales, cuyas capas finales generan representaciones complejas y altamente útiles para distintas aplicaciones.
- Técnicas de aprendizaje no supervisado, como los autoencoders, que permiten extraer embeddings sin necesidad de etiquetas ni supervisión directa.
- Modelos preentrenados como VGG, ResNet y MobileNetV2 que, al eliminar su capa de clasificación, funcionan como extractores de características potentes. Además, pueden ajustarse mediante fine-tuning para tareas específicas y así generar representaciones más especializadas.

En particular, en el estudio de Elharrouss et al. (2024) se destaca que las CNN actúan como una columna vertebral en muchos modelos de aprendizaje profundo, permitiendo la extracción automática de características relevantes sin necesidad de diseñar filtros manualmente. Esto ha impulsado mejoras en la precisión y eficiencia en áreas como el reconocimiento de imágenes y el procesamiento del lenguaje natural.

2.2.7. Optimización de la búsqueda de similitud

La búsqueda tradicional de imágenes similares requiere comparar cada nuevo vector con todos los existentes por medio de fuerza bruta, lo cual resulta poco eficiente en bases de datos grandes. Para solucionar este problema, se usan algoritmos de búsqueda de vecinos más cercanos aproximados (ANN). Según Hyvönen et al. (2024), estos algoritmos permiten identificar rápidamente los k elementos más cercanos a un vector de consulta, operando en tiempos sublineales. A diferencia de la búsqueda exacta, los métodos ANN aceptan un pequeño margen de error a cambio de una mejora considerable en velocidad. Entre las técnicas más comunes se encuentran estructuras de indexación como árboles, funciones hash, cuantificación de vectores y grafos.

2.2.7.1. Facebook AI Similarity Search

Facebook AI Similarity Search, conocido como **FAISS**, es una librería diseñada para realizar búsquedas eficientes de similitud y agrupamiento en vectores densos. Según su página oficial (2025), es capaz de manejar conjuntos de vectores de cualquier tamaño, incluso aquellos que superan la capacidad de la memoria RAM, y ofrece herramientas para ajustar parámetros y evaluar el rendimiento. De acuerdo con el sitio My Scale (2024), FAISS utiliza técnicas de ANN mediante la creación de índices que organizan los vectores, lo cual evita la comparación exhaustiva con cada elemento. En lugar de una búsqueda lineal $O(N)$, FAISS consulta un índice preconstruido que reduce significativamente el número de comparaciones, enfocándose en los candidatos más relevantes. Además, el índice puede guardarse en disco, evitando su reconstrucción constante y facilitando tareas de registro masivo de imágenes. Estas características hacen de FAISS una herramienta sumamente eficaz para sistemas que requieren búsquedas rápidas y precisas sobre grandes volúmenes de datos vectoriales.

2.2.8. Principales desafíos en el reconocimiento de imágenes de perros

El reconocimiento de imágenes de perros mediante inteligencia artificial enfrenta varios desafíos importantes:

- **Variabilidad de las diferentes razas de perros:** los perros son una de las especies con mayor variabilidad morfológica. Según Boyko y Quignon (2010) existen genes que regulan cerca de 100 rasgos morfológicos en perros, como son el tamaño, el pelaje, el color, entre

otros. Esta amplia variabilidad, complica el reconocimiento de imágenes ya que las características pueden cambiar considerablemente entre diferentes ejemplares.

- **Perros mestizos:** Los perros que resultan de la cruce entre varias razas tienen características físicas mezcladas que pueden asemejarse a distintas razas, complicando su identificación exacta.
- **Calidad y condiciones de la imagen:** Se prevé que, en el futuro, el modelo se utilice junto con aplicaciones móviles en escenarios reales, donde las imágenes suelen ser capturadas por usuarios con teléfonos móviles en exteriores y bajo condiciones variables de iluminación, ángulo y fondo. Estas variaciones representan un desafío para que el algoritmo mantenga una precisión adecuada en el reconocimiento.

3. Objetivos y metodología de trabajo

En el siguiente apartado, se presentan los objetivos del proyecto y la metodología utilizada para implementar el modelo de reconocimiento de perros perdidos.

3.1. Objetivo general

La pérdida de perros representa un problema emocional y social importante, especialmente en México, donde la cantidad de perros en situación de calle y la limitada eficiencia de los métodos actuales dificultan el reencuentro entre mascotas y sus dueños. Por ello, es necesario desarrollar soluciones tecnológicas que optimicen y automaticen el proceso de búsqueda, mejorando las probabilidades de éxito y reduciendo el tiempo de respuesta.

Por tanto, el objetivo general de este trabajo es desarrollar un sistema para la identificación de imágenes de perros, optimizado para entornos con recursos limitados.

3.2. Objetivos específicos

Para alcanzar el objetivo general, se definen los siguientes objetivos específicos:

1. Analizar y seleccionar la arquitectura de modelo de reconocimiento de imágenes más adecuada para la extracción de características justificando la elección en base a criterios de precisión, eficiencia y compatibilidad considerando las limitaciones propias de entornos con recursos limitados.
2. Recopilar y conformar un conjunto de datos de imágenes de perros diverso y representativo a partir de múltiples fuentes.
3. Diseñar e implementar una arquitectura de software modular y mantenible que estructure los componentes del sistema para facilitar su desarrollo, escalabilidad y futura evolución.
4. Implementar el modelo de reconocimiento de imágenes seleccionado como un extractor de características, asegurando el preprocesamiento adecuado y la generación de embeddings normalizados.
5. Diseñar e implementar un sistema robusto de gestión de base de datos para el almacenamiento persistente y el registro automático de imágenes de perros.

6. Desarrollar e integrar un mecanismo de detección de duplicados optimizado en el proceso de registro.
7. Desarrollar un módulo de búsqueda que permita identificar imágenes similares de manera rápida y eficiente en grandes volúmenes de datos, utilizando técnicas avanzadas de indexación y consulta.
8. Evaluar y validar el rendimiento global del sistema para justificar la efectividad de los mecanismos de búsqueda de similitud y detección de duplicados, mediante la aplicación de un protocolo de pruebas y el análisis de métricas de desempeño.

3.3. Metodología de trabajo

Este apartado describe las fases principales para desarrollar el sistema de búsqueda de similitud de imágenes de perros perdidos.

Fase 1. Análisis y planificación (3 semanas): estudio del problema y de las soluciones existentes, incluyendo una comparación de modelos de inteligencia artificial para reconocimiento de imágenes. Definición de objetivos y requisitos del sistema (funcionales y no funcionales). Inicio de la redacción de capítulos introductorios del TFG.

Fase 2. Recopilación y preparación de datos (4 semanas): Recolección de imágenes de perros desde diversas fuentes. Se establecerán criterios para asegurar variabilidad en pose, iluminación y entorno, y se definirá un ground truth *inicial* para futuras evaluaciones.

Fase 3. Diseño e implementación de arquitectura (3 semanas) : Planificación de la estructura del sistema y organización modular del código para facilitar su mantenimiento y escalabilidad.

Fase 4. Implementación del modelo de extracción de características (3 semanas): Configuración e implementación de un modelo preentrenado basado en redes neuronales convolucionales. Incluye la preparación de imágenes y la generación de embeddings normalizados.

Fase 5. Diseño e implementación de la base de datos (3 semanas): Planificación, diseño e implementación de una base de datos relacional para almacenar imágenes y vectores de características. Se implementará un sistema eficiente para el registro automático.

Fase 6. Desarrollo e integración del mecanismo de detección de duplicados (3 semanas):

Implementación y optimización de un sistema que prevenga la inserción de imágenes duplicadas o muy similares, asegurando la calidad y unicidad de los datos.

Fase 7. Desarrollo e implementación del módulo de búsqueda de similitud (4 semanas):

Desarrollo e integración del componente principal del sistema, basado en técnicas de indexación que permitan identificar rápidamente imágenes similares.

Fase 8. Pruebas y evaluación (4 semanas): Verificación exhaustiva de la funcionalidad del sistema y evaluación cuantitativa del rendimiento (latencia, escalabilidad, precisión) de los mecanismos de búsqueda de similitud y detección de duplicados. Se documentarán métricas y se justificarán los umbrales definidos.

Fase 9. Documentación final y conclusiones (2 semanas): Sistematización de resultados, análisis de cumplimiento de objetivos, identificación de limitaciones y propuesta de mejoras.

La redacción de la memoria será un proceso continuo durante todo el proyecto, y no se limitará a esta fase.

3.3.1. Cronograma

La Figura 13 muestra el cronograma estimado con la duración y secuencia de las fases.

Figura 13. Cronograma



4. Requisitos del sistema

Este capítulo describe los requisitos funcionales y no funcionales que debe cumplir el sistema, los cuales son clave para guiar el desarrollo, asegurar que se alcancen los objetivos del proyecto y definir cómo se evaluará su rendimiento.

4.1. Requisitos Funcionales

En este apartado se describen las funciones específicas que el sistema debe cumplir para satisfacer las necesidades de los usuarios y garantizar el correcto desempeño de sus operaciones.

RF1: Registro automático de imágenes: El sistema debe ser capaz de procesar un directorio o una lista de directorios de imágenes de perros.

- **RF1.1: Validación de formato:** Debe verificar que los archivos sean formatos de imagen válidos (JPG, JPEG, PNG).
- **RF1.2: Extracción de características:** Debe extraer vectores de características (embeddings) de cada imagen utilizando un modelo preentrenado.
- **RF1.3: Almacenamiento en base de datos:** Debe almacenar los datos de las imágenes como ruta, nombre y características en una base de datos persistente.

RF2: Detección de duplicados: El sistema debe prevenir la inserción de imágenes duplicadas en la base de datos.

- **RF2.1: Verificación por ruta:** Debe omitir la adición de imágenes cuya ruta de archivo ya exista en la base de datos.
- **RF2.2: Verificación por similitud de características (preexistentes):** Debe identificar y omitir la adición de imágenes que sean visualmente idénticas a alguna ya almacenada en la base de datos, aún si la ruta o el nombre son diferentes.
- **RF2.3: Verificación por similitud de características (en lote):** Debe identificar y omitir la adición de imágenes que sean idénticas a alguna imagen ya procesada y aceptada dentro del mismo lote de adición.

- **RF2.4: Umbral de Duplicados:** Debe utilizar un umbral de similitud establecido para determinar si una imagen es un duplicado por características.

RF3: Búsqueda de perros similares: El sistema debe permitir al usuario buscar perros visualmente parecidos a una imagen de consulta.

- **RF3.1: Entrada de consulta:** Debe aceptar una imagen como consulta para la búsqueda.
- **RF3.2: Recuperación de resultados:** Debe devolver una lista de perros almacenados en la base de datos cuya similitud visual con la imagen de consulta supere el umbral definido.
- **RF3.3: Ordenamiento de resultados:** Los resultados de la búsqueda deben ser presentados ordenados de mayor a menor similitud.
- **RF3.4: Umbral de similitud para búsqueda:** Debe utilizar un umbral de similitud establecido para filtrar los resultados de la búsqueda.

RF4: Gestión del índice de búsqueda: El sistema debe manejar el índice de similitud para asegurar búsquedas eficientes.

- **RF4.1: Construcción del índice:** Debe ser capaz de construir un índice de búsqueda de similitud a partir de las características de las imágenes almacenadas.
- **RF4.2: Persistencia del índice:** Debe guardar el índice de búsqueda en disco para evitar reconstrucciones costosas en cada inicio.
- **RF4.3: Carga del índice:** Debe cargar el índice de búsqueda desde disco a la memoria
- **RF4.4: Sincronización del índice:** Debe asegurar que el índice de búsqueda se mantenga sincronizado con los cambios en la base de datos.

4.2. Requisitos no funcionales.

Definen atributos de calidad y restricciones operativas que el sistema debe cumplir para ser considerado exitoso.

RNF1: Rendimiento: El sistema debe ser capaz de realizar búsquedas de similitud en grandes volúmenes de datos en un tiempo aceptable para aplicaciones interactivas.

- **RNF1.1: Latencia de búsqueda:** La latencia para una consulta de búsqueda de similitud no debe exceder 0.005 segundos para una base de datos con 20,000 imágenes.

- **RNF1.2: Escalabilidad del registro:** El proceso de registro de nuevas imágenes debe ser escalable, permitiendo la adición de 20,000 imágenes en 20 minutos.

RNF2: Precisión (exactitud de similitud): La búsqueda de similitud debe arrojar resultados relevantes de acuerdo con el Ground Truth definido.

- **RNF2.1: Precisión en detección de duplicados:** La detección de duplicados debe alcanzar una precisión de al menos 98%.
- **RNF2.2: F1-score en búsqueda de similitud:** La función de búsqueda de similitud debe alcanzar un F1-score promedio de al menos 0.85 en el conjunto de prueba de Ground Truth.

RNF3: Mantenibilidad: El código fuente debe ser modular, legible y estar documentado para facilitar futuras modificaciones y mejoras.

RNF4: Disponibilidad de datos: El sistema debe poder acceder a los datasets de imágenes desde ubicaciones definidas

RNF5: Persistencia.

- **RNF5.1: Almacenamiento de datos:** La base de datos debe asegurar la persistencia de la información de los perros entre ejecuciones.
- **RNF5.2: Persistencia del índice:** El índice de búsqueda debe ser guardado y cargado para evitar reconstrucciones costosas en cada inicio.

RNF6: Optimización para entornos de recursos limitados: El sistema debe utilizar los recursos computacionales de forma eficiente para que pueda implementarse en dispositivos con capacidad limitada de procesamiento y memoria.

- **RNF6.1: Huella de memoria del modelo:** El modelo de extracción de características debe tener una huella de memoria inferior a 150 MB en RAM, adecuada para entornos con memoria limitada.
- **RNF6.2: Eficiencia de inferencia del modelo:** El tiempo de extracción de características por imagen debe ser bajo en CPU, no excediendo los 50 ms por imagen.
- **RNF6.3: Tamaño del modelo:** El tamaño del modelo de extracción de características en disco debe ser compacto para facilitar su distribución, inferior a 25 MB.

5. Implementación del sistema de similitud de imágenes

Este apartado describe las etapas del proyecto, desde el análisis y elección del modelo de reconocimiento de imágenes, hasta la implementación de la base de datos y los mecanismos de búsqueda por similitud.

5.1. Evaluación y elección del modelo de reconocimiento de imágenes

Se compararon dos modelos de reconocimiento de imágenes de perros para determinar cuál era el más adecuado, considerando las limitaciones del proyecto: tiempo, recursos computacionales y nivel técnico.

5.1.1. Análisis de DogFaceNet

DogFaceNet, diseñado específicamente para el reconocimiento facial de perros, fue una opción considerada. No obstante, tras revisar en detalle su documentación y los repositorios disponibles, se identificó como una limitación importante el hecho de que no cuenta con un modelo previamente entrenado. Esto implicaría entrenarlo desde cero, lo cual requiere una alta capacidad computacional y una inversión significativa de tiempo para realizar pruebas, ajustes y validaciones. Además, el nivel técnico necesario para entrenar y optimizar DogFaceNet excede los conocimientos adquiridos en una formación de grado en informática, resultando más apropiado para estudios de posgrado o máster en inteligencia artificial.

5.1.2. Justificación de la elección de MobileNetV2

MobileNetV2 fue seleccionado debido a que cuenta con un modelo preentrenado con el conjunto de datos ImageNet, que incluye aproximadamente 14 millones de imágenes. Esto permite utilizarlo como un extractor de características eficaz sin necesidad de entrenarlo desde cero, lo que reduce considerablemente el tiempo y los recursos requeridos. Si bien no está especializado en perros, su capacidad para identificar patrones visuales generales es suficiente para tareas de comparación de imágenes. Además, está optimizado para funcionar en dispositivos móviles, lo cual es especialmente relevante considerando que, tras el TFG, se prevé el desarrollo de una aplicación móvil.

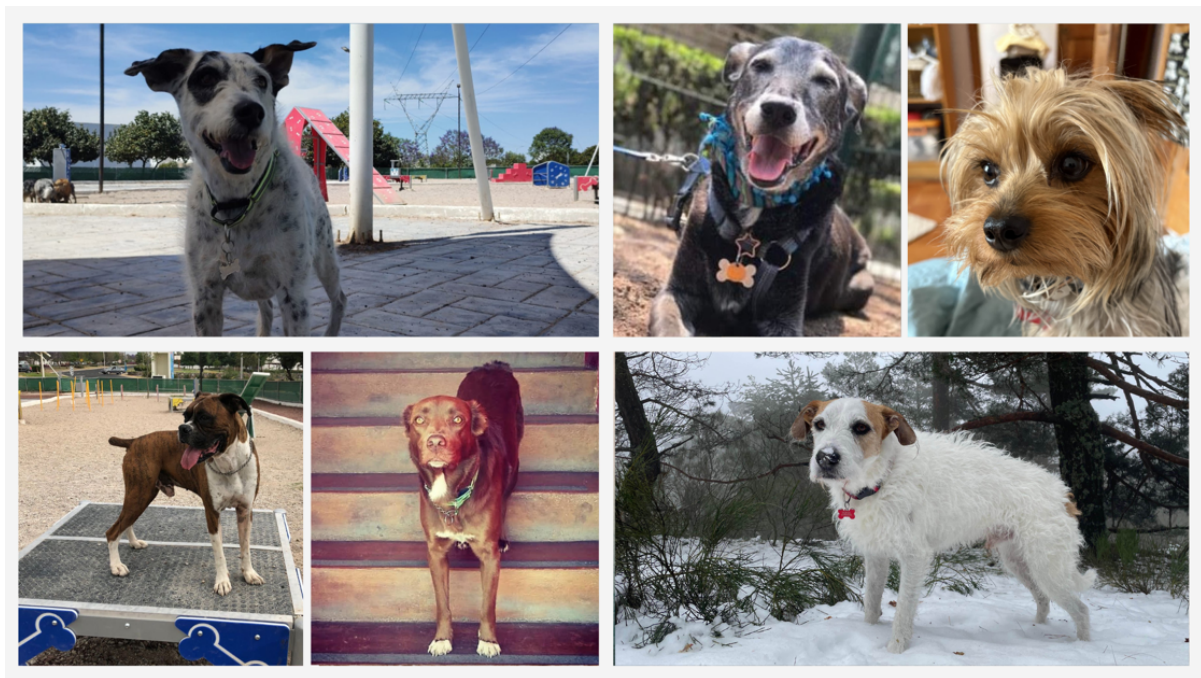
Aunque DogFaceNet representa una alternativa enfocada específicamente en reconocimiento facial canino, sus altas exigencias técnicas y la necesidad de entrenarlo desde cero lo hacen

poco viable para este proyecto. En contraste, MobileNetV2 ofrece un equilibrio adecuado entre precisión, simplicidad y eficiencia, lo que lo convierte en la opción más apropiada para esta implementación.

5.2. Recolección y preparación del dataset de imágenes

Para evaluar la efectividad del sistema en diferentes condiciones, se conformó un conjunto de datos sólido y variado que incluye imágenes propias y datasets públicos. Las imágenes fueron seleccionadas considerando distintas condiciones de iluminación, poses, fondos y calidad, con el fin de simular escenarios reales. En la Figura 14 se muestran ejemplos representativos de las imágenes de los datasets empleados.

Figura 14. Ejemplos de imágenes de diferentes datasets utilizados.



Fuente: Dataset Dog_Mx_Dataset, dataset Stanford Dogs Dataset

El conjunto de datos utilizados fue:

- **Dog_Mx_Dataset:** 314 imágenes recopiladas de perros de amigos y familiares, con entre 3 y 5 fotos por mascota. Incluye variaciones en ángulos y entornos. Se agregó intencionalmente una imagen duplicada, por lo que hay 313 imágenes únicas.
- **Stanford Dogs Dataset:** conjunto público con 20,580 imágenes clasificadas en 120 razas (Aditya, Nityananda, Bangpeng, & Li, 2011).

- **NonDogs:** contiene solo 2 imágenes (una manzana y un gato), utilizadas para validar la precisión del sistema ante entradas no caninas.
- **Duplicate_test_subset:** contiene 20 imágenes divididas en 6 originales, 6 duplicadas exactas, 6 similares y 2 distintas. Este dataset fue utilizado para pruebas de detección de similitud.
- **Test_images_limitations:** contiene 9 imágenes con condiciones desafiantes (poca luz, baja calidad, pelaje oscuro o perros mestizos), diseñadas para evaluar los límites del modelo.

En la Tabla 3, se resumen las características principales de cada conjunto:

Tabla 3. Descripción de los datasets utilizados para la búsqueda de similitud de imágenes

Dataset	Número de Imágenes	Número de Razas	Fuente / Descripción
Dog_Mx_Dataset	314	N/A	Imágenes recolectadas con amigos y familiares.
Stanford Dogs Dataset	20,580	120	Dataset público con imágenes etiquetadas con razas de perros.
NonDogs	2	N/A	Imágenes que no son perros.
Duplicate_test_subset	20	N/A	Imágenes de originales, duplicadas, similares y únicas.
Test_images_limitations	9	N/A	Imágenes con características desafiantes para el modelo.

En total, los datasets Dog_Mx_Dataset y Stanford Dogs Dataset suman aproximadamente 20,894 imágenes, proporcionando una base robusta y representativa. Los otros tres conjuntos fueron creados para realizar pruebas específicas que permitan analizar el comportamiento del sistema frente a retos como duplicidad, similitud parcial o entradas irrelevantes.

5.2.1. Control y depuración del conjunto de imágenes

Antes de ingresar las imágenes al modelo de extracción de características, se optó por una estrategia que aprovecha el diseño del sistema para controlar la calidad y eliminar la redundancia de los datos. La fase de preparación de los datasets se complementa con dos mecanismos fundamentales implementados en etapas posteriores del sistema:

- **Preprocesamiento automático por el modelo:** El modelo basado en MobileNetV2 ajusta automáticamente las dimensiones, formato y valores de píxeles de cada imagen para cumplir con los requisitos de entrada, garantizando características consistentes y de alta calidad. Los detalles se explican en la sección Preprocesamiento de imágenes.

- **Detección y prevención automática de duplicados:** En lugar de depurar manualmente, el sistema incluye un módulo que detecta y evita el registro de imágenes idénticas o muy similares durante el almacenamiento, asegurando un dataset único y depurado, incluso si los datos originales contienen duplicados. La implementación de este proceso se describe en la sección Detección de Duplicados

Esta metodología garantiza que la solidez del conjunto de imágenes se logre no solo mediante la variedad de fuentes, sino también gracias a procesos de filtrado y preprocesamiento incorporados directamente en el sistema.

5.3. Estructura modular del desarrollo y organización del código

Con el objetivo de facilitar la escalabilidad, reutilización y mantenimiento del sistema, se adoptó una estructura modular, dividiendo la lógica del proyecto en archivos Python y un notebook principal. Todo el código fuente está disponible públicamente en el repositorio de GitHub del proyecto: <https://github.com/dasniela/TFG-DogFinderV2>

La implementación se compone de los siguientes elementos principales:

- **El notebook:** *main_dog_finder.ipynb*, funciona como el entorno central de pruebas e integración, donde se ejecutan las funcionalidades del sistema y se validan los resultados.
- **El archivo Python:** *faiss_utils.py*, que contiene funciones específicas para la construcción y carga del índice FAISS, la conversión de características a vectores, el guardado y recuperación del índice, y el manejo del mapeo de identificadores. Este módulo se encarga de toda la lógica relacionada con la búsqueda de similitud entre imágenes a nivel de vectores.
- **El archivo Python:** *feature_extraction.py*, donde se concentran las funciones responsables de cargar el modelo de reconocimiento de imágenes, preparar los datos de entrada, extraer las características (embeddings) de cada imagen, y normalizar los vectores resultantes. Este archivo gestiona todo el procesamiento visual basado en el modelo previamente elegido.

- **El archivo Python: *memory_usage_profile.py***, permite medir el consumo de memoria durante la inferencia, lo cual resulta útil para evaluar el rendimiento del sistema en entornos con recursos limitados.

Esta estructura modular permite una separación clara entre la lógica funcional del sistema, implementada en módulos reutilizables, y el entorno de pruebas, contenido en el notebook. Gracias a esta organización, el proyecto puede ampliarse o modificarse sin comprometer su estabilidad.

5.4. Implementación del modelo de reconocimiento

Para implementar el modelo de reconocimiento se optó por MobileNetV2, una arquitectura eficiente preentrenada con el dataset ImageNet. El objetivo fue usarla como extractor de características, sin necesidad de entrenarla desde cero. Este enfoque reduce significativamente los recursos computacionales y el tiempo requerido, lo que resulta ideal para proyectos con alcance limitado como un TFG.

5.4.1. Preparación del entorno y bibliotecas

Se utilizó Python junto con el framework TensorFlow y su API de alto nivel, Keras. Estas herramientas facilitan el desarrollo de redes neuronales, incluso para usuarios sin conocimientos avanzados en inteligencia artificial. Además, se emplearon librerías clave como son:

- **numpy** para manejo de datos numéricos
- **OpenCV** para el procesamiento de imágenes
- **FAISS** para la búsqueda eficiente de similitud entre vectores,
- **sqlalchemy, datetime, json y matplotlib** para tareas de base de datos, gestión de datos y visualización.

5.4.2. Uso de MobileNetV2 como extractor de características

Para adaptar MobileNetV2 a esta tarea se siguieron los siguientes pasos:

1. **Congelamiento de capas:** se desactivó el entrenamiento de las capas preentrenadas para preservar los conocimientos adquiridos con ImageNet.

2. **Eliminación de la capa de clasificación:** se retiró la última capa, que originalmente predecía clases, para usar únicamente el modelo como extractor de características.
3. **Obtención de vectores (embeddings):** se generó un vector por imagen, que resume su contenido visual de forma numérica.
4. **Carga de pesos preentrenados:** se reutilizaron los pesos ya aprendidos por el modelo para detectar patrones visuales generales como formas y texturas.

5.4.3. Preprocesamiento de imágenes

Antes de alimentar al modelo con una imagen, es necesario preprocesarla para cumplir con sus requisitos específicos y asegurar resultados precisos. Cada arquitectura puede requerir un preprocesamiento distinto, por lo que usar uno genérico puede causar errores.

Este proceso incluyó:

1. Cargar la imagen con OpenCV.
2. Convertirla de formato BGR a RGB, que es el formato que espera MobileNetV2.
3. Redimensionar la imagen a 224x224 píxeles.
4. Convertir a float32 para cálculos en TensorFlow.
5. Añadir una dimensión para representar el lote, aunque solo se procese una imagen. Como se observa en la Figura 15, el array inicialmente contiene la altura, el ancho y los tres canales de la imagen. Al añadir la dimensión en la posición cero, se representa el tamaño del lote, que en este caso es de un solo elemento.

Figura 15. Expansión de dimensiones para el lote

$$(224, 224, 3) \rightarrow (1, 224, 224, 3)$$

tamaño del lote
 altura
 ancho
 canales

6. Aplicar la normalización específica para MobileNetV2, asegurando que los valores de los píxeles estén en el rango esperado por el modelo.

Estos pasos garantizan que las imágenes estén en el formato correcto para extraer información visual de forma precisa.

5.4.4. Extracción de características

Con la imagen ya preprocesada, se obtiene el embedding, es decir, un vector que representa sus características más relevantes. Para facilitar comparaciones, se normaliza este vector dividiéndolo por su norma L2, lo que garantiza una magnitud uniforme en todos los casos. Este vector será la base para buscar las similitudes entre las imágenes de perros.

5.5. Implementación de la base de datos

Para organizar la información de los perros registrados y facilitar su consulta, el sistema cuenta con una base de datos sencilla y funcional. Esta permite almacenar automáticamente los datos generados al procesar cada imagen.

5.5.1. Elección de tecnología

Para el desarrollo del sistema se eligió SQLite como sistema de gestión de base de datos, en conjunto con la librería SQLAlchemy. SQLite no requiere instalación ni configuración de un servidor, ya que almacena toda la información en un único archivo con extensión .db, lo que facilita su configuración y le confiere alta portabilidad. Además, permite almacenar estructuras complejas como vectores de características en formato JSON, aunque no soporta operaciones directas sobre estos datos dentro de la base de datos.

Por su parte, SQLAlchemy es una librería de Python que actúa como un Object-Relational Mapper (ORM), lo que simplifica la interacción con la base de datos al permitir trabajar con objetos de Python en lugar de escribir sentencias SQL.

5.5.2. Estructura de la Base de Datos en SQLite.

La base de datos contiene una sola tabla, llamada dogs, que guarda la información esencial de cada imagen registrada:

- **id:** identificador único.
- **name:** nombre de la imagen.
- **location:** lugar donde fue encontrado el perro.
- **found_date:** fecha en la que fue encontrado el perro.
- **image_path:** ruta local de la imagen.
- **features:** vector de características extraídas (almacenado como JSON).

5.5.3. Creación y configuración de la Base de Datos

Primero se definió una clase base para crear las tablas de la base de datos. Luego se conectó SQLite mediante un motor que genera un archivo local donde se guarda toda la información.

Después, se creó automáticamente la tabla dogs si no existía y se configuró una sesión para interactuar con la base de datos, permitiendo insertar, consultar, actualizar o eliminar datos usando Python, sin escribir SQL directamente.

5.6. Registro automático de imágenes

El sistema permite registrar imágenes de forma automática: cada imagen se procesa, se extraen sus características visuales y se almacena junto con su información. Para asegurar la calidad del registro, el sistema incorpora un mecanismo que evita guardar imágenes duplicadas, lo cual es fundamental para mantener una base de datos limpia y eficiente, especialmente en tareas posteriores como la búsqueda por similitud.

Aunque este mecanismo se usó inicialmente con fines de prueba, está pensado también para entornos reales. Por ejemplo, en un caso de uso orientado al público general, los usuarios cargarían pocas imágenes en cada ocasión. En contraste, para refugios de animales, que gestionan grandes volúmenes, la capacidad de realizar cargas masivas es esencial.

5.6.1. Desafíos encontrados

Durante las primeras pruebas, realizadas con el dataset Dog_Mx_Dataset, compuesto por 314 imágenes el sistema funcionó sin problemas. Sin embargo, al trabajar con datasets más grandes como Stanford Dogs Dataset, surgieron dificultades de rendimiento. Por cada nueva imagen, se comparaban sus características con todas las almacenadas, lo que resultaba lento y poco escalable.

Este enfoque funcionaba bien con pocos datos, pero no era viable cuando el número de imágenes crecía, ya que el tiempo de comparación aumentaba de forma exponencial. Este problema abrió la necesidad de buscar soluciones más eficientes, que se abordan en el siguiente apartado.

5.7. Detección y prevención de duplicados

Este apartado detalla el mecanismo para asegurar que la base de datos no contenga imágenes duplicadas. Se implementó un sistema optimizado que mejora la eficiencia y la escalabilidad del registro automático.

5.7.1. Optimización con FAISS para la detección de duplicados

Para superar las limitaciones del método inicial, se utilizó FAISS, una herramienta especializada en búsquedas rápidas de similitud en grandes colecciones de vectores. En lugar de comparar cada nueva imagen con todas las existentes, FAISS organiza las características almacenadas en un índice que permite encontrar rápidamente imágenes similares.

Este proceso funciona en dos etapas:

1. **Construcción del índice FAISS:** Al iniciar una sesión o proceso de registro, se cargan todas las características previamente almacenadas en la base de datos y se utilizan para construir un índice FAISS en memoria. Este índice actúa como una estructura especializada que permite realizar búsquedas eficientes por similitud. Además, puede ser guardado en disco y reutilizado en futuras sesiones, lo que evita tener que reconstruirlo cada vez.
2. **Detección eficiente de duplicados:** Al registrar una nueva imagen, en lugar de compararla directamente con cada vector de la base de datos, se consulta el índice FAISS. Esta consulta devuelve los vectores más cercanos al nuevo, permitiendo detectar rápidamente si existe una imagen similar ya registrada. De esta forma, el proceso de verificación de duplicados se realiza de manera mucho más ágil y escalable.

5.7.2. Proceso de registro y detección de duplicados

El proceso de registro de imágenes combina la detección de duplicados por ruta de archivo con una verificación de características visuales mediante FAISS.

El procedimiento comienza con la sincronización del sistema con la base de datos existente: se carga un índice de similitud (FAISS) que contiene todas las características de las imágenes ya registradas. Si este índice no existe o está desactualizado, se reconstruye automáticamente, asegurando que esté en consonancia con la base de datos.

Durante el procesamiento de un nuevo conjunto de imágenes, se crea una copia temporal del índice para gestionar el lote actual. Esta copia incluye los vectores existentes y se irá actualizando conforme se añadan nuevas imágenes válidas.

Durante la carga de nuevas imágenes, primero se verifica si la imagen ya está registrada mediante su ruta para evitar repeticiones obvias. Si no está registrada, se extraen sus características y se comparan contra el índice para detectar duplicados visuales en el índice. Cuando una imagen no es duplicada, se guarda su información (incluido el vector de características) en la base de datos. Además, su vector se añade al índice temporal para que pueda ser considerado en las siguientes comparaciones del mismo lote.

Finalmente, en lugar de guardar cada imagen de forma individual, se agrupan los cambios y se confirman todos juntos, lo que mejora el rendimiento general. Si ocurre algún error durante el proceso, todos los cambios se revierten para mantener la base de datos en un estado consistente.

5.8. Implementación de la búsqueda de similitud

La búsqueda de similitud es una función esencial del sistema, diseñada para relacionar una imagen de entrada con aquellas almacenadas en la base de datos y encontrar coincidencias visuales. Para ello, cada imagen se representa mediante un vector numérico que resume sus características más relevantes. El sistema compara este vector con los ya registrados y devuelve solo los resultados que superan un umbral mínimo de similitud.

5.8.1. Fundamentos de la Similitud por Vectores de Características

El cálculo que se empleará para medir la similitud entre dos vectores es la similitud del coseno. Esta métrica evalúa la distancia entre dos vectores en el espacio. Cuanto menor es el ángulo entre ellos, más similares son. Un valor cercano a 1 indica una similitud alta, mientras que un valor cercano a 0 refleja que comparten pocas o ninguna característica visual.

La similitud del coseno entre los vectores A y B, siendo A el vector de la imagen de consulta y B el vector de la imagen a comparar de la base de datos es:

$$\cos(\theta) = \frac{\vec{A} \cdot \vec{B}}{|\vec{A}| |\vec{B}|}$$

Es importante destacar que los vectores de características generados por nuestro modelo son previamente normalizados, por lo que el denominador se convierte en:

$$|\vec{A}||\vec{B}| = 1 \times 1 = 1$$

Por lo tanto, la fórmula de la similitud del coseno se simplifica al producto punto de los vectores normalizados:

$$\cos(\theta) = \vec{A} \cdot \vec{B}$$

Esta simplificación es computacionalmente eficiente y permite que la similitud coseno se interprete directamente como el producto punto entre los vectores aplanados. La decisión del umbral, es decir, qué tan cercano a 1 debe ser un valor para considerar las imágenes similares se justifica en detalle en la sección F1-score en Búsqueda de Similitud y determinación del umbral óptimo.

5.8.2. Optimización de la Búsqueda con FAISS

Para agilizar la búsqueda de imágenes similares en grandes volúmenes de datos, el sistema emplea FAISS. Si bien, en una sección anterior se detalló su uso en la detección de duplicados, aquí se describe su aplicación general en la recuperación eficiente de imágenes similares.

El proceso se organiza en las siguientes etapas clave:

Construcción y persistencia del índice

Cuando el sistema se inicia o detecta cambios importantes en la base de datos, como la incorporación de muchas nuevas imágenes, se extraen los vectores de características de todas las imágenes registradas. Estos vectores son representaciones numéricas de alta dimensión generadas a partir del contenido visual de cada imagen. Para organizarlos de manera eficiente se construye un índice especializado que permite calcular similitudes de forma rápida. Los vectores han sido previamente normalizados para que la métrica utilizada, basada en el producto interno, sea equivalente a la similitud del coseno. Esto permite identificar imágenes similares con alta precisión sin requerir transformaciones adicionales.

Una vez creado, el índice se guarda en disco para poder reutilizarlo en futuras sesiones sin necesidad de reconstruirlo. Junto con este índice se almacena también un archivo adicional que vincula cada vector con su identificador correspondiente en la base de datos. Este vínculo

garantiza que los resultados obtenidos en las búsquedas puedan asociarse fácilmente con la información completa de cada registro.

Carga y Mantenimiento del Índice en Memoria

Para que las búsquedas se realicen de forma inmediata, el índice de similitud y su información asociada se cargan en memoria cada vez que el sistema inicia. Al tener este índice disponible en tiempo real, se evita reconstruirlo para cada consulta, lo que reduce considerablemente el tiempo de respuesta. El sistema también verifica constantemente que el índice esté sincronizado con la base de datos. Si detecta que hay una diferencia entre la cantidad de registros almacenados y los vectores disponibles en el índice, se realiza una reconstrucción automática. Esto asegura que las búsquedas y validaciones se hagan siempre sobre datos actualizados y consistentes.

5.8.3. Proceso de búsqueda de similitud

El sistema cuenta con un mecanismo especializado para encontrar perros visualmente similares a partir de una imagen proporcionada por el usuario. Esta búsqueda, que en condiciones normales sería una tarea intensiva en recursos, se realiza de forma rápida gracias al uso del índice de similitud previamente cargado en memoria.

Flujo de Proceso Detallado:

1. **Extracción de características:** La imagen de consulta se somete al mismo proceso estandarizado de preprocesamiento y extracción de características que se aplica a todas las imágenes almacenadas en la base de datos. Esto asegura que el vector de consulta sea compatible con los vectores en el índice FAISS. El resultado es un vector de características L2 normalizado de 1280 dimensiones
2. **Consulta en el índice FAISS:** Una vez obtenido el vector, se compara contra el índice de similitud previamente cargado en memoria. Este índice permite identificar, de forma eficiente, los vectores más cercanos sin necesidad de revisar todos los registros uno por uno. Los resultados incluyen las puntuaciones de similitud y los índices internos de los vectores encontrados dentro del índice FAISS.
3. **Filtrado por umbral:** Para garantizar la relevancia de los resultados, se aplica un umbral de similitud. Solo se conservan las coincidencias que superan este valor mínimo, mientras

que las demás se descartan por considerarse insuficientemente parecidas. La justificación de este umbral se aborda en la Sección F1-score en Búsqueda de Similitud y determinación del umbral óptimo.

4. **Recuperación de información detallada del perro:** Para cada vector que supera el umbral, se accede a la base de datos mediante un identificador vinculado al vector encontrado. Esto permite recuperar toda la información asociada al perro correspondiente, como su nombre, ubicación, fecha de hallazgo y la ruta de su imagen.
5. **Ordenamiento y preparación de la salida:** Finalmente, los perros identificados como similares se ordenan de mayor a menor según su nivel de similitud. Esta presentación prioriza los resultados más relevantes, facilitando al usuario la identificación de coincidencias visuales cercanas a su imagen de referencia.

En conjunto, la integración de FAISS y el diseño eficiente del índice de similitud permiten que el sistema realice búsquedas visuales de forma ágil, precisa y escalable. Esta capacidad es clave para ofrecer una experiencia de usuario efectiva, tanto en escenarios con pocas imágenes como en contextos donde se manejan grandes volúmenes de datos.

6. Pruebas y evaluación

Este capítulo detalla el proceso de pruebas y evaluación realizado para validar la funcionalidad, el rendimiento y la precisión del sistema. Dada la complejidad y la interconexión de sus componentes, se ha adoptado una estrategia de pruebas agrupadas que permite una verificación eficiente y exhaustiva de los requisitos funcionales y no funcionales. La evaluación busca no solo confirmar que el sistema opera como se espera, sino también cuantificar su desempeño mediante métricas clave. Las pruebas fueron organizadas en un Jupyter Notebook y se estructuran en las siguientes secciones:

1. **Preparación inicial y modelo**: Incluye los pasos necesarios para configurar el entorno, cargar bibliotecas, inicializar el modelo de extracción de características y conectar con la base de datos. Aunque no constituyen pruebas en sí mismas, su correcta ejecución es imprescindible para el resto del proceso.
2. **Pruebas funcionales y carga de datos**: Reúne la validación de varios requisitos clave, incluyendo el registro automático de imágenes (RF1), la detección de duplicados (RF2), la gestión del índice de búsqueda FAISS (RF4), la disponibilidad de datos (RNF5) y la persistencia de la información (RNF6). Las pruebas se realizan de forma secuencial para simular un flujo de trabajo realista y observar la interacción entre los componentes.
3. **Pruebas de búsqueda de similitud**: Evalúa la funcionalidad principal del sistema, centrada en encontrar imágenes de perros visualmente similares. Se valida el ingreso de consultas, la recuperación de resultados, el ordenamiento y la aplicación de umbrales de similitud, como se especifica en RF3.
4. **Evaluación de rendimiento y precisión**: Mide el desempeño del sistema mediante métricas como la latencia de búsqueda y la escalabilidad del registro (RNF1), la precisión en la detección de duplicados (RNF2.1), y el F1-score en la búsqueda de similitud (RNF2.2). También se evalúa la optimización para entornos con recursos limitados (RNF7), aportando evidencia del cumplimiento de atributos de calidad del sistema.

Esta organización permite una comprensión clara del proceso de validación, asegurando que cada aspecto del sistema sea evaluado de manera sistemática y se proporcione evidencia

cuantificable de su funcionamiento. En el Anexo C se presenta una matriz que relaciona cada prueba con el requisito o los requisitos con los que se relaciona.

6.1. Preparación inicial y modelo

Aunque no constituyen pruebas propiamente dichas, las acciones descritas en esta sección son prerequisite para todas las validaciones posteriores. Abarcan la configuración del entorno, la carga de bibliotecas, la inicialización del modelo de extracción de características y el establecimiento de conexión con la base de datos.

6.2. Pruebas funcionales y carga de datos

Esta sección verifica que el sistema cumple con los requisitos funcionales y no funcionales relacionados con el registro, almacenamiento y gestión de imágenes. Para ello, se llevaron a cabo pruebas utilizando conjuntos de datos reales y públicos bajo condiciones controladas, con el fin de evaluar tanto el comportamiento esperado como los límites operativos del sistema.

6.2.1. Carga Inicial de imágenes y persistencia

Se evaluó la capacidad del sistema para procesar directorios completos de imágenes, extraer sus características mediante MobileNetV2 y almacenarlas de forma persistente en una base de datos, así como construir correctamente el índice de similitud.

Esta prueba valida los siguientes requisitos: RF1. Registro de imágenes, RF4.1 y RF4.2. Persistencia del índice FAISS, RNF5.1. Disponibilidad del sistema

Para garantizar un entorno limpio, se reinició la base de datos y se eliminaron archivos previamente generados. Posteriormente, se ejecutó el registro masivo utilizando los datasets *Stanford Dogs* y *Dog_Mx_Dataset*. Como resultado, se almacenaron correctamente las imágenes no duplicadas, y se generaron los archivos `dog_features.faiss` y `dog_id_map.json`, que confirman la construcción exitosa del índice FAISS.

La Figura 16 muestra cómo el sistema omite automáticamente los duplicados internos durante una carga de imágenes. Los registros (logs) detallan cuántas imágenes fueron añadidas y cuántas descartadas, lo que demuestra que el sistema es capaz de manejar grandes

volúmenes de datos de forma confiable, construir un índice de similitud y mantener la persistencia de los datos.

Figura 16. Salida del sistema durante carga inicial de imágenes y persistencia.

```

=== 2.1 Carga Inicial de Imágenes y Persistencia ===
Objetivo: Verificar el registro automático, creación y persistencia de la DB y FAISS.

Acción: Reseteando la base de datos para una prueba de carga inicial limpia...
Borrando todos los registros de la tabla 'dogs'...
Registros borrados.
Archivo de índice FAISS 'dog_features.faiss' borrado.
Archivo de mapeo de IDs 'dog_id_map.json' borrado.
Base de datos e índice FAISS reiniciados.
Acción: Añadiendo imágenes desde ['Stanford_images', 'Dog_Mx_Dataset'] por primera vez (incluye detección de duplicados por característica
s)...
Reconstruyendo índice FAISS desde la base de datos...
Base de datos vacía o sin características válidas, no se puede construir índice FAISS. Borrando archivos de índice si existen.
DEBUG: Índice de batch iniciado vacío, ya que la DB existente está vacía.
Iniciando procesamiento de imágenes...
Se añadieron 20755 nuevas imágenes. Reconstruyendo índice FAISS global para sincronizar con la DB.
Reconstruyendo índice FAISS desde la base de datos...
Índice FAISS guardado en dog_features.faiss con 20755 vectores.
Mapeo de IDs guardado en dog_id_map.json.
Índice FAISS global actualizado y guardado.

--- Resumen del procesamiento ---
Nuevas imágenes añadidas a la BD: 20755
Imágenes omitidas (ya existen por ruta): 0
Imágenes omitidas (duplicados por características): 140

Resultados de Carga Inicial:
Imágenes esperadas en disco (total): 20895
Imágenes añadidas a la DB (después de omitir duplicados internos): 20755
Archivos FAISS creados: True (Index), True (ID Map)
Criterio RF1 / RF4.1/RF4.2 / RNF5.1 / RNF6: ¡ÉXITO!
Imágenes añadidas, FAISS creado/guardado, datos accesibles. Proceso de registro inicial exitoso.
FAISS Index count coincide con DB count. Persistencia inicial verificada.

```

6.2.2. Detección de Duplicados

El sistema está diseñado para evitar el registro de imágenes duplicadas, ya sea porque ya existen o porque son visualmente idénticas.

6.2.2.1. Detección por ruta de archivo.

Esta prueba valida el requisito: RF2.1. Evitar duplicados por ruta

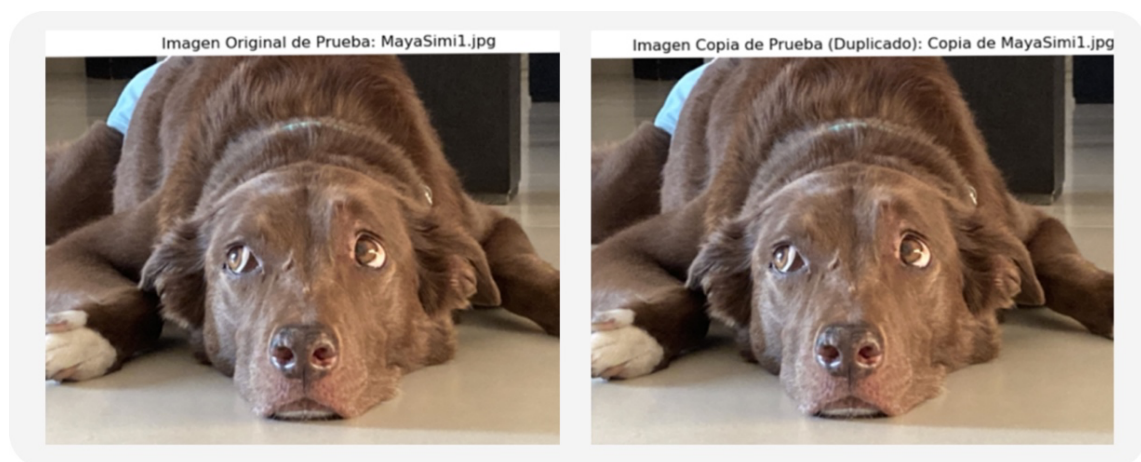
Se intentó añadir nuevamente el mismo conjunto de imágenes previamente registrado. El sistema detectó automáticamente que las rutas ya existían en la base de datos y no añadió ninguna imagen nueva. Esto confirma que la verificación por ruta funciona correctamente y evita duplicaciones innecesarias.

6.2.2.2. Detección por similitud de características

Esta prueba valida los siguientes requisitos: RF2.2. Detección por similitud, RF2.3. Umbral de similitud configurable, RF2.4. Registro condicionado a duplicación, RNF2.1. Precisión de detección de duplicados.

Para esta prueba, se usó un conjunto controlado que contenía una imagen original y su copia exacta pero con distinto nombre. Al extraer sus características y compararlas, la similitud coseno fue de 1.00, indicando coincidencia total. El sistema detectó correctamente que se trataba de un duplicado, y omitió su registro. Esta verificación demuestra que el sistema es capaz de identificar imágenes idénticas, incluso cuando su nombre o ruta cambian. Las dos imágenes utilizadas para esta prueba pueden verse en la Figura 17.

Figura 17. Imágenes con similitud por características



Fuente: Dataset Dog_Mx_Dataset

6.2.2.3. Sincronización del índice FAISS

Esta prueba valida los siguientes requisitos: RF4.3. Carga del índice desde disco, RF4.4. Actualización automática del índice.

Primero, se reinició el entorno de ejecución para simular una nueva sesión y se cargó el índice desde el disco. El sistema pudo reconstruir correctamente el índice en memoria, y la cantidad de vectores coincidía con los registros existentes en la base de datos. Luego, se añadió un pequeño lote de nuevas imágenes. El sistema reconstruyó automáticamente el índice FAISS para incluir las nuevas entradas, y el conteo final de vectores reflejó correctamente el nuevo total de registros.

La Figura 18 muestra la salida generada durante esta prueba, donde se evidencia la correcta actualización del índice y la gestión automática de duplicados. Esto valida que el sistema mantiene sincronizado el índice FAISS de forma eficiente ante cambios, sin requerir intervención manual.

Figura 18. Salida del sistema durante carga y sincronización del índice FAISS

```

=== 2.2.4 Carga y sincronización del índice FAISS ===
Objetivo: Verificar que FAISS carga desde disco y se sincroniza al añadir nuevas imágenes.

Acción: Reiniciando el índice FAISS global para verificar la carga desde disco...
Índice FAISS cargado desde dog_features.faiss con 20755 vectores.
Mapeo de IDs cargado desde dog_id_map.json.
Índice FAISS cargado y sincronizado con la base de datos.
Criterio RF4.3 – Carga del índice: ¡ÉXITO! El índice FAISS se cargo correctamente desde disco y este sincronizado.

Acción: Preparando y añadiendo nuevas imágenes para probar sincronización...
Imágenes en DB antes de añadir nuevas: 20755
Índice FAISS cargado desde dog_features.faiss con 20755 vectores.
Mapeo de IDs cargado desde dog_id_map.json.
Índice FAISS cargado y sincronizado con la base de datos.
DEBUG: Índice de batch inicializado con 20755 elementos de la DB existente.
Iniciando procesamiento de imágenes...
Se añadieron 4 nuevas imágenes. Reconstruyendo índice FAISS global para sincronizar con la DB.
Índice FAISS cargado desde dog_features.faiss con 20755 vectores.
Mapeo de IDs cargado desde dog_id_map.json.
ADVERTENCIA: La DB tiene 20759 registros, pero el índice FAISS cargado tiene 20755. Reconstruyendo índice para sincronizar.
Reconstruyendo índice FAISS desde la base de datos...
Índice FAISS guardado en dog_features.faiss con 20759 vectores.
Mapeo de IDs guardado en dog_id_map.json.
Índice FAISS global actualizado y guardado.

--- Resumen del procesamiento ---
Nuevas imágenes añadidas a la BD: 4
Imágenes omitidas (ya existen por ruta): 0
Imágenes omitidas (duplicados por características): 0
Imágenes en DB después de añadir nuevas: 20759
Criterio RF4.4 – Sincronización FAISS: ¡ÉXITO! FAISS se reconstruyó y sincronizó.
(FAISS: 20759 vectores == DB: 20759 registros)

```

6.3. Pruebas de búsqueda de similitud

Este apartado presenta la demostración de la funcionalidad central del sistema: la búsqueda de perros visualmente similares (RF3). El objetivo es confirmar que, a partir de una imagen de consulta, el sistema puede recuperar imágenes relevantes de la base de datos que presenten una alta similitud visual, ordenadas por relevancia y filtradas por un umbral de similitud coseno.

6.3.1. Procesamiento de la imagen de consulta

Esta prueba valida el requisito: RF3.1. Aceptación y procesamiento de imagen de consulta.

Se seleccionó una imagen específica para ser usada como consulta. Esta fue cargada, visualizada y procesada mediante preprocesamiento y extracción de características con MobileNetV2. El sistema generó exitosamente el vector de características (embedding) sin errores, cumpliendo con el requisito de procesamiento de imágenes de consulta.

La Figura 19 muestra este resultado, mostrando tanto la imagen utilizada como los mensajes de la consola que confirman la finalización del proceso sin errores. Con esto, se demuestra la capacidad del sistema para aceptar y procesar correctamente las imágenes de consulta, cumpliendo con el requisito especificado.

Figura 19. Salida del sistema durante procesamiento de imagen de consulta

```

--- INICIANDO: Pruebas de búsqueda de similitud ---
--- Requisitos relacionados: RF3 ---

===Procesamiento de la imagen de consulta (RF3.1) ===
Objetivo: Verificar que el sistema acepta una imagen como consulta para la búsqueda.

Acción: Seleccionando imagen de búsqueda: Dog_Mx_Dataset/oliver2.jpg

Imagen de consulta para RF3

```



```

Acción: Extrayendo características de la imagen de búsqueda...
Características de la imagen de búsqueda extraídas.

```

6.3.2. Recuperación de resultados y aplicación del umbral

Esta prueba valida los siguientes requisitos: RF3.2. Recuperación de imágenes similares, RF3.4. Aplicación de umbral de similitud configurable.

Se ejecutó la búsqueda solicitando 15 resultados con un umbral de similitud coseno 0.50. El sistema devolvió un listado de imágenes que superaban el umbral, confirmando que la búsqueda respeta el criterio configurado.

6.3.3. Ordenamiento y Visualización de Resultados

Esta prueba valida el requisito: RF3.3. Ordenamiento descendente por similitud.

Se comprobó que los resultados estuvieran ordenados correctamente de mayor a menor similitud y se mostró información relevante para cada resultado. La mayoría de las imágenes recuperadas resultaron visualmente coherentes con la imagen de entrada.

6.3.4. Ejemplos de resultados de la búsqueda de similitud

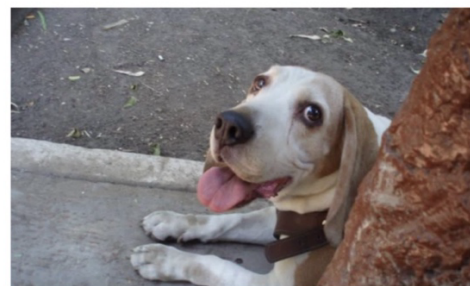
En esta sección se presentan ejemplos gráficos que ilustran la efectividad del sistema en la búsqueda de imágenes visualmente similares. Se muestran la imagen de consulta junto con las imágenes recuperadas por el modelo, ordenadas según el nivel de similitud calculado. Estas representaciones permiten corroborar visualmente que el sistema cumple con los requisitos

de recuperación y ordenamiento, facilitando una evaluación cualitativa del desempeño en escenarios reales. Como complemento, en el Anexo A se incluyen dos ejemplos adicionales.

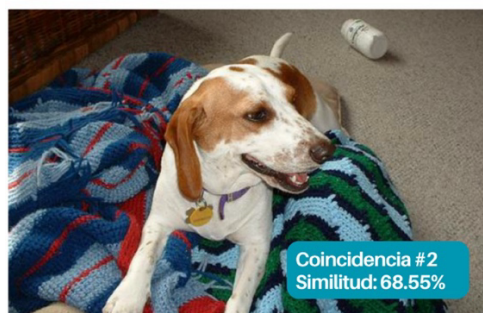
En la Figura 20, se muestran seis imágenes recuperadas tras ingresar una imagen de un perro blanco con manchas café. Las similitudes oscilan entre el 69.78% y el 66.93%. A pesar de las diferencias en los fondos, se aprecia visualmente que todos los perros presentan una apariencia física similar a la del perro de consulta.

Figura 20. Resultados de búsqueda de un perro blanco con manchas café

Imagen de consulta:



Coincidencias:



Como segundo ejemplo, en la Figura 21 se muestra el resultado de ingresar una imagen de un perro husky siberiano. Aunque la imagen presenta baja calidad de iluminación y el perro no está mirando al frente, su cuerpo es visible. Las similitudes obtenidas van del 73.37% al 70.24%. Se observa que, al tratarse de una raza específica y reconocible, los porcentajes de coincidencia tienden a ser más altos.

Figura 21. . Resultados de búsqueda de un perro de raza husky siberiano.

Imagen de consulta:



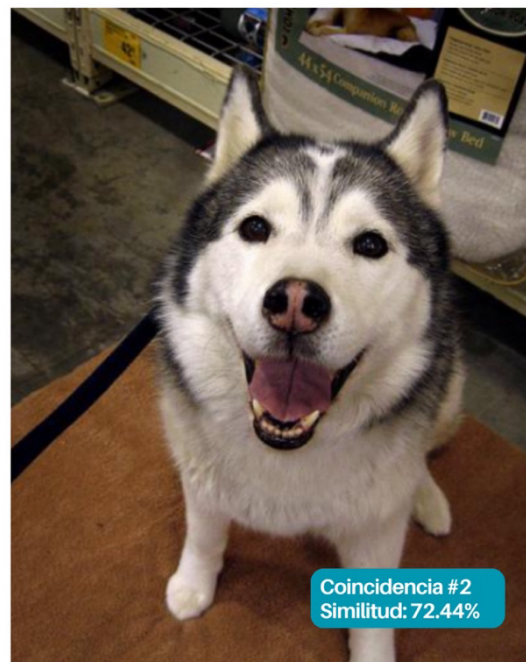
Coincidencias:



Coincidencia #1
Similitud: 73.37%



Coincidencia #3
Similitud: 67.70%



Coincidencia #2
Similitud: 72.44%



Coincidencia #4
Similitud: 70.76%



Coincidencia #5
Similitud: 70.38%



Coincidencia #6
Similitud: 70.24%

6.3.5. Pruebas con imágenes desafiantes

Esta prueba valida el requisito: RNF2. Rendimiento con imágenes desafiantes.

Se evaluó el comportamiento del sistema ante imágenes difíciles, como perros mestizos, mala iluminación, baja calidad o pelaje oscuro. En el primer caso, se usó como imagen de entrada un perro mestizo. Aunque el sistema recuperó algunas coincidencias, estas pertenecían a perros de razas específicas con rasgos similares, lo que sugiere que el modelo tiende a asociar características con razas conocidas. Este ejemplo puede observarse en la Figura 22.

Figura 22. Prueba con imagen de un perro mestizo.



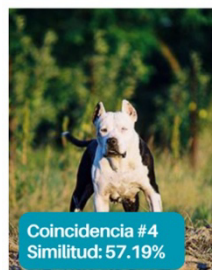
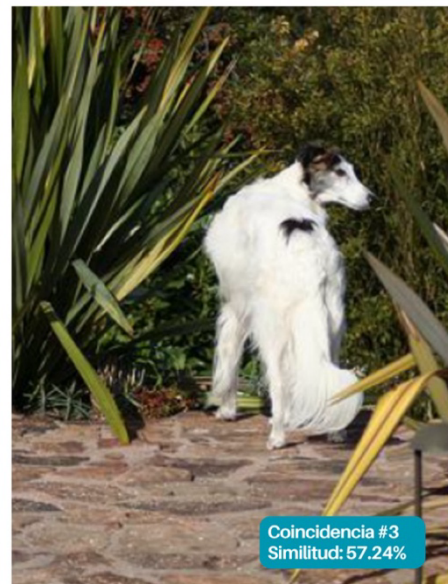
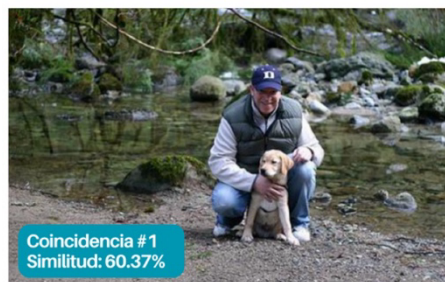
En el segundo ejemplo, se empleó una imagen con mala iluminación, en la que los rasgos del perro estaban poco definidos debido a la falta de luz. En este caso, el sistema no logró recuperar coincidencias relevantes, lo que evidencia una limitación importante en contextos con calidad visual reducida.

Figura 23. Prueba con imagen de perro con mala iluminación.

**Imagen de consulta
con mala iluminación:**



Coincidencias:



Como conclusión, los resultados fueron satisfactorios para pelaje oscuro y perros mestizos, pero no para baja calidad de imagen ni mala iluminación, evidenciando limitaciones para cumplir plenamente con el requisito RNF2. Ejemplos adicionales de estos casos se pueden consultar en el Anexo B.

6.4. Evaluación de rendimiento y precisión

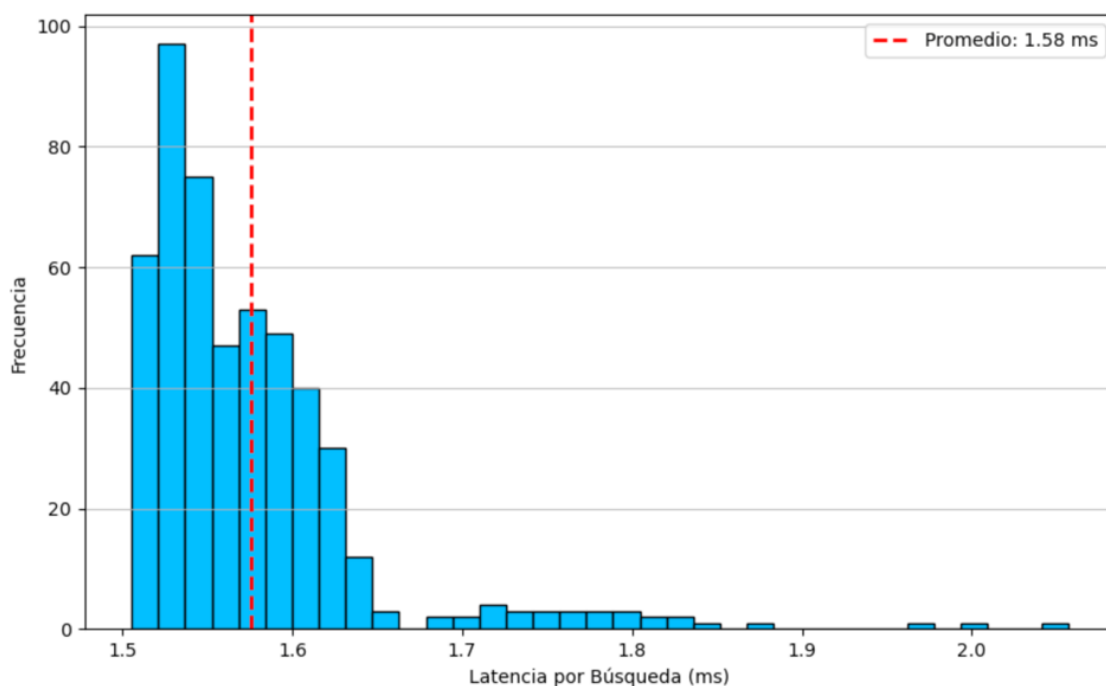
Este apartado mide el desempeño del sistema en aspectos clave como la latencia de búsqueda, escalabilidad del registro masivo y precisión en la detección de duplicados y búsqueda de similitud. Se validan los requisitos no funcionales relacionados con rendimiento, precisión y optimización en entornos con recursos limitados.

6.4.1. Latencia de búsqueda

Esta prueba valida el requisito: RNF1.1. Latencia de búsqueda.

Se midió el tiempo que tarda una consulta de similitud en una base de datos con 20,755 imágenes. Para obtener una medida estadísticamente robusta y confiable, la prueba realizó 500 mediciones individuales de la función de búsqueda. En cada consulta se solicitaron los 15 resultados más similares, utilizando un umbral de similitud coseno de 0.50. Los resultados mostraron un tiempo promedio de respuesta de 1.58 ms con una desviación estándar de 0.07 ms, cumpliendo ampliamente el límite de 5 ms establecido por el requisito RNF1.1. Como se observa en la Figura 24, la mayoría de las consultas se completan entre 1.5 y 1.62 ms.

Figura 24. Distribución de latencia de búsqueda



6.4.2. Escalabilidad del registro

Esta prueba valida el requisito: RNF1.2. Escalabilidad del registro.

Se evaluó el tiempo requerido para registrar más de 20,000 imágenes utilizando el dataset Stanford_images. Tras reinicializar la base de datos y el índice FAISS para un entorno limpio, se ejecutó el registro masivo completo y se midió la duración total. El sistema añadió exitosamente 20,755 imágenes en 11.88 minutos, superando el objetivo de completar el proceso en menos de 20 minutos. Esto confirma que el sistema puede manejar registros de imágenes con eficiencia y escalabilidad adecuada para aplicaciones reales.

6.4.3. Evaluación de Precisión

Este apartado evalúa la precisión del sistema en la detección de duplicados, búsqueda de similitud y manejo de imágenes fuera de la clase objetivo, asegurando que cumple con los requisitos de exactitud y confiabilidad establecidos.

6.4.3.1. Precisión en Detección de Duplicados

Esta prueba valida el requisito: RNF2.1. Precisión mínima del 98% en detección de duplicados.

Se utilizó un conjunto controlado de 20 imágenes, compuesto por 6 imágenes originales, 6 imágenes duplicadas, 6 imágenes visualmente similares y 2 imágenes únicas. El registro automático fue ejecutado en un entorno limpio para evaluar la capacidad del sistema de evitar registros redundantes. El sistema detectó correctamente las 6 copias exactas como duplicados y las omitió, registrando únicamente las imágenes originales, las visualmente similares y la únicas dando un total de 14 imágenes registradas y 6 omitidas.

La precisión observada fue del 100%, cumpliendo y superando el requisito establecido. La Figura 25 muestra la salida del sistema durante esta prueba, donde se confirma la correcta detección y omisión de duplicados.

Figura 25. Salida del sistema durante prueba de detección de duplicados

```

===4.3.1: Precision en detección de duplicados (RNF2.1) ===
Objetivo: Cuantificar la efectividad de la detección de duplicados al momento del registro.

Accion: Reseteando la DB para la prueba de precisión de duplicados...
Borrando todos los registros de la tabla 'dogs'...
Registros borrados.
Archivo de índice FAISS 'dog_features.faiss' borrado.
Archivo de mapeo de IDs 'dog_id_map.json' borrado.
Base de datos e índice FAISS reiniciados.
--- Iniciando adición de imagenes desde 'duplicate_test_subset' a la base de datos ---
Reconstruyendo índice FAISS desde la base de datos...
Base de datos vacía o sin características válidas, no se puede construir índice FAISS. Borrando archivos de índice si existen.
DEBUG: Índice de batch iniciado vacío, ya que la DB existente está vacía.
Iniciando procesamiento de imágenes...
Se añadieron 14 nuevas imágenes. Reconstruyendo índice FAISS global para sincronizar con la DB.
Reconstruyendo índice FAISS desde la base de datos...
Índice FAISS guardado en dog_features.faiss con 14 vectores.
Mapeo de IDs guardado en dog_id_map.json.
Índice FAISS global actualizado y guardado.

--- Resumen del procesamiento ---
Nuevas imágenes añadidas a la BD: 14
Imágenes omitidas (ya existen por ruta): 0
Imágenes omitidas (duplicados por características): 6

--- Resumen Final ---
Total de imágenes en la base de datos después de la adición: 14
--- Adición de imagenes completada ---
--- El numero de imagenes duplicadas en la DB es: 6 ---
Criterio RNF2.1 - Precision en Detección de Duplicados : ¡ÉXITO! El porcentaje de precisión es de 100% .

```

6.4.3.2. F1-score en Búsqueda de Similitud y determinación del umbral óptimo

Esta prueba valida el requisito: RNF2.2. F1-score mínimo de 0.85 en búsqueda de similitud.

Este apartado tiene como objetivo establecer el umbral óptimo de similitud visual que permita maximizar el rendimiento del sistema en la recuperación de imágenes relevantes. Se busca lograr un equilibrio entre la precisión (minimizar falsas coincidencias) y el recall (maximizar resultados relevantes), empleando como métrica principal el F1-score.

Para esta evaluación se construyó un conjunto de referencia manualmente etiquetado, conocido como *Ground Truth*, que permite medir con objetividad el rendimiento del sistema. Este conjunto incluye 313 imágenes organizadas en 39 grupos, donde cada grupo agrupa imágenes del mismo perro o de perros muy similares físicamente.

Dentro de este contexto, se consideran dos categorías:

- **Imágenes similares esperadas:** pertenecientes al mismo grupo que la imagen de consulta, ya sea por ser el mismo perro en diferentes condiciones o perros muy parecidos.
- **Imágenes no similares esperadas:** tomadas de grupos diferentes, sin similitudes físicas relevantes con la consulta.

Esta clasificación permite simular escenarios realistas y medir cómo se comporta el sistema ante búsquedas con diferentes grados de exigencia. A continuación, se muestran tres ejemplos representativos de las 39 agrupaciones realizadas: en la Figura 26 se muestran

imágenes de perros con pelaje dorado, en la Figura 27 se observan perros con pelaje blanco con manchas negras o café y en la Figura 28 agrupa imágenes de perros de la raza dachshund.

Figura 26. Grupo de perros con pelaje dorado.



Fuente: Dataset Dog_Mx_Dataset

Figura 27. Grupo de perros con pelaje blanco con manchas.



Fuente: Dataset Dog_Mx_Dataset

Figura 28. Grupo de perros de la raza dachshund.



Fuente: Dataset Dog_Mx_Dataset

Se cargaron todas las imágenes del conjunto de prueba, se extrajeron sus vectores de características con MobileNetV2, y se construyó un índice FAISS completo. A partir de esto, se ejecutaron búsquedas para cada imagen de consulta del Ground Truth variando el umbral de similitud coseno desde 0.20 hasta 0.95. Para cada búsqueda se compararon los resultados obtenidos con los grupos del Ground Truth, registrando los verdaderos positivos, falsos positivos y falsos negativos, como se muestra en la Tabla 4.

Tabla 4. Resultados simplificados por imagen y umbral.

	query_image	threshold	true_positives	false_positives	false_negatives	\
0	alek4.JPG	0.20	8	5	0	0
1	alek4.JPG	0.30	8	5	0	0
2	alek4.JPG	0.40	8	4	0	0
3	alek4.JPG	0.50	8	1	0	0
4	alek4.JPG	0.55	8	0	0	0
..	...	...	...	...	...	...
415	tugger1.JPG	0.60	1	0	0	0
416	tugger1.JPG	0.70	0	0	1	1
417	tugger1.JPG	0.80	0	0	1	1
418	tugger1.JPG	0.90	0	0	1	1
419	tugger1.JPG	0.95	0	0	1	1

A partir de esto se calcularon las métricas promedio de precisión, recall y F1-score para cada valor de umbral. La Tabla 5 presenta los valores promedio agregados por umbral.

Tabla 5. Resultados de análisis de umbral de similitud.

threshold	avg_precision	avg_recall	avg_f1_score
0.20	0.485511	0.980952	0.630871
0.30	0.525862	0.980952	0.665403
0.40	0.651564	0.971270	0.764871
0.50	0.930830	0.834553	0.858890
0.55	0.915873	0.638606	0.715020
0.60	0.745238	0.434720	0.515740
0.70	0.285714	0.088955	0.127874
0.80	0.047619	0.013889	0.021429
0.90	0.000000	0.000000	0.000000
0.95	0.000000	0.000000	0.000000

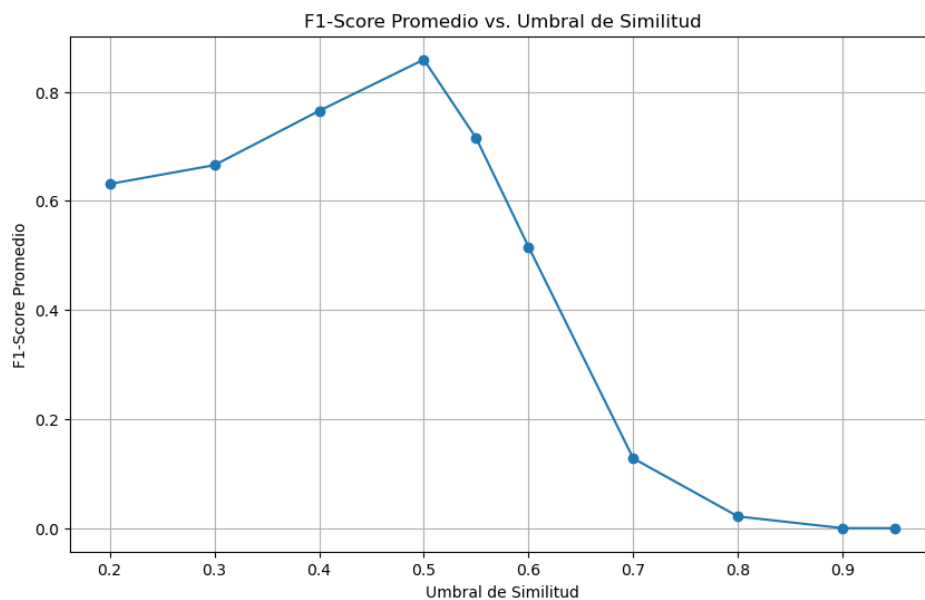
El comportamiento observado fue el siguiente:

- Para umbrales bajos (0.20–0.30), el sistema recuperó casi todas las imágenes similares, con un recall cercano al 100%, pero también devolvió muchos falsos positivos, lo que redujo la precisión a valores entre el 48% y 52%.
- En umbrales intermedios, particularmente en 0.40, la precisión aumentó considerablemente a 65.1% mientras el recall se mantuvo alto a 97.1%, mejorando significativamente el F1-score.

- El punto óptimo se alcanzó con un umbral de 0.50, donde se logró el F1-score promedio más alto: 0.858. En este umbral, el sistema alcanzó una precisión de 93.0% y un recall de 83.4%, logrando un equilibrio.
- A partir del umbral 0.55, el rendimiento comenzó a caer abruptamente. El recall bajó a 63.8% en 0.55 y 43.4% en 0.60, lo que redujo el F1-score. Para umbrales superiores a 0.90, el sistema prácticamente no encontró resultados.

La Figura 29 muestra con claridad la curva de rendimiento y confirma que 0.50 es el umbral ideal para maximizar el desempeño general del sistema.

Figura 29. F1-Score Promedio vs Umbral de Similitud



A partir de la evaluación realizada se observó que el sistema de búsqueda de perros similares alcanza su mejor rendimiento general con un umbral de 0.50. En este umbral, el sistema demuestra un balance óptimo entre la precisión, con el 96.1% de los resultados encontrados relevantes y el recall, el 86.6% de los resultados esperados son recuperados.

6.4.3.3. Robustez a Imágenes No Pertenecientes a la Clase.

Esta prueba valida el requisito: RNF2.3. Robustez ante imágenes irrelevantes.

La prueba tiene como objetivo verificar que el sistema no devuelve resultados de similitud cuando se le consulta con imágenes que claramente no pertenecen a la clase objetivo (perros), como objetos, otros animales o imágenes irrelevantes. La ausencia de coincidencias ante este

tipo de entradas garantiza que el modelo no genere falsos positivos frente a datos fuera del dominio, lo cual es esencial para su confiabilidad en uso real.

Para llevar a cabo esta validación, se seleccionaron dos imágenes que no corresponden a perros: una imagen de un gato (*cat.jpeg*) y otra de una manzana (*apple.jpg*). Ambas fueron colocadas en un directorio de prueba negativo llamado *nonDogs*. Las imágenes se cargaron utilizando el mismo procesamiento y extracción de características aplicado a imágenes válidas. A continuación, se ejecutó el proceso de búsqueda de similitud, evaluando si el sistema encontraba alguna coincidencia por encima del umbral establecido (0.50). En ambos casos, el sistema no devolvió resultados, confirmando que estas imágenes no fueron clasificadas incorrectamente como similares a perros almacenados. La prueba realizada con la imagen del gato se muestra en la Figura 30.

Figura 30. Prueba con imagen de entrada de un gato.



6.4.4. Optimización para Entornos de Recursos Limitados

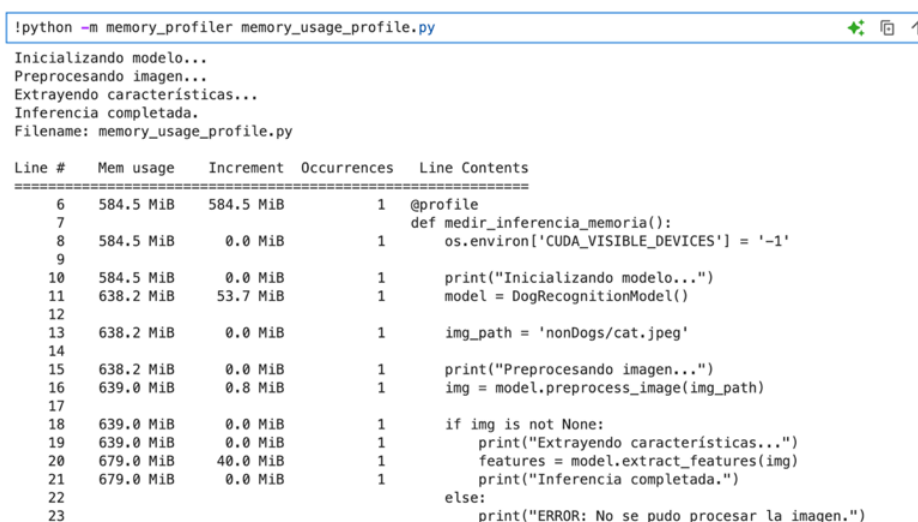
En este apartado se evalúan las optimizaciones realizadas para asegurar que el modelo funcione eficientemente en entornos con recursos limitados.

6.4.4.1. Huella de Memoria del Modelo

Esta prueba valida el requisito: RNF6.1. El uso de memoria durante la inferencia debe ser inferior a 150 MB de RAM.

El objetivo de esta prueba es evaluar el uso de memoria RAM durante el proceso de inferencia del modelo, con el fin de comprobar que no se supera el umbral de 150 MB, establecido para garantizar la compatibilidad del sistema con entornos de recursos limitados. Para simular un escenario basado únicamente en CPU, se configuró el entorno para deshabilitar el uso de GPU. A continuación, se ejecutó un análisis detallado del uso de memoria mientras se cargaba el modelo y se procesaba una imagen de prueba. Como se observa en la Figura 31, al momento de cargar el modelo, la memoria utilizada se incrementa en 49.1 MiB, lo que corresponde al uso necesario para mantener el modelo en memoria. Posteriormente, durante el proceso de inferencia y extracción de características, se registra un aumento adicional de 43.1 MiB. En conjunto, el uso total de memoria pasa de aproximadamente 583.4 MiB a 676.4 MiB, reflejando el consumo combinado asociado tanto a la carga del modelo como al procesamiento de la imagen de entrada. Este valor supera de forma considerable el límite establecido por el requisito.

Figura 31. Medición de huella de memoria del modelo



```
!python -m memory_profiler memory_usage_profile.py
```

```
Inicializando modelo...
Preprocesando imagen...
Extrayendo características...
Inferencia completada.
Filename: memory_usage_profile.py
```

Line #	Mem usage	Increment	Occurrences	Line Contents
6	584.5 MiB	584.5 MiB	1	@profile
7				def medir_inferencia_memoria():
8	584.5 MiB	0.0 MiB	1	os.environ['CUDA_VISIBLE_DEVICES'] = '-1'
9				
10	584.5 MiB	0.0 MiB	1	print("Inicializando modelo...")
11	638.2 MiB	53.7 MiB	1	model = DogRecognitionModel()
12				
13	638.2 MiB	0.0 MiB	1	img_path = 'nonDogs/cat.jpeg'
14				
15	638.2 MiB	0.0 MiB	1	print("Preprocesando imagen...")
16	639.0 MiB	0.8 MiB	1	img = model.preprocess_image(img_path)
17				
18	639.0 MiB	0.0 MiB	1	if img is not None:
19	639.0 MiB	0.0 MiB	1	print("Extrayendo características...")
20	679.0 MiB	40.0 MiB	1	features = model.extract_features(img)
21	679.0 MiB	0.0 MiB	1	print("Inferencia completada.")
22				else:
23				print("ERROR: No se pudo procesar la imagen.")

6.4.4.2. Eficiencia de Inferencia del Modelo

Esta prueba valida el requisito: RNF6.2. El tiempo de extracción de características por imagen no debe exceder los 50 ms por imagen.

Esta prueba tiene como objetivo evaluar la velocidad del modelo durante la extracción de características en CPU, asegurando que no supere los 50 milisegundos por imagen. Esta métrica es crítica para sistemas embebidos que no disponen de aceleración por GPU. Se

seleccionó una imagen válida, se encapsuló la inferencia en una función auxiliar y midió el tiempo promedio de procesamiento en 100 ejecuciones. El resultado fue de 25.36 ms por imagen, lo que se encuentra muy por debajo del umbral.

6.4.4.3. Tamaño del Modelo

Esta prueba valida el requisito: RNF6.3. El tamaño en disco debe ser inferior a 25 MB.

Aquí se evalúa si el modelo puede almacenarse eficientemente para facilitar su despliegue en dispositivos con capacidad de almacenamiento reducida. Se guardó el modelo MobileNetV2 utilizado para la extracción de características en formato .keras y se midió su tamaño en disco.

El resultado fue de 9.16 MB, muy por debajo del umbral máximo de 25 MB, lo que lo hace apto para implementaciones en sistemas embebidos.

6.4.5. Limitaciones del sistema

Si bien el sistema ha demostrado ser eficaz en la búsqueda de similitud de imágenes de perros y en la gestión de duplicados, se identificaron algunas limitaciones durante su desarrollo y evaluación. Una de las principales es el consumo de memoria del modelo MobileNetV2 durante la inferencia, que alcanzó los 676.4 MiB, superando ampliamente el umbral deseado de 150 MB. Esto podría limitar su implementación en dispositivos con recursos restringidos, como plataformas móviles de gama baja o sistemas embebidos, sin una optimización adicional. Otra limitación importante está relacionada con la sensibilidad del sistema ante imágenes con condiciones visuales desafiantes. En fotografías con mala iluminación, baja resolución, contraluz o sombras marcadas, la calidad de los resultados disminuye considerablemente, dificultando la extracción de características relevantes.

Por último, si bien MobileNetV2 está diseñado para funcionar en dispositivos móviles, su rendimiento puede variar ampliamente entre distintos modelos, versiones de sistema operativo y librerías de inferencia. La falta de pruebas específicas en entornos móviles reales impide anticipar su comportamiento exacto, lo que representa una limitación adicional a considerar en futuros despliegues.

7. Conclusiones y trabajo futuro

En esta sección se resumen los principales resultados y se evalúa cómo se han cumplido los objetivos del proyecto, mostrando cómo la implementación y las pruebas contribuyen a desarrollar un sistema eficiente para la búsqueda visual de similitud entre imágenes de perros.

Para facilitar la comprensión de la relación entre los objetivos, los requisitos, las secciones que describen su implementación, las pruebas realizadas y su estado de cumplimiento, se incluye en el Anexo D una matriz de trazabilidad. Esta matriz organiza la información de forma clara y sirve como guía para entender cómo cada parte del trabajo aporta al cumplimiento de los requisitos.

Finalmente, se señalan las limitaciones detectadas durante el desarrollo y la evaluación, y se proponen futuras líneas de trabajo para mejorar el rendimiento, la eficiencia y la robustez del sistema, especialmente en escenarios más complejos o con recursos limitados.

7.1. Conclusiones del trabajo

A continuación, se presentan las conclusiones asociadas a cada objetivo específico del proyecto, con el fin de mostrar cómo cada fase del trabajo aportó al desarrollo del sistema propuesto.

Conclusión sobre el Objetivo Específico 1

Objetivo: *Analizar y seleccionar la arquitectura de modelo de reconocimiento de imágenes más adecuada para la extracción de características justificando la elección en base a criterios de precisión, eficiencia y compatibilidad considerando las limitaciones propias de entornos con recursos limitados.*

Se compararon varias arquitecturas y se eligió MobileNetV2 por su eficiencia y modelo preentrenado. Las pruebas mostraron que cumple con el tamaño en disco y tiempo de inferencia, aunque el consumo de memoria en RAM excede el límite esperado. Por tanto, el objetivo se considera parcialmente alcanzado, con margen para optimizar el uso de memoria en futuras versiones.

Conclusión sobre el Objetivo Específico 2

Objetivo: *Recopilar y conformar un conjunto de datos de imágenes de perros diverso y representativo a partir de múltiples fuentes.*

Se buscó conformar un dataset diverso y representativo de imágenes de perros a partir de varias fuentes. Se recopiló un conjunto de aproximadamente 20,925 imágenes provenientes de cinco datasets, que fue procesado y almacenado automáticamente, asegurando calidad y unicidad con mecanismos internos. Por ello, este objetivo se considera alcanzado, con una base sólida para el desarrollo del sistema.

Conclusión sobre el Objetivo Específico 3

Objetivo: *Diseñar e implementar una arquitectura de software modular y mantenible que estructure los componentes del sistema para facilitar su desarrollo, escalabilidad y futura evolución.*

El objetivo consistió en diseñar una arquitectura modular y mantenible para facilitar desarrollo y escalabilidad. Se implementó una estructura con notebooks y módulos Python especializados que separan funcionalidades clave. La modularidad permitió integrar librerías complejas y optimizar procesos, validándose indirectamente mediante el rendimiento logrado. Se considera, por tanto, que el objetivo fue alcanzado.

Conclusión sobre el Objetivo Específico 4

Objetivo: *Implementar el modelo de reconocimiento de imágenes seleccionado como un extractor de características, asegurando el preprocesamiento adecuado y la generación de embeddings normalizados.*

Se implementó MobileNetV2 como extractor de características con un preprocesamiento adecuado para generar embeddings normalizados. El sistema demostró alta eficiencia de inferencia y calidad en la representación vectorial, validada con un F1-score de 0.858 en búsqueda de similitud. Este objetivo se considera alcanzado.

Conclusión sobre el Objetivo Específico 5

Objetivo: *Diseñar e implementar un sistema robusto de gestión de base de datos para el almacenamiento persistente y el registro automático de imágenes de perros.*

Se diseñó e implementó un sistema robusto de gestión de base de datos con SQLite y SQLAlchemy para almacenamiento persistente y registro automático. El sistema procesó y almacenó con éxito más de 20,700 imágenes, manteniendo la persistencia y sincronización del índice de búsqueda. Por tanto, el objetivo fue alcanzado.

Conclusión sobre el Objetivo Específico 6

Objetivo: *Desarrollar e integrar un mecanismo de detección de duplicados optimizado en el proceso de registro.*

Se desarrolló un mecanismo optimizado de detección de duplicados usando FAISS, combinando verificación por ruta y similitud de embeddings. Las pruebas mostraron una precisión del 100% en un conjunto reducido y una alta escalabilidad en el registro masivo. El objetivo se considera parcialmente alcanzado, aunque se recomienda validación en escenarios más amplios.

Conclusión sobre el Objetivo Específico 7

Objetivo: *Desarrollar un módulo de búsqueda que permita identificar imágenes similares de manera rápida y eficiente, utilizando técnicas avanzadas de indexación y consulta.*

El módulo de búsqueda implementado usa embeddings y FAISS para consultas rápidas y escalables. Se validó la correcta gestión del índice y la robustez frente a imágenes no caninas. La búsqueda es funcional y eficiente, aunque la calidad de resultados depende de la imagen de entrada. Este objetivo se considera parcialmente alcanzado.

Conclusión sobre el Objetivo Específico 8

Objetivo: *Evaluar y validar el rendimiento global del sistema para justificar la efectividad de los mecanismos de búsqueda de similitud y detección de duplicados, mediante la aplicación de un protocolo de pruebas y el análisis de métricas de desempeño.*

Se evaluó el rendimiento global del sistema con métricas de latencia, escalabilidad y precisión. Los resultados superaron la mayoría de los requisitos, incluyendo un tiempo de respuesta

promedio de 1.58 ms y un F1-score de 0.858. Se identificaron limitaciones en la huella de memoria y manejo de imágenes difíciles, marcando áreas para mejora. El objetivo se considera parcialmente alcanzado.

Conclusión general

El objetivo general de desarrollar un sistema para la identificación de imágenes de perros optimizado para entornos con recursos limitados se considera parcialmente alcanzado. El sistema integra componentes eficientes como MobileNetV2, FAISS y SQLite, y ha demostrado buen desempeño en precisión, latencia y escalabilidad. Aunque se identificaron limitaciones en el uso de memoria y en el manejo de imágenes complejas, estos resultados ofrecen una base sólida para futuras mejoras y validaciones en escenarios más exigentes.

7.2. Líneas de trabajo futuro

Con el fin de evolucionar el sistema desarrollado hacia una solución aplicable en contextos reales, se proponen distintas líneas de mejora. En primer lugar, se plantea su despliegue en una aplicación móvil o web que permita registrar perros mediante imágenes y datos, facilitando la búsqueda en caso de extravío mediante coincidencias visuales, mapas interactivos y canales de contacto. Para mantener la compatibilidad con entornos de recursos limitados, será esencial optimizar el rendimiento en dispositivos móviles y asegurar una baja demanda computacional.

Además, se contemplan mejoras técnicas como el ajuste fino del modelo MobileNetV2 con datos específicos de perros, el uso de funciones de pérdida como Triplet Loss para mejorar la calidad de los embeddings, y la implementación de detección automática de objetos para centrar el análisis en el perro y no en el fondo. También se propone ampliar y diversificar el dataset con más razas, condiciones visuales variadas y múltiples imágenes por individuo.

Finalmente, se sugiere integrar imágenes y datos de redes sociales, así como utilizar metadatos como ubicación, color, tamaño y fecha para enriquecer y refinar las búsquedas, aumentando la efectividad del sistema en contextos reales, dinámicos y potencialmente masivos, sin comprometer la eficiencia del sistema.

Referencias bibliográficas

- Aditya, K., Nityananda, J., Bangpeng, Y., & Li, F.-F. (2011). *Novel dataset for fine-grained image categorization: Stanford Dogs*. En *2011 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 3513-3520). IEEE.
- American Humane. (2022, 19 de abril). *Attention pet owners: It's National Pet ID Week!* <https://www.americanhumane.org/blog/attention-pet-owners-its-national-pet-id-week/>
- Andrade, H. A. (2021). Modelo para detectar el uso correcto de mascarillas en tiempo real utilizando redes neuronales convolucionales. *Revista de Investigación en Tecnologías de la Información*, 9(18), 70-83. <https://doi.org/10.36825/RITI.09.17.011>
- Arathoon, J., Allen, D., & Hallatt, A. (2024). "Too hot to handle": Making lost and stolen pets present in virtual space. *Symbolic Interaction*, 47(1), 127-151. <https://doi.org/10.1016/j.geoforum.2024.104013>
- Awofeso, Z. (2024, 21 de enero). *A summary of the MobileNetV2 (inverted residuals and linear bottlenecks) paper*. Medium. <https://medium.com/codex/a-summary-of-the-mobilenetv2-inverted-residuals-and-linear-bottlenecks-paper-e19b187cb78a>
- Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8), 1798-1828. <https://doi.org/10.48550/arXiv.1206.5538>
- Bericat, E. (2025). Personas y perros. La relación humano-perro en la sociedad contemporánea: Imágenes, emociones y experiencias. *Revista Española de Sociología*, 33(2), a135. <https://doi.org/10.22325/fes/res.2025.251>
- Binta Islam, S., Valles, D., Hibbitts, T., Ryberg, W., Walkup, D., & Forstner, M. (2023). Animal species recognition with deep convolutional neural networks from ecological camera trap images. *Animals*, 13(13), 2191. <https://doi.org/10.3390/ani13091526>
- Bouraya, S., & Belangour, A. (2024). Evaluating the real-world application efficacy of MobileNet models. *International Journal of Engineering Trends and Technology*, 72(4), 197-202. <https://doi.org/10.14445/22315381/IJETT-V72I9P116>

- Boyko, A. R., Quignon, P., Li, L., Schoenebeck, J. J., Degenhardt, J. D., Lohmueller, K. E., Zhao, K., Brisbin, A., Parker, H. G., vonHoldt, B. M., Cargill, M., Auton, A., Reynolds, A., Elk, C. E., Ostrander, E. A., & Bustamante, C. D. (2010). A simple genetic architecture underlies morphological variation in dogs. *PLoS Biology*, 8(8), e1000451.
<https://doi.org/10.1371/journal.pbio.1000451>
- Chu, J. C. (2019, 29 de julio). *Say hello to dog facial recognition technology*. Slate.
<https://slate.com/technology/2019/07/dog-facial-recognition-privacy-finding-rover-megvii.html>
- Cook, A. (2010, 14 de julio). *Tips for finding a lost pet*. WebMD.
<https://pets.webmd.com/features/tips-for-finding-a-lost-pet>
- Das, S. (2013, 11 de diciembre). *Finding Rover app tracks lost dogs using facial recognition*. CNET. <https://www.cnet.com/tech/mobile/finding-rover-app-tracks-lost-dogs-using-facial-recognition/>
- Datta, R., Joshi, D., Li, J., & Wang, J. Z. (2008). Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys*, 40(2), 1–60.
<https://doi.org/10.1145/1348246.134824>
- Deshmukh, S. C. (2023). Study of image recognition using machine learning. *International Journal for Research in Applied Science and Engineering Technology*, 11(5), 4583–4586. <https://doi.org/10.22214/ijraset.2023.54063>
- Elharrouss, O., Akbari, Y., Almaadeed, N., & Al-Maadeed, S. (2024). Backbones-review: Feature extraction networks for deep learning and deep reinforcement learning approaches. *Neurocomputing*, 574, 127239.
<https://doi.org/10.1016/j.cosrev.2024.100645>
- Faiss. (s. f.). *Welcome to Faiss documentation*. Recuperado el 7 de julio de 2024, de <https://faiss.ai/index.html>
- Faster Capital. (2025, 31 de marzo). *The role of social media in lost pet recovery*.
<https://fastercapital.com/topics/the-role-of-social-media-in-lost-pet-recovery.html>
- Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.

- Google. (s. f.). *¿Qué es la inteligencia artificial?* Google Cloud. Recuperado el 7 de julio de 2024, de <https://cloud.google.com/learn/what-is-artificial-intelligence?hl=es-419>
- He, K., Zhang, X., Ren, S., & Sun, J. (2015). *Deep residual learning for image recognition*. ArXiv. <https://doi.org/10.48550/arXiv.1512.03385>
- Howard, A., Sandler, M., Chu, G., Chen, L. C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., Le, Q. V., & Adam, H. (2019). Searching for MobileNetV3. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 1314-1324. <https://doi.org/10.1109/ICCV.2019.00140>
- Humane World for Animals. (s.f.). *How a microchip can keep you and your pet together*. <https://www.humaneworld.org/en/resources/how-microchip-can-keep-you-and-your-pet-together>
- Hyvönen, V., Jaasaari, E., & Roos, T. (2024). A multilabel classification framework for approximate nearest neighbor search. *Journal of Machine Learning Research*, 25(9), 1-36. <http://jmlr.org/papers/v25/22-0359.html>
- Kanat-Maymon, Y., Wolfson, S., Cohen, R., & Roth, G. (2021). The benefits of giving as well as receiving need support in human–pet relations. *Journal of Happiness Studies*, 22, 3247–3263. <https://doi.org/10.1007/s10902-020-00279-9>
- Kurdek, L. A. (2008). Pet dogs as attachment figures. *Journal of Social and Personal Relationships*, 25(2), 247–266. <https://doi.org/10.1177/026540750708795>
- Lord, L. K., Wittum, T. E., Ferketich, A. K., Funk, J. A., & Rajala-Schultz, P. (2007). Search methods that people use to find lost dogs. *Journal of the American Veterinary Medical Association*, 230(2), 211–216. <https://doi.org/10.2460/javma.230.2.211>
- Loriente, M. R. (2022). *Aprendizaje automático para la identificación de razas caninas* [Tesis de grado, Universitat Politècnica de València]. Repositorio Institucional UPV. <http://hdl.handle.net/10251/187212>
- Mars Petcare. (2020). *Resultados del estudio actitudinal - México: Índice de las mascotas sin hogar*. https://www.mars.com/sites/g/files/dfsbufz106/files/2024-11/Indice_de_Mascotas_sin_Hogar_Actitudinal.pdf

- Missing Animal Response Network. (s.f.). *Neon & fluorescent posters*. Recuperado el 7 de julio de 2024, de <https://www.missinganimalresponse.com/neon-posters/>
- Mougeot, G., Li, D., & Jia, S. (2019). A deep learning approach for dog face verification and recognition. En *2019 IEEE International Conference on Image Processing (ICIP)* (pp. 418-430). IEEE. https://doi.org/10.1007/978-3-030-29894-4_34
- MyScale. (2024, 2 de febrero). *Mastering efficient similarity search with Faiss vector database*. <https://myscale.com/blog/mastering-efficient-similarity-search-faiss-vector-database/>
- NMAS. (2024, 29 de febrero). *¿Cuántos perros y gatos callejeros hay en México en 2024? Esta es la cifra de abandonados*. <https://www.nmas.com.mx/nacional/cuantos-perros-gatos-callejeros-hay-mexico-2024-cifra-abandonados/>
- Pan, H., Pang, Z., Wang, Y., Wang, Y., & Chen, L. (2020). A new image recognition and classification method combining transfer learning algorithm and MobileNet model for welding defects. *IEEE Access*, 8, 118845-118855.
- Pan, S. J., & Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10), 1345-1359. <https://doi.org/10.1109/TKDE.2009.191>
- PawMaw. (s.f.). *Lost pet statistics: Understanding the numbers*. Recuperado el 7 de julio de 2024, de <https://www.pawmaw.com/blog/lost-pet-statistics>
- Perros perdidos, encontrados y en adopción Querétaro. (2025, 6 de julio). [Publicación sobre perro encontrado en El Pueblito, Querétaro]. Facebook. <https://www.facebook.com/share/p/1Ca1iiUCqc/>
- Perros perdidos Mexico. (s.f.). *Pestaña de Inicio* [Página de Facebook]. Facebook. Recuperado el 4 de abril de 2025, de <https://www.facebook.com/groups/perrosperdidosmexico/>
- Perros perdidos C.D.M.X(D.F) y Edo. Mex. (s.f.). *Pestaña de Inicio* [Página de Facebook]. Facebook. Recuperado el 4 de abril de 2025, de <https://www.facebook.com/groups/150252431688929/>

- Perros perdidos, encontrados y en adopción de Querétaro. (s.f.). *Pestaña de Inicio* [Página de Facebook]. Facebook. Recuperado el 4 de abril de 2025, de <https://www.facebook.com/groups/161669840908850/>
- Petco Love. (s.f.). *Petco Love Lost*. Recuperado el 7 de julio de 2024, de <https://petcolove.org/>
- Petco Love. (2025, 2 de abril). *Free national lost and found pet database, Petco Love Lost, wins three prestigious Anthem Awards for outstanding technology*. <https://petcolove.org/newsroom/news-story/free-national-lost-and-found-pet-database-petco-love-lost-wins-three/>
- Pietikäinen, M., & Silven, O. (2022). *Challenges of artificial intelligence—from machine learning and computer vision to emotional intelligence*. ArXiv. <https://urn.fi/URN:ISBN:9789526231990>
- Quraishi, A. H. (2025, 17 de abril). *How AI is using facial recognition to help bring lost pets home*. CBS News. <https://www.cbsnews.com/news/ai-lost-pets-petco-love-facial-recognition/>
- Roboflow. (2023, 29 de noviembre). *OpenCV color spaces and conversion: An introduction*. <https://blog.roboflow.com/opencv-color-spaces/>
- Rousseau, F. A., & Arsigny, V. (2020). Residual networks as flows of diffeomorphisms. *Journal of Mathematical Imaging and Vision*, 62, 753–764. <https://doi.org/10.1007/s10851-019-00890-3>
- Rui, Y., Huang, T. S., & Chang, S. F. (1999). Image retrieval: Current techniques, promising directions, and open issues. *Journal of Visual Communication and Image Representation*, 10(1), 39-62. <https://doi.org/10.1006/jvci.1999.0413>
- Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). MobileNetV2: Inverted residuals and linear bottlenecks. En *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 4510-4520). IEEE. <https://doi.org/10.1109/CVPR.2018.00474>

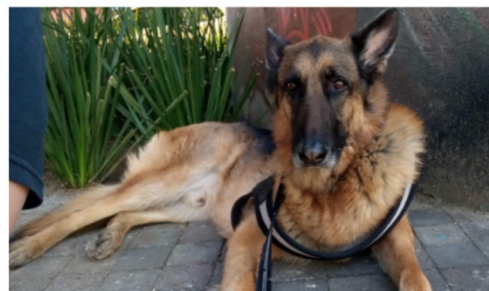
- Seidaliyeva, U., Akhmetov, D., Ilipbayeva, L., & Matson, E. T. (2020). Real-time and accurate drone detection in a video with a static background. *Sensors*, 20(17), 4936.
<https://doi.org/10.3390/s20174936>
- Sukri, S. S. A., & Ruhaiyem, N. I. R. (2019). iFR: A new framework for real-time face recognition with deep learning. En A. A. Abdullah, N. M. F. Qurniawan, A. W. B. A. W. G., & T. S. S. T. Jemrin (Eds.), *Advances in Visual Informatics* (pp. 279-290). Springer.
<https://doi.org/10.3390/s20143856>
- Sundeeep, T., Divyasree, U., Tejaswi, K., Vinithanjali, U. R., & Kumar, A. K. (2023). Feature extraction of ophthalmic images using deep learning and machine learning algorithms. *Applied Sciences*, 13(10), 6214. <https://doi.org/10.3390/ASEC2023-15231>
- Superb AI. (2023, 14 de marzo). *A practical guide to feature embeddings for ML engineers*.
<https://archive.superb-ai.com/blog/a-practical-guide-to-feature-embeddings-for-ml-engineers/>
- Taye, M. M. (2023). Theoretical understanding of convolutional neural network: Concepts, architectures, applications, future directions. En *Computation and Recognition* (Vol. 1). EAI. <https://doi.org/10.3390/computation11030052>
- TensorFlow. (s.f.). *TensorFlow*. Recuperado el 7 de julio de 2024, de <https://www.tensorflow.org>
- Ubicapet. (s.f.). *Ubicapet*. Recuperado el 7 de julio de 2024, de <https://ubicapet.com.mx/>
- Villagrán, A. (2021). *Iniciativa con proyecto de decreto, por el que se reforman y adicionan diversas disposiciones de la ley de protección a los animales de la Ciudad de México*. Congreso de la Ciudad de México.
<https://www.congresocdmx.gob.mx/media/documentos/10614da616026d42c4f3eb1049722bddd1f1d7ab.pdf>
- Weiss, E., Slater, M., & Lord, L. K. (2012). Frequency of lost dogs and cats in the United States and the methods used to locate them. *Animals*, 2(2), 226-243.
<https://doi.org/10.3390/ani2020301>

- Zahnow, R., Verrier, J., Hames, S., & Corcoran, J. (2024). Mapping and measuring neighbourhood social media groups: The case of Facebook. *Computers, Environment and Urban Systems*, 108, 102061. <https://doi.org/10.1016/j.apgeog.2024.103415>
- Zou, D. N., Zhang, S. H., Mu, T. J., & Zhang, M. (2020). A new dataset of dog breed images and a benchmark for finegrained classification. *Computational Visual Media*, 6, 155–168. <https://doi.org/10.1007/s41095-020-0184-6>

Anexo A. Resultados de búsqueda de similitud en imágenes

Este anexo complementa la sección de pruebas gráficas presentando dos casos adicionales de búsqueda de similitud. En cada ejemplo se muestra la imagen de consulta junto con las imágenes recuperadas por el sistema, ordenadas por nivel de similitud. Estos casos permiten observar visualmente el comportamiento del modelo en distintos escenarios y evaluar la coherencia de los resultados obtenidos.

Imagen de consulta:



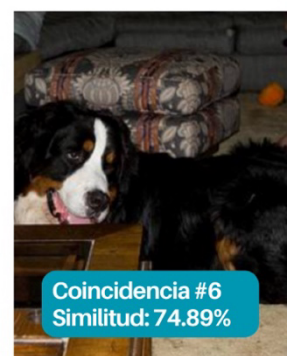
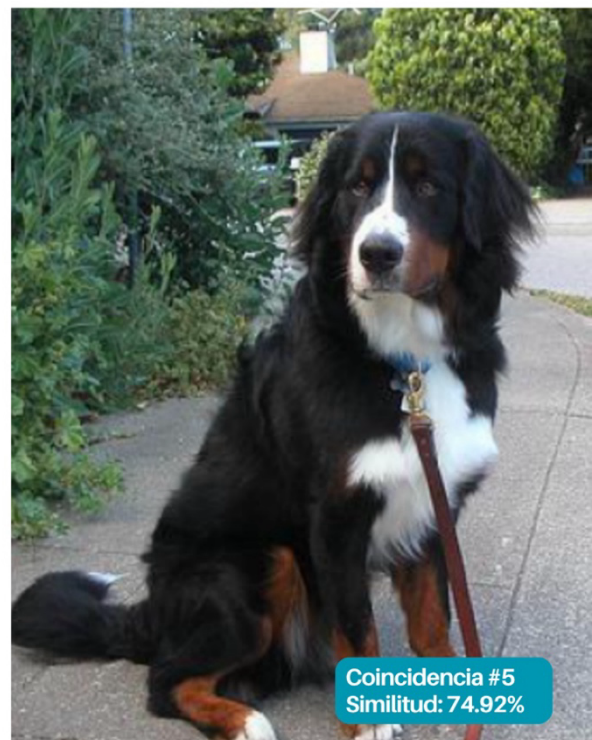
Coincidencias:



Imagen de consulta:



Coincidencias:

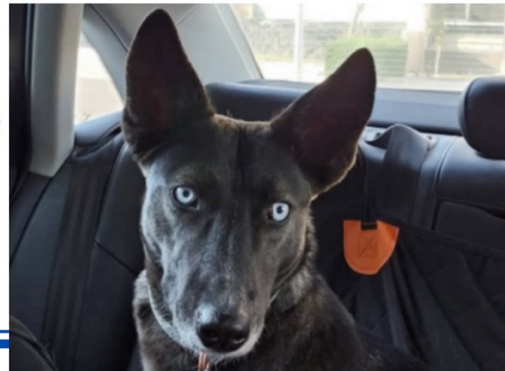


Volver a sección de Ejemplos de resultados de la búsqueda de similitud.

Anexo B. Resultados pruebas con imágenes desafiantes

Este anexo presenta los resultados visuales obtenidos durante las pruebas con imágenes consideradas desafiantes, como aquellas con mala iluminación, baja calidad, perros mestizos o de pelaje oscuro.

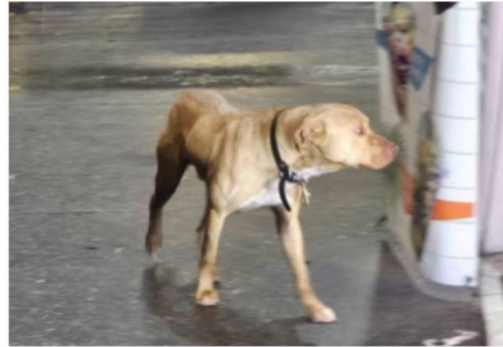
Imagen de consulta de perro de color negro:



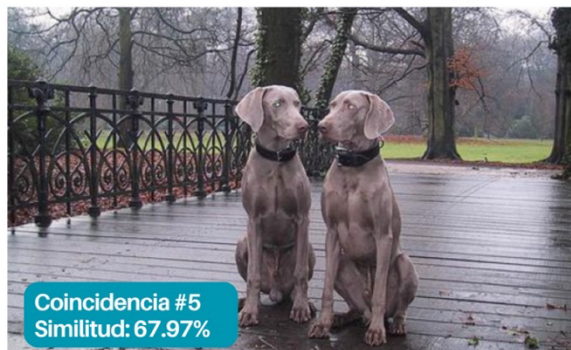
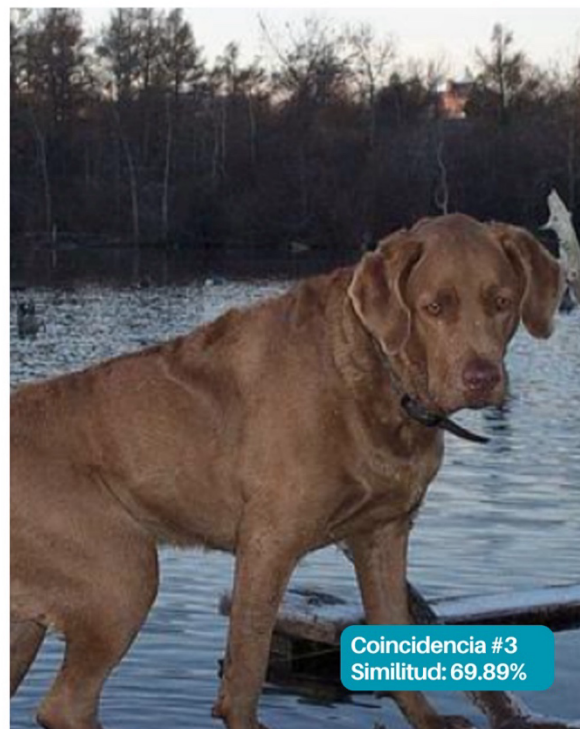
Coincidencias:



Imagen de consulta con mala calidad:



Coincidencias:



Volver a sección de Pruebas con imágenes desafiantes.

Anexo C. Matriz de pruebas y requisitos relacionados

A continuación, se presenta una matriz que facilita la visualización rápida del alcance de las pruebas y su correspondencia con los requisitos establecidos, evidenciando cómo se ha verificado el cumplimiento integral de los objetivos del sistema.

N	Secciones de pruebas	Pruebas	Requisitos relacionados
1	Preparación inicial y modelo	No aplica	No aplica
2	Pruebas funcionales y carga de datos	Carga Inicial de Imágenes y Persistencia	RF1, RF4.1, RF4.2, RNF5.1
		Detección de Duplicados - Verificación por Ruta	RF2.1
		Detección de Duplicados - Verificación por similitud de características	RF2.2, RF2.3, RF2.4, RNF2.1
		Detección de Duplicados - Carga y Sincronización del Índice FAISS	RF4.3, RF4.4
3	Pruebas de Búsqueda de Similitud:	Procesamiento de la imagen de consulta	RF3.1
		Recuperación de Resultados y Aplicación del Umbral	RF3.2, RF3.4
		Ordenamiento y Visualización de Resultados	RF3.3
		Pruebas con imágenes desafiantes	RNF2
4	Evaluación de Rendimiento y Precisión:	Latencia de Búsqueda	RNF1.1
		Escalabilidad del Registro	RNF1.2
		Evaluación de Precisión - Precisión en Detección de Duplicados	RNF2.1
		Evaluación de Precisión - F1-score en Búsqueda de Similitud y determinación del umbral óptimo	RNF2.2
		Evaluación de Precisión - Robustez a Imágenes No Pertenecientes a la Clase	RNF2.3
		Optimización para Entornos de Recursos Limitados - Huella de Memoria del Modelo	RNF6.1
		Optimización para Entornos de Recursos Limitados - Eficiencia de Inferencia del Modelo	RNF6.2
		Optimización para Entornos de Recursos Limitados - Tamaño del Modelo	RNF6.3

Volver a la sección de Pruebas y evaluación.

Anexo D. Matriz de relación: requisitos, pruebas y resultados.

A continuación, se presenta una matriz de trazabilidad que muestra de manera clara cómo se relacionan los objetivos del proyecto con los requisitos, las secciones donde se explica su implementación, las pruebas realizadas y el estado de cumplimiento.

N	Objetivo específico	Requisitos relacionados	Apartado en la memoria	Pruebas realizadas	Estado de cumplimiento
1	Investigar y seleccionar la arquitectura de modelo de reconocimiento de imágenes más adecuada para la extracción de características, justificando la elección en base a precisión, eficiencia y compatibilidad considerando las limitaciones propias de entornos con recursos limitados.	RNF6: Optimización para Entornos de Recursos Limitados	5.1. Análisis comparativo de modelos de reconocimiento de imágenes y justificación del modelo elegido	6.4.4 Optimización para Entornos de Recursos Limitados	Parcialmente alcanzado
2	Recopilar y conformar un conjunto de datos de imágenes de perros diverso y representativo a partir de múltiples fuentes.	RF1.1 - RF1.4 Registro automático de imágenes RNF4: Disponibilidad de datos	5.2. Recolección y preparación del dataset de imágenes 5.4.4. Preprocesamiento de imágenes 5.7. Detección y prevención de duplicados	6.2.1 Carga Inicial de Imágenes y Persistencia	Alcanzado
3	Diseñar e implementar una arquitectura de software modular y mantenible que estructure los componentes del sistema para facilitar su desarrollo, escalabilidad y futura evolución.	RNF3: Mantenibilidad RNF1.2: Escalabilidad del Registro	5.3. Estructura modular del desarrollo y organización del código	6.4.2 Escalabilidad del Registro	Alcanzado
4	Implementar el modelo de reconocimiento de imágenes seleccionado como un extractor de características, asegurando el preprocesamiento adecuado y la generación de embeddings normalizados.	RF1.2: Extracción de Características RNF6.2: Eficiencia de Inferencia del Modelo RNF2.2: F1-score en Búsqueda de similitud	5.4. Implementación del modelo de reconocimiento	6.4.3.2 F1-score en Búsqueda de Similitud y determinación del umbral óptimo, 6.4.4.2 Eficiencia de Inferencia del Modelo	Alcanzado
5	Diseñar e implementar un sistema robusto de gestión de base de datos para el almacenamiento persistente y el registro automático de imágenes de perros.	RF1.3: Almacenamiento en Base de Datos RNF5: Persistencia	5.5. Implementación de la base de datos 5.6 Registro automático de imágenes	6.2 Pruebas funcionales y carga de datos	Alcanzado
6	Desarrollar e integrar un mecanismo de detección de duplicados optimizado en el proceso de registro	RF2.1 – RF2.4: Detección de Duplicados RNF2.1: Precisión en Detección de Duplicados RNF1.2: Escalabilidad del registro	5.7. Detección y prevención de duplicados	6.2.2 Detección de Duplicados, 6.4.3.1 Precisión en Detección de Duplicados, 6.4.4.2 Eficiencia de Inferencia del Modelo	Parcialmente alcanzado
7	Desarrollar un módulo de búsqueda que permita identificar imágenes similares de manera rápida y eficiente en grandes volúmenes de datos, utilizando técnicas avanzadas de indexación y consulta.	RF3.1- RF3.4: Búsqueda de Perros Similares RF4.1- RF4.4: Gestión del Índice de Búsqueda RNF2.3: Robustez a imágenes no pertenecientes	5.8. Implementación de la búsqueda de similitud	6.3 Pruebas de búsqueda de similitud 6.3.5 Pruebas con imágenes desafiantes 6.4.2.3 Carga y Sincronización del Índice FAISS	Parcialmente alcanzado
8	Evaluar y validar el rendimiento global del sistema para justificar la efectividad de los mecanismos de búsqueda de similitud y detección de duplicados, mediante la aplicación de un protocolo de pruebas y el análisis de métricas de desempeño	RNF1: Rendimiento RNF2: Precisión RNF6.1 – RNF6.4: Optimización para entornos de recursos limitados	6.4 Evaluación y rendimiento	6.4.4. Optimización para entornos de recursos limitados	Parcialmente alcanzado