

# Sistema de Monitoreo y Alerta Temprana basado en Inteligencia Artificial para Áreas Protegidas

Autores:

Fabrizio Martin Contigiani  
Gabriel Orlando Da Silva Schmies

Tutor:

Dr. Ing. Sergio Eduardo Moya

27 de diciembre de 2025

## Resumen

El presente trabajo describe el diseño e implementación de un sistema de monitoreo y alerta temprana para áreas protegidas de la Selva Misionera, que combina tecnologías de Internet de las Cosas (IoT) con Inteligencia Artificial (IA) para la detección automática de fauna silvestre, personas y vehículos.

El sistema está compuesto por una red de nodos de captura basados en microcontroladores ESP32 equipados con cámaras, los cuales se comunican mediante el protocolo ESP-MESH para transmitir imágenes hacia un nodo raíz. Este nodo actúa como puerta de enlace, reenviando las imágenes a un servidor remoto a través de Transmission Control Protocol (TCP)/IP.

En el servidor, las imágenes son procesadas por un servicio de inferencia basado en SpeciesNet, un modelo de detección de objetos desarrollado por Google que utiliza YOLOv5. Este modelo permite identificar y clasificar especies de fauna silvestre, así como detectar la presencia de humanos y vehículos, generando alertas automáticas ante posibles intrusiones.

La arquitectura del servidor incluye una aplicación web desarrollada en Django para la gestión de imágenes, un bot de Telegram para el envío de notificaciones en tiempo real, y una base de datos PostgreSQL para el almacenamiento persistente. Todo el sistema está contenedorizado mediante Docker para facilitar su despliegue.

Los resultados demuestran la viabilidad de implementar un sistema de vigilancia inteligente de bajo costo para áreas protegidas, capaz de operar de manera autónoma y alertar a los administradores ante eventos relevantes.

**Palabras Clave** - Cámaras Trampa, Internet de las Cosas, Inteligencia Artificial, Monitoreo de Fauna, Detección de Intrusos, ESP-MESH, SpeciesNet, Selva Misionera

# Índice general

Índice de figuras	5
Índice de tablas	6
Glosario	7
Siglas	10
<b>1. Introducción</b>	<b>11</b>
1.1. Contexto y motivación . . . . .	11
1.2. Estructura del documento . . . . .	12
<b>2. Antecedentes</b>	<b>13</b>
2.1. Trabajos relacionados . . . . .	13
2.2. Estado del arte . . . . .	14
2.3. Soluciones comerciales existentes . . . . .	15
2.4. Análisis comparativo . . . . .	16
<b>3. Planteamiento del Problema</b>	<b>17</b>
3.1. Contexto regional . . . . .	17
3.1.1. La Selva Misionera y el Bosque Atlántico . . . . .	17
3.1.2. Biodiversidad y especies en peligro . . . . .	17
3.1.3. Importancia ecológica de la región . . . . .	17
3.2. Problemáticas de conservación en la región . . . . .	18
3.2.1. Deforestación y fragmentación del hábitat . . . . .	18
3.2.2. Caza furtiva y tráfico de fauna . . . . .	18
3.2.3. Intrusión en áreas protegidas . . . . .	18
3.2.4. Recursos limitados para vigilancia . . . . .	18
3.3. Sistemas de monitoreo actuales . . . . .	18
3.3.1. Cámaras trampa convencionales . . . . .	19
3.3.2. Limitaciones de los sistemas existentes . . . . .	19
3.4. Justificación del proyecto . . . . .	19
<b>4. Objetivos y Alcance</b>	<b>20</b>
4.1. Objetivo general . . . . .	20
4.2. Objetivos específicos . . . . .	20
4.3. Alcance del proyecto . . . . .	20

4.4. Limitaciones . . . . .	20
<b>5. Marco Teórico</b>	<b>21</b>
5.1. Internet de las Cosas (IoT) . . . . .	21
5.1.1. Arquitecturas IoT . . . . .	21
5.1.2. Protocolos de comunicación inalámbrica . . . . .	21
5.2. Redes Mesh . . . . .	22
5.2.1. Topologías de red . . . . .	22
5.2.2. ESP-MESH y Mwifi . . . . .	22
5.3. Inteligencia Artificial aplicada a visión por computadora . . . . .	22
5.3.1. Redes neuronales convolucionales (CNN) . . . . .	22
5.3.2. Detección de objetos con YOLO . . . . .	22
5.3.3. SpeciesNet de Google . . . . .	23
5.4. Tecnologías de desarrollo . . . . .	23
5.4.1. Microcontroladores ESP32 . . . . .	23
5.4.2. ESP-IDF y ESP-MDF . . . . .	23
5.4.3. Contenedorización con Docker . . . . .	23
5.4.4. Framework Django . . . . .	23
5.5. Sensores y actuadores . . . . .	24
5.5.1. Sensores de movimiento PIR . . . . .	24
5.5.2. Módulos de cámara . . . . .	24
5.6. Diseño y fabricación digital . . . . .	24
5.6.1. Modelado 3D . . . . .	24
5.6.2. Fabricación aditiva . . . . .	24
<b>6. Metodología</b>	<b>25</b>
6.1. Enfoque metodológico . . . . .	25
6.2. Etapas del desarrollo . . . . .	25
6.3. Herramientas y tecnologías utilizadas . . . . .	25
6.4. Métricas de evaluación . . . . .	25
<b>7. Diseño del Sistema</b>	<b>26</b>
7.1. Arquitectura general . . . . .	27
7.2. Diseño del hardware . . . . .	27
7.2.1. Selección de componentes . . . . .	27
7.2.2. Nodo de captura con cámara . . . . .	27
7.2.3. Sensor de movimiento PIR . . . . .	27
7.2.4. Carcasa para impresión 3D . . . . .	27
7.2.5. Nodo raíz . . . . .	27
7.2.6. Alimentación y consumo energético . . . . .	27
7.3. Diseño de la red mesh . . . . .	27
7.3.1. Topología de la red . . . . .	27
7.3.2. Protocolo de comunicación . . . . .	27
7.3.3. Formato de datos . . . . .	27
7.4. Diseño del servicio de detección . . . . .	27
7.4.1. Servidor de inferencia con SpeciesNet . . . . .	27
7.4.2. Detección de animales, humanos y vehículos . . . . .	27

7.4.3.	Anotación de imágenes con bounding boxes . . . . .	27
7.5.	Diseño del servidor de aplicación . . . . .	27
7.5.1.	Arquitectura de servicios . . . . .	27
7.5.2.	Gestión de imágenes . . . . .	27
7.5.3.	Interfaz web . . . . .	27
7.5.4.	Bot de Telegram y sistema de alertas . . . . .	27
<b>8.</b>	<b>Implementación</b>	<b>28</b>
8.1.	Nodo mesh (mesh-node) . . . . .	29
8.1.1.	Firmware del nodo de captura . . . . .	29
8.1.2.	Integración del sensor PIR . . . . .	29
8.1.3.	Captura de imágenes . . . . .	29
8.1.4.	Compresión y transmisión . . . . .	29
8.1.5.	Fabricación de la carcasa 3D . . . . .	29
8.2.	Nodo raíz (root-node) . . . . .	29
8.2.1.	Firmware del nodo raíz . . . . .	29
8.2.2.	Conexión con servidor TCP . . . . .	29
8.2.3.	Gestión de la red mesh . . . . .	29
8.3.	Servicio de detección (wildlife-detection) . . . . .	29
8.3.1.	Contenedor Docker con SpeciesNet . . . . .	29
8.3.2.	API de inferencia con LitServe . . . . .	29
8.3.3.	Procesamiento de imágenes . . . . .	29
8.4.	Servidor de aplicación (server) . . . . .	29
8.4.1.	Aplicación Django . . . . .	29
8.4.2.	Integración con SpeciesNet . . . . .	29
8.4.3.	Bot de Telegram y sistema de alertas . . . . .	29
8.4.4.	Base de datos PostgreSQL . . . . .	29
8.4.5.	Despliegue con Docker Compose . . . . .	29
<b>9.</b>	<b>Pruebas y Resultados</b>	<b>30</b>
9.1.	Ambiente de pruebas . . . . .	30
9.2.	Pruebas de conectividad y red mesh . . . . .	30
9.2.1.	Alcance de la red . . . . .	30
9.2.2.	Latencia de transmisión . . . . .	30
9.2.3.	Estabilidad de la conexión . . . . .	30
9.3.	Pruebas de detección . . . . .	30
9.3.1.	Detección de fauna silvestre . . . . .	30
9.3.2.	Detección de humanos . . . . .	30
9.3.3.	Detección de vehículos . . . . .	30
9.4.	Evaluación del modelo de IA . . . . .	30
9.4.1.	Precisión y recall . . . . .	30
9.4.2.	Tiempo de inferencia . . . . .	30
9.5.	Pruebas de consumo energético . . . . .	30
9.6.	Pruebas del sistema de alertas . . . . .	30
9.6.1.	Tiempo de respuesta . . . . .	30
9.7.	Análisis de resultados . . . . .	30

<b>10. Conclusiones</b>	<b>31</b>
10.1. Conclusiones generales . . . . .	31
10.2. Aportes del trabajo . . . . .	31
10.3. Trabajos futuros . . . . .	31
10.4. Recomendaciones . . . . .	31
<b>11. Bibliografía</b>	<b>32</b>
<b>A. Esquemáticos del hardware</b>	<b>34</b>
<b>B. Código fuente relevante</b>	<b>35</b>
B.1. Firmware del nodo mesh . . . . .	35
B.2. Firmware del nodo raíz . . . . .	35
B.3. Servidor de detección . . . . .	35
B.4. Aplicación Django . . . . .	35
<b>C. Manual de instalación y configuración</b>	<b>36</b>
C.1. Configuración del firmware . . . . .	36
C.2. Despliegue del servidor . . . . .	36
C.3. Configuración del bot de Telegram . . . . .	36
<b>D. Manual de usuario</b>	<b>37</b>
<b>E. Especificaciones técnicas</b>	<b>38</b>
<b>F. Análisis de viabilidad económica</b>	<b>39</b>

# Índice de figuras

# Índice de tablas

2.1. Comparativa de soluciones de monitoreo y trabajos relacionados. . . . .	16
--	----



# Glosario

- aprendizaje automático** Rama de la inteligencia artificial que permite a los sistemas aprender y mejorar automáticamente a partir de datos sin ser programados explícitamente, también conocido como machine learning. 13, 14, 22
- Bosque Atlántico** Ecorregión de bosques subtropicales húmedos que se extiende por Brasil, Paraguay y Argentina, conocida también como Mata Atlántica, uno de los biomas más biodiversos del planeta. 17
- bot de Telegram** Programa automatizado que interactúa con usuarios a través de la plataforma de mensajería Telegram. 1, 11
- bounding box** Rectángulo que delimita la ubicación de un objeto detectado en una imagen. 23
- CAD** Diseño Asistido por Computadora (Computer-Aided Design), software utilizado para crear modelos 2D y 3D de objetos. 24
- cámara trampa** Dispositivo de captura de imágenes activado por movimiento, utilizado comúnmente para monitoreo de fauna silvestre. 11, 13, 14, 15, 19, 23
- carcasa** Estructura protectora que alberga los componentes electrónicos de un dispositivo, protegiéndolos del ambiente externo. 24
- caza furtiva** Práctica ilegal de captura o matanza de animales silvestres sin autorización, a menudo de especies protegidas o en peligro de extinción. 11, 12, 16, 18, 19
- clasificación taxonómica** Proceso de identificar y categorizar organismos biológicos en grupos jerárquicos como familia, género y especie. 13, 15, 23
- código abierto** Software cuyo código fuente está disponible públicamente para que cualquiera pueda estudiarlo, modificarlo y distribuirlo. 12, 14, 15, 19, 23
- detección de objetos** Técnica de visión por computadora que identifica y localiza objetos dentro de una imagen, generalmente mediante bounding boxes. 11, 13
- Django** Framework de desarrollo web de alto nivel escrito en Python, que sigue el patrón modelo-vista-plantilla. 1, 21, 23
- Docker** Plataforma de contenedorización que permite empaquetar aplicaciones junto con sus dependencias para facilitar el despliegue. 1, 23

**Docker Compose** Herramienta para definir y ejecutar aplicaciones Docker multi-contenedor mediante archivos YAML. 23

**edge computing** Paradigma de computación distribuida que acerca el procesamiento y el almacenamiento de datos a la ubicación donde se necesitan, con el fin de mejorar los tiempos de respuesta y ahorrar ancho de banda, también conocido como computación en el borde. 14, 16

**ESP-IDF** Espressif IoT Development Framework, entorno oficial de desarrollo para microcontroladores ESP32. 23

**ESP-MDF** Espressif Mesh Development Framework, framework para desarrollo de redes mesh sobre ESP-IDF. 22, 23

**ESP-MESH** Protocolo de red mesh desarrollado por Espressif para microcontroladores ESP32, basado en Wi-Fi. 1, 11, 21, 22

**ESP32** Microcontrolador de bajo costo y bajo consumo con Wi-Fi y Bluetooth integrados, fabricado por Espressif Systems. 1, 11, 14, 16, 19, 21, 22, 23

**impresión 3D** Proceso de fabricación aditiva que crea objetos tridimensionales mediante la deposición de material capa por capa. 19, 24

**inferencia** Proceso de utilizar un modelo de aprendizaje automático entrenado para realizar predicciones sobre nuevos datos. 1, 14

**LoRa** Tecnología de comunicación inalámbrica de largo alcance (Long Range) y bajo consumo de energía, que utiliza modulación por espectro ensanchado. 14, 16

**LoRaWAN** Protocolo de red que utiliza la tecnología LoRa para conectar dispositivos a internet, optimizado para bajo consumo y comunicaciones de largo alcance. 14

**MegaDetector** Modelo de detección de objetos desarrollado por Microsoft para identificar automáticamente animales, personas y vehículos en imágenes de cámaras trampa. 15

**mesh** Topología de red donde cada nodo puede conectarse con múltiples nodos vecinos, permitiendo rutas alternativas para la transmisión de datos. 12, 13, 14, 16, 19, 21, 22

**Mwifi** Componente del ESP-MDF que proporciona APIs de alto nivel para comunicación en redes ESP-MESH. 22

**nodo** Dispositivo individual que forma parte de una red mesh. 1, 22

**nodo raíz** Nodo principal de una red mesh que actúa como puerta de enlace hacia redes externas. 1, 22

**PostgreSQL** Sistema de gestión de bases de datos relacional de código abierto. 1, 23

- PyTorch** Biblioteca de aprendizaje automático de código abierto basada en la biblioteca Torch, utilizada para aplicaciones como visión computacional y procesamiento de lenguaje natural. 15
- Selva Misionera** Porción argentina del Bosque Atlántico, ubicada principalmente en la Provincia de Misiones, caracterizada por su alta biodiversidad y vegetación subtropical. 1, 11, 16, 17
- sensor PIR** Sensor de Infrarrojo Pasivo que detecta cambios en la radiación infrarroja emitida por objetos en movimiento, comúnmente utilizado para detectar presencia de animales o personas. 24
- SpeciesNet** Modelo de detección y clasificación de fauna silvestre desarrollado por Google, basado en YOLOv5. 1, 11, 13, 15, 19, 23
- tiempo real** Procesamiento de datos que ocurre con latencia mínima, permitiendo respuestas inmediatas a eventos. 11, 12, 13, 14, 16, 19
- visión por computadora** Campo de la inteligencia artificial que permite a las computadoras interpretar y comprender información visual del mundo real. 12, 13, 22
- YOLOv5** Versión 5 del modelo You Only Look Once, arquitectura de red neuronal para detección de objetos en tiempo real. 1, 13, 14, 23

# Siglas

**CNN** Red Neuronal Convolucional. 13, 22

**IA** Inteligencia Artificial. 1, 11, 12, 13, 14, 15, 16, 19, 23

**IoT** Internet de las Cosas. 1, 11, 12, 21

**TCP** Transmission Control Protocol. 1

**YOLO** You Only Look Once. 13, 14, 22

# Capítulo 1

## Introducción

La conservación de la fauna silvestre y la protección de áreas naturales representan desafíos críticos en la actualidad. En regiones de alta biodiversidad como la Selva Misionera, la pérdida de hábitat, la caza furtiva y la intrusión humana en ecosistemas protegidos amenazan el equilibrio ecológico y la supervivencia de numerosas especies. Ante esta problemática, surge la necesidad de implementar sistemas de monitoreo que permitan vigilar estas áreas de manera continua y eficiente.

Tradicionalmente, el monitoreo de fauna silvestre se ha realizado mediante cámara trampa, dispositivos que capturan imágenes cuando detectan movimiento. Sin embargo, estos sistemas convencionales presentan limitaciones significativas: requieren revisión manual periódica, no permiten alertas en tiempo real, y generan grandes volúmenes de datos que deben ser analizados manualmente por expertos.

El avance de tecnologías como el IoT y la IA ofrece nuevas posibilidades para superar estas limitaciones. La combinación de redes de sensores inalámbricos con algoritmos de detección de objetos permite desarrollar sistemas capaces de identificar automáticamente fauna silvestre, personas y vehículos, generando alertas inmediatas ante eventos relevantes.

El presente trabajo, desarrollado en el marco de la Universidad Nacional de Misiones, propone el diseño e implementación de un sistema de monitoreo y alerta temprana para áreas protegidas de la región, que integra una red de nodos de captura basados en ESP32 comunicados mediante ESP-MESH, un servidor de procesamiento con IA basado en SpeciesNet, y un sistema de notificaciones a través de bot de Telegram.

### 1.1. Contexto y motivación

Las áreas protegidas enfrentan amenazas constantes que van desde la caza ilegal de especies en peligro hasta la intrusión de personas no autorizadas y vehículos en zonas restringidas. Los guardaparques y administradores de estas áreas frecuentemente carecen de los recursos humanos y tecnológicos necesarios para mantener una vigilancia efectiva sobre extensas superficies de terreno, muchas veces en ubicaciones remotas con acceso limitado a infraestructura de comunicaciones.

Las cámaras trampa tradicionales, aunque útiles para la investigación científica, no fueron diseñadas para la vigilancia en tiempo real. Las imágenes capturadas permanecen almacenadas en tarjetas de memoria que deben ser recolectadas físicamente, lo que implica visitas frecuentes al campo y retrasos significativos entre la captura de un evento y su descubrimiento. Para

cuando se detecta una intrusión o un acto de caza furtiva, los responsables ya se encuentran lejos del área.

Es cierto que en la actualidad existen cámaras trampa más modernas que incorporan conectividad celular o satelital, permitiendo el envío remoto de imágenes. Sin embargo, estos dispositivos suelen tener un costo significativamente mayor, lo que limita su adopción masiva especialmente en países en vías de desarrollo, donde paradójicamente se concentra gran parte de la biodiversidad mundial. Además, muchas de estas soluciones comerciales dependen de servicios en la nube propietarios con costos de suscripción recurrentes.

La motivación de este proyecto surge de la necesidad de transformar el paradigma del monitoreo pasivo hacia un sistema activo de vigilancia inteligente, pero de manera accesible y de bajo costo. Un sistema que no solo capture imágenes, sino que las transmita en tiempo real, las analice automáticamente mediante algoritmos de IA, y genere alertas inmediatas cuando se detecten eventos de interés, ya sea la presencia de fauna silvestre para fines de investigación, o la detección de intrusos para fines de seguridad. Todo esto utilizando componentes de hardware económicos y software de código abierto.

## 1.2. Estructura del documento

El presente documento se organiza en diez capítulos que describen de manera progresiva el desarrollo del sistema propuesto:

**Capítulo 1 - Introducción:** Presenta el contexto general del proyecto, la motivación y la estructura del documento.

**Capítulo 2 - Antecedentes:** Revisa trabajos relacionados, soluciones comerciales existentes y el estado del arte en sistemas de monitoreo de fauna y detección de intrusos.

**Capítulo 3 - Planteamiento del Problema:** Describe la problemática de las áreas protegidas, las limitaciones de los sistemas tradicionales y la justificación del proyecto.

**Capítulo 4 - Objetivos y Alcance:** Define los objetivos generales y específicos, así como el alcance y las limitaciones del trabajo.

**Capítulo 5 - Marco Teórico:** Presenta los fundamentos teóricos sobre IoT, redes mesh, IA aplicada a visión por computadora, y las tecnologías utilizadas.

**Capítulo 6 - Metodología:** Describe el enfoque metodológico, las etapas de desarrollo y las herramientas empleadas.

**Capítulo 7 - Diseño del Sistema:** Detalla la arquitectura general, el diseño del hardware, la red mesh, el servicio de detección y el servidor de aplicación.

**Capítulo 8 - Implementación:** Presenta la implementación de cada componente: nodos mesh, nodo raíz, servicio de detección y servidor de aplicación.

**Capítulo 9 - Pruebas y Resultados:** Documenta las pruebas realizadas y analiza los resultados obtenidos en conectividad, detección, rendimiento y consumo energético.

**Capítulo 10 - Conclusiones:** Resume las conclusiones generales, los aportes del trabajo, trabajos futuros y recomendaciones.

# Capítulo 2

## Antecedentes

En los últimos años, el campo de la IA aplicada a la visión por computadora ha experimentado avances significativos. El desarrollo de arquitecturas de Redes Neuronales Convolucionales (CNNs) cada vez más eficientes, junto con la disponibilidad de grandes conjuntos de datos de entrenamiento, ha permitido crear modelos capaces de detectar y clasificar objetos en imágenes con una precisión sin precedentes [1]. Algoritmos como You Only Look Once (YOLO) [2] y sus sucesivas versiones, incluyendo YOLOv5 [3], han revolucionado la detección de objetos, permitiendo el procesamiento en tiempo real incluso en dispositivos con recursos limitados.

En el ámbito específico del monitoreo de fauna silvestre, estos avances han dado lugar a modelos especializados como SpeciesNet, desarrollado por Google [4], que combina detección de objetos con clasificación taxonómica para identificar especies a partir de imágenes de cámara trampa. Estos desarrollos han abierto nuevas posibilidades para automatizar el análisis de las enormes cantidades de imágenes que generan los sistemas de monitoreo.

Paralelamente, la tecnología de cámara trampa ha evolucionado desde dispositivos autónomos que almacenan imágenes localmente, hasta sistemas más sofisticados con conectividad celular o satelital que permiten la transmisión remota de datos. Este capítulo examina tanto los avances en IA aplicada a la visión por computadora, como la evolución de las soluciones de monitoreo de fauna, incluyendo sistemas comerciales y proyectos de investigación relacionados.

### 2.1. Trabajos relacionados

En el ámbito local, Barrero y Schmunck [5] desarrollaron en la Universidad Nacional de Misiones una microcámara de vigilancia orientada a la protección de fauna salvaje. Este trabajo propuso el diseño de un dispositivo compacto basado en microcontroladores capaz de capturar imágenes en áreas naturales. El proyecto sentó las bases para el desarrollo de soluciones de bajo costo adaptadas a las necesidades específicas de la región misionera, demostrando la viabilidad de utilizar hardware económico para aplicaciones de monitoreo ambiental. El presente trabajo extiende estos conceptos incorporando comunicación en red mesh, procesamiento con IA y un sistema de alertas en tiempo real.

En el contexto regional argentino, González et al. [6] presentaron en el Workshop de Investigadores en Ciencias de la Computación (WICC 2022) un sistema de IA para la multi-clasificación de fauna en fotografías automáticas utilizadas en investigación científica. Este trabajo demuestra el creciente interés en la comunidad científica argentina por aplicar técnicas de aprendizaje automático al análisis de imágenes de cámara trampa, estableciendo un

precedente importante para proyectos de monitoreo de fauna en el país.

En el contexto internacional, diversos trabajos han explorado el uso de YOLOv5 para el análisis automatizado de imágenes de cámara trampa. Abood et al. [7] presentaron un enfoque innovador en la conferencia IEEE ICMNWC 2023, demostrando que los modelos de la familia YOLO pueden adaptarse eficazmente para la detección y clasificación de fauna silvestre. Los autores destacaron las ventajas de YOLOv5 en términos de velocidad de inferencia y precisión. De manera similar, Njathi et al. [8] propusieron en IEEE AFRICON 2023 un sistema eficiente de anotación de imágenes de cámaras trampa basado en YOLOv5, enfocándose en reducir el tiempo y esfuerzo requerido para etiquetar grandes conjuntos de datos. Ambos trabajos evidencian la idoneidad de los modelos YOLO para aplicaciones de monitoreo de fauna que requieren procesamiento eficiente de grandes volúmenes de imágenes.

Un antecedente particularmente relevante para el presente trabajo es el sistema propuesto por Whytock et al. [9], quienes desarrollaron cámaras trampa con IA capaces de enviar alertas en tiempo real a través de la red satelital Iridium. El estudio, realizado en Gabón (África Central), demostró la viabilidad de integrar procesamiento con IA directamente en dispositivos de campo para detectar fauna y enviar notificaciones inmediatas a investigadores y guardaparques. Aunque su solución utiliza conectividad satelital —con costos operativos significativos—, el concepto de alertas en tiempo real basadas en detección automática constituye un pilar fundamental del sistema propuesto en este trabajo. La diferencia principal radica en que nuestra solución emplea redes mesh locales y conectividad Wi-Fi/TCP, reduciendo considerablemente los costos de operación.

Otro proyecto directamente comparable es AiCatcher, desarrollado por Mallya [10], que propone una cámara trampa inteligente capaz de realizar inferencia directamente en el dispositivo de campo. AiCatcher utiliza una Raspberry Pi como unidad de procesamiento para ejecutar modelos de detección en el borde (edge computing), y emplea módulos LoRa (Adafruit LoRa Bonnets) configurados en una red LoRaWAN para la transmisión de datos, convirtiendo las señales recibidas en mensajes SMS mediante Twilio. Si bien este enfoque es innovador, el uso de Raspberry Pi presenta un consumo energético significativamente mayor, limitando la autonomía del dispositivo en campo. En contraste, nuestra solución emplea microcontroladores ESP32 de bajo consumo para la captura y transmisión, delegando el procesamiento de IA a un servidor remoto. Además, el uso de redes mesh en nuestra propuesta permite una topología de red más flexible y autoorganizada.

## 2.2. Estado del arte

Vélez et al. [11] realizaron una evaluación exhaustiva de las plataformas disponibles para el procesamiento de imágenes de cámara trampa utilizando IA. Su estudio, publicado en *Methods in Ecology and Evolution*, comparó múltiples herramientas en términos de precisión, facilidad de uso y requisitos computacionales. Los autores concluyeron que, si bien existen diversas opciones de código abierto y comerciales, la elección de la plataforma adecuada depende del contexto específico de cada proyecto, incluyendo el volumen de imágenes, las especies objetivo y los recursos disponibles. Este análisis proporciona un marco de referencia valioso para la selección de tecnologías en proyectos de monitoreo de fauna.

Tabak et al. [12] demostraron la aplicabilidad del aprendizaje automático para clasificar especies animales en imágenes de cámaras trampa, alcanzando precisiones superiores al 90 % en la identificación de especies comunes. Su trabajo destacó la importancia de contar con



conjuntos de datos de entrenamiento representativos de las condiciones locales para optimizar el rendimiento de los modelos.

Por su parte, Steenweg et al. [13] abordaron el desafío de escalar las redes de cámara trampa para el monitoreo de biodiversidad a nivel global. Los autores argumentaron que la integración de sensores remotos en redes coordinadas, combinada con técnicas de análisis automatizado, representa el futuro del monitoreo de fauna silvestre, permitiendo obtener datos de biodiversidad a escalas espaciales y temporales sin precedentes.

Entre las plataformas más recientes, Hernández et al. [14] presentaron PyTorch-Wildlife, un framework colaborativo de aprendizaje profundo desarrollado por Microsoft AI for Good. Esta plataforma, similar en objetivos a SpeciesNet de Google, ofrece modelos preentrenados para la detección y clasificación de fauna silvestre en imágenes de cámara trampa. PyTorch-Wildlife se distingue por su enfoque en la facilidad de uso y la integración con el ecosistema PyTorch, facilitando tanto el despliegue de modelos existentes como el entrenamiento de modelos personalizados para especies específicas. La existencia de múltiples frameworks de código abierto respaldados por grandes empresas tecnológicas evidencia la relevancia del problema y la madurez de las soluciones disponibles.

Cabe destacar también MegaDetector, desarrollado originalmente por Microsoft AI for Earth [15], que se ha convertido en una herramienta fundamental en el procesamiento de imágenes de cámara trampa. A diferencia de los clasificadores de especies, MegaDetector se especializa en la detección genérica de animales, humanos y vehículos, funcionando como un primer filtro que reduce drásticamente el volumen de imágenes a analizar. Esta herramienta ha sido adoptada por más de 60 organizaciones de conservación a nivel mundial, demostrando reducciones de hasta el 90 % en el tiempo de procesamiento de datos. Su arquitectura modular permite integrarlo como etapa previa a clasificadores más específicos como SpeciesNet.

## 2.3. Soluciones comerciales existentes

En el mercado actual, existen diversas soluciones comerciales diseñadas para el monitoreo remoto de fauna. Las cámaras trampa con conectividad celular, como las series REVEAL de Tactacam [16] y las líneas Flex de Spypoint [17], son las más extendidas. Estos dispositivos permiten capturar imágenes de alta resolución y enviarlas a través de redes LTE a una aplicación móvil propietaria. Algunas de sus características principales incluyen disparo rápido (menor a 0.5 segundos), visión nocturna por infrarrojos y, en modelos recientes, la capacidad de solicitar fotografías o videos bajo demanda.

Sin embargo, estas soluciones presentan limitaciones críticas para su adopción masiva en proyectos de conservación a gran escala o en regiones remotas. En primer lugar, dependen enteramente de la infraestructura de red celular; en áreas de selva densa como la de Misiones, la falta de cobertura inalámbrica en el interior de las reservas suele ser la norma, invalidando el uso de estos dispositivos para alertas remotas. En segundo lugar, el costo operativo es elevado, ya que además de que cada cámara en sí tiene un costo de adquisición significativo, cada dispositivo requiere un plan de datos independiente, con suscripciones mensuales que pueden oscilar entre los 5 y 15 dólares por unidad [16], [17].

En cuanto a la gestión de datos, plataformas como Wildlife Insights [18] ofrecen una infraestructura robusta basada en la nube para el almacenamiento y análisis de imágenes mediante IA. Esta herramienta permite filtrar imágenes vacías automáticamente y realizar la clasificación taxonómica de especies, facilitando la colaboración entre investigadores. No obstante, Wildlife

Insights está orientada principalmente al procesamiento posterior (post-hoc) de los datos recolectados manualmente de las tarjetas SD, y no está diseñada para la vigilancia y respuesta inmediata ante eventos de intrusión o caza furtiva en tiempo real.

## 2.4. Análisis comparativo

A fin de situar las necesidades identificadas frente a las alternativas y trabajos discutidos anteriormente, se presenta a continuación un análisis comparativo que resume las principales diferencias técnicas y operativas. En la tabla 2.1 se contrastan las características de las categorías de soluciones analizadas: cámaras trampa tradicionales, soluciones comerciales celulares [16], [17], el sistema satelital de Whytock et al. [9] y el enfoque basado en edge computing de Mallya (AiCatcher) [10].

**Tabla 2.1:** Comparativa de soluciones de monitoreo y trabajos relacionados.

Característica	Tradicional	Celular [16]	Satelital [9]	LoRa [10]
Alertas en tiempo real	No	Sí	Sí	Sí
Infraestructura requerida	Ninguna	Operador Celular	Red Satelital	Puerta de enlace LoRa
Procesamiento de IA	Post-hoc	No / Limitado	Edge (Hardware Dedicado)	Edge (Raspberry Pi)
Costo de adquisición	Bajo-Medio	Medio-Alto	Muy Alto	Medio-Alto
Costo Operativo	Bajo	Alto (Plan mensual)	Muy Alto (Iridium)	Bajo (SMS/Twilio)
Autonomía energética	Muy Alta	Media	Media-Baja	Baja (Raspberry Pi)
Escalabilidad	Laboriosa	Por suscripción	Costo-prohibitiva	Media (LoRa)
Dependencia de terceros	No	Muy Alta	Muy Alta	Media (Twilio)

Como se desprende de la comparación, si bien existen soluciones de vanguardia, el acceso a alertas inmediatas suele estar condicionado por elevados costos operativos o dependencia de infraestructura de terceros (celular o satelital). Los trabajos de investigación como AiCatcher [10] y el sistema de Whytock et al. [9] demuestran la viabilidad de la IA para el filtrado en tiempo real, pero enfrentan desafíos en cuanto a consumo energético y costos de transmisión.

En este sentido, se identifica una oportunidad para un sistema que combine la eficiencia en el consumo de los microcontroladores ESP32 con la capacidad de extensión de cobertura de las redes mesh, permitiendo un monitoreo inteligente y de bajo costo operativo adaptado a las condiciones reales observadas en la Selva Misionera.

# Capítulo 3

## Planteamiento del Problema

### 3.1. Contexto regional

La provincia de Misiones, ubicada en el extremo noreste de Argentina, alberga uno de los ecosistemas más diversos y amenazados del continente. El marco geográfico de este proyecto se sitúa en el corazón de la selva paranaense, una región que requiere estrategias de conservación urgentes y tecnificadas.

#### 3.1.1. La Selva Misionera y el Bosque Atlántico

La Selva Misionera representa el remanente continuo más extenso del Bosque Atlántico en el Cono Sur. Originalmente, este bioma cubría una vasta superficie desde el este de Brasil hasta el noreste de Argentina y el este de Paraguay. Sin embargo, debido a la expansión agrícola y forestal, hoy solo sobrevive una pequeña fracción de su extensión original. Misiones conserva cerca de un millón de hectáreas de este bosque, lo que la posiciona como un refugio crítico para la biodiversidad regional y global.

#### 3.1.2. Biodiversidad y especies en peligro

Esta región es hogar de una variedad excepcional de flora y fauna. Entre las especies más emblemáticas se encuentra el yagüareté (*Panthera onca*), monumento natural nacional y símbolo de la selva, cuya población se encuentra en peligro crítico debido a la pérdida de hábitat y la persecución directa. Otras especies de importancia incluyen el tapir, el oso hormiguero gigante, y una gran diversidad de aves y primates que dependen de la conectividad del bosque para su supervivencia. La presencia de estos grandes felinos y herbívoros es un indicador clave del estado de salud del ecosistema, pero su monitoreo en densas zonas de selva es extremadamente complejo.

#### 3.1.3. Importancia ecológica de la región

La importancia de la Selva Misionera trasciende la mera preservación de especies; actúa como regulador climático, protector de cuencas hídricas y proveedor de servicios ecosistémicos esenciales para la región. La Ley de Ordenamiento Territorial de Bosques Nativos en Argentina clasifica a gran parte de la superficie de Misiones en categorías de máxima conservación

(Categoría I - Rojo), lo que impone la necesidad de una vigilancia constante para evitar la degradación de estas áreas protegidas frente a actividades ilícitas.

## **3.2. Problemáticas de conservación en la región**

A pesar de los esfuerzos institucionales, las áreas protegidas de Misiones enfrentan amenazas persistentes que comprometen la integridad de sus ecosistemas.

### **3.2.1. Deforestación y fragmentación del hábitat**

La pérdida de cobertura boscosa por actividades agrícolas no autorizadas y la extracción ilegal de madera noble son problemas recurrentes. La fragmentación del hábitat obliga a las especies de gran tamaño a desplazarse por áreas no protegidas, aumentando el riesgo de conflictos con humanos y reduciendo la viabilidad genética de las poblaciones aisladas.

### **3.2.2. Caza furtiva y tráfico de fauna**

La caza furtiva sigue siendo uno de los mayores desafíos. A pesar de la prohibición total de la caza en la provincia, la incursión de cazadores en el interior de parques provinciales y reservas privadas es frecuente. Los cazadores utilizan técnicas de acecho y trampas que no solo afectan a las especies de interés comercial o cinegético, sino que degradan la fauna en su totalidad y ponen en riesgo la seguridad de los guardaparques.

### **3.2.3. Intrusión en áreas protegidas**

La falta de un control perimetral efectivo permite la entrada de personas y vehículos no autorizados para actividades como la pesca ilegal, la extracción de palmitos o el desmonte encubierto. Sin un sistema de alerta temprana, estas intrusiones solo se descubren a posteriori, cuando el daño ambiental ya ha sido perpetrado.

### **3.2.4. Recursos limitados para vigilancia**

El sistema de guardaparques de Misiones, aunque altamente capacitado, es insuficiente en número para cubrir los cientos de miles de hectáreas bajo su custodia. Las patrullas terrestres consumen tiempo y combustible, y están limitadas por la geografía accidentada y la densidad de la selva. La tecnología actual de monitoreo (uso de cámaras trampa tradicionales) genera datos valiosos pero no permite una respuesta operativa rápida para detener ilícitos en curso.

## **3.3. Sistemas de monitoreo actuales**

Actualmente, el monitoreo biológico y de vigilancia en Misiones depende de métodos que, si bien son efectivos para la investigación, presentan brechas operativas para la seguridad ambiental.

### 3.3.1. Cámaras trampa convencionales

El uso de cámara trampa pasivas es el estándar de oro para la investigación de fauna. Se instalan en puntos estratégicos y capturan imágenes activadas por sensores de calor y movimiento. Sin embargo, estas cámaras son dispositivos aislados; las imágenes se almacenan localmente en tarjetas SD que deben ser retiradas físicamente por personal especializado cada varios meses.

### 3.3.2. Limitaciones de los sistemas existentes

La principal limitación es el retraso temporal entre el evento y su análisis. Un acto de caza furtiva detectado por una cámara tradicional solo será descubierto meses después, impidiendo cualquier acción disuasoria o legal inmediata. Además, el alto volumen de imágenes capturadas (frecuentemente disparadas por vegetación en movimiento) requiere cientos de horas-hombre para su clasificación manual. Las soluciones comerciales con conectividad celular, aunque existen, son inviables debido a la falta de señal en zonas profundas y los altos costos operativos de mantenimiento de múltiples líneas móviles.

## 3.4. Justificación del proyecto

Dada la crítica situación de biodiversidad en Misiones y las limitaciones de los sistemas actuales, se justifica el desarrollo de una solución tecnológica que sea:

1. **De Bajo Costo:** Permitiendo un despliegue masivo con presupuesto limitado, utilizando hardware basado en ESP32 y componentes de código abierto.
2. **Con Conectividad Independiente:** Utilizando redes mesh que no requieran infraestructura celular previa, adaptándose a la topología de la selva.
3. **Inteligente y Automatizada:** Integrando modelos de IA como SpeciesNet para filtrar imágenes relevantes en tiempo real, notificando inmediatamente ante eventos críticos (intrusos o fauna en peligro).
4. **Fabricable Localmente:** Empleando tecnologías como la impresión 3D para la creación de carcasas duraderas y personalizadas.

Este proyecto no solo representa un aporte técnico en el área de redes de sensores e inteligencia artificial, sino que constituye una herramienta de aplicación directa para fortalecer la vigilancia de las áreas protegidas de la región.

# Capítulo 4

## Objetivos y Alcance

- 4.1. Objetivo general
- 4.2. Objetivos específicos
- 4.3. Alcance del proyecto
- 4.4. Limitaciones

# Capítulo 5

## Marco Teórico

### 5.1. Internet de las Cosas (IoT)

El IoT se define como la red de objetos físicos que incorporan sensores, software y otras tecnologías con el fin de conectar e intercambiar datos con otros dispositivos y sistemas a través de internet. En el contexto de la conservación ambiental, el IoT permite la creación de redes de monitoreo autónomas que pueden operar en ubicaciones remotas con mínima intervención humana.

#### 5.1.1. Arquitecturas IoT

Una arquitectura típica de IoT se compone de tres capas fundamentales:

1. **Capa de Percepción:** Formada por los sensores y actuadores que interactúan directamente con el entorno físico.
2. **Capa de Red:** Encargada de la transmisión de datos desde los nodos sensores hacia los sistemas de procesamiento.
3. **Capa de Aplicación:** Donde los datos son procesados, almacenados y presentados al usuario final.

El sistema desarrollado en esta tesis abarca estas tres capas, utilizando nodos ESP32 en la percepción, ESP-MESH en la red y un servidor Django en la aplicación.

#### 5.1.2. Protocolos de comunicación inalámbrica

La elección del protocolo de comunicación es crítica en entornos rurales. Mientras que protocolos como Bluetooth y ZigBee son ideales para corto alcance y baja potencia, y LoRa es excelente para muy largo alcance con bajo ancho de banda, las redes Wi-Fi (802.11) ofrecen un equilibrio adecuado para la transmisión de imágenes si se gestionan adecuadamente mediante una topología mesh, permitiendo extender la cobertura sin depender de una única estación base.

## 5.2. Redes Mesh

Una red de malla o mesh es una topología en la que los nodos se conectan entre sí de forma dinámica y no jerárquica, cooperando para propagar los datos a través de la red. Esta arquitectura es especialmente robusta frente a fallos de nodos individuales, ya que la red puede reconfigurarse automáticamente para encontrar nuevas rutas.

### 5.2.1. Topologías de red

A diferencia de las topologías en estrella (comunes en redes Wi-Fi domésticas donde todos los dispositivos dependen de un mismo Punto de Acceso), las redes mesh permiten que cada nodo actúe también como un repetidor. Esto facilita el despliegue en terrenos difíciles o boscosos, donde los obstáculos físicos limitan la línea de vista. La red se organiza en una estructura de árbol donde existe un nodo raíz que actúa como puerta de enlace, y nodos intermedios que transportan los datos de los nodos hoja.

### 5.2.2. ESP-MESH y Mwifi

ESP-MESH es el protocolo propietario de Espressif basado en Wi-Fi que permite conectar hasta miles de dispositivos ESP32 en una sola red. Su principal ventaja es la capacidad de autoorganización y autocuración: si un nodo intermedio falla, sus nodos descendientes buscan automáticamente un nuevo padre. Por otro lado, Mwifi es un componente del ESP-MDF que simplifica el desarrollo de aplicaciones mesh al proporcionar una capa de abstracción de alto nivel, facilitando la transmisión de paquetes TCP/UDP y la gestión de la topología desde el código de usuario.

## 5.3. Inteligencia Artificial aplicada a visión por computadora

La visión por computadora busca emular la capacidad humana de interpretar imágenes digitales. Los avances en este campo están impulsados por el aprendizaje automático y, más específicamente, por el aprendizaje profundo (deep learning).

### 5.3.1. Redes neuronales convolucionales (CNN)

Las CNNs son arquitecturas especializadas en procesar datos con una topología de cuadrícula, como las imágenes. Funcionan mediante la aplicación de filtros o núcleos que extraen características fundamentales (bordes, texturas, formas compleja) en distintas capas jerárquicas. Esta capacidad de extracción automática de características las hace ideales para la clasificación de fauna y la detección de intrusos.

### 5.3.2. Detección de objetos con YOLO

YOLO es uno de los algoritmos de detección de objetos más populares debido a su velocidad y precisión. A diferencia de otros métodos que analizan una imagen en múltiples pasadas, YOLO (*You Only Look Once*) trata la detección como un problema de regresión único, prediciendo



las bounding boxes y las probabilidades de clase simultáneamente para toda la imagen. El uso de YOLOv5 en este proyecto se justifica por su excelente equilibrio entre rendimiento computacional y exactitud, permitiendo procesar múltiples flujos de imágenes en el servidor.

### **5.3.3. SpeciesNet de Google**

SpeciesNet es un sistema de IA de código abierto desarrollado por Google [4] diseñado específicamente para el análisis de fotos de cámara trampa. Utiliza modelos preentrenados potentes para identificar cientos de especies globales, permitiendo realizar la clasificación taxonómica de manera automatizada. En este sistema, SpeciesNet actúa como el núcleo de procesamiento que valida si una imagen capturada por los nodos contiene fauna de interés, personas o vehículos.

## **5.4. Tecnologías de desarrollo**

Para la implementación del sistema se han seleccionado herramientas que garantizan la modularidad, facilidad de mantenimiento e interoperabilidad.

### **5.4.1. Microcontroladores ESP32**

El ESP32 es un System on Chip (SoC) de bajo costo y bajo consumo con Wi-Fi y Bluetooth integrados. Perteneció a la familia de microcontroladores de Espressif Systems y destaca por su procesador de doble núcleo, rica variedad de periféricos y su idoneidad para aplicaciones de IoT industriales y ambientales.

### **5.4.2. ESP-IDF y ESP-MDF**

El ESP-IDF es el entorno de desarrollo oficial para el ESP32, basado en FreeRTOS, que permite un control preciso sobre el hardware y la gestión de la energía. Sobre este entorno se construye el ESP-MDF (Mesh Development Framework), que proporciona las herramientas específicas para la creación y gestión de redes mesh, incluyendo capacidades de actualización remota (OTA).

### **5.4.3. Contenedorización con Docker**

Docker es una plataforma que permite empaquetar software en unidades estandarizadas llamadas contenedores, las cuales incluyen todo lo necesario para que la aplicación se ejecute correctamente. El uso de Docker y Docker Compose en este proyecto facilita el despliegue del servidor de procesamiento (SpeciesNet, base de datos y backend) en cualquier máquina, asegurando que el entorno de ejecución sea idéntico al de desarrollo.

### **5.4.4. Framework Django**

Django es un framework web de alto nivel escrito en Python que fomenta el desarrollo rápido y un diseño limpio. Se utiliza para construir el servidor central que recibe las imágenes de la red mesh, interactúa con el sistema de IA y gestiona la base de datos PostgreSQL. Su

arquitectura MVT (Modelo-Vista-Plantilla) permite una separación clara entre la lógica de datos y la interfaz de usuario.

## **5.5. Sensores y actuadores**

Los componentes físicos periféricos permiten la interacción del nodo con el entorno, detectando eventos y capturando información visual.

### **5.5.1. Sensores de movimiento PIR**

El sensor PIR (Passive Infrared) detecta movimiento mediante la medición de los cambios en los niveles de radiación infrarroja emitida por objetos cercanos. En este proyecto, el sensor PIR actúa como el disparador del sistema: cuando se detecta calor en movimiento (característico de humanos o animales), el microcontrolador sale de su modo de bajo consumo y activa la cámara para la captura.

### **5.5.2. Módulos de cámara**

Se utilizan módulos de cámara compatibles con el ESP32 (como la OV2640), que permiten capturar imágenes en resoluciones de hasta 2 megapíxeles. Estos módulos están integrados en placas de desarrollo como la ESP32-CAM, proporcionando una solución compacta para la adquisición visionaria de datos con un consumo controlado.

## **5.6. Diseño y fabricación digital**

La protección física del hardware es tan importante como el software en entornos de selva subtropical.

### **5.6.1. Modelado 3D**

El diseño de la carcasa se realiza mediante herramientas de CAD, permitiendo crear estructuras precisas que protejan los componentes de la humedad y la lluvia, a la vez que proporcionan aperturas para la lente de la cámara y el ángulo de visión del sensor PIR.

### **5.6.2. Fabricación aditiva**

La impresión 3D es la tecnología utilizada para materializar los diseños CAD. Permite iterar rápidamente sobre prototipos y fabricar carcasas robustas utilizando filamentos resistentes como el PETG o el ASA, que soportan mejor la radiación UV y las altas temperaturas exteriores en comparación con el PLA.

# Capítulo 6

## Metodología

- 6.1. Enfoque metodológico
- 6.2. Etapas del desarrollo
- 6.3. Herramientas y tecnologías utilizadas
- 6.4. Métricas de evaluación



# Capítulo 7

## Diseño del Sistema

### 7.1. Arquitectura general

### 7.2. Diseño del hardware

#### 7.2.1. Selección de componentes

#### 7.2.2. Nodo de captura con cámara

#### 7.2.3. Sensor de movimiento PIR

#### 7.2.4. Carcasa para impresión 3D

#### 7.2.5. Nodo raíz

#### 7.2.6. Alimentación y consumo energético

### 7.3. Diseño de la red mesh

#### 7.3.1. Topología de la red

#### 7.3.2. Protocolo de comunicación

#### 7.3.3. Formato de datos

### 7.4. Diseño del servicio de detección

#### 7.4.1. Servidor de inferencia con SpeciesNet

#### 7.4.2. Detección de animales, humanos y vehículos

#### 7.4.3. Anotación de imágenes con bounding boxes

### 7.5. Diseño del servidor de aplicación

#### 7.5.1. Arquitectura de servicios

#### 7.5.2. Gestión de imágenes

#### 7.5.3. Interfaz web

#### 7.5.4. Bot de Telegram y sistema de alertas



# Capítulo 8

## Implementación

### 8.1. Nodo mesh (mesh-node)

#### 8.1.1. Firmware del nodo de captura

#### 8.1.2. Integración del sensor PIR

#### 8.1.3. Captura de imágenes

#### 8.1.4. Compresión y transmisión

#### 8.1.5. Fabricación de la carcasa 3D

### 8.2. Nodo raíz (root-node)

#### 8.2.1. Firmware del nodo raíz

#### 8.2.2. Conexión con servidor TCP

#### 8.2.3. Gestión de la red mesh

### 8.3. Servicio de detección (wildlife-detection)

#### 8.3.1. Contenedor Docker con SpeciesNet

#### 8.3.2. API de inferencia con LitServe

#### 8.3.3. Procesamiento de imágenes

### 8.4. Servidor de aplicación (server)

#### 8.4.1. Aplicación Django

#### 8.4.2. Integración con SpeciesNet

#### 8.4.3. Bot de Telegram y sistema de alertas

#### 8.4.4. Base de datos PostgreSQL<sub>29</sub>

#### 8.4.5. Despliegue con Docker Compose

# Capítulo 9

## Pruebas y Resultados

### 9.1. Ambiente de pruebas

### 9.2. Pruebas de conectividad y red mesh

#### 9.2.1. Alcance de la red

#### 9.2.2. Latencia de transmisión

#### 9.2.3. Estabilidad de la conexión

### 9.3. Pruebas de detección

#### 9.3.1. Detección de fauna silvestre

#### 9.3.2. Detección de humanos

#### 9.3.3. Detección de vehículos

### 9.4. Evaluación del modelo de IA

#### 9.4.1. Precisión y recall

#### 9.4.2. Tiempo de inferencia

### 9.5. Pruebas de consumo energético

### 9.6. Pruebas del sistema de alertas

#### 9.6.1. Tiempo de respuesta

### 9.7. Análisis de resultados



# Capítulo 10

## Conclusiones

10.1. Conclusiones generales

10.2. Aportes del trabajo

10.3. Trabajos futuros

10.4. Recomendaciones

# Capítulo 11

## Bibliografía

- [1] S. Schneider, G. W. Taylor y S. Kremer, «Deep Learning Object Detection Methods for Ecological Camera Trap Data,» en *2018 15th Conference on Computer and Robot Vision (CRV)*, IEEE, 2018, págs. 321-328. DOI: [10.1109/CRV.2018.00052](https://doi.org/10.1109/CRV.2018.00052).
- [2] J. Redmon, S. Divvala, R. Girshick y A. Farhadi, «You Only Look Once: Unified, Real-Time Object Detection,» *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, págs. 779-788, 2016.
- [3] G. Jocher. «YOLOv5 by Ultralytics.» Repositorio oficial de YOLOv5, visitado 27 de dic. de 2024. dirección: <https://github.com/ultralytics/yolov5>.
- [4] T. Gadot et al., «To crop or not to crop: Comparing whole-image and cropped classification on a large dataset of camera trap images,» *IET Computer Vision*, 2024.
- [5] G. Barrero y A. Schmunck, «Desarrollo de una microcámara de vigilancia para la protección de la fauna salvaje,» Tesis de Grado, Universidad Nacional de Misiones, 2023. dirección: <https://drive.google.com/file/d/1jJDM-Mc8kJw3Rcgl0tf4Mq3MxNyqXW5>.
- [6] F. González et al., «Inteligencia artificial para la multi-clasificación de fauna en fotografías automáticas utilizadas en investigación científica,» en *XXIV Workshop de Investigadores en Ciencias de la Computación (WICC 2022)*, Mendoza, Argentina, abr. de 2022, ISBN: 978-987-48222-3-9. dirección: <https://sedici.unlp.edu.ar/handle/10915/144851>.
- [7] B. S. Z. Abood, M. B. M, Z. A. Almoussawi, N. Shilpa y A. M. Shakir, «Revolutionizing Wildlife Monitoring: A Novel Approach to Camera Trap Image Analysis with YOLOv5,» en *2023 3rd International Conference on Mobile Networks and Wireless Communications (ICMNWC)*, IEEE, dic. de 2023. DOI: [10.1109/ICMNWC60182.2023.10435785](https://doi.org/10.1109/ICMNWC60182.2023.10435785). dirección: <http://dx.doi.org/10.1109/ICMNWC60182.2023.10435785>.
- [8] Y. Njathi, L. Wanjiku, L. Mugambi, J. N. Kabi, G. Kiarie y C. w. Maina, «Efficient Camera Trap Image Annotation Using YOLOv5,» en *2023 IEEE AFRICON*, IEEE, sep. de 2023. DOI: [10.1109/AFRICON55910.2023.10293724](https://doi.org/10.1109/AFRICON55910.2023.10293724). dirección: <http://dx.doi.org/10.1109/AFRICON55910.2023.10293724>.
- [9] R. C. Whytock et al., «Real-time alerts from AI-enabled camera traps using the Iridium satellite network: A case-study in Gabon, Central Africa,» *Methods in Ecology and Evolution*, vol. 14, n.º 3, págs. 867-874, 2023, ISSN: 2041-210X. DOI: [10.1111/2041-210X.14036](https://doi.org/10.1111/2041-210X.14036). dirección: <http://dx.doi.org/10.1111/2041-210X.14036>.

- [10] D. Mallya. «AiCatcher: Extending machine intelligence into the wild,» visitado 27 de dic. de 2024. dirección: <https://deepakmallya.com/aicatcher>.
- [11] J. Vélez et al., «An evaluation of platforms for processing camera-trap data using artificial intelligence,» *Methods in Ecology and Evolution*, vol. 14, n.º 2, págs. 459-477, 2022, ISSN: 2041-210X. DOI: [10.1111/2041-210X.14044](https://doi.org/10.1111/2041-210X.14044). dirección: <http://dx.doi.org/10.1111/2041-210X.14044>.
- [12] M. A. Tabak et al., «Machine learning to classify animal species in camera trap images: Applications in ecology,» *Methods in Ecology and Evolution*, vol. 10, n.º 4, págs. 585-590, 2019. DOI: [10.1111/2041-210X.13120](https://doi.org/10.1111/2041-210X.13120).
- [13] R. Steenweg et al., «Scaling-up camera traps: monitoring the planet’s biodiversity with networks of remote sensors,» *Frontiers in Ecology and the Environment*, vol. 15, n.º 1, págs. 26-34, 2017. DOI: [10.1002/fee.1448](https://doi.org/10.1002/fee.1448).
- [14] A. Hernandez, Z. Miao, L. Vargas, R. Dodhia y J. Lavista, *Pytorch-Wildlife: A Collaborative Deep Learning Framework for Conservation*, 2024. arXiv: [2405.12930](https://arxiv.org/abs/2405.12930) [cs.CV]. dirección: <https://arxiv.org/abs/2405.12930>.
- [15] S. Beery, D. Morris y S. Yang, «Efficient Pipeline for Camera Trap Image Review,» *arXiv preprint arXiv:1907.06772*, 2019. dirección: <https://arxiv.org/abs/1907.06772>.
- [16] Tactacam. «Tactacam REVEAL Cellular Cameras,» visitado 27 de dic. de 2024. dirección: <https://www.revealcellcam.com/>.
- [17] Spypoint. «Spypoint Cellular Trail Cameras,» visitado 27 de dic. de 2024. dirección: <https://www.spypoint.com/>.
- [18] Wildlife Insights. «Wildlife Insights: A cloud-based platform for camera trap data,» visitado 27 de dic. de 2024. dirección: <https://www.wildlifeinsights.org/>.

## Apéndice A

### Esquemáticos del hardware

# Apéndice B

## Código fuente relevante

B.1. Firmware del nodo mesh

B.2. Firmware del nodo raíz

B.3. Servidor de detección

B.4. Aplicación Django

## Apéndice C

### Manual de instalación y configuración

C.1. Configuración del firmware

C.2. Despliegue del servidor

C.3. Configuración del bot de Telegram

## Apéndice D

### Manual de usuario

## Apéndice E

### Especificaciones técnicas



## Apéndice F

### Análisis de viabilidad económica