
This is the **published version** of the bachelor thesis:

Martínez García, Adrián. *Detección y clasificación automatizada de fauna salvaje mediante imágenes de cámaras trampa fotográficas*. Treball de Final de Grau (Universitat Autònoma de Barcelona), 2026 (Enginyeria Informàtica)

This version is available at <https://ddd.uab.cat/record/334375>

under the terms of the  license.

Detección y clasificación automatizada de fauna salvaje mediante imágenes de cámaras trampeo fotográficas

Adrián Martínez García

Resumen—En estudios de monitorización de fauna salvaje el análisis y clasificación manual de las fotografías mediante cámaras de trampeo es una tarea muy costosa. En este trabajo se desarrolla un sistema automático para la detección y clasificación de diez especies de animales diferentes mediante el uso de técnicas de aprendizaje profundo. Para ello, se utilizan diferentes métodos de aprendizaje por transferencia en modelos preentrenados basados en la arquitectura YOLOv5, junto con un preprocesado y aumento de los datos para mejorar los resultados. También se analizan diferentes configuraciones del entrenamiento y tamaños del conjunto de datos con el objetivo de optimizar el rendimiento y precisión del sistema. Además, el código desarrollado durante el proyecto se encuentra disponible en un repositorio GitHub público.

Palabras claves—Aprendizaje por transferencia, aprendizaje profundo, cámaras trampeo, clasificación de especies, inteligencia artificial.

Abstract—In wildlife monitoring studies, the manual analysis and classification of the photographs taken by camera traps is a very costly task. This work develops an automated system for the detection and classification of ten different animal species using deep learning techniques. To achieve this, different transfer learning methods are used on pretrained models based on the YOLOv5 architecture, along with data preprocessing and augmentation to improve results. Different training configurations and dataset sizes are also analysed to optimize the system's performance and accuracy. In addition, the code developed during this project is available in a public GitHub repository.

Index Terms—Artificial intelligence, camera trap, deep learning, species classification, transfer learning.



1 INTRODUCCIÓN

El uso de las cámaras de trampeo en el estudio biológico de los animales es de suma importancia, proporcionando grandes cantidades de imágenes de numerosas especies en su hábitat natural. Sin embargo, el análisis manual de este elevado número de imágenes supone un alto coste de tiempo y recursos humanos. Este proceso puede ralentizar los resultados de las investigaciones y dificultar el estudio de proyectos a gran escala.

Los recientes avances tecnológicos y popularización de la inteligencia artificial en visión por computador han abierto nuevas posibilidades en la automatización de la detección y clasificación de especies animales en este tipo de problemas. En concreto, las técnicas de aprendizaje profundo han demostrado ser capaces de resolver esta tarea y tener unos resultados con un nivel de precisión alto.



Fig. 1: Ejemplo de una cámara de trampeo utilizada en la monitorización de fauna salvaje

1.1 Objetivos

El objetivo principal de este Trabajo Final de Grado consiste en la investigación, desarrollo y análisis de un sistema basado en técnicas de aprendizaje profundo para la detección y clasificación automática de especies animales a partir de imágenes capturadas en cámaras trampa. Para ello,

- E-mail de contacto: 1601959@uab.cat
- Menció n realizada: Computaci3n
- Trabajo tutorizado por: Mehmet Oguz Mulayim (Departamento de Ciencias de la Computaci3n)
- Curso 2025/26

se utilizará la arquitectura de red neuronal convolucional (CNN) [1] YOLOv5, con el propósito de adaptar el modelo preentrenado mediante técnicas de aprendizaje por transferencia, aprovechando el conocimiento adquirido del modelo en tareas generales y especializándolo en un problema más concreto, y evaluar su capacidad de detección y clasificación en diferentes especies en entornos naturales. Este trabajo no pretende desarrollar un sistema de detección de objetos que supere los existentes, sino comprender en profundidad cómo funcionan los modelos basados en redes neuronales convolucionales aplicados a un problema de visión por computador, como es el de la monitorización de la fauna salvaje.

En el desarrollo del proyecto también se aprenderá a utilizar herramientas especializadas en aprendizaje profundo y procesamiento de imágenes, como es el caso de PyTorch o Ultralytics YOLO. También se utilizará una infraestructura proporcionada por la universidad con GPU dedicadas a acelerar el proceso de entrenamiento del modelo.

Para lograr este objetivo, se ha dividido el proyecto en varias tareas principales, como la investigación previa de los datos y arquitectura, el desarrollo del modelo de detección y clasificación y la evaluación final del sistema. Por lo tanto, el proyecto se ha dividido de la siguiente forma:

1. Investigación del estado del arte relacionado con sistemas de detección y clasificación de fauna salvaje, así como investigación y selección del conjunto de datos de cámaras trampa y diseño e implementación del preprocesado de las imágenes y las anotaciones necesarias.
2. Configuración, entrenamiento y optimización del modelo YOLOv5 mediante técnicas de aprendizaje por transferencia.
3. Evaluación y análisis de los resultados obtenidos.

Este documento está dividido en varias secciones las cuales son:

- Sección 2: Incluye el estado del arte actual del proyecto.
- Sección 3: Se muestra la metodología que se ha usado para el desarrollo del sistema, como la preparación del dataset utilizado y el entrenamiento de la red neuronal.
- Sección 4: Se detalla y se analizan los resultados obtenidos con los experimentos realizados.
- Sección 5: Se describen las conclusiones obtenidas en el proyecto.

2 ESTADO DEL ARTE

En la actualidad, el avance tecnológico en técnicas de inteligencia artificial y el incremento de la capacidad computacional han permitido que numerosas tareas relacionadas con el análisis de imágenes se puedan automatizar. Uno de los campos que se aprovecha de manera significativa estos avances tecnológicos es el del estudio y monitorización de la fauna salvaje mediante el uso de cámaras de trapeo. Estos estudios utilizan cámaras que generan grandes cantidades de imágenes que deben ser procesadas para poder identificar y clasificar las especies presentes, haciendo que el coste de tiempo y recursos fuese muy elevado en el pasado.

Los primeros enfoques que permitieron desarrollar sistemas capaces de detectar y clasificar especies animales estaban basados mayoritariamente en técnicas clásicas de Machine Learning, donde es necesario extraer manualmente características relevantes de las imágenes, como bordes, texturas o patrones geométricos [2][3]. Estos métodos consiguieron resultados aceptables, pero presentaban bastantes limitaciones al enfrentarse a escenarios más complejos, como la variabilidad de la iluminación, diferentes perspectivas o animales parcialmente ocultos.

La evolución hacia modelos basados en Deep Learning permitió superar estas limitaciones. En concreto, las redes neuronales convolucionales (CNN) han permitido eliminar la necesidad de definir manualmente los atributos relevantes, demostrando una gran capacidad de aprendizaje automático de características complejas de las imágenes. Este tipo de arquitecturas han demostrado grandes resultados en los últimos años en proyectos relacionadas con clasificación de imágenes y detección de objetos [4][5].

Uno de los proyectos más destacados es el de CameraTrapDetector [6], donde desarrollan un modelo para la detección y clasificación de un total de 75 especies distintas. Este trabajo utiliza una arquitectura basada en Faster-CNN con ResNet-50 como base, obteniendo resultados con una precisión y capacidad de detección alta.

Otro proyecto relevante es el estudio de Reconocimiento de especies en un bosque europeo [7], en el cual se plantea un problema a una menor escala, centrándose en la clasificación de un total de 11 especies animales. En este proyecto se utiliza la arquitectura YOLOv5 (You Only Look Once) debido a su bajo coste computacional y a la rapidez del entrenamiento comparado con otros modelos más complejos. También se destaca el proyecto DeepFauna [8], siendo una de sus principales aportaciones el uso de técnicas de aprendizaje por transferencia. Concretamente, emplean un modelo ConvNext-Base para clasificación y MegaDetectorV5, basado en YOLOv5, para la detección. Al utilizar Transfer Learning lograron reducir considerablemente el tiempo de entrenamiento y mejoraron el rendimiento del sistema.

En estos proyectos se puede observar varias aproximaciones diferentes para resolver este tipo de problemas. Algunos enfoques hacen uso de arquitecturas diferentes para detección y clasificación [8], mientras que otras integran

ambas en un solo modelo [6][7]. También se aprecia la utilización de aprendizaje por transferencia de forma recurrente, debido a su capacidad de aprovechar el conocimiento y adaptarlo en tareas más específicas [8]. Considerando las ventajas y características de estos trabajos, se decidió utilizar para el desarrollo del proyecto una arquitectura basada en YOLOv5 utilizando técnicas de aprendizaje por transferencia. Esta elección se basa en buscar el equilibrio entre la precisión del modelo, el tiempo de entrenamiento y los requisitos computacionales disponibles.

3 METODOLOGÍA

3.1 Preparación de los datos

Para el desarrollo del proyecto se ha utilizado el conjunto de datos Caltech Camera Traps [9], una base de datos pública utilizada en muchos proyectos y concursos de detección y clasificación automática de fauna salvaje. Este dataset contiene más de 240.000 imágenes diferentes obtenidas de 140 cámaras de trapeo localizadas en el suroeste de los Estados Unidos de America, contando con un total de 18 especies diferentes en las fotografías. Una de las principales ventajas de este conjunto de datos es que cuenta con 66.000 anotaciones asociadas a las imágenes, proporcionando información tanto de que especie o especies se encuentran en la imagen como de la localización aproximada de los animales en la fotografía. Las imágenes que se encuentran en este conjunto de datos tienen características que podemos encontrar en escenarios reales. Como se puede ver en la figura 1, algunas de estas características son las variaciones de iluminación entre fotografías realizadas durante el día o la noche, animales parcialmente ocultos por vegetación, animales en diferentes posiciones y tamaños, movimiento o ruido durante la captura de la imagen y presencia de muchas imágenes donde no se presenta ningún animal.



Fig. 2: Ejemplos de imágenes de la base de datos [9].

Antes de empezar a entrenar el modelo, fue necesario un proceso de preprocesado para adaptar las imágenes y anotaciones a los requisitos de entrada del modelo utilizado. La primera etapa consistió en redimensionar todas las imágenes a una resolución de 640 x 640 píxeles, tamaño estándar utilizado por la arquitectura YOLOv5 [10]. También se transformaron las anotaciones originales al formato utilizado en YOLOv5. Este formato representa la posición de los animales mediante una caja delimitadora definida por cuatro parámetros: posición horizontal y vertical del centro de la caja y anchura y altura de la misma caja.



Fig 3: Ejemplos de imágenes preprocesadas con las cajas delimitadoras de las anotaciones.

Inicialmente se realizaron diferentes pruebas con conjuntos de datos y número de especies reducido. Esto permitió poder hacer múltiples experimentos preliminares con tiempos de ejecución total menores. Sin embargo, el tamaño final del conjunto de datos se incrementó progresivamente hasta alcanzar un total de 50.000 imágenes. Estas imágenes se dividen en un total del 80% para el entrenamiento del modelo y el 20% restante para la validación. Esta separación permite entrenar el modelo utilizando un alto número de imágenes y hacer una validación mediante imágenes no utilizadas durante el proceso de aprendizaje. Para la selección de especies se priorizó aquellas con un mayor número de imágenes disponibles. En la figura 3 se puede observar el número de imágenes con anotaciones de las 10 especies más representativas en la base de datos. El resto de las especies tienen un número de imágenes mucho menor que estas, y por tanto, su elección dificultaría el entrenamiento del modelo en gran medida.

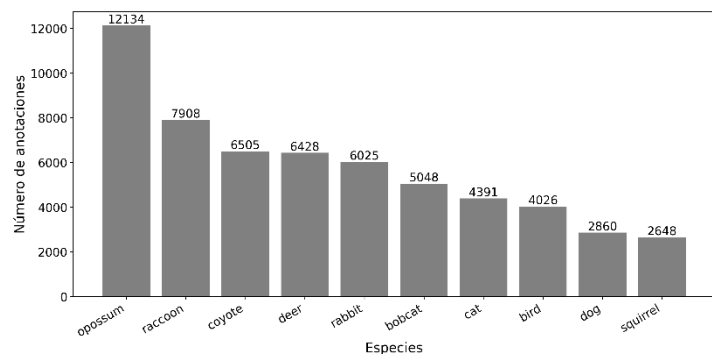


Fig. 4: Gráfica con la distribución de las anotaciones de las 10 especies más representadas.

Habiendo analizado el conjunto de datos, se observó un importante desbalanceo entre las especies, habiendo con un número de imágenes superior respecto a otras. Esta situación puede provocar que el modelo tienda a empeorar su capacidad de generalización y favorezca a las especies más representadas.

Con el objetivo de reducir este problema se incorporaron varias técnicas de aumento de datos (Data augmentation) [11]. Estas técnicas permiten generar nuevas muestras artificiales a partir de las imágenes originales aplicando diferentes transformaciones. Entre las operaciones utilizadas destacan el cambio de contraste y brillo, inversiones del eje horizontal (flip) y recortes de la imagen (crop).



Fig. 5: A la izquierda una imagen de la base de datos y a la derecha una imagen aumentada con flip.

Estas transformaciones se realizan únicamente en el entrenamiento y permiten balancear y aumentar la variabilidad de los datos.

3.2 Desarrollo del sistema de detección y clasificación

El desarrollo y entrenamiento del sistema de detección y clasificación se realizó utilizando Python como lenguaje de programación, junto con las librerías PyTorch y Ultralytics YOLO [10]. PyTorch proporciona las herramientas necesarias para el desarrollo de redes neuronales, mientras que Ultralytics permite implementar el entrenamiento del modelo YOLOv5 de manera más optimizada.

Para poder realizar los experimentos y entrenamientos de este modelo también se utilizó como GPU principal una NVIDIA GeForce GTX 1660 SUPER. Sin embargo, con el aumento de los experimentos y el tamaño del conjunto de datos en estos, los tiempos y recursos de ejecución comenzaron a incrementar notablemente.

Para solucionar este problema se utilizó una infraestructura proporcionada por la universidad, donde se puede hacer ejecuciones de manera remota, equipada con GPU dedicadas. Concretamente, se utilizó una tarjeta gráfica NVIDIA RTX 2080 Ti, que permitió reducir los tiempos de ejecución. Esto permitió poder realizar un mayor número de experimentos y entrenar modelos utilizando un número de datos de mayor tamaño.

La arquitectura YOLOv5, elegida para el desarrollo del modelo, es una red neuronal convolucional (CNN) diseñada específicamente para tareas de detección de objetos. A diferencia de otros métodos de detección, como Faster R-CNN, que realizan el proceso de detección y

clasificación en varias etapas independientes, YOLO realiza de manera simultánea la localización y clasificación de los objetos con una sola pasada a la red (You Only Look Once).

El funcionamiento de YOLO se basa en dividir las imágenes de entrada en cuadrículas y analizar al mismo tiempo cada región mediante la red neuronal, para después predecir en cada región una caja delimitadora.

La arquitectura YOLOv5 [12] se puede dividir principalmente en tres bloques:

- Backbone (Columna vertebral): extrae las características principales de las imágenes mediante diferentes capas convolucionales.
- Neck (Cuello): combina las características recopiladas por el backbone para enviarlas al head.
- Head (Cabeza): genera las predicciones de la localización y clasificación finales de los objetos detectados.

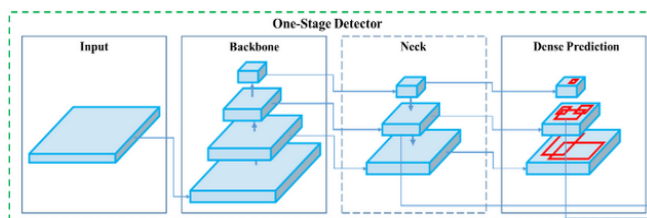


Fig. 6: Arquitectura de YOLOv5, donde se aprecian los tres bloques principales: backbone, neck y head.

Este modelo YOLOv5 está preentrenado sobre un dataset que contiene un total de 80 clases de diferentes objetos comunes (COCO) [13]. Entre estos objetos se encuentran varios tipos de vehículos (coches, bicicletas, trenes, ...), objetos relacionados con actividades deportivas (raquetas de tenis, bastos de béisbol, ...) y animales (perro, caballo, oveja, ...). Analizando todas las clases, se puede observar como solo una pocas relacionadas con animales son útiles para el proyecto.

Para poder adaptar el modelo al problema deseado se utilizaron técnicas de aprendizaje por transferencia. Esta estrategia permite reutilizar el modelo previamente entrenado y adaptarlo a problemas más específicos con menos coste computacional.

Se emplearon dos aproximaciones diferentes de aprendizaje por transferencia [14]:

- Fine-Tuning: modifica completamente todos los pesos de la red durante el entrenamiento.
- Feature Extraction: congela la mayoría de las capas de la red, y solo ajusta las capas finales responsables de la predicción.

El sistema desarrollado se compone de tres etapas principales: una fase de entrenamiento del modelo utilizando técnicas de aprendizaje por transferencia sobre el conjunto de datos de entrenamiento previamente procesado, una fase de validación del modelo, donde se evalúa el rendimiento del modelo imágenes no empleadas en el entrenamiento, y una última fase de predicción sobre las imágenes de validación para detectar la localización de los animales y clasificar las especies.

Finalmente, se realizaron múltiples experimentos variando hiperparámetros, como el número de épocas (epochs), el tamaño del conjunto de datos, el tamaño del lote de entrenamiento (batch) y diferentes tasas de aprendizaje (learning rates), con el objetivo de evaluar cada configuración y determinar una combinación que proporcionase el mejor equilibrio entre precisión, capacidad de generalización y rendimiento computacional.

El código desarrollado se encuentra disponible de manera pública en el siguiente repositorio de GitHub: <https://github.com/Adrian020/TFG-Project>.

4 RESULTADOS Y ANÁLISIS

Una vez completados numerosos procesos de entrenamiento y validación del modelo, se van a presentar y analizar los resultados obtenidos. Para ello, se emplean diversas métricas de evaluación del rendimiento, matrices de confusión para el análisis de la clasificación y ejemplos de inferencia utilizando imágenes del conjunto de validación.

4.1 Resultados de predicciones

La primera evaluación del modelo consistió en analizar los resultados de la fase de predicción, utilizando imágenes del conjunto de datos de validación. En esa fase, el sistema genera una predicción compuesta por una caja delimitadora (bounding box), que indica donde se localiza el animal (o animales) dentro de la imagen, y un porcentaje asociado a la confianza del modelo sobre la localización del animal y clasificación de la especie.

En la figura 7 pueden observarse varios ejemplos de fotografías donde el modelo detecta y clasifica correctamente los animales presentes en las imágenes con un nivel de confianza elevado. Se aprecia como el modelo mantiene un comportamiento satisfactorio incluso ante variaciones de posición y tamaño de los animales, condiciones de iluminación diferentes y varios animales presentes en las imágenes. Además, los valores de confianza obtenidos demuestran una elevada seguridad en las predicciones realizadas.

Sin embargo, el modelo también puede cometer errores durante las predicciones, y resulta importante estudiar estos casos.

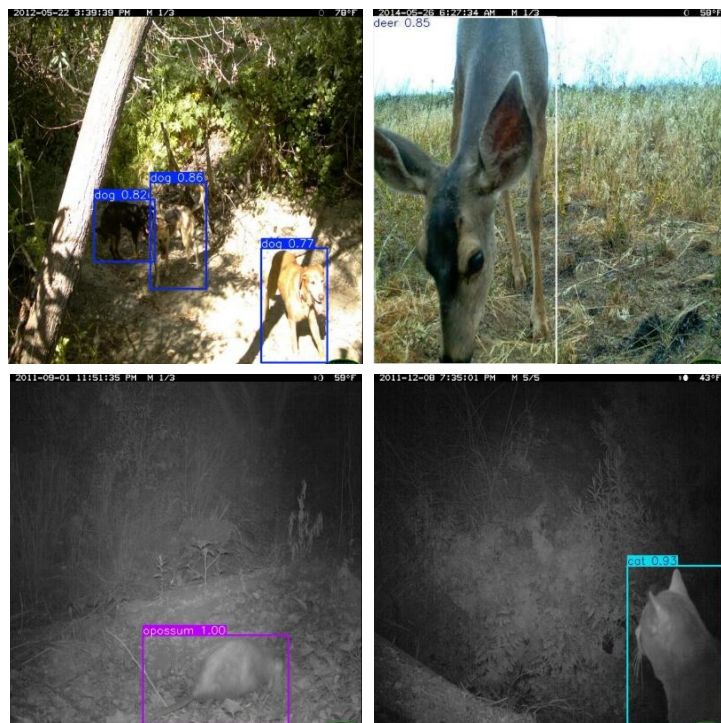


Fig. 7: Ejemplos de predicciones correctas realizadas del modelo junto con el porcentaje de confianza.



Fig. 8: Ejemplos de predicciones erróneas del modelo.

Como se puede observar en las predicciones del modelo de la figura 8, realizadas por modelos en una fase temprana del desarrollo, este también puede cometer errores en la identificación y clasificación de los animales. En las primeras dos imágenes se puede observar como el modelo confunde vegetación y objetos por un animal, principalmente

porque confunde características similares de estos objetos por los de un animal. En las otras dos imágenes se puede observar como el modelo no ha localizado, y por tanto tampoco clasificado, dos animales que están en un primer plano. El modelo también puede tener errores donde determinar una especie a un animal incorrectamente.

4.2 Evolución del entrenamiento y comparación entre estrategias de aprendizaje

Una vez analizado los resultados visuales del sistema, se estudió la evolución del entrenamiento del modelo utilizando diferentes estrategias de aprendizaje por transferencia.

Para medir el rendimiento del sistema se utilizan diferentes métricas para la detección y clasificación. Para poder evaluar la detección del modelo entrenado se emplean principalmente las métricas mAP50 y mAP50-95 (mean Average Precision) [15], comúnmente utilizadas en problemas de detección de objetos. La métrica mAP50 evalúa la capacidad del modelo para localizar correctamente animales presentes en las imágenes utilizando un umbral de IoU (Intersección sobre Unión) de un 50%, mientras que mAP50-95 utiliza varios umbrales de IoU entre un 50 y 95%. De esta forma mAP50 permite observar una visión general de rendimiento del sistema en detección y mAP50-95 una evaluación más precisa en cuanto a calidad y exactitud de las detecciones.

Por otro lado, para evaluar el rendimiento de la clasificación del modelo entrenado se utilizan las métricas de precisión (precision) y sensibilidad (recall) [16]. La precisión es la proporción de predicciones positivas correctas respecto al número total de predicciones positivas realizadas por el modelo y la sensibilidad representa la proporción de casos positivos reales que han sido identificados correctamente. Mientras que la precisión indica la fiabilidad de las predicciones realizadas por el modelo, la sensibilidad evalúa su capacidad para detectar todos los casos positivos reales. Ambas métricas permiten analizar el equilibrio entre falsos positivos y falsos negativos en el entrenamiento del modelo.

Inicialmente, durante las primeras etapas de desarrollo se utilizaron conjuntos de datos reducidos con el objetivo de realizar entrenamientos con bajos tiempos de ejecución y poder validar el funcionamiento del sistema. En esta fase del proyecto tampoco se utilizó ningún tipo de técnica de aumento de datos.

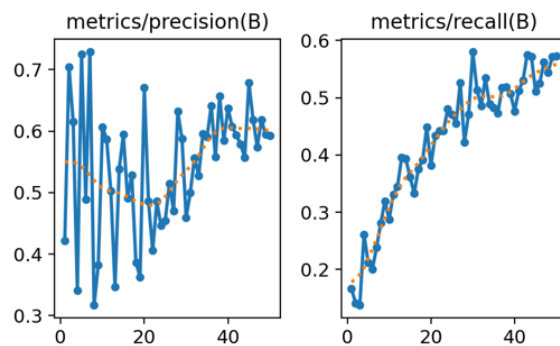
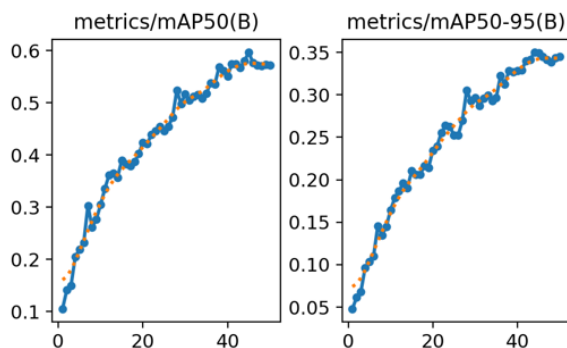


Fig. 9: Métricas del entrenamiento de un modelo en las primeras fases de desarrollo.

Posteriormente el tamaño del conjunto de datos fue ampliándose progresivamente hasta alcanzar un total de 50.000 imágenes. Se hicieron pruebas con dos técnicas de aprendizaje por transferencias, el fine-tuning y el feature extraction [14]. Para feature extraction se congelaron todas las capas del modelo preentrenado menos la última y para el fine-tuning se cambiaron los pesos en todas las capas de la red neuronal. También se hicieron un gran número de pruebas para encontrar los mejores parámetros de entrenamiento.

Las siguientes figuras corresponden a los mejores resultados de modelos con diferentes técnicas de aprendizaje por transferencia entrenados con 50 épocas (epochs) y lotes de entrenamiento de 32 imágenes (batches).

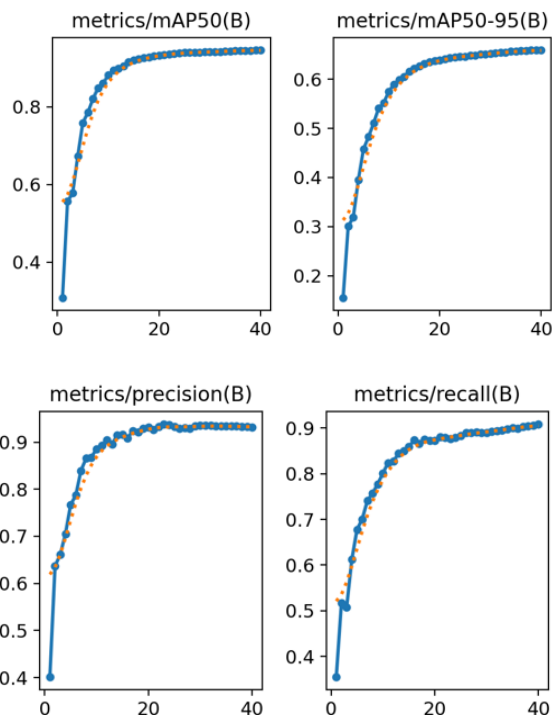


Fig. 10: Métricas del modelo entrenado con fine-tuning.

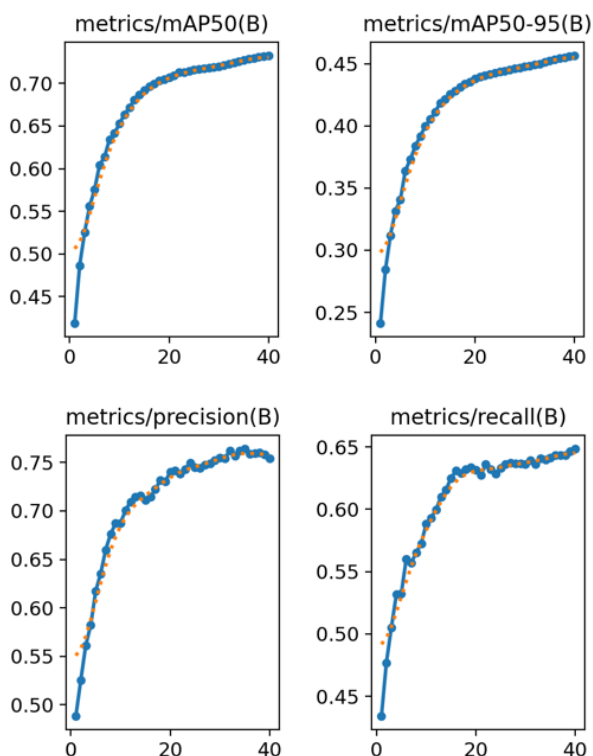


Fig. 11: Métricas del modelo con feature extraction.

Comparando los modelos de las figuras 10 y 11 pueden apreciarse diferencias significativas entre las dos estrategias. El modelo entrenado mediante feature extraction presenta una mejora progresiva de rendimiento adecuada, pero las métricas alcanzan nivel de convergencia limitado. En cambio, el modelo con fine-tuning presenta un rendimiento superior en todas las métricas, teniendo también una evolución muy estable. Al hacer un reajuste completo de los pesos de la red neuronal, el modelo puede adaptar de forma más precisa las características extraídas durante el preentrenamiento a las especies presentes en el conjunto de datos utilizado. Como resultado, se obtienen unos mejores resultados tanto en las métricas de detección como de clasificación respecto a la técnica de feature extraction.

Asimismo, comparando los resultados de la figura 9 con la 10, se observa una mejora significativa entre las dos fases de desarrollo. Por tanto, se puede decir que el aumento progresivo del conjunto de datos, la incorporación de técnicas de aumento de datos y el ajuste de parámetros del entrenamiento ha permitido una mejora más que notable del rendimiento global del sistema.

4.3 Evaluación del comportamiento del modelo

Además de los resultados observado durante la fase de entrenamiento y predicción, es necesario un análisis con diferentes indicadores de rendimiento que permiten evaluar con mayor detalle el comportamiento del modelo. Entre las diferentes herramientas de análisis utilizadas se encuentran las curvas de relación entre métricas, las matrices de confusión y los mapas de activación. Estas técnicas

permiten analizar el rendimiento desde otra perspectiva y ayudan a comprender el comportamiento del modelo.

La figura 12 muestra la relación entre el valor F1, que combina la precisión y sensibilidad del modelo, y los niveles de confianza de las clasificaciones durante la fase de validación.

Esto nos permite analizar como varía el rendimiento del modelo en función del umbral de confianza.

En este caso, para valores de confianza bajos, el modelo acepta un mayor número de predicciones, aumentando la capacidad de clasificación al mismo tiempo que aumenta el número de falsos positivos. Por el contrario, al haber un nivel de confianza superior el sistema se vuelve más estricto y rechaza más clasificaciones, reduciendo el número de falsos positivos, pero pudiendo aumentar el número de falsos negativos. El punto más álgido de la curva, entre 20% y 40% de confianza, es el valor que mejor equilibrio entre precisión y sensibilidad del modelo.

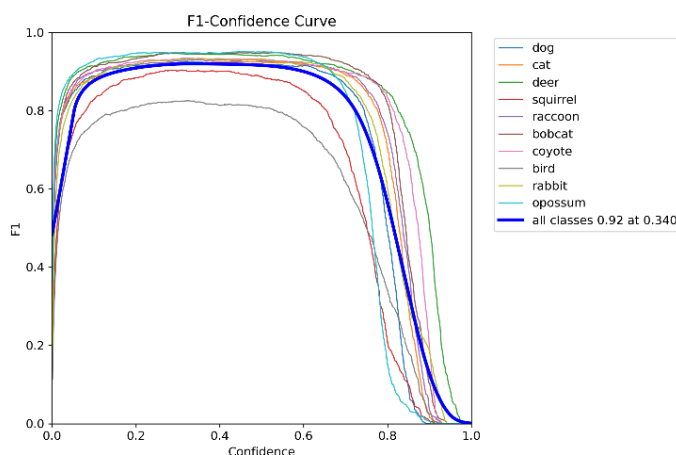


Fig. 12: Curva de F1-Confidence del modelo.

La figura 13 representa la matriz de confusión correspondiente al mejor modelo seleccionado (el mismo que en la figura 10). Esta matriz permite estudiar el comportamiento de cada especie del modelo. En esta matriz, los valores situados en la diagonal nos indican las clasificaciones correctas de cada especie y el resto de los elementos corresponden a errores de clasificación.

Se puede observar como la mayoría de las especies presenta un alto porcentaje de aciertos de clasificación, a excepción de la especie de pájaro. Esto puede deberse a el reducido tamaño que puedan presentar en la imagen o la similitud entre el terreno, entorno o vegetación de la imagen. La figura 14 muestra varios mapas de activación generados utilizando las capas 18 y 19 del modelo entrenado. Estas capas corresponden a una etapa final de la red neuronal (head), y por tanto, extraen características más complejas relacionadas con la forma o estructura del animal detectado.

En este caso se puede observar como la capa se centra su atención en el cuerpo del animal. Esto indica que la red neuronal ha sido capaz de aprender patrones visuales para poder diferenciar especies y que las características extraídas resultan coherentes con el problema abordado.

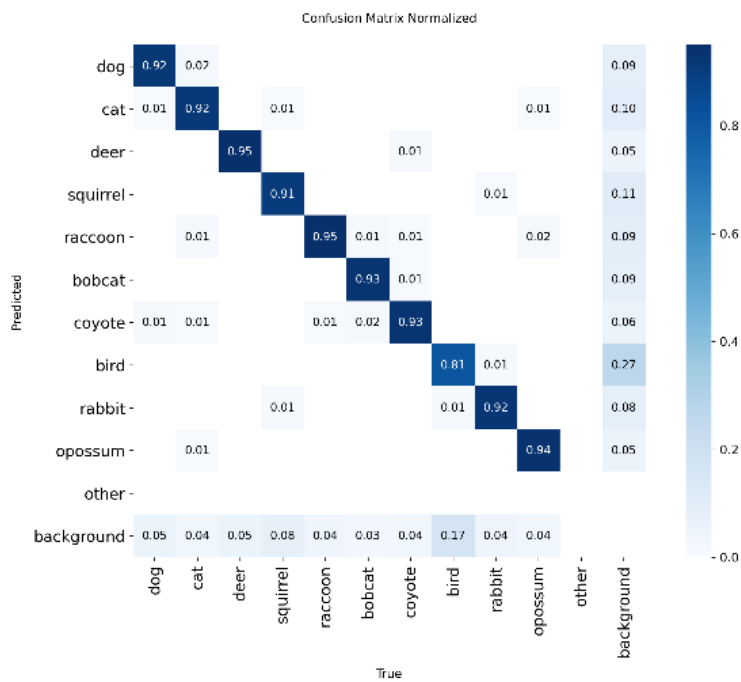


Fig. 13: Matriz de confusión de las especies del modelo.

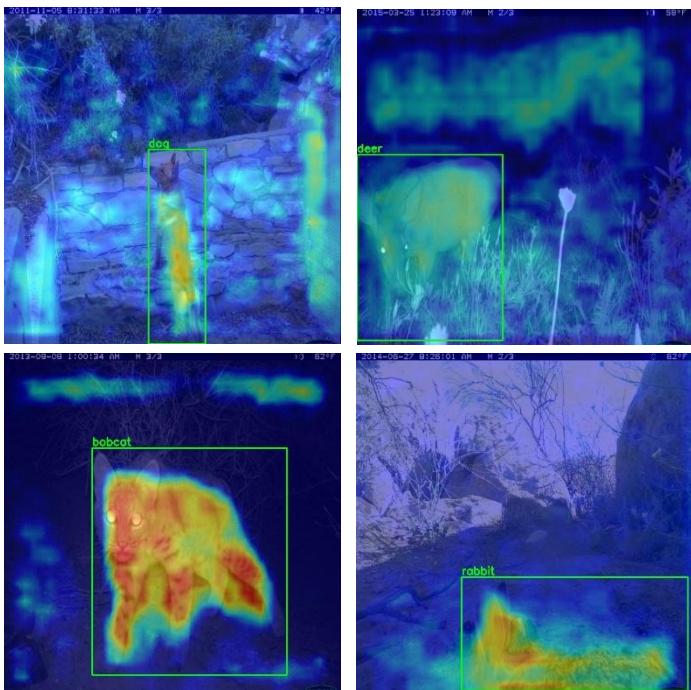


Fig. 14: Mapas de activación sobre imágenes del conjunto de datos.



Fig. 15: Predicción de localización y clasificación de los animales de las imágenes utilizadas en la figura 14.

5 CONCLUSIÓN

Con el desarrollo de este proyecto se puede concluir que las técnicas de aprendizaje profundo son muy útiles para resolver y automatizar problemas relacionados con la detección y clasificación de fauna salvaje mediante el uso de imágenes capturas en cámaras trampa. Los resultados

obtenidos muestran que es posible desarrollar un sistema con un nivel de rendimiento más que satisfactorio utilizando modelos preentrenados y técnicas de aprendizaje por transferencia. Por otra parte, se ha podido observar la importancia del tamaño y calidad del conjunto de datos, que permitió mejorar considerablemente el rendimiento y reducir los errores del modelo.

Finalmente, este trabajo me ha permitido adquirir muchos conocimientos relacionados con la visión por computador, las redes neuronales convolucionales y como tratar un problema real con técnicas de aprendizaje profundo.

Asimismo, con el fin de garantizar la accesibilidad y reproducibilidad del trabajo realizado, el código desarrollado se encuentra disponible de manera pública en el siguiente repositorio de GitHub: <https://github.com/Adrian020/TFG-Project>.

USO DE LA IA GENERATIVA

Durante el desarrollo de este Trabajo Final de Grado se han utilizado herramientas de Inteligencia Artificial Generativa en contadas ocasiones. Estas herramientas fueron empleadas principalmente para la búsqueda de información útil del tema del proyecto, ayuda puntual en la corrección de errores en el código y ayuda a la formalización y búsqueda de errores en este documento.

AGRADECIMIENTOS

Mis más sinceros agradecimientos al tutor de este trabajo, Mehmet Oguz Mulayim, ya que me ha ayudado en gran medida a realizar y organizar el proyecto y ha ofrecido todas las herramientas que tenía a su disposición. También agradecer a mi familia y amigos por apoyarme y animarme en el transcurso de desarrollo del proyecto.

BIBLIOGRAFÍA

- [1] Insights and approaches using deep learning to classify wildlife, Zhongqi Miao (2019). https://www.researchgate.net/publication/333515959_Insights_and_approaches_using_deep_learning_to_classify_wildlife
- [2] Identifying animal species in camera trap images using deep learning and citizen science, Marco Willi (2018). <https://besjournals.onlinelibrary.wiley.com/doi/10.1111/2041-210X.13099>
- [3] Active Learning Approaches to Identifying Animals in Camera Trap Data, Talfan Evans (2014). https://www.homepages.ucl.ac.uk/~ucbpste/MP2_Talfan_Evans.pdf
- [4] Addressing significant challenges for animal detection in camera trap images: a novel deep learning-based approach, Margarita Mulero-Pázmány (2025). <https://www.nature.com/articles/s41598-025-90249-z>
- [5] Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning, Mohammad Sadegh Norouzzadeh (2018). <https://www.pnas.org/doi/10.1073/pnas.1719367115>
- [6] CameraTrapDetector: Automatically detect, classify, and count animals in camera trap images using artificial intelligence, Michael A Tabak (2022). https://www.researchgate.net/publication/358488817_CameraTrapDetector_Automatically_detect_classify_and_count_animals_in_camera_trap_images_using_artificial_intelligence
- [7] A First Step Towards Automated Species Recognition from Camera Trap Images of Mammals Using AI in a European Temperate Forest, Mateusz Chojński (2021). https://link.springer.com/chapter/10.1007/978-3-030-84340-3_24
- [8] The DeepFaune initiative: a collaborative effort towards the automatic identification of the French fauna in camera-trap images, Noa Rigoudy (2022). https://www.researchgate.net/publication/359351666_The_DeepFaune_initiative_a_collaborative_effort_towards_the_automatic_identification_of_the_French_fauna_in_camera-trap_images
- [9] Caltech Camera Traps. <https://lila.science/datasets/caltech-camera-traps/>
- [10] Ultralytics YOLOv5 (2023). <https://docs.ultralytics.com/models/yolov5>
- [11] A survey on Image Data Augmentation for Deep Learning, Connor Shorten (2019). <https://link.springer.com/article/10.1186/s40537-019-0197-0>
- [12] Ultralytics YOLOv5 Architecture (2026) <https://docs.ultralytics.com/yolov5/tutorials/architecture-description>
- [13] Microsoft COCO: Common Objects in Context, Tsung-Yi Lin (2014). https://link.springer.com/chapter/10.1007/978-3-319-10602-1_48
- [14] Transfer Learning: Feature Extraction and Fine tuning, Alessandro Bosco (2024). <https://medium.com/data-reply-it-data-tech/transfer-learning-feature-extraction-and-fine-tuning-db7d82767992>
- [15] Mean Average Precision (mAP) (2023). <https://www.ultralytics.com/glossary/mean-average-precision-map>
- [16] Performance Metrics Deep Dive (2023). <https://docs.ultralytics.com/guides/yolo-performance-metrics>