

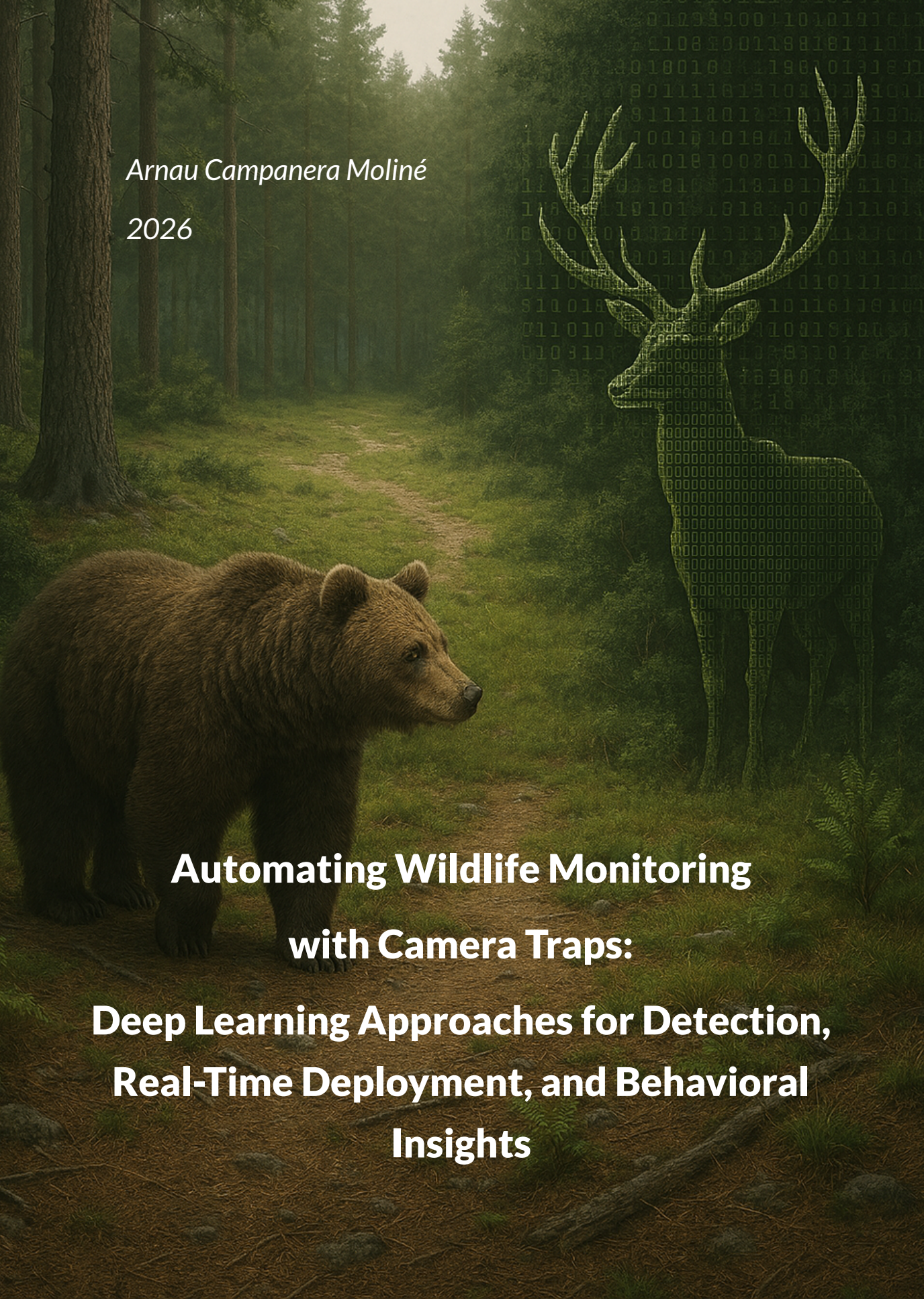
**Automating Wildlife Monitoring with Camera Traps:
Deep Learning Approaches for Detection, Real-Time Deployment,
and Behavioral Insights**

Arnau Campanera Moliné

ADVERTIMENT. L'accés als continguts d'aquesta tesi queda condicionat a l'acceptació de les condicions d'ús establertes per la següent llicència Creative Commons:  <https://creativecommons.org/licenses/?lang=ca>

ADVERTENCIA. El acceso a los contenidos de esta tesis queda condicionado a la aceptación de las condiciones de uso establecidas por la siguiente licencia Creative Commons:  <https://creativecommons.org/licenses/?lang=es>

WARNING. The access to the contents of this doctoral thesis it is limited to the acceptance of the use conditions set by the following Creative Commons license:  <https://creativecommons.org/licenses/?lang=en>

A brown bear stands in a forest, looking towards a deer. The deer is composed of a digital grid pattern, with its antlers also made of this pattern. The background is a misty forest with tall trees and a dirt path.

Arnau Campanera Moliné

2026

**Automating Wildlife Monitoring
with Camera Traps:
Deep Learning Approaches for Detection,
Real-Time Deployment, and Behavioral
Insights**

Automating Wildlife Monitoring with Camera Traps:

Deep Learning Approaches for Detection, Real-Time Deployment, and Behavioral Insights

Dissertation presented by
Arnau Campanera Moliné
to obtain the degree of Doctor

Directed by

Dr. Lluís Brotons

Forest Science and
Technology Centre of
Catalonia (CTFC)

Dr. Víctor Sazatornil

Forest Science and
Technology Centre of
Catalonia (CTFC)

Dr. Daniel Ponsa

Computer Vision
Center (CVC)

Tutored by

Dr. Jordi Martinez

Ecological and Forestry Applications Research Centre (CREAF)

Doctoral program of Terrestrial Ecology

Ecological and Forestry Applications Research Centre, CREAM

Autonomous University of Barcelona

Solsona, November 2025

The natural world is the greatest source of excitement; the greatest source of visual beauty; the greatest source of intellectual interest. It is the greatest source of so much in life that makes life worth living.

David Attenborough

Agraïments

Aquesta tesi no hauria estat possible sense el suport i la perseverança dels meus directors.

Gràcies, Víctor, per obrir-me la porta a treballar en el món de les càmeres de fototrampeig i poder explotar la meua inquietud sobre el creixent món de la intel·ligència artificial aplicada al món de l'Ecologia. Gràcies per trobar dades, contactes i projectes per poder seguir fent créixer tot això que estem desenvolupant.

Gràcies també al Dani Ponsa, qui ha hagut de lidiar amb algú que jugava fora de casa, en un camp canviant i complex. Les teves perseverança, solucions i ganes de tirar endavant aquest projecte ha fet que sigui molt millor del que hagués sigut. Estenc aquesta gratitud al Felipe i al Robert, qui han sofert del mateix mal però sempre amb bona actitud. Les vostres contribucions van posar llum a un moment crític on tot semblava impossible.

I finalment al Lluís qui, ja farà sis anys, va confiar en mi per forma part del grup d'Ecoland i em va donar la oportunitat de poder tornar a Catalunya i seguir dedicant-me al que m'agrada. Tot i que a partir d'ara prenguem camins separats, l'oportunitat de fer un doctorat no hauria estat possible sense tu.

Ahora, vull donar les gràcies a la gent del meu despatx; al David, al Guillem, al Gabri, l'Anna i també a la Mar amb qui he compartit molts riures, ens hem acompanyat i escoltat en moments complicats, i hem fet que, sobretot, valgués la pena venir presencialment al Centre, així com al Dani i a la Magda per sempre tenir paraules càlides i acollidores.

A la resta de la gent, fora del despatx que sempre ha tingut bones paraules amb mi, i s'ha preocupat per aquelles coses que van més enllà de la feina. Als que encara hi son i als que han agafat altres camins. En especial a l'Àngel o a la Carla Fuentes, amb qui sempre hi pots comptar.

A l'Elena Pi, el Noel, la Maguiña i la Cristina, qui han fet una feina enorme amb hores i hores d'etiquetatge perquè els models poguessin créixer.

Agraeixo també a la gent del comitè i a la gent del personal, qui sempre et reben amb un somriure.

Finalment, vull agrair a totes aquelles persones fora del CTFC que han fet i fan que la vida sigui molt millor, especialment mentre fas un doctorat. Al Ferran, Salva, Sergio i Uri, per ser uns antidepressius naturals infal·libles. Al Dani Macias, sempre present i disposat a solucionar i escoltar problemes. A la gent de Gironella qui encara no saben explicar en què consisteix la meva feina, i als Elais amb qui se t'obliden tots els problemes de doctorat.

I evidentment, als de casa. Als que sempre hi són. Als meus pares, i a la meva germana, qui m'han acompanyat en els moments més complicats. A la família política, qui sempre m'han acollit i facilitat tot el que he necessitat.

I a tu, Judit. Sense tu, això no hauria estat ni de bon tros possible. Gràcies per la teva paciència, per adaptar-te, per cuidar-me i per animar-me. Per tirar-ho tot endavant quan jo no hi podia ser, per cuidar del Nil i per sempre buscar la manera de poder fer les coses.

Tot i que ell no ho entengui encara, no puc deixar d'agrar al Nil tot el que m'ha donat i em dona cada dia. Recordar el que és essencial i important, fer-te tocar de peus a terra i relativitzar-ho tot. Tot i arribar l'últim estic segur que sense tu tot hauria estat molt diferent.

A tots vosaltres, mil gràcies.

Table of contents

Preface.....	i
Abstract.....	iii
Resum	iv
A Brief Introduction to Computer Vision for Ecologists	1
What is Computer Vision?	3
Digital Image Basics	3
Core Tasks in Computer Vision	4
Machine Learning Foundations	6
Challenges in Ecological Context.....	9
Integration with Ecology.....	10
Summary	10
1 General Introduction	13
1.1 Context & State of the Art.....	15
1.2 General Objectives and Structure.....	22
2 Benchmarking deep learning YOLO Object Detection models for Wildlife	
Detection in Camera Trap Imagery	29
2.1 Introduction	33
2.2 Materials and Methods	37
2.3 Results	53
2.4 Discussion.....	61
2.5 Conclusions	64
2.6 Supplementary Material	65
3 Enhancing Wildlife Detection in Small Camera Trap Datasets using	
YOLOv9e.....	91
3.1 Introduction	95
3.2 Materials and Methods	98

3.3	Results.....	106
3.4	Discussion.....	112
3.5	Conclusions.....	116
4	Real-time wildlife detection using 4G camera traps and Deep Learning ..	117
4.1	Introduction.....	121
4.2	Materials and Methods	124
4.3	Results.....	133
4.4	Discussion.....	141
4.5	Conclusions.....	144
5	Inferring Brown Bear hair snare interactions by automatically detecting bipedalism on camera trap images using Pose Estimation and a Multi-Layer Perceptron	147
5.1	Introduction.....	151
5.2	Materials and Methods	154
5.3	Results.....	172
5.4	Discussion.....	180
5.5	Conclusions.....	184
6	General Discussion.....	185
6.1	General Discussion	187
6.2	Contributions to Ecology and Conservation.....	190
6.3	Contributions to Computer Vision.....	191
6.4	Strengths and limitations	192
6.5	Future Research directions	194
7	Conclusions.....	197
	REFERENCES.....	201

Preface

The present thesis is the result of the work carried out by Arnau Campanera between 2020 and 2025. The thesis was developed at the Forest Science and Technology Centre of Catalonia (Solsona, Spain). As a result of the work derived from this thesis, the following articles are intended to be published or are under revision to be published in international scientific journals:

Campanera-Moliné, A., Sazatornil, V., Brotons, L., Ramilo-Henry, M., Pi, E., Caparroz, N., Lumbreras, F., Benavente, R., Sayol, F. Palazon, S. López-Bao, JV., & Ponsa, D. Benchmarking deep learning YOLO Object Detection models for Wildlife Detection in Camera Trap Imagery. In preparation.

Campanera-Moliné, A., Ponsa, D., Sazatornil, V., Lumbreras, F., Benavente, R. & Brotons, L. Enhancing Wildlife Detection in Small Camera Trap Datasets using YOLOv9e. In preparation.

Campanera-Moliné, A., Brotons, L., Ponsa, D., Pi, E., Caparroz, N. & Sazatornil, V. Real-time wildlife detection using 4G camera traps and Deep Learning. In preparation.

Campanera-Moliné, A., Sazatornil, V., Brotons, L., Lumbreras, F., Benavente, R., Ramilo-Henry, M., Ramos, C., Palazon, S. & Ponsa, D. Inferring Brown Bear hair snare interactions by automatically detecting bipedalism on camera trap images using Pose Estimation and a Multi-Layer Perceptron. Submitted to *Remote Sensing in Ecology and Conservation*. Under revision.

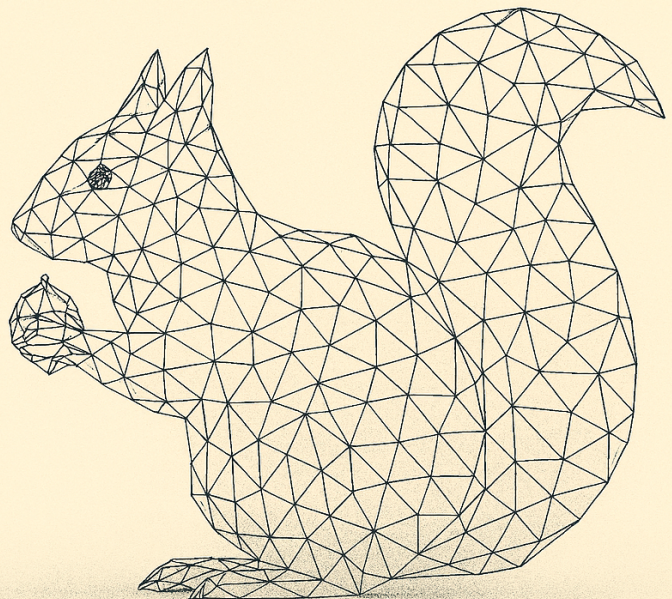
Abstract

Wildlife monitoring is a cornerstone of ecological research and conservation, supporting both retrospective, data-driven management strategies and active, real-time decision making. It provides essential information on species distributions, behavior, and population dynamics in the context of accelerating biodiversity loss. However, the rapid expansion of camera trap deployments, together with the increasing temporal resolution and precision of modern monitoring technologies, has led to unprecedented volumes of data, creating substantial analytical challenges. In recent years, machine learning and deep learning approaches have emerged as promising tools to address these limitations. This thesis evaluates the potential of YOLO-based computer vision models across a range of ecological applications. A custom wildlife image dataset spanning multiple species was assembled, and recent variations of the YOLO architecture were benchmarked to establish baseline detection performance. Several strategies were subsequently investigated to optimize the best-performing model without increasing training data requirements. To assess real-world applicability, a real-time wildlife monitoring system integrating cellular-connected camera traps and the optimized model was deployed. In addition, pose-estimation models were combined with a multi-layer perceptron to infer bear behavior through an innovative analytical framework. The results demonstrate that YOLO-based detection models perform robustly for wildlife monitoring, with performance gains achieved through hyperparameter optimization, while increased domain specificity through color-based dataset partitioning did not improve outcomes. The deployed system achieved near-real-time detection capabilities suitable for a wide range of ecological applications, and adapted pose-estimation models successfully inferred bipedal behavior. Overall, this thesis highlights the strong potential of deep learning approaches to address persistent challenges in wildlife monitoring and to enable automated, real-time ecological observation, while providing practical guidance on the strengths and limitations of these methods for ecological research.

Resum

El monitoratge de la fauna salvatge és un element fonamental de la recerca ecològica i de la conservació, ja que dona suport tant a estratègies de gestió retrospectives basades en dades com a la presa de decisions en temps real. Proporciona informació essencial sobre la distribució de les espècies, el comportament i la dinàmica poblacional en un context de pèrdua accelerada de biodiversitat. Tanmateix, la ràpida expansió de l'ús de càmeres trampa, juntament amb l'augment de la resolució temporal i la precisió de les tecnologies modernes de monitoratge, ha generat volums de dades sense precedents, cosa que planteja reptes analítics importants. En els darrers anys, els enfocaments d'aprenentatge automàtic i d'aprenentatge profund han emergit com a eines prometedores per abordar aquestes limitacions. Aquesta tesi avalua el potencial dels models de visió per computador basats en YOLO en una varietat d'aplicacions ecològiques. Per a aquest objectiu, es va confeccionar un conjunt de dades d'imatges de fauna que abasta múltiples espècies, i es van avaluar diferents variants recents de l'arquitectura YOLO amb l'objectiu d'establir un rendiment base de detecció. Posteriorment, es van investigar diverses estratègies per optimitzar el model amb millor rendiment sense incrementar els requeriments de dades d'entrenament. Per avaluar l'aplicabilitat en condicions reals, es va desplegar un sistema de monitoratge de fauna en temps real que integra càmeres trampa amb connectivitat cel·lular i el model optimitzat. A més, es van combinar models d'estimació de postura amb un perceptró multicapa per inferir el comportament de l'ós mitjançant un enfocament analític innovador. Els resultats mostren que els models de detecció basats en YOLO ofereixen un rendiment robust per al monitoratge de fauna, amb millores obtingudes mitjançant l'optimització d'hiperparàmetres, mentre que l'increment de l'especificitat del domini mitjançant la partició del conjunt de dades segons criteris cromàtics no va aportar beneficis addicionals. El sistema desplegat va demostrar capacitats de detecció quasi en temps real adequades per a una àmplia gamma d'aplicacions ecològiques, i els models d'estimació de postura adaptats van permetre inferir amb èxit el comportament bípede. En conjunt, aquesta tesi posa de manifest el gran potencial dels enfocaments d'aprenentatge profund per afrontar reptes persistents en el monitoratge de fauna i per permetre l'observació ecològica automatitzada en temps real, alhora que ofereix orientacions pràctiques

sobre els punts forts i les limitacions d'aquestes metodologies en l'àmbit de la recerca ecològica.



A Brief Introduction to Computer Vision for Ecologists

A Brief Introduction to Computer Vision for Ecologists

Computer vision is a field of artificial intelligence that enables computers to interpret and analyze visual information from the world, much like humans do. In the context of ecology, computer vision provides tools to automatically process images and videos. This capability is transformative for camera trap workflows, where millions of images are generated across sites, seasons, and habitats. Object detection models can identify and localize animals, count individuals per image, and provide species-level labels—accelerating biodiversity monitoring, occupancy estimation, and behavioral studies at scales that would be impractical otherwise.

What is Computer Vision?

At its core, computer vision involves algorithms that transform raw pixels into meaningful information. A digital image is essentially a grid of numbers representing color and brightness. Computer vision techniques extract patterns from these numbers to identify objects, classify species, or detect movement.

Digital Image Basics

Pixels, color channels, and image representation

Images are composed of pixels, small units that store color and brightness information. Understanding this representation is fundamental because all computer vision algorithms operate on these numerical values. For example, a camera trap image of a bear is just a grid of numbers to a computer; algorithms learn to interpret these numbers as meaningful patterns.

Most ecological imagery uses RGB (Red–Green–Blue), that is 3 color channels. Each number in the channel grid represents the brightness of that color at that pixel position. Combining the three-color channels one can obtain the rest of the colors

present in the image. In grayscale images, there is either one single channel or three identical channels representing brightness (from black to white).

Concept of features (edges, textures, shapes)

Traditional computer vision relied on detecting simple features such as edges, corners, and textures. These features were combined to recognize objects. Modern approaches, however, use deep learning to automatically learn complex patterns from data. For instance, a convolutional neural network (CNN) can learn to distinguish the fur patterns of a jaguar from those of a leopard just from training examples, without the need of an explicit model of their respective appearances.

Core Tasks in Computer Vision

Image Preprocessing: noise reduction, normalization

Preprocessing images improves data quality and model robustness before feature extraction or learning. Some image processing techniques include:

- **Denoising:** Reduce sensor noise, especially in low-light camera trap images without blurring animal edges.
- **Contrast enhancement:** Enhances details under shadows or glare common in forest understory or snow/ice scenes.
- **Color normalization:** White balance correction and illumination normalization mitigate color cast variation across devices and times of day.
- **Geometric corrections:** Lens distortion correction, rectification, stabilization, and alignment for drone, or fisheye lenses.
- **Background subtraction:** Useful in fixed camera setups to highlight moving animals and reduce empty frames.
- **Resizing and normalization:** Standardize image dimensions and scale pixel values (e.g., to $[0,1]$) for consistent model input.

In ecological pipelines, preprocessing is tuned to preserve subtle cues (e.g., ear notches, pelage markings) while reducing variability due to environment and hardware.

Classification: species identification

Image classification algorithms are typically designed to assign a single class label to an entire image. This approach presents inherent limitations in ecological contexts, particularly in scenarios where multiple distinct species are present within the same image. As a result, conventional classification models are unable to provide multiple labels per image or quantify the number of individual instances.

However, image-level classification is useful for triage, for example, filtering empty frames, “animal vs. non-animal,” (Beery et al., 2019) or assigning a dominant species to a frame. However, it cannot count individuals or handle multi-species co-occurrence reliably.

Object Detection: locating animals in images

Object detection provides a more sophisticated solution. Object detection algorithms not only identify and localize multiple individuals within an image by putting a bounding box around each detected animal but also assign class labels to each individual based on the trained species. This enables both species-level classification and individual-level enumeration within a single image.

Segmentation: outlining objects for detailed analysis

Segmentation assigns a label to each pixel, enabling fine-grained analysis:

- **Semantic segmentation:** Labels pixels by class (e.g., “animal,” “vegetation,” “water”), without distinguishing individual instances.
- **Instance segmentation:** Separates individual animals, even if overlapping—critical for counting colonial seabirds or herd mammals.
- **Panoptic segmentation:** Combines semantic and instance segmentation for complete scene understanding.

Machine Learning Foundations

Difference between traditional CV and deep learning

Traditional computer vision relied on extracting hand-crafted features from images, such as edges, corners, and textures. These features were then combined to recognize objects. Common examples include:

- Edges and gradients: outline animal silhouettes against background.
- Corners and keypoints: support matching and tracking (e.g., repeated site visits).
- Texture descriptors: capture fur/feather patterns and body outlines.

Strengths: transparency, low data requirements, and robustness in constrained settings.

Limitations: brittle under environmental variability; feature design is task specific.

Contemporary machine vision systems follow a data-driven approach, leveraging deep neural networks (i.e., architectures composed of numerous layers) that learn hierarchical and complex features from training data to optimize task-specific performance.

Strengths: high accuracy, adaptability across tasks (classification / detection / segmentation), and transfer learning using pretrained models.

Limitations: requires more data/compute, less inherently interpretable, vulnerable to domain shift if training data is narrow.

For ecologists, transfer learning (fine-tuning pretrained models on ImageNet or wildlife datasets) often yields strong performance with modest labeled datasets (Yang et al., 2024).

Neural networks and convolutional neural networks (CNNs)

CNNs-based systems have existed in computer vision for many years, offering great performance in object detection and classification. Recent advances are driven by deep learning models, which learn hierarchical representations of visual data, starting from simple edges to complex shapes and textures. Modern detectors (e.g., YOLO-v5/v8, Faster/Mask R-CNN, DETR-family) learn multi-scale features and predict boxes, classes, and confidences in a single pass. CNNs have proven highly effective for species identification and even individual recognition. For example, CNN-based models have been used to identify individual whales from tail fluke patterns and snow leopards from their unique spot patterns.

Training, Validation and Test Sets

Computer vision models require large datasets of labeled images for training. The diversity and quality of these datasets directly influence model performance. For ecologists, this means that careful dataset curation is critical.

When developing computer vision models, the dataset is typically divided into three subsets: training, validation, and test. Each serves a distinct purpose in ensuring that the model learns effectively and generalizes well to new data.

Training Set

- **Purpose:** The training set is used to teach the model. During training, the algorithm adjusts its internal parameters (weights) to minimize prediction errors on this data.
- **Characteristics:** It should be the largest portion of the dataset (commonly 70–80%) to provide enough examples for the model to learn patterns.
- **Ecological Example:** If you are building a species classifier for camera trap images, the training set contains labeled images of animals (e.g., deer, fox, lynx) under various conditions (lighting, seasons, habitats). The model uses these examples to learn what visual features distinguish each species.

Validation Set

- **Purpose:** The validation set is used during training to monitor performance and tune hyperparameters (e.g., learning rate, number of layers). It helps detect overfitting, where the model memorizes the training data instead of learning general patterns.
- **Characteristics:** Typically, 10–20% of the dataset. It should represent the same distribution as the training set but remain unseen by the model.
- **Ecological Example:** While training a detection model for camera trap images, the validation set might include images from different camera stations or times of day. If performance drops on validation data while improving on training data, it signals overfitting.

Test Set

- **Purpose:** The test set is used after training is complete to evaluate the model's ability to generalize to unseen data. It provides an unbiased estimate of real-world performance.
- **Characteristics:** Usually 10–15% of the dataset. Ideally, it should include data from different sites, seasons, or devices to simulate deployment conditions.
- **Ecological Example:** For a camera trap project, the test set could include images from new locations not used in training or validation. This ensures the model is robust to habitat variability and sensor differences.

The importance of splitting the data:

- **Avoiding Data Leakage:** If images from the same site or time appear in both training and test sets, performance metrics will be inflated and not reflect true generalization.
- **Ecological Bias:** Splits should consider spatial and temporal independence. For example, separating by camera station or survey period prevents the model from exploiting site-specific cues (background vegetation, lighting) instead of species features.

- **Transferability:** Proper splits help assess whether a model trained in one region can work in another—a critical question for large-scale biodiversity monitoring.

Challenges in Ecological Context

Environmental variability (lighting, occlusion)

Wildlife imagery spans day (color) and night (grayscale), seasons, weather, and habitats with clutter (vegetation, shadows, glare). Animals are often partially occluded or camouflaged. Motion blur from low light conditions further complicates recognition. These challenges can be mitigated by illumination normalization, robust data augmentations and domain adaptation to shift models between habitats or devices.

Class imbalance and rare species

Many datasets contain vast numbers of empty frames or common species, while target species tend to be rare. In these conditions, models may overfit to dominant classes and miss rare events.

Data annotation difficulties

To teach a computer vision what to learn, labeled data must be provided. Labeling is time-consuming and requires expertise (e.g., distinguishing cryptic or similar species). Ambiguous images, partial views, and juveniles complicate annotation, while inter-annotator variability can introduce noise to the dataset (Plank, 2022).

Integration with Ecology

How CV outputs translate into ecological insights

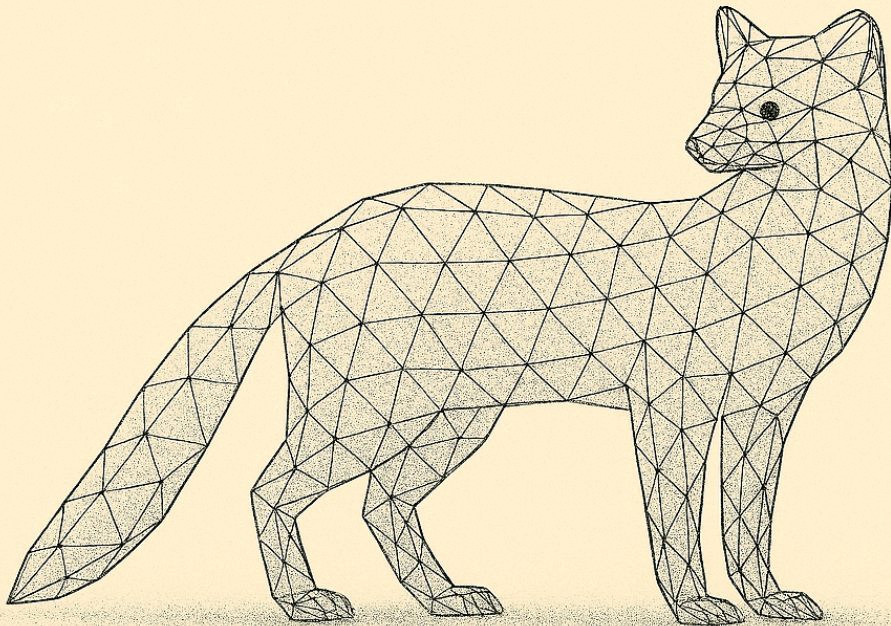
Computer vision outputs (detections, counts, species labels, masks, timestamps, locations) feed directly into ecological models:

- **Occupancy models:** Binary presence from camera traps (after filtering empty frames) informs species distribution and site use, accounting for detection probability (MacKenzie, 2018).
- **Abundance and density:** Individual counts from detection/segmentation combined with survey effort, area, and detection functions (e.g., distance sampling in aerial surveys) (McIvor et al., 2024).
- **Mark–recapture / individual ID:** Re-identification via distinct markings (flukes, stripes, spots) supports demographic estimates (survival, movement) (McIvor et al., 2024).
- **Behavioral ecology:** Activity states from pose estimation/segmentation (foraging, resting) enable time budgets and disturbance assessments (Mathis et al., 2018).
- **Habitat associations:** link detections with site-level covariates (vegetation, elevation, human proximity) for species distribution modeling (Niedballa et al., 2015).

Summary

For camera-trap ecology, object detection is the most informative and scalable computer vision task: it enables species identification, individual enumeration, and event-based analyses aligned to ecological models. With careful data splitting, robust augmentation, and domain-aware evaluation, detectors can substantially reduce manual effort while improving the timeliness and scope of wildlife monitoring.

CHAPTER 1



1 General Introduction

1.1 Context & State of the Art

1.1.1 Biodiversity crisis: Monitoring & Conservation Challenges

Biodiversity has undergone substantial fluctuations throughout Earth's history, including five major mass extinction events over the past 450 million years. These events, which eliminated approximately 70-95% of the existing species, were triggered by profound environmental perturbations (Benton, 1995). In recent decades, the concept of a human-driven sixth mass extinction has gained prominence within the scientific community (Barnosky et al., 2011; Brondizio et al., 2019). Current extinction rates are estimated to be up to 1,000 times higher than background rates inferred from the fossil record (Pimm et al., 2014). There is growing consensus that we are entering a sixth mass extinction episode, one that may constitute the most critical environmental threat for our civilization, as it would be irreversible (Ceballos et al., 2020).

These anthropogenically driven extinction dynamics underscore the urgent need for coordinated, large-scale actions to halt biodiversity loss. This urgency is reflected in international agreements such as the Kunming-Montreal Global Biodiversity Framework (GBF). The agreement, also known as the “Paris Agreement for Nature”, sets out four overarching goals to be achieved by 2050: (1) ensuring healthy, resilient, and interconnected ecosystems; (2) promoting the sustainable use of biodiversity; (3) securing fair and equitable sharing of benefits; and (4) providing adequate means for implementation.

At the European scale, the European Union (EU) has developed a suite of policies to operationalize the GBF, reflected in the EU Biodiversity Strategy for 2030, the derived nature directives, the Nature Restoration regulation, and complementary initiatives under the European Green Deal.

1.1.2 Biodiversity Monitoring

Effective biodiversity conservation requires robust, timely, and reliable information on species distributions, trends, and ecological processes (Sutherland et al., 2004). As global biodiversity continues to decline and anthropogenic pressures intensify, systematic and scalable wildlife monitoring becomes indispensable. However, biodiversity monitoring efforts are frequently fragmented, temporally inconsistent, and hindered by limited data harmonization, resulting in substantial information gaps (Proença et al., 2017). Furthermore, traditional monitoring approaches rely heavily on field observers whose inherent subjectivity may introduce biases into ecological datasets.

Initiatives such as the Europa Biodiversity Observation Network (EuropaBON) seek to address these challenges by designing a coordinated EU-wide framework for biodiversity and ecosystem service monitoring. In 2023, expert surveys conducted by EuropaBON identified key gaps, bottlenecks, and limitations in current monitoring workflows (Kissling, Breeze, et al., 2024). The project has also catalogued emerging monitoring technologies and assessed their maturity and suitability for large scale implementation across Europe (Dornelas et al., 2023).

Beyond conservation, wildlife monitoring also underpins ecological research, behavioral studies, and efforts to mitigate human-wildlife conflicts. In this regard, camera traps have become a fundamental, non-invasive tool for wildlife monitoring (Oliver R. Wearn & Paul Glover-Kapfer, 2017). Their widespread adoption has transformed the estimation of species presence, density abundance by enabling cost-effective, low-disturbance, long-term, and remote data collection (Howe et al., 2017). Over the last decade, the deployment of large-scale camera-trap networks has expanded rapidly, generating millions of images annually, and overwhelming traditional data processing capacities. Additional technologies and sensors, such as acoustic loggers or GPS tracking devices, have simultaneously improved in speed, accuracy, and affordability, further increasing the volume and complexity of ecological while at the same time easing to address the urgent international demands (Pimm et al., 2015; T. Dalton et al., 2022).

The exponential rise in data generation has introduced a major challenge: data overload. Manually processing vast quantities of imagery, video, or acoustic data is slow, labor-intensive, and prone to human error. Converting these high-volume

datasets into actionable ecological insights has become a substantial operational bottleneck in contemporary ecological research (Keitt & Abelson, 2021). Consequently, efficient automated data-processing pipelines now represent one of the most pressing needs for researchers, practitioners, and conservation managers (Hahn et al., 2022).

Advances in hardware and software have led to the rapid emergence of data-driven computational tools, particularly in the field of artificial intelligence (AI) (Hampton et al., 2013; Perry et al., 2022). Over the past decade, machine learning (ML) and deep learning (DL) have shown great promise in addressing the challenges posed by data overload. However, their successful deployment in ecological workflows require strong interdisciplinary collaboration between ecologists and ML experts (Tuia et al., 2022). Indeed, the capacity to foster such collaboration has been identified as a key factor constraining the widespread automation of ecological data collection and analysis (Besson et al., 2022).

1.1.3 The Technological Opportunity

The era of “big data” has also reached the field of wildlife monitoring (Hampton et al., 2013; Larsen et al., 2020). In recent years, an increasing number of researchers have adopted ML and DL approaches (Thompson, 2023), to automate and accelerate segments of data processing that would otherwise require extensive manual effort. DL models have already demonstrated high performance across a wide range of image- and audio-based tasks (Pichler & Hartig, 2023), and in some cases have even surpassed human experts in species identification or bird call recognition (Kahl et al., 2021; Van Horn et al., 2018). Fennell et al., in 2022 quantified the benefits of ML and DL integration, reporting increases in processing speed exceeding 500%, and reductions in required manual effort by a factor of 8.4.

Computer vision is among the domains that have benefited most from the rapid development of deep learning. This progress is largely attributed to the availability of large, well-annotated datasets (Deng et al., 2009; Lin et al., 2014) and significant advances in neural network architectures. In ecological research, automated classification of camera-trap images was already explored over a decade ago, achieving an accuracy of 38.32% (G. Chen et al., 2014a). Since then, modern wildlife-monitoring models have consistently exceeded 90% accuracy, sometimes

leveraging temporal information by processing image sequences (Rigoudy et al., 2023).

1.1.4 You Only Look Once (YOLO)

One of the most widely adopted model families for wildlife monitoring is YOLO (You Only Look Once; Redmon et al., 2016). YOLO represents a state-of-the-art deep learning architecture for object detection and is known for its real-time inference capabilities. Although originally developed for general computer-vision applications, YOLO models have rapidly gained traction within ecological workflows due to their efficiency, detection accuracy, and speed (Kristo et al., 2020; X. Lei et al., 2025; Roy et al., 2023).

Object detection offers several advantages over standard image classification. While deep convolutional neural networks (DCNNs) trained for classification assign a single label to an entire image, this approach becomes limiting when multiple individuals or species appear simultaneously. Object detection overcomes these constraints by localizing each individual with bounding boxes and assigning species-level labels to each detection based on the trained species.

The hallmark of the YOLO family is its single-stage detection pipeline, in which bounding boxes and class probabilities are predicted in a single forward pass. This provides considerable speed advantages over two-stage detectors such as Faster R-CNN (Ren et al., 2016) or RetinaNet (Lin et al., 2018), which require sequential region-proposal and classification steps. While early YOLO versions traded off accuracy for speed, recent iterations consistently achieve competitive or superior performance compared to alternative object-detection architectures. Moreover, the YOLO framework has expanded to support additional computer-vision tasks, including oriented bounding boxes, semantic segmentation, and pose estimation (Jocher et al., 2023a; Jocher & Qiu, 2024).

The recent acceleration of applied artificial intelligence has resulted in the release of multiple YOLO variants within a short period (M. Lei et al., 2025; Tian et al., 2025; A. Wang et al., 2024; C.-Y. Wang et al., 2024). Despite sequential version names, these variants are not strictly built upon one another; rather, each introduces distinct architectural design choices. To date, these differences have not been

systematically benchmarked using camera trap datasets. Understanding their respective strengths and limitations is therefore essential for ecologists aiming to select appropriate deep learning solutions for wildlife monitoring.

1.1.5 Computer Vision Challenges

Despite remarkable progress, contemporary algorithms still fall short of human performance across many ecological applications (Leorna & Brinkman, 2022). A central challenge is the scarcity of large, domain-specific annotated datasets (Pardo et al., 2021). Although large public datasets exist (Beery et al., 2018; Swanson et al., 2015), the species they include are relevant to only a limited subset of ecological studies. Consequently, most ecologists must assemble custom datasets, which tend to be small, exhibit severe class imbalance, and encompass substantial variability in lighting, background, and environmental conditions.

Species behavior further exacerbates these imbalances. Elusive or nocturnal species are often underrepresented, producing long-tailed class distributions. As training examples become scarce, model performance decreases sharply, whereas well-represented species benefit from substantially improved accuracy (Van Horn & Perona, 2017). Such ideal data conditions are rare in real-world ecological monitoring.

Generalization capability is another key requirement across scientific disciplines. Popular benchmark datasets (ImageNet, Pasca-VOC, OpenImages, COCO) contain millions of well-curated images, typically capturing human-dominated or urban environments (Deng et al., 2009; Everingham et al., 2010; Kuznetsova et al., 2020; Lin et al., 2014). Models trained on these datasets perform exceptionally well on in-domain test sets, but their accuracy often drops substantially on custom, ecologically relevant datasets.

The challenge is even more pronounced in ecology due to extreme variability in background composition, seasonality, weather conditions, sensor placement, and illumination. High-performing models often fail to generalize across ecosystems, camera-trap brands, or geographical regions (Beery et al., 2018). The rapid deployment of new sensors in new locations each year introduces additional sources of variation, making the generalization problem particularly severe. As a

result, the development of curated, region-specific datasets with balanced class representation and high-quality annotations remains both a fundamental requirement and a formidable methodological challenge.

1.1.6 Remote monitoring

Traditional camera traps rely on batteries and local data storage, typically SD cards, which necessitate periodic field visits every three to six months depending on storage capacity, battery autonomy, and trigger frequency. In contrast, some recent camera-trap models incorporate cellular communication capabilities, enabling data transmission via LTE/4G/3G/2G networks. These systems allow near-instantaneous access to images and videos. However, all incoming media, including empty frames, must still be manually inspected. Suboptimal field placement, such as vegetation swaying in the camera's field of view or incorrect orientation toward direct sunlight, can produce hundreds or even thousands of false-trigger images per day. In large-scale monitoring programs deploying hundreds of camera units, this rapid accumulation of data can create a severe interpretation bottleneck.

The emergence of computer-vision models capable of real-time inference greatly expands the potential of remote wildlife monitoring especially when combined with wireless camera traps. Integrating image-recognition algorithms into the data pipeline of 4G-enabled cameras can automate data processing and alleviate interpretation bottlenecks, thereby improving both cost-effectiveness and analytical accuracy (Kissling, Evans, et al., 2024; Smith et al., 2024). Additionally, real-time notifications for target species can support rapid conservation responses or assist in mitigating human-wildlife conflict, all without the need for manual image inspection. This capability substantially enhances resource efficiency.

Nonetheless, the deployment of wireless camera traps is strongly constrained by environmental conditions. Topography, canopy cover, and habitat structure critically influence the availability and strength of cellular signals (Laughlin et al., 2023). For real-time systems, the presence of high-band width connections (e.g., 4G) and adequate signal strength determine the feasibility and speed of data transmission. To date, no published study has quantitatively evaluated the relationship between network-band availability and actual transmission

performance in wildlife-monitoring contexts, representing a significant knowledge gap.

1.1.7 Edge computing

Despite the advantages of real-time automated alerts via cellular networks, dependence on mobile connectivity imposes important limitations. Many wildlife species sensitive to human presence inhabit remote or rugged regions where cellular coverage is sparse or absent, rendering 4G camera traps impractical. Satellite communication offers global coverage through multiple active constellations; however, the associated data bandwidth is extremely limited or prohibitively expensive, making the transmission of images infeasible or economically unjustifiable.

Edge computing provides a promising partial solution to these constraints. In edge-based systems, data processing occurs directly at the point of collection rather than being transmitted to centralized servers. Small, energy-efficient computing devices, such as Raspberry Pi units, NVIDIA Jetson Nano boards or Google Coral accelerators are capable of running deep learning models locally with high computational efficiency (Saravanan et al., 2024). Processing raw image data on-device allows systems to transmit only detection results, for example via compact text files, through low-bandwidth satellite connections (Zuolkernan et al., 2022). This architecture enables real-time monitoring in locations lacking cellular infrastructure while minimizing transmission costs and energy expenditure.

1.2 General Objectives and Structure

The overarching aim of this dissertation is to develop, benchmark, and enhance computer-vision tools for the automated monitoring of wildlife. The primary focus is on deep learning methods applied to data collected through camera traps, mostly in the Catalan Pyrenees but also in other regions of Catalonia and north-western Spain.

This work relies predominantly on YOLO-based computer-vision architectures, exploring and evaluating their performance across multiple ecological tasks, environmental contexts, and deployment conditions.

1.2.1 Wildlife camera trap dataset

A custom dataset was assembled and used throughout the thesis to develop, test, and refine the computer-vision models. The dataset comprises 20 classes, representing 19 mammal species and one general “vehicle” class (Figure 1). This curated dataset forms the foundation for the benchmark analyses and model improvements presented in the subsequent chapters.

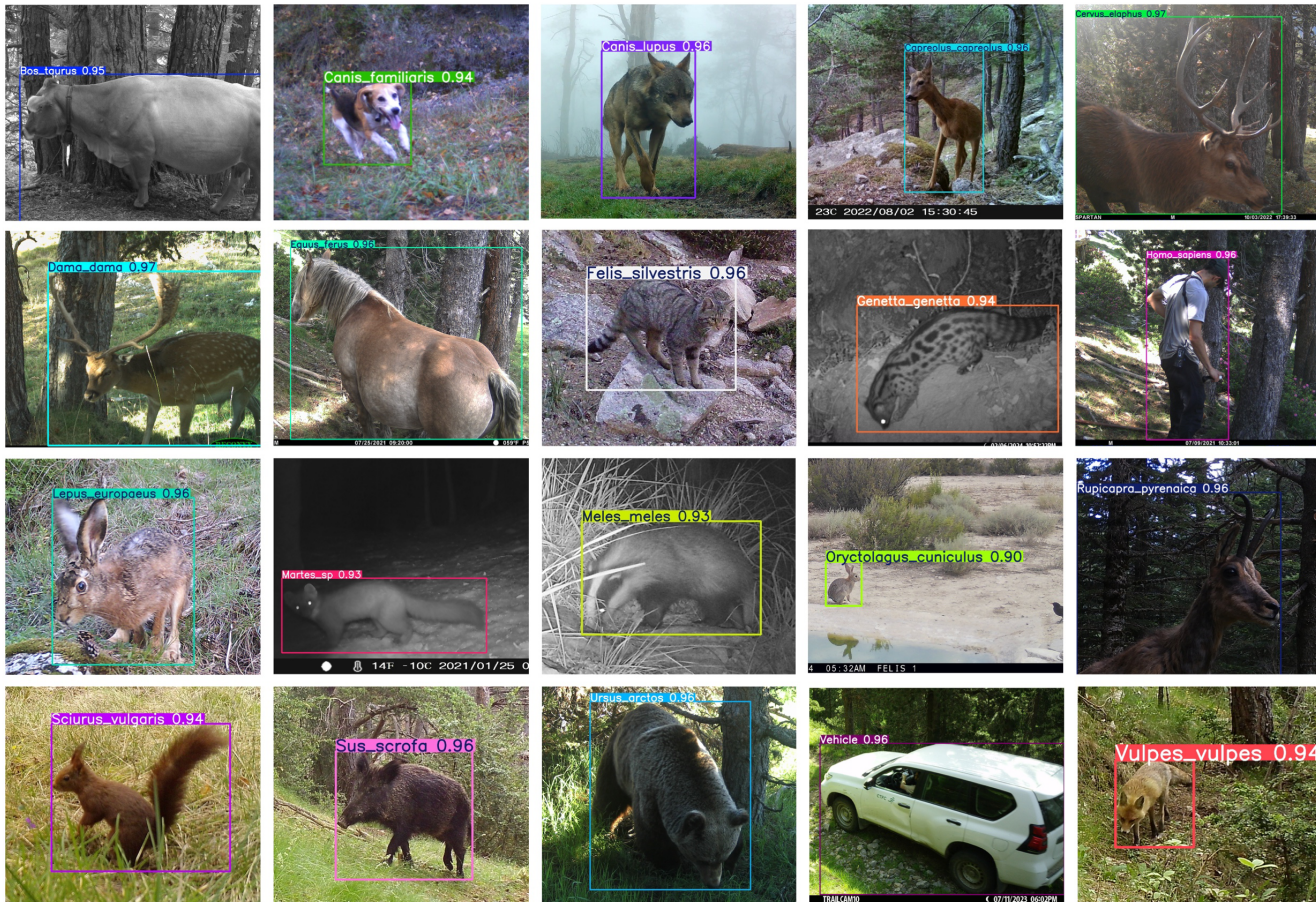


Figure 1. Mosaic of the twenty classes present in the custom dataset used throughout the thesis.

1.2.2 Chapters

Chapter 1: Benchmarking deep learning YOLO Object Detection models for Wildlife Detection in Camera Trap Imagery

Chapter 1 evaluates several versions of the YOLO family, from YOLOv8 to YOLOv12, within the specific context of camera-trap imagery. It provides a comprehensive benchmark of their respective strengths, limitations, and suitability for ecological applications. To achieve this, a region-specific dataset of approximately 50,000 carefully annotated images was developed. The imagery was collected primarily in the Catalan Pyrenees and other regions of Catalonia and north-western Spain.

The objectives of Chapter 1 are to:

1. **Benchmark YOLOv8 to YOLOv12 model variants** and provide a systematic comparison of their detection performance on camera-trap data.
2. **Assemble a curated, region-specific wildlife camera trap dataset** suitable for ecological monitoring applications in the Pyrenees and Catalonia.

Chapter 2: Enhancing Wildlife Detection in Small Camera Trap Datasets using YOLOv9e

Building on the findings from Chapter 1, Chapter 2 addresses challenges commonly associated with small and imbalanced datasets. Focusing on the best-performance model identified earlier (the YOLOv9e architecture), the chapter explores strategies to enhance detection performance without increasing dataset size. These strategies include hyperparameter optimization and domain-specific model training achieved by splitting the dataset based on image palettes (diurnal and nocturnal).

The aims of Chapter 2 are to:

1. **Evaluate the benefits of hyperparameter tuning** under small-dataset conditions.
2. **Investigate the role of domain specificity** by training separate models tailored to distinct image-palette subsets within the camera-trap dataset.

Chapter 3: Real-time wildlife detection using 4G camera traps and Deep Learning

Chapter 3 focuses on the development of a real-time wildlife monitoring system using 4G-enabled camera traps paired with the optimized model developed in Chapters 1 and 2 (Figure 2). A notification integrated through the Telegram application enables real-time alerts when target species are detected. To assess system suitability across field scenarios, the chapter evaluates the relationship between cellular signal intensity and data-transmission time, and benchmarks the processing performance of several computing devices, including the Raspberry Pi as an edge-computing solution.



Figure 2. Deployment of one of the 4G camera traps used in Chapter 3

The goals of Chapter 3 are to:

1. **Develop a real-time monitoring system** that integrates 4G camera traps with a computer-vision model.
2. **Evaluate how cellular signal intensity influences transmission times** for real-time data transfer.
3. **Compare processing speeds across four computing devices** to assess their feasibility within a real-time detection workflow.

Chapter 4: Inferring Brown Bear hair snare interactions by automatically detecting bipedalism on camera trap images using Pose Estimation and a Multi-Layer Perceptron

Chapter 4 expands the scope beyond species detection to explore behavioral inference through pose-estimation techniques. Focusing on the brown bear (*Ursus arctos*) population of the Catalan Pyrenees, it proposes a methodology for detecting interactions with hair snares by classifying bear postures as either bipedalism or quadrupedy. A YOLO-based pose-estimation model is trained and optimized using hyperparameter tuning and transfer learning. Model outputs are subsequently processed using a multilayer perceptron to classify postures. Given the scarcity of annotated bear imagery in the region, transfer learning from other annotated animal pose datasets plays a central role in model development.

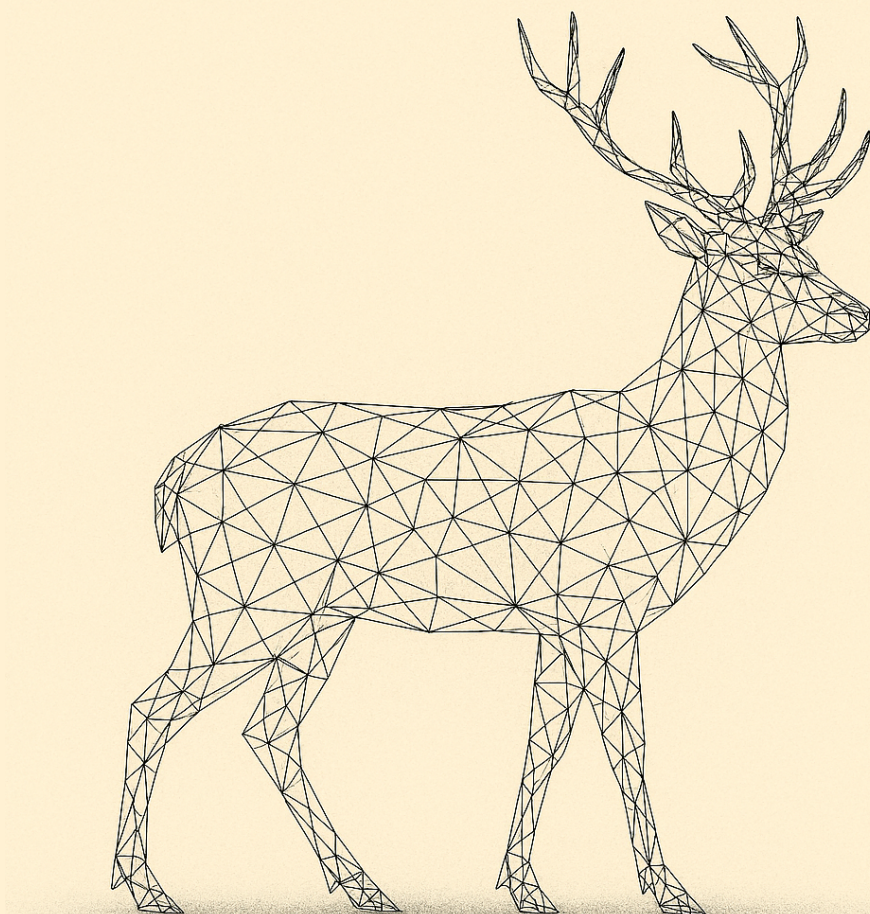
Chapter 4 aims to:

1. Develop a methodology to infer bear back-rubbing behavior at hair snares by predicting bipedalism versus quadrupedy.
2. Adapt a human pose-estimation model to estimate pose in bears.
3. Leverage transfer learning to compensate for limited brown bear imagery by drawing on related annotated datasets.
4. Propose an edge-compatible, lightweight inference system capable of transmitting pose information over low-bandwidth satellite connections, enabling deployment in remote environments.

1.2.3 Summary of dissertation structure

Collectively, the chapters of this dissertation present a holistic pipeline for integrating artificial intelligence into wildlife monitoring, from model benchmarking and performance enhancement to field deployment and behavioral analysis. The work highlights both the opportunities and challenges involved in applying computer-vision methods to ecological datasets and underscores the importance of developing systems that are robust, efficient, and tailored to their ecological context. Ultimately, this research contributes to the expanding field of ecological AI applications by demonstrating how carefully adapted deep learning approaches can enhance wildlife monitoring, improve behavioral interpretation, and support data-driven conservation decision-making.

CHAPTER 2



2 Benchmarking deep learning YOLO Object Detection models for Wildlife Detection in Camera Trap Imagery

Arnau Campanera-Moliné, Víctor Sazatornil, Lluís Brotons, Maguiña Ramilo-Henry, Elena Pi,
Noel Caparroz, Felipe Lumbreras, Robert Benavente, Santiago Palazón, Ferran Sayol, José
Vicente López-Bao, Daniel Ponsa

ABSTRACT

Advances in deep learning have substantially transformed wildlife monitoring by enabling the automated analysis of camera-trap imagery. Despite the rapid development of object detection architectures, comparative assessments of their performance under realistic ecological conditions remain limited, hindering informed model selection by practitioners. In this study, we benchmark a set of recent YOLO (You Only Look Once) object detection models for wildlife detection in camera-trap images: YOLOv8, YOLOv9, YOLOv10, YOLOv11 and YOLOv12. To address limitations of existing datasets in terms of regional relevance and class balance, we assembled a curated camera-trap dataset of ~48,800 images and 20 classes representative of Pyrenean wildlife, encompassing multiple mammal species. All models were trained under identical conditions using pre-trained weights and evaluated on an independent test partition using standard object detection performance indicators. Results show that all YOLO variants achieved robust detection performance on this challenging dataset, while notable differences emerged among model architectures. In particular, an earlier YOLO variant outperformed more recent versions, indicating that architectural novelty does not necessarily translate into improved detection accuracy in ecological applications. Performance was not strongly associated with model size or computational complexity, suggesting that design priorities in newer architectures may favor efficiency over detection reliability. These findings demonstrate that YOLO-based object detection provides a reliable and effective approach for wildlife monitoring with camera traps, while highlighting the continued relevance of well-established architectures. The curated dataset and benchmarking results offer practical guidance for ecologists seeking to deploy deep learning tools tailored to real-world wildlife monitoring tasks. Future work should explore hyperparameter optimization and domain-specific training strategies, as well as alternative architectures such as RT-DETR or hybrid models to further improve detection performance in ecological contexts.

2.1 Introduction

Since their introduction over a century ago, camera traps have become a cornerstone of wildlife research and conservation (Oliver R. Wearn & Paul Glover-Kapfer, 2017). They have revolutionized the way presence, density, and abundance of animals are estimated (Howe et al., 2017) offering low-disturbance, long-term deployment and even remote monitoring capabilities. Additionally, advances in monitoring technologies are enabling more efficient data collection and improving compliance with conservation requirements. Sensors and systems capable of identifying animal species, individuals, and tracking their movements have become faster, more accurate and cost-effective, helping address the urgent demands of international biodiversity conventions (Pimm et al., 2015; T. Dalton et al., 2022).

However, transforming the vast volumes of data generated by these technologies into meaningful ecological insights remains a major challenge and represents a crucial operational bottleneck (Hampton et al., 2013; Keitt & Abelson, 2021). In addition to the high cost of technology, reliable and efficient automating data processing is one of the primary constraints and the largest emerging need for practitioners, researchers and technologists (Hahn et al., 2022).

Recent advancements in hardware and software have led to the emergence of a new generation of data-driven tools in the field of artificial intelligence (AI) (Hampton et al., 2013; Perry et al., 2022). In recent years, an increasing number of researchers are turning to Machine Learning (ML) and Deep Learning (DL) to monitor biodiversity (Thompson, 2023), aiming to automatize and speed parts of the data processing that otherwise must be done manually. Some studies, as the one conducted by Fennel M., et al. (Fennell et al., 2022) quantified the gains and observed an increasing processing speed of over 500%, and the time required for manual processing reduced by a factor of 8.4. The era of “big data” has also reached ecological studies (Hampton et al., 2013), with DL algorithms already demonstrating strong performance in image- and audio-based tasks (Pichler &

Hartig, 2023). While AI-based solutions offer notable improvements, successful implementation still requires a tight collaboration between ecologists and ML specialists to address key challenges (Tuia et al., 2022).

DL methods have been applied to camera trap image classification for over a decade. In 2014, Chen G. (G. Chen et al., 2014b) and collaborators developed the first species recognition algorithm for wild animal classification on camera trap imagery. They used a Deep Convolutional Neural Network (DCNN) with only six layers, trained on 23,876 images and 20 North American species, achieving an overall accuracy of 38.32%. Following their approach, subsequent studies have explored the use of DCNN for animal classification. In 2016, Gomez A. (Gomez et al., 2016) and collaborators trained on the Snapshot Serengeti dataset (3.2 million images) with 26 classes out of 48, using various CNN architectures, achieving 88.9% accuracy with a ResNet. Using the same dataset, Norouzzadeh (Norouzzadeh et al., 2018) and collaborators extended the task to include inference of additional attributes such as the number of individuals and life stage, achieving 94.9% accuracy for classification and 63.1% accuracy for description. In recent years, many researchers have embraced DCNNs to hasten their camera trap imagery classification (Choiński et al., 2021; Rigoudy et al., 2023; Schneider et al., 2020; Tabak et al., 2022; Whytock et al., 2023; Zualkernan et al., 2022).

Although significant advancements have been achieved, and some recent DCNN algorithms may outperform human expertise in visual classification tasks (Van Horn et al., 2018), current algorithms still fall short of human performance in complex tasks, especially when dealing with small or far away objects (Leorna & Brinkman, 2022). Additionally, data is often unevenly distributed, resulting in long-tailed datasets. As the number of training examples decreases for some classes, model accuracy tends to drop sharply. Conversely, when classes are well-represented, models reach peak performance (Van Horn & Perona, 2017). However, such data-ideal conditions are rarely present in nature.

In Ecology, as well as in other scientific domains, the ability to generalize to unseen data is essential. High performing models are frequently trained on data from different contexts, biomes or species, than those in which they are ultimately

applied. Additionally, new camera traps are deployed in new locations each year, further emphasizing the need for robust generalization. Yet, most wildlife recognition models fail to do so (Beery et al., 2018). Consequently, finding a curated, region-specific dataset with well-annotated and balanced class representation, is both crucial and challenging.

These challenges in generalization are compounded by the limitations of standard image classification tasks, where DCNN are typically designed to assign a single class label to an entire image. This approach presents inherent limitations in ecological contexts, particularly in scenarios where multiple distinct species are present within the same image. As a result, conventional classification models are unable to provide multiple labels per image or quantify the number of individual instances.

To address these constraints, object detection provides a more sophisticated solution. Object detection algorithms not only identify and localize multiple individuals within an image by putting a bounding box around each detected animal but also assign class labels to each individual based on the trained species. This enables both species-level classification and individual-level enumeration within a single image.

The recent surge in artificial intelligence development has led to rapid improvements in both accuracy and computational speed, with new model variants emerging only in the past few years. Within the YOLO family, versions from YOLOv8 through YOLOv12 have been released, each aiming to enhance inference speed while minimizing traditional trade-offs between accuracy and efficiency. Despite their sequential naming, these YOLO variants are not strictly built upon one another; rather, each introduces its own architectural innovations.

Although these newer YOLO variants offer improved performance at comparable computational cost, they have not been systematically benchmarked on camera trap datasets. Nevertheless, understanding their strengths and limitations is crucial

for ecologists seeking to select and deploy the most suitable deep learning tools for wildlife monitoring.

This study aims to fill this gap by (1) benchmarking YOLOv8 to YOLOv12 model variants and provide a systematic performance comparison between them. To do so, (2) a curated, wildlife camera trap dataset containing Pyrenees' wildlife images is assembled to control data quality, variability and class balance.

2.2 Materials and Methods

2.2.1 Dataset

2.2.1.1 Data acquisition

For this study, a custom dataset of camera trap images was assembled. These cameras are equipped with passive infrared sensors (PIR) that can detect movement and trigger the shooter if an animal passes in front of them. Moreover, they can use white or infrared flash to keep operating at night. This generates either color or grayscale images, depending on light conditions.

The imagery was compiled from two sources: the Catalan Government's Flora and Fauna Services and the FELIS-ICHN group (Figure 3).

The decision to create a custom dataset rather than relying on publicly available, wildlife camera trap datasets was based on several considerations:

- **Quality control:** Building a custom dataset ensures control over image quality, annotation accuracy and class balance.
- **Regional relevance:** The dataset reflects local fauna and habitat, providing a valuable resource for the scientific community and support for region-specificity ecological research.
- **Application suitability:** The will to deployed trained models for real-time monitoring made existing datasets with non-local species unsuitable for this purpose.

The Flora and Fauna Services, supported by the Catalan Rangers Body, maintain an extensive network of camera trap stations across the Catalan Upper Pyrenees region, primarily in the counties of Pallars Sobirà (42° 31' 18" N, 1° 11' 18" E) and Alta Ribagorça (42° 27' 48" N, 0° 49' 47" E). This network has expanded over the years to include more than 98 active stations in 2023.

The data provided by FELIS-ICHN comes from a long-term mesocarnivore monitoring scheme, which operates through a network of cameras traps distributed across Catalonia. This initiative is based on citizen science, involving approximately 150 volunteers, including naturalists and park rangers, who independently acquired and deployed camera traps in their local environments.

Although the primary focus is on mesocarnivore species, the monitoring scheme records any medium and large mammal (see Fernández-Cabello et al., 2025 for examples of species and area covered). Images are manually classified by volunteers and reviewed by experts in cases of difficult imagery or taxonomic uncertainty, providing high-quality validated data on mammal species across the study area.

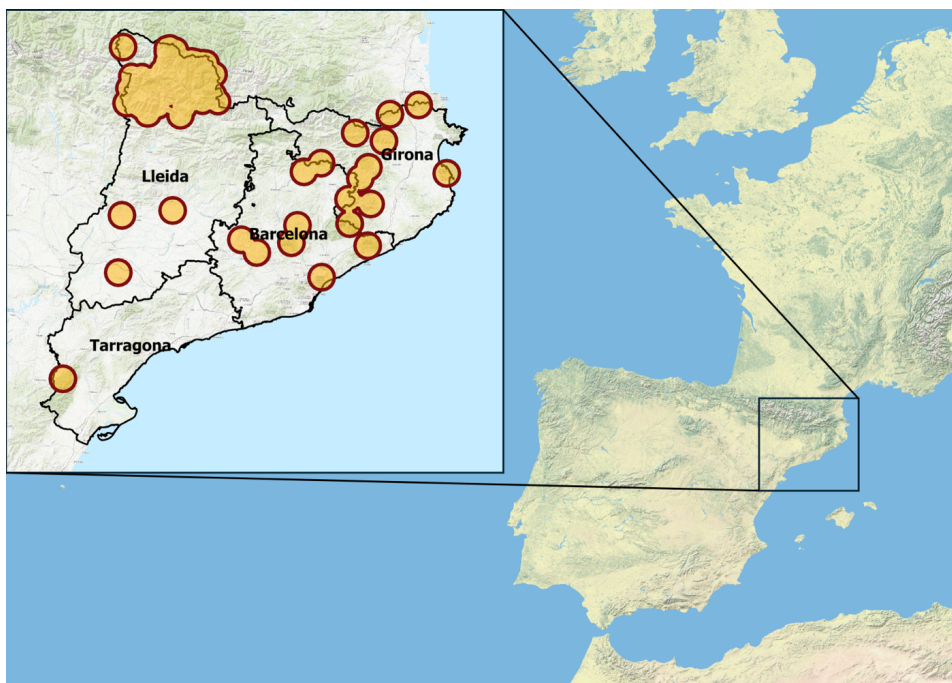


Figure 3. Map of Catalonia, Spain, showing the source of the imagery present in the custom dataset.

2.2.1.2 Data set Composition and Characteristics

A total of 49,302 images, collected between 2017 and 2024, were manually classified by wildlife technicians into 20 distinct classes: 19 mammal species and a “vehicle” class. Species selection was restricted to those most detected by camera traps in the study area. The classes included were: cow (*Bos taurus*), roe deer (*Capreolus capreolus*), European wild cat (*Felis silvestris*), European hare (*Lepus europaeus*), Pyrenean chamois (*Rupicapra pyrenaica*), wild boar (*Sus scrofa*), red fox (*Vulpes vulpes*), European badger (*Meles meles*), red deer (*Cervus elaphus*), red squirrel (*Sciurus vulgaris*), bear (*Ursus arctos*), fallow deer (*Dama dama*),

horse (*Equus ferus*), dog (*Canis familiaris*), wolf (*Canis lupus*), human (*Homo sapiens*), the Martes genus (*Martes sp.*), common genet (*Genetta genetta*), rabbit (*Oryctolagus cuniculus*) and the class vehicle. *Martes sp.* class contained two species: the European pine marten (*Martes martes*) and the Beech marten (*Martes foina*), also known as stone marten, house marten or white breasted marten. The decision of grouping them into one single class was due to the impossibility of correctly distinguishing between both species in the majority of the images.

The wolf is immersed in a remarkable expansion process, recolonizing many parts of its former range in Western Europe (Chapron et al. 2014). Wolf presence in the Pyrenees, composed mostly of solitary, dispersing males, has been regularly monitored since the first observations in the early 2000s, with the first reproduction event documented in 2025. The wolf is a species of particular interest because of the potential conflict around its presence in rural landscapes, and because of the expected population growth after the first packs have been established. For that reason, the species was included in the dataset. Due to limited image dataset from the Pyrenees, the class was complemented with wolf pictures from wolf areas in NW Spain to ensure sufficient representation, and to maintain a relatively balanced dataset. This was the only data sourced from outside of Catalonia.

The dataset included images resolutions ranging from 0.5 to 36 megapixels, with image dimensions spanning from 965x576 to 8000x4500, stored primarily in JPG, with some in PNG format.

To prevent data leakage, the dataset was partitioned at a camera-station level to ensure that all images from a given station were assigned entirely to either training, validation or test subsets. The classes with less pictures (Badgers, Martes & Red squirrels) defined which stations went to training and which to validation, as only a defined number of combinations allowed for an ~80/20% split. From there, the rest of stations were subsequently distributed around this first choice to ensure proper representation of all classes in training and validation subsets.

- Training subset: 83 stations, 33,390 images, 36,892 instances
- Validation subset: 52 stations, 9,075 images, 9,911 instances
- Test subset: 30 stations, 6,837 images, 8,997 instances.

This distribution represented a final ~79/21% proportion between the training and validation subsets. Camera stations in the test set were different from those of training and validation subsets and were primarily sourced from the FELIS-ICHN group. Assessing the performance of a model on unseen data allows to evaluate its generalization capabilities. All classes were present in the test set except for ‘wolf’ and ‘rabbit’ due to the scarcity of available local data. Due to species’ circadian habits, some classes were only present in grayscale images (e.g., the common genet), others only in color images (e.g., the red squirrel), and some in both (e.g., the wild boar, red deer, etc.). Species active both day and night, had more images overall (Figure 4). Between 5 to 10 images per station with no species were included when null detections were kept and not deleted, resulting in 5.6% empty, background images in the total dataset.

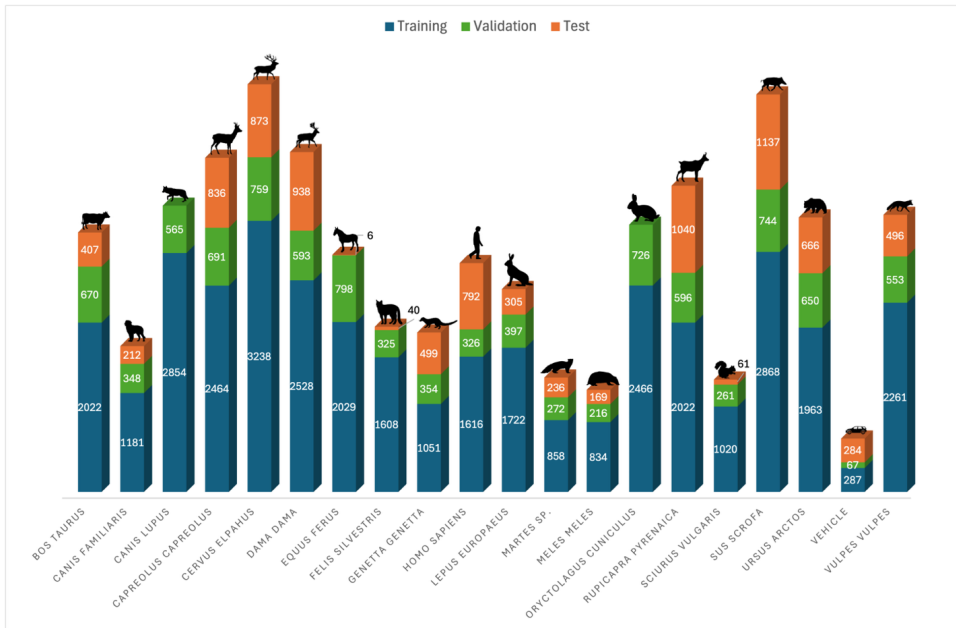


Figure 4. Training, validation and test instances of each class in the custom dataset.

2.2.1.3 Data curation and annotation

Corrupted images were removed from the dataset, along with those that were overexposed, clipped, blurred or affected by intense lens flare.

Annotation tasks were created grouping images by species and stations. This approach facilitated the redistribution of station-class blocks between the subsets.

In computer vision object detection, labeling consists of manually drawing pixel-tight bounding boxes around all the individuals present in an image. The annotation process was carried out using the Computer Vision Annotation Tool (CVAT) (Sekachev et al., 2020), an open source, web-based platform originally developed by Intel. CVAT can be self-hosted via Docker (Merkel, 2014) or accessed through cloud-based subscription plans.

2.2.2 YOLO-based object detection models

2.2.2.1 YOLO Variants

The YOLO (You only look once) family of models represent a state-of-the-art deep learning architecture for real-time object detection. Its hallmark is its single stage detector, where bounding boxes and class labels are predicted in one single network pass. Other model families such as Faster R-CNN (Ren et al., 2016) or RetinaNet (Lin et al., 2018) are two-stage detectors, where in the first stage the network proposes candidate regions where objects might be found, and in the second stage a classifier and a box regressor refines those proposals. This single stage detector made earlier versions of YOLO the only networks capable of inference in real-time, although it came at the cost of slightly lower accuracy. Although other model families such as Faster R-CNN (Ren et al., 2016) and RetinaNet (Lin et al., 2018) have been widely used (Tabak et al., 2019; Whytock et al., 2023; Zualkernan et al., 2022), older YOLO versions continue to appear in recent literature (Choiński et al., 2021; Rigoudy et al., 2023; Roy et al., 2023).

This study benchmarks the latest YOLO variants released from 2023 until early 2025: YOLOv8, YOLOv9, YOLOv10, YOLOv11 and YOLOv12.

Recent versions have introduced significant architectural improvements to enhance object detection accuracy, computational efficiency and speed of inference. In the following, the main characteristics of each model are summarized:

- YOLOv8 (2023) – Developed by Ultralytics (Jocher et al., 2023a), an open-source organization and software development company, upon previous YOLO versions. It introduced advancements in backbone and neck architectures resulting in **improved feature extraction capabilities**. The model also introduced **anchor-free split heads** for more efficient detection. Each head oversees a prediction task and works independently. This approach improves efficiency and accuracy, as each task is optimized separately, and interference between tasks is reduced. These improvements translate to better overall accuracy as well as better performance on irregular and smaller objects (Jocher et al., 2023b).
- YOLOv9 (2024) - Two major innovations were added: **Programmable Gradient Information** and **Generalized Efficiency Layer Aggregation Network**. These additions solved the Information Bottleneck Principle (Tishby & Zaslavsky, 2015), a common issue in deep learning where important data gets lost as it passes through the layers. Also, **aggregating features from multiple layers**, efficiency and flexibility was improved (C.-Y. Wang et al., 2024). These innovations meant a new landmark in object detection accuracy.
- YOLOv10 (2024) - Developed by researchers at Tsinghua University, YOLOv10 addressed post-processing and model deficiencies found in previous YOLO versions by introducing key innovations such as ‘**One-to-many heads**’ or **eliminating non-maximum suppression**. YOLOv10 did not outperform v9 in accuracy, but it showed similar results with significantly reduced computational overhead (A. Wang et al., 2024).

- YOLOv11 (2024) – Developed by Ultralytics. Built upon their previous versions (v5 and v8), YOLOv11 focused on **optimizing different parts of the architecture**, achieving higher accuracy than v10 with fewer parameters. It also enhanced support for tasks like segmentation, pose estimation, classification and oriented object detection. Overall, v11 was **optimized for edge devices and cloud development** (Jocher & Qiu, 2024).
- YOLOv12 (2025) – Developed by researchers at the University at Buffalo, the last member of the YOLO family lineage, YOLOv12, departed from the traditional CNN-based approaches and introduced an **attention-centric architecture** and a **Residual Efficient Layer Aggregation Network** alongside other architectural improvements. YOLOv12 showed marginal gains in accuracy compared to YOLOv9 but added training and inference costs due to its complex architecture (Tian et al., 2025).

Since their initial release in 2016, YOLO versions have all had different network size variants. Small sizes, also known as ‘tiny’, ‘small’ or ‘nano’, are very light and fast, primarily targeted for real-time tasks running on edge devices or machines with low computational power. However, they come with a trade-off in accuracy and robustness. Medium sized models offer a balance between accuracy and real-time capabilities, being slower and heavier than small sized versions. Finally, large and extra-large model sizes offer highest accuracy for tasks where speed is less important, with strong performance on challenging datasets but preventing any real-time video inference application on most devices.

In this study, the largest sizes of the models were used (Table 1). Larger model sizes are better suited for complex visual environments and challenging datasets than their smaller counterparts, but tend to overfit on small datasets, as the custom one used. However, in this study models were not trained from scratch, but rather

initialized with the pre-trained weights from the COCO 2017 Dataset (more than 200,000 images), as described in section 2.2.2.

Table 1. Comparison of benchmarked YOLO variants by release year, key innovations, number of parameters, number of layers and GFlops. GFlops are the number of mathematical operations that a model needs to perform to predict one single image. Higher GFlops indicate higher computational cost per prediction.

Model	Year	Key innovations	Parameters	Layers	GFlops
YOLOv8x	2023	• Backbone & neck improvements • Anchor-free split heads	68,2 M	209	258,1
YOLOv9e	2024	• PGI (Programmable Gradient Information) • GELAN (Generalized Efficiency Layer Aggregation Network)	58,1 M	721	192,7
YOLOv10x	2024	• NMS-free • One-to-many heads • Significantly reduced computational overhead	31,7 M	400	171
YOLOv11x	2024	• Architectural improvements for higher accuracy and efficiency • Support for segmentation, pose estimation, classification and oriented object detection tasks • Optimized for edge devices and cloud development	56,9 M	357	195,5
YOLOv12x	2025	• Attention-centric architecture • Residual Efficient Layer Aggregation Network (R-ELAN)	59,1 M	488	199,8

2.2.2.2 Pre-trained weights

Ultralytics maintains a repository of pre-trained weights for all YOLO variants and sizes. These models are trained on the Common Objects in Context (COCO) dataset (Lin et al., 2014), which includes ~118k color images and ~886k annotated everyday objects across 80 different classes. Although most COCO classes are unrelated to ecology, using pre-trained weights can accelerate model training and reduce computational costs, particularly when working with small datasets. When using pre-trained weights, the model is already capable of finding very basic and common patterns to form parts and structures that lastly lead to object detection (Kolesnikov et al., 2020). For this study, models initialized with COCO 2017 Dataset pre-trained weights were used.

2.2.3 Experimental setup

2.2.3.1 Model training

Training was conducted on a Supermicro GPU server equipped with 768 GB of RAM and 4x NVidia L40S GPUs (totaling 184 GB of VRAM). The software environment was standardized across all experiments, using CUDA version 12.5 and Ultralytics YOLO version 8.3.193. Also, the same default hyperparameters were maintained throughout training; input resolution was scaled to 896 pixels and batch size was maintained at a value of 24.

The popular AdamW (Adaptive Moment Estimation with Weight Decay) (Loshchilov & Hutter, 2019) optimizer was selected to ensure stable and efficient convergence. Compared to SGD (Stochastic Gradient Descend), which often requires extensive hyperparameter tuning and large datasets to achieve optimal generalization (Bottou, 2010). AdamW typically provides faster convergence and robust performance in small to medium datasets.

To prevent overfitting, an early stopping function was applied with a patience of 100 epochs. This form of regularization halts the training if the model shows no

improvement on the validation set over a defined number of consecutive epochs. This method helps the model to better generalize on unseen data.

2.2.3.2 Loss function

To measure how well a machine learning model is acquiring new knowledge loss functions are used. These are mathematical formulas that measure model's predictions with the actual target values. It acts as a guide for the training process: the lower the loss, the better the model is performing.

There exist several different loss functions to measure different tasks. Ultralytics YOLO implementation uses a combination of different loss functions to measure performance on bounding box placement and classification. Then, they are averaged in a weighted sum to give a final score that guides the training process.

2.2.3.3 Data augmentation

To improve model generalization and robustness data augmentation was performed. This technique consists of a series of modifications applied to the original data to augment the amount of trainable images. The following set of data augmentation methods were automatically applied during training by the pipeline of Ultralytics YOLO implementation.:

- Color space adjustments: hue (hsv_h), saturation (hsv_s), and brightness (hsv_v).
- Geometric transformations: translation (translate) and scaling (scale).
- Mosaic augmentation (mosaic): combining four images into one.
- Flipping images horizontally (fliplr). Vertical flipping was excluded due to its impracticality in real-world scenarios.

2.2.4 Evaluation metrics

Computer vision models usually use different metrics to keep track of a model's performance. In object detection tasks there are two crucial abilities that need to be evaluated: how precise is the placement of the inferred bounding box and assigning the correct class to that box.

2.2.4.1 Intersection over Union

A popular metric to quantify how precise a model can place a bounding box around an object is *Intersection over Union* (IoU). This metric quantifies the overlap between the ground truth and the predicted bounding box and compares it to the total area of both ground truth and prediction.

$$\text{Intersection over Union} = \frac{A \cap B}{A \cup B}$$

An IoU of 1 indicates a perfect match whereas a score of 0 indicates no overlapping between the boxes (Figure 5).

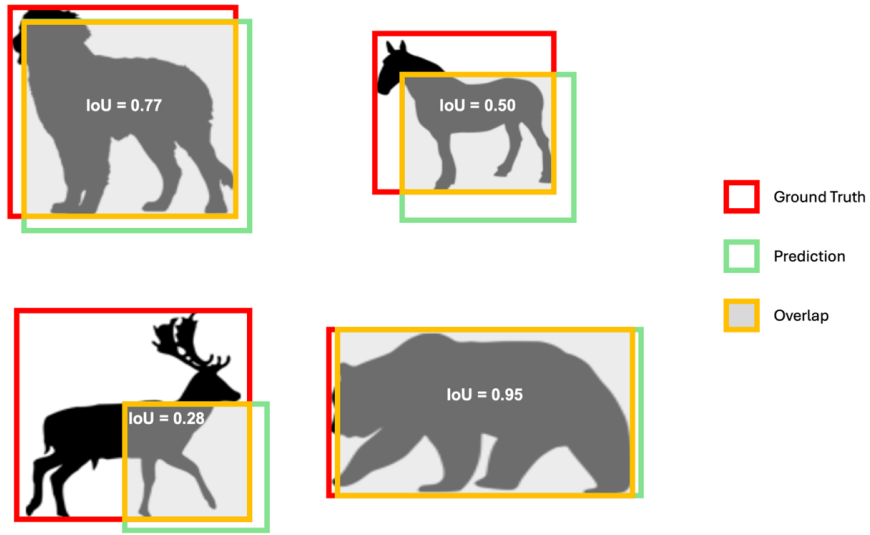


Figure 5. Different Intersection over Union thresholds.

2.2.4.2 Precision, recall and F1-score

Assigning the correct label to a predicted box is also crucial in object detection tasks. To assess this, precision, recall and F1-score metrics are computed.

- Precision: measures out of all the detections, how many were correct. In other words, the ratio between true positives (TP) and all predicted positives (TP + False Positives (FP)).

$$Precision = \frac{TP}{(TP + FP)}$$

- Recall: measures how many labels were correctly predicted out of all the ground truth labels. Also, in other words, the ratio between true positives and all the positives that should have been predicted (TP + False Negatives (FN)).

$$Recall = \frac{TP}{(TP + FN)}$$

- F1 score: It combines precision and recall metrics into a single value balancing the trade-off between both. F1 score represents the harmonic mean between these two metrics, and it is usually given at different confidence values.

To consider a True Positive the predicted IoU between the ground truth annotation and the predicted bounding box must be ≥ 0.5 . To compute the global metrics, the unweighted average of all per-class metrics is calculated.

2.2.4.3 mean Average Precision (mAP)

But the gold standard metric in object detection tasks is mAP (mean Average Precision) as it captures both localization quality and classification correctness. Two mAP metrics are commonly used: mAP@50 and mAP@50-95. The difference lies in the IoU used to calculate the metric.

- mAP@50: The detection is considered correct if the class matches at an IoU threshold ≥ 0.5
- mAP@50-95: uses 10 different thresholds, from 0.5 to 0.95 in 0.05 steps and then averages them all.

Being easier, mAP@50 scores tend to be much higher than mAP@50-95 scores, which are much stricter. However, the latter measures model quality more realistically.

For the aforementioned reasons, mAP@50-95 was used as the main evaluation metric for this experiment.

2.2.5 Benchmarking protocol

2.2.5.1 Training

All YOLO variants were trained under identical conditions using Ultralytics YOLO framework on the dataset described in section 2.1 (Table 2). Models were initialized with COCO 2017 pre-trained weights, trained using the exact same training partition, validated on the same validation split and evaluated on a held-out test set to prevent data leakage and ensure unbiased evaluation.

The same data preprocessing and augmentation were applied by Ultralytics package to keep the same conditions across all the variants. All training sessions lasted until no improvements were observed on the validation subset.

Table 2. Summary of metaparameters kept during all training sessions.

Parameter	Value	Parameter	Value
<i>optimizer</i>	AdamW	<i>cls</i>	0.5
<i>batch size</i>	24	<i>dfl</i>	1.5
<i>Input size</i>	896	<i>hsv_h</i>	0.015
<i>lr0</i>	0.01	<i>hsv_s</i>	0.7
<i>lrf</i>	0.01	<i>hsv_v</i>	0.4
<i>momentum</i>	0.937	<i>translate</i>	0.1
<i>weight decay</i>	0.0005	<i>scale</i>	0.5
<i>warmup epochs</i>	3.0	<i>fliplr</i>	0.5
<i>warmup momentum</i>	0.8	<i>mosaic</i>	1
<i>box</i>	7.5		

2.2.5.2 Evaluation

Following training, models were evaluated on the test set. Standard detection metrics were computed, including mAP@50, mAP@50-95, Precision, Recall and F1-score. A confusion matrix showing classification performance was also computed for each model.

Although speed of inference can be crucial in other tasks or applications, detection performance is the primary focus for camera trap image classification.

2.3 Results

The validation and tests subsets were both considered to detect generalization or overfitting problems.

2.3.1 Models Performance on validation and test set

All models yielded > 0.75 in $mAP@50-95$ on the validation set, which is considered excellent for object detection tasks given the complexity of the dataset. Similarly, precision values were consistently > 0.85 , and recall values > 0.78 , indicating robust detection and classification performance across all variants on the validation (Table 3).

Table 3. Scores of the principal metrics achieved by each YOLO model variant on the validation set. Shaded cells show best model performance for a specific metric.

	<i>mAP@50</i>	<i>mAP@50-95</i>	<i>Precision</i>	<i>Recall</i>	<i>F1</i> (score at conf.)
<i>YOLOv8x</i>	0.877	0.763	0.885	0.801	0.83 @ 0.69
<i>YOLOv9e</i>	0.882	0.77	0.873	0.818	0.84 @ 0.58
<i>YOLOv10x</i>	0.876	0.767	0.869	0.797	0.83 @ 0.48
<i>YOLOv11x</i>	0.87	0.754	0.861	0.791	0.82 @ 0.55
<i>YOLOv12x</i>	0.86	0.753	0.852	0.788	0.81 @ 0.62

YOLOv9e obtained the best overall validation metrics, with a $mAP@50-95$ score of 0.77, a $mAP@50$ score of 0.882 and a recall of 0.818. YOLOv8x achieved better precision than YOLOv9e (0.885 vs 0.873) but felt behind on the rest of metrics. The lowest-performing model was YOLOv12x, with a $mAP@50-95$ score of 0.753, nearly identical to YOLOv11x (0.754). Overall, performance across models

was fairly similar, without major performing differences between YOLO versions. Training evolution is shown on Figure 6.



Figure 6. Training evolution of each YOLO model variant. Performance scores are based on the validation set. The last value represents the best performing epoch found before overfitting.

When evaluated on the test set, all models exhibited an approximate 10% performance drop compared to validation results, indicating a moderate generalization gap. This outcome is expected for a challenging wildlife dataset characterized by occluded animals, variable background and diverse species. Test performance metrics are summarized in Table 4.

Table 4. Scores of the principal metrics achieved by each YOLO model variant on the test set. Shaded cells show best model performance for a specific metric.

	<i>mAP@50</i>	<i>mAP@50-95</i>	<i>Precision</i>	<i>Recall</i>	<i>F1</i> (score at conf.)
<i>YOLOv8x</i>	0.763	0.667	0.754	0.725	0.70 @ 0.64
<i>YOLOv9e</i>	0.787	0.693	0.753	0.738	0.71 @ 0.56
<i>YOLOv10x</i>	0.746	0.646	0.744	0.72	0.68 @ 0.46
<i>YOLOv11x</i>	0.709	0.613	0.74	0.686	0.66 @ 0.6
<i>YOLOv12x</i>	0.718	0.624	0.704	0.718	0.67 @ 0.51

Despite the performance drop, all models maintained a mAP@50-95 score > 0.61, which is still competitive for wildlife detection tasks.

Similar to the validation results, YOLOv9e delivered the best overall performance, leading most of the metrics' scores on the test set; a mAP@50-95 score of 0.693, a mAP@50 score of 0.787, a recall of 0.738 and an F1-score of 0.71 at a 0.56 confidence threshold. YOLOv8x was the second-best performance model, with the best overall score of precision, at 0.747. YOLOv10 and subsequent versions did not achieve better scores at any performance metric. YOLOv11x achieved the worst scores in mAP@50 with 0.709, in mAP@50-95 with 0.613, in recall with 0.686 and at F1-score with 0.66 at a 0.6 confidence threshold. However, it trained the fastest (0.78 days) and reached its best score within the lowest epoch count (79). Finally, YOLOv12x did not achieve the best score at any performance metric and trained for the longest, reaching 2.58 days.

2.3.2 Model- and species-wise performance on the test set

All YOLO variants exhibited similar per-class performance on the test set, which is expected when training models of the same lineage on identical data. However, some notable differences emerged:

YOLOv8x and YOLOv9e had similar performance across the majority of classes. YOLOv8 outperformed YOLOv9e in detecting hares (90% vs. 85%), Pyrenean chamois (77% vs 74%), wild boars (64% vs. 62%), red deer (66% vs. 63%), red squirrel (87% vs. 85%), humans (88% vs. 86%) and vehicles (89% vs. 88%). For other classes, YOLOv9e had superior recall by margins of 1-10%. These small differences reflect YOLOv9's incremental improvements over YOLOv8, rather than a radical architectural shift.

YOLOv10x and YOLOv11x also share similar architectural designs, with the latter being a refinement on v10's concepts. However, YOLOv10x generally performed better on the test set except in detecting hares (87% vs. 84%), foxes (54% vs. 50%), red squirrels (89% vs. 84%), brown bears (81% vs. 80%) and fallow deer (51% vs. 48%). The rest of classes were better predicted by YOLOv10x with significantly higher performance for cows (29% vs 19%), red deer (68% vs. 58%) and wildcats (80% vs 72%).

Finally, YOLOv12x performance level was between YOLOv10 and YOLOv11 for most classes. Unexpectedly, YOLOv12x achieved the highest overall recall for red squirrels (92% vs. 89% of YOLOv11x) and fallow deer (56% vs. 52% of YOLOv9e).

Figure 7 shows the amount of classes grouped by performance levels.

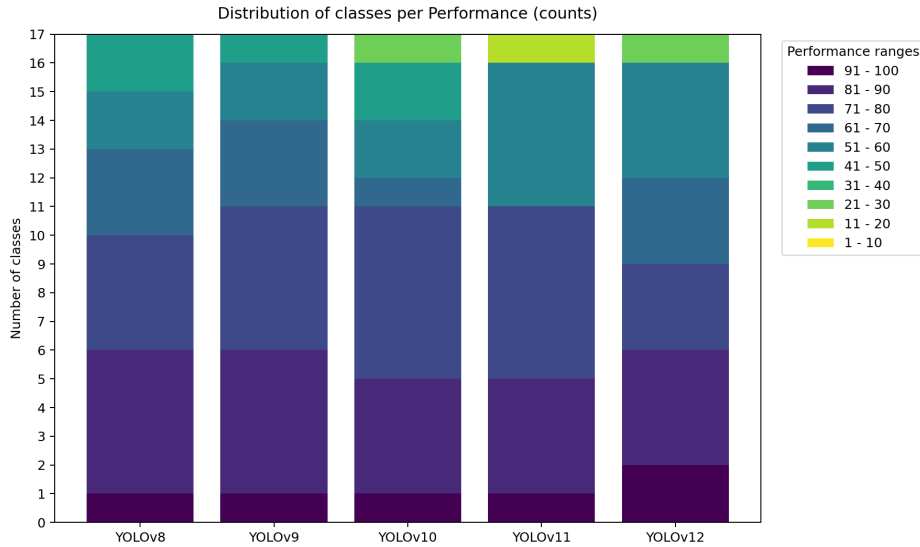


Figure 7. Stacked bar graph showing the number of classes in each performance range for each YOLO version.

Confusion matrix and detailed performance charts for each model are provided in the supplementary material.

Per-class recall is shown in Table 5. for the best-performing model, YOLOv9e:

- The five most accurately detected classes were: common genet (95%), vehicle (88%), brown bear (87%), human (86%), red squirrel (85%) and hare (85%).
- The five least accurately detected classes were: wild boar (62%), dog (61%), fox (60%), fallow deer (52%) and cow (50%).

Similarly, for the worst-performing model, YOLOv11x:

- The five most accurately detected classes were: common genet (95%), red squirrel (89%), vehicle (87%), hare (87%) and horse (83%).
- The five worst accurately detected classes were: wild boar (58%), fox (54%), dog (51%), fallow deer (51%), and cow (19%).

Despite some significant performance differences in specific classes, these results show that all model versions had a robust and competitive performance.

Table 5. Per-class recall by model. The classes ‘wolf’ (*Canis lupus*) and ‘rabbit’ (*Oryctolagus cuniculus*) were not present in the test set. Similarly, class ‘horse’ (*Equus ferus*) only had 6 instances in the test set, which made their results not relevant. Green- and orange-colored cells show the best and worst performing model respectively, for that particular species.

	Yv8x	Yv9e	Yv10x	Yv11x	Yv12x
<i>Bos taurus</i>	42	50	29	19	24
<i>Canis familiaris</i>	58	61	57	51	54
<i>Canis lupus</i>					
<i>Capreolus capreolus</i>	78	78	77	71	68
<i>Cervus elpahus</i>	66	63	68	58	61
<i>Dama dama</i>	50	52	48	51	56
<i>Equus ferus</i>	83	83	1	83	1
<i>Felis silvestris</i>	67	72	80	72	67
<i>Genetta genetta</i>	95	95	96	95	95
<i>Homo sapiens</i>	88	86	84	80	83
<i>Lepus europaeus</i>	90	85	84	87	84
<i>Martes sp.</i>	77	77	75	74	76
<i>Meles meles</i>	79	80	76	73	75
<i>Oryctolagus cuniculus</i>					
<i>Rupicapra pyrenaica</i>	77	74	78	75	77
<i>Sciurus vulgaris</i>	87	85	84	89	92
<i>Sus scrofa</i>	64	62	58	58	60
<i>Ursus arctos</i>	86	87	80	81	87
<i>Vehicle</i>	89	88	89	87	87
<i>Vulpes vulpes</i>	60	60	50	54	53

Models YOLOv8x and YOLOv9e achieved the best performance in 65% of the classes. YOLOv10x and YOLOv12x led in 23% and 12% classes respectively. YOLOv11x had the worst performance in 47% of the classes, followed by YOLOv10x in 35%.

The common genet was the most accurately detected species across all models, achieving $\geq 95\%$ accuracy in all of them. Its singular pattern with black spots and

a long-stripped tail may facilitate recognition and classification. On the contrary, the cow was the most misclassified species by far, never exceeding 50% accuracy, with YOLOv11x scoring as low as 19%. When analyzing the images, most of the partially occluded cows were confused by horses.

2.3.3 Best model's Performance Comparison between validation set and test set

Analyzing the differences in performance between the validation and the test set can provide insights about shortcomings of the model. For example, a performance drop between the validation and the test set may indicate a generalization gap or overfitting for certain classes.

For YOLOv9e, performance metrics between the validation and test set had a mean drop of 0.084 in mAP@50-95, a drop of 0.097 in mAP@50, a drop of 0.114 in precision and a drop of 0.09 in recall. Some classes showed a bigger drop while others increased their detection rate. Classes with the largest drops included wildcat (0.804 → 0.542), Martes sp. (0.756 → 0.476), dog (0.777 → 0.534), fox (0.804 → 0.557) and fallow deer (0.785 → 0.589). Conversely, some classes improved their metrics between the validation and the test set; human (0.768 → 0.81), bear (0.839 → 0.876), hare (0.651 → 0.776) and vehicle (0.693 → 0.851).

For example, the wildcat class suffered a -0,262 drop, likely due to the small number of examples in the test set compared to validation (40 vs 325), which may lead to poor statistical generalization of results in the validation. In contrast, the vehicle class improved due to a larger test sample (284 vs 67 in validation), which helped the results to better generalize statistically. Classes with similar sample sizes but showing significant drops may reflect more challenging test images than in validation, or some degree of overfitting.

Focusing on the best performing model YOLOv9e, results on validation and test set were as shown on Table 6.

Table 6. Principal results on the validation and test sets. As described previously, the shaded classes ‘wolf’ (Canis lupus) and ‘rabbit’ (Oryctolagus cuniculus) were not present in the test set. Similarly, class ‘horse’ (Equus ferus) only had 6 instances in the test set, which made their results not relevant.

YOLOv9e	Training		Validation						Test					
	Images	Instances	Images	Instances	mAP@50	mAP@50-95	Precision	Recall	Images	Instances	mAP@50	mAP@50-95	Precision	Recall
<i>Bos taurus</i>	1609	2022	524	670	0.838	0.744	0.866	0.737	285	407	0.676	0.6	0.836	0.487
<i>Canis familiaris</i>	901	1181	323	348	0.831	0.777	0.717	0.793	178	212	0.6	0.534	0.566	0.592
<i>Canis lupus</i>	2183	2854	445	565	0.815	0.726	0.887	0.669						
<i>Capreolus capreolus</i>	2474	2464	680	691	0.898	0.818	0.785	0.855	814	836	0.826	0.753	0.715	0.787
<i>Cervus elpahus</i>	2716	3238	678	759	0.853	0.785	0.805	0.769	551	873	0.806	0.723	0.875	0.638
<i>Dama dama</i>	2075	2528	498	593	0.853	0.785	0.847	0.771	674	938	0.683	0.589	0.836	0.499
<i>Equus ferus</i>	1179	2029	548	798	0.853	0.714	0.905	0.716	5	6	0.754	0.743	0.0663	1
<i>Felis silvestris</i>	1618	1608	325	325	0.884	0.804	0.826	0.825	40	40	0.589	0.542	0.27	0.714
<i>Genetta genetta</i>	1041	1051	354	354	0.995	0.941	0.992	0.999	499	499	0.971	0.861	0.962	0.925
<i>Homo sapiens</i>	1131	1616	298	326	0.899	0.768	0.969	0.862	451	792	0.943	0.81	0.97	0.865
<i>Lepus europaeus</i>	1869	1722	397	397	0.787	0.651	0.666	0.907	289	305	0.888	0.776	0.841	0.866
<i>Martes sp.</i>	939	858	272	272	0.922	0.756	0.904	0.832	193	236	0.661	0.476	0.619	0.691
<i>Meles meles</i>	906	834	212	216	0.925	0.814	0.908	0.87	169	169	0.897	0.789	0.924	0.834
<i>Oryctolagus cuniculus</i>	2318	2466	707	726	0.903	0.687	0.974	0.779						
<i>Rupicapra pyrenaica</i>	1655	2022	417	596	0.842	0.754	0.895	0.721	618	1040	0.791	0.717	0.788	0.718
<i>Sciurus vulgaris</i>	1088	1020	260	261	0.926	0.758	0.959	0.806	61	61	0.774	0.614	0.729	0.77
<i>Sus scrofa</i>	2281	2868	620	744	0.917	0.779	0.911	0.851	776	1137	0.765	0.665	0.874	0.593
<i>Ursus arctos</i>	1954	1963	522	650	0.928	0.839	0.958	0.869	523	666	0.948	0.876	0.975	0.874
<i>Vehicle</i>	212	287	60	67	0.879	0.693	0.911	0.836	275	284	0.934	0.851	0.941	0.877
<i>Vulpes vulpes</i>	2354	2261	549	553	0.899	0.804	0.78	0.893	496	496	0.669	0.557	0.76	0.556

2.4 Discussion

This study benchmarks YOLO variants from v8 to v12 in the specific task of detecting wildlife in camera-trap images. YOLOv9e achieved the highest overall performance, followed by YOLOv8x, YOLOv10x, YOLOv12x and YOLOv11x.

Additionally, a region-specific dataset was ensembled, containing 49,302 images across 20 classes. Constructing a region-specific dataset can be challenging as data is often scattered across small partners and organizations or insufficient to achieve acceptable model performance.

2.4.1 Interpretation of Model and Class Performance

Performance differences among models did not correlate with parameter nor layer count, necessary GFLOPs, or release novelty, suggesting that architectural design could be the primary cause.

This could arguably be the case of model variants YOLOv10x and YOLOv11x. YOLOv10x had the lowest parameter count (31,7 M) and the lowest GFlops (171) per inference, indicating a holistic efficiency-accuracy driven model design, mostly optimized for real-time end-to-end object detection (A. Wang et al., 2024). Despite its pruned architecture, YOLOv10x achieved the best score in five different classes, demonstrating a very well-designed trade-offs between performance and computational cost. Similarly, YOLOv11 was optimized for edge devices, reducing its parameter count from its previous version, while trying to maintain accuracy and performance. Even though it demonstrated a 0.3 mAP@50-95 score increase in the COCO Dataset compared to YOLOv10x, it still remained far from the most accurate model, YOLOv9e (+0.9). This performance improvement was not achieved on this study's dataset, where it performed worse than YOLOv10x. Finally, YOLOv12x introduced a new attention-centric architecture. Despite achieving the best detectability rate for the red squirrel and the bear it had a mediocre performance on the rest of the classes.

2.4.2 Relevance for Wildlife Monitoring

The most common operational workflow when working with camera trap imagery involves manually classifying images or, more recently, using computer vision algorithms to detect wildlife in it, at the office or lab. This allows using the largest, more computationally-demanding models instead of leaning towards efficiency and speed. Yet, recent YOLO models seem to prioritize edge-device optimization, trading-off accuracy for efficiency and speed. These are versatile, generalist models, not specifically targeted to wildlife detection, but adaptable to a wide range of different applications. Ecological monitoring demands maximum accuracy and recall, with speed of inference and computational cost being secondary. According to this, older models like YOLOv8 and YOLOv9 may remain preferable for wildlife studies, in spite of the newer, more efficient but less accurate YOLO versions. Although being more computationally expensive, older models provide more robust and accurate detections, reducing manual classification workload in wildlife monitoring and conservation.

However, successful deployment still requires tight collaboration between AI specialists and ecologists to maximize the potential of DL and ML solutions for conservation needs(Besson et al., 2022).

2.4.3 Limitations

Despite model performance metrics being within an excellent range, the size of the used dataset, alongside some degree of class imbalance likely constrained overall performance. Previous published studies using DL in wildlife camera trap imagery have used millions of images(Rigoudy et al., 2023; Tabak et al., 2019), ensuring better representation of the classes, thus achieving better performance levels. However, when using DL models for specific tasks in a particular region, there is often a mismatch between the detectable species of a big model trained with millions of images and the local fauna. Thus, custom datasets must often be ensembled and annotated from scratch, rarely meeting the aforementioned ideal conditions, presenting most of them, some degree of class imbalance due to the natural, intrinsic habits of the studied species.

Additionally, test data had a subtle domain-shift as most of the images came from the citizen science initiative, which did not represent the majority of the training data. Camera traps from this network were positioned lower than usual (knee height, around 40 cm) as their main focus was monitoring mesocarnivore species. Conversely, most images in the training set came from camera trap positioned at a more common height (~150 cm) recommended by camera trap manufacturers. This shift in perspective may have had some impact in detectability for images in the test set, thus hampering model performance.

2.4.4 Future Directions

Although this study is focused on showing the benchmark results of several model variants trained under the same conditions on the same dataset, standard machine learning practices can further improve model performance without the need of acquiring more data. These enhancements may include performing an optimal hyperparameters search, where several meta-parameters affecting the training behavior are tuned to tailor the training procedure to a particular dataset's characteristics. Finding the optimal hyperparameters for a given dataset is a very heavy computational task that was out of the scope of this study. The benchmarked models are required to detect both daytime and nighttime images similarly. Other, more technical approaches could involve improving the specificity of the training domain by, for example, creating two separate datasets according to the palette of each image (grayscale and color). Finally, other model families and architectures such as Faster R-CNN (Ren et al., 2016), Mask R-CNN (He et al., 2018), RetinaNet (Lin et al., 2018), RTDETR (Lv et al., 2024), DiffusionDet (S. Chen et al., 2023), or hybrid models (X. Lei et al., 2025) could also be explored.

Further research should also include the exploration of the different YOLO size variants (n, s, m, l) for edge deployment. YOLO's inherent speed and efficiency make it a strong candidate for on-device wildlife monitoring. Although other studies have already used computer vision models in edge devices for wildlife monitoring (Whytock et al., 2023; Zualkernan et al., 2022), YOLO variants were not employed in these cases.

2.5 Conclusions

Understanding which tools deliver the best results for wildlife monitoring is essential for the ecological research community to optimize their often-limited time and resources in an era of rapidly evolving AI technologies.

In this study, five different YOLO variants were benchmark on a relatively small custom camera trap dataset of ~50,000 images across 20 classes. YOLOv9e achieved the highest overall performance, despite not being the newest model, nor the one with the largest parameter count. For applications where maximum reliability is prioritized over inference speed, researchers should use more robust models such as YOLOv9 or YOLOv8 for camera trap image analysis.

Additionally, building a custom dataset ensured control over annotation quality and class balance while creating a very valuable asset for the local scientific community.

Although this study focused on YOLO variants, further testing should explore alternative architectures such as RT-DETR or hybrid models, which may offer competitive performance.

2.6 Supplementary Material

2.6.1 Per-class model performance for each model

YOLOv9e	Validation						Test					
	<i>Images</i>	<i>Instances</i>	<i>mAP@50</i>	<i>mAP@50-95</i>	<i>Precision</i>	<i>Recall</i>	<i>Images</i>	<i>Instances</i>	<i>mAP@50</i>	<i>mAP@50-95</i>	<i>Precision</i>	<i>Recall</i>
<i>Bos taurus</i>	524	670	0.849	0.763	0.87	0.727	285	407	0.668	0.583	0.889	0.393
<i>Canis familiaris</i>	323	348	0.846	0.786	0.809	0.79	178	212	0.543	0.471	0.619	0.59
<i>Canis lupus</i>	445	565	0.809	0.725	0.942	0.623						
<i>Capreolus capreolus</i>	680	691	0.874	0.803	0.76	0.849	814	836	0.82	0.755	0.721	0.772
<i>Cervus elpahus</i>	678	759	0.827	0.76	0.823	0.725	551	873	0.79	0.704	0.867	0.625
<i>Dama dama</i>	498	593	0.838	0.765	0.892	0.728	674	938	0.674	0.572	0.84	0.478
<i>Equus ferus</i>	548	798	0.843	0.707	0.926	0.669	5	6	0.504	0.499	0.069	0.833
<i>Felis silvestris</i>	325	325	0.868	0.79	0.844	0.806	40	40	0.537	0.488	0.314	0.7
<i>Genetta genetta</i>	354	354	0.995	0.936	0.969	0.997	499	499	0.979	0.878	0.979	0.923
<i>Homo sapiens</i>	298	326	0.898	0.759	0.972	0.849	451	792	0.946	0.8	0.976	0.867
<i>Lepus europaeus</i>	397	397	0.786	0.65	0.619	0.912	289	305	0.864	0.756	0.794	0.873
<i>Martes sp.</i>	272	272	0.914	0.746	0.918	0.82	193	236	0.632	0.464	0.548	0.725
<i>Meles meles</i>	212	216	0.922	0.8	0.928	0.856	169	169	0.899	0.794	0.927	0.828
<i>Oryctolagus cuniculus</i>	707	726	0.903	0.678	0.988	0.698						
<i>Rupicapra pyrenaica</i>	417	596	0.843	0.758	0.89	0.747	618	1040	0.811	0.725	0.788	0.743
<i>Sciurus vulgaris</i>	260	261	0.884	0.716	0.939	0.77	61	61	0.818	0.609	0.712	0.82
<i>Sus scrofa</i>	620	744	0.899	0.761	0.888	0.852	776	1137	0.776	0.68	0.883	0.603
<i>Ursus arctos</i>	522	650	0.946	0.869	0.971	0.882	523	666	0.941	0.85	0.969	0.855
<i>Vehicle</i>	60	67	0.906	0.705	0.949	0.837	275	284	0.936	0.868	0.969	0.863
<i>Vulpes vulpes</i>	549	553	0.885	0.792	0.799	0.882	496	496	0.596	0.506	0.711	0.56
<i>All</i>			0.877	0.763	0.885	0.801			0.763	0.667	0.754	0.725

YOLOv9e	Validation						Test					
	Images	Instances	mAP@50	mAP@50-95	Precision	Recall	Images	Instances	mAP@50	mAP@50-95	Precision	Recall
<i>Bos taurus</i>	524	670	0.838	0.744	0.866	0.737	285	407	0.787	0.693	0.753	0.738
<i>Canis familiaris</i>	323	348	0.831	0.777	0.717	0.793	178	212	0.6	0.534	0.566	0.592
<i>Canis lupus</i>	445	565	0.815	0.726	0.887	0.669						
<i>Capreolus capreolus</i>	680	691	0.898	0.818	0.785	0.855	814	836	0.826	0.753	0.715	0.787
<i>Cervus elpahus</i>	678	759	0.853	0.785	0.805	0.769	551	873	0.806	0.723	0.875	0.638
<i>Dama dama</i>	498	593	0.853	0.785	0.847	0.771	674	938	0.683	0.589	0.836	0.499
<i>Equus ferus</i>	548	798	0.853	0.714	0.905	0.716	5	6	0.754	0.743	0.066	1
<i>Felis silvestris</i>	325	325	0.884	0.804	0.826	0.825	40	40	0.589	0.542	0.27	0.714
<i>Genetta genetta</i>	354	354	0.995	0.941	0.992	0.999	499	499	0.971	0.861	0.962	0.925
<i>Homo sapiens</i>	298	326	0.899	0.768	0.969	0.862	451	792	0.943	0.81	0.97	0.865
<i>Lepus europaeus</i>	397	397	0.787	0.651	0.666	0.907	289	305	0.888	0.776	0.841	0.866
<i>Martes sp.</i>	272	272	0.922	0.756	0.904	0.832	193	236	0.661	0.476	0.619	0.691
<i>Meles meles</i>	212	216	0.925	0.814	0.908	0.87	169	169	0.897	0.789	0.924	0.834
<i>Oryctolagus cuniculus</i>	707	726	0.903	0.687	0.974	0.779						
<i>Rupicapra pyrenaica</i>	417	596	0.842	0.754	0.895	0.721	618	1040	0.791	0.717	0.788	0.718
<i>Sciurus vulgaris</i>	260	261	0.926	0.758	0.959	0.806	61	61	0.774	0.614	0.729	0.77
<i>Sus scrofa</i>	620	744	0.917	0.779	0.911	0.851	776	1137	0.765	0.665	0.874	0.593
<i>Ursus arctos</i>	522	650	0.928	0.839	0.958	0.869	523	666	0.948	0.876	0.975	0.874
<i>Vehicle</i>	60	67	0.879	0.693	0.911	0.836	275	284	0.934	0.851	0.941	0.877
<i>Vulpes vulpes</i>	549	553	0.899	0.804	0.78	0.893	496	496	0.669	0.557	0.76	0.556
<i>All</i>			0.882	0.77	0.873	0.818			0.787	0.693	0.753	0.738

Benchmarking deep learning YOLO Object Detection models for Wildlife Detection in Camera Trap Imagery

YOLOv10x	Validation						Test					
	Images	Instances	mAP@50	mAP@50-95	Precision	Recall	Images	Instances	mAP@50	mAP@50-95	Precision	Recall
<i>Bos taurus</i>	524	670	0.835	0.753	0.859	0.698	285	407	0.59	0.504	0.881	0.291
<i>Canis familiaris</i>	323	348	0.815	0.746	0.774	0.741	178	212	0.5	0.438	0.537	0.486
<i>Canis lupus</i>	445	565	0.824	0.74	0.911	0.678						
<i>Capreolus capreolus</i>	680	691	0.883	0.807	0.748	0.867	814	836	0.819	0.748	0.705	0.773
<i>Cervus elpahus</i>	678	759	0.848	0.78	0.802	0.755	551	873	0.833	0.74	0.91	0.68
<i>Dama dama</i>	498	593	0.838	0.769	0.889	0.727	674	938	0.694	0.59	0.886	0.489
<i>Equus ferus</i>	548	798	0.849	0.722	0.905	0.715	5	6	0.283	0.282	0.045	1
<i>Felis silvestris</i>	325	325	0.868	0.795	0.78	0.831	40	40	0.568	0.522	0.263	0.825
<i>Genetta genetta</i>	354	354	0.994	0.938	0.962	0.994	499	499	0.978	0.869	0.947	0.938
<i>Homo sapiens</i>	298	326	0.888	0.769	0.95	0.828	451	792	0.929	0.787	0.969	0.838
<i>Lepus europaeus</i>	397	397	0.81	0.669	0.623	0.899	289	305	0.873	0.764	0.741	0.846
<i>Martes sp.</i>	272	272	0.91	0.742	0.941	0.766	193	236	0.735	0.522	0.678	0.716
<i>Meles meles</i>	212	216	0.91	0.791	0.902	0.852	169	169	0.875	0.756	0.874	0.799
<i>Oryctolagus cuniculus</i>	707	726	0.933	0.717	0.972	0.776						
<i>Rupicapra pyrenaica</i>	417	596	0.849	0.751	0.901	0.732	618	1040	0.797	0.715	0.791	0.756
<i>Sciurus vulgaris</i>	260	261	0.888	0.715	0.961	0.756	61	61	0.785	0.562	0.686	0.787
<i>Sus scrofa</i>	620	744	0.905	0.774	0.898	0.841	776	1137	0.769	0.667	0.925	0.572
<i>Ursus arctos</i>	522	650	0.934	0.852	0.965	0.846	523	666	0.905	0.807	0.954	0.782
<i>Vehicle</i>	60	67	0.834	0.696	0.83	0.761	275	284	0.935	0.87	0.936	0.873
<i>Vulpes vulpes</i>	549	553	0.911	0.818	0.808	0.877	496	496	0.57	0.488	0.663	0.504
<i>All</i>			0.876	0.767	0.869	0.767			0.746	0.646	0.744	0.72

YOLOv11x	Validation						Test					
	Images	Instances	mAP@50	mAP@50-95	Precision	Recall	Images	Instances	mAP@50	mAP@50-95	Precision	Recall
<i>Bos taurus</i>	524	670	0.834	0.74	0.872	0.681	285	407	0.507	0.422	0.846	0.175
<i>Canis familiaris</i>	323	348	0.849	0.775	0.785	0.758	178	212	0.432	0.371	0.42	0.486
<i>Canis lupus</i>	445	565	0.796	0.708	0.843	0.653						
<i>Capreolus capreolus</i>	680	691	0.865	0.788	0.785	0.805	814	836	0.769	0.7	0.776	0.702
<i>Cervus elpahus</i>	678	759	0.795	0.731	0.734	0.717	551	873	0.764	0.68	0.875	0.559
<i>Dama dama</i>	498	593	0.844	0.763	0.836	0.747	674	938	0.687	0.577	0.855	0.478
<i>Equus ferus</i>	548	798	0.839	0.704	0.906	0.702	5	6	0.149	0.149	0.034	1
<i>Felis silvestris</i>	325	325	0.849	0.766	0.815	0.788	40	40	0.528	0.493	0.276	0.75
<i>Genetta genetta</i>	354	354	0.994	0.94	0.98	0.997	499	499	0.975	0.873	0.972	0.922
<i>Homo sapiens</i>	298	326	0.895	0.767	0.969	0.847	451	792	0.94	0.793	0.977	0.807
<i>Lepus europaeus</i>	397	397	0.773	0.64	0.625	0.889	289	305	0.857	0.753	0.789	0.882
<i>Martes sp.</i>	272	272	0.899	0.733	0.926	0.737	193	236	0.576	0.409	0.567	0.66
<i>Meles meles</i>	212	216	0.911	0.793	0.876	0.829	169	169	0.838	0.729	0.903	0.718
<i>Oryctolagus cuniculus</i>	707	726	0.915	0.676	0.97	0.799						
<i>Rupicapra pyrenaica</i>	417	596	0.8	0.715	0.825	0.718	618	1040	0.787	0.698	0.807	0.723
<i>Sciurus vulgaris</i>	260	261	0.897	0.719	0.947	0.751	61	61	0.83	0.621	0.742	0.803
<i>Sus scrofa</i>	620	744	0.904	0.757	0.897	0.83	776	1137	0.761	0.661	0.919	0.531
<i>Ursus arctos</i>	522	650	0.944	0.864	0.966	0.869	523	666	0.91	0.81	0.981	0.788
<i>Vehicle</i>	60	67	0.926	0.73	0.938	0.836	275	284	0.923	0.855	0.958	0.842
<i>Vulpes vulpes</i>	549	553	0.861	0.77	0.727	0.866	496	496	0.521	0.436	0.624	0.522
<i>All</i>			0.87	0.754	0.861	0.791			0.709	0.613	0.74	0.686

Benchmarking deep learning YOLO Object Detection models for Wildlife Detection in Camera Trap Imagery

YOLOv12x	Validation						Test					
	Images	Instances	mAP@50	mAP@50-95	Precision	Recall	Images	Instances	mAP@50	mAP@50-95	Precision	Recall
<i>Bos taurus</i>	524	670	0.779	0.691	0.855	0.621	285	407	0.537	0.45	0.745	0.266
<i>Canis familiaris</i>	323	348	0.808	0.747	0.692	0.776	178	212	0.393	0.334	0.37	0.518
<i>Canis lupus</i>	445	565	0.83	0.75	0.891	0.68						
<i>Capreolus capreolus</i>	680	691	0.853	0.779	0.75	0.818	814	836	0.734	0.676	0.693	0.684
<i>Cervus elpahus</i>	678	759	0.828	0.761	0.787	0.755	551	873	0.78	0.688	0.863	0.615
<i>Dama dama</i>	498	593	0.831	0.758	0.851	0.731	674	938	0.688	0.587	0.786	0.555
<i>Equus ferus</i>	548	798	0.808	0.676	0.86	0.729	5	6	0.328	0.328	0.038	1
<i>Felis silvestris</i>	325	325	0.85	0.776	0.865	0.785	40	40	0.467	0.436	0.215	0.7
<i>Genetta genetta</i>	354	354	0.995	0.94	0.961	0.997	499	499	0.696	0.866	0.944	0.938
<i>Homo sapiens</i>	298	326	0.898	0.765	0.965	0.843	451	792	0.942	0.807	0.96	0.857
<i>Lepus europaeus</i>	397	397	0.798	0.665	0.59	0.889	289	305	0.796	0.703	0.727	0.852
<i>Martes sp.</i>	272	272	0.905	0.75	0.942	0.779	193	236	0.657	0.451	0.606	0.737
<i>Meles meles</i>	212	216	0.929	0.815	0.929	0.866	169	169	0.846	0.739	0.845	0.776
<i>Oryctolagus cuniculus</i>	707	726	0.881	0.656	0.962	0.692						
<i>Rupicapra pyrenaica</i>	417	596	0.757	0.674	0.81	0.695	618	1040	0.743	0.665	0.726	0.76
<i>Sciurus vulgaris</i>	260	261	0.878	0.72	0.86	0.77	61	61	0.79	0.595	0.629	0.82
<i>Sus scrofa</i>	620	744	0.881	0.749	0.866	0.834	776	1137	0.796	0.686	0.941	0.594
<i>Ursus arctos</i>	522	650	0.938	0.865	0.94	0.873	523	666	0.941	0.877	0.954	0.868
<i>Vehicle</i>	60	67	0.897	0.739	0.926	0.791	275	284	0.93	0.864	0.961	0.859
<i>Vulpes vulpes</i>	549	553	0.862	0.775	0.74	0.837	496	496	0.581	0.485	0.669	0.521
<i>All</i>			0.86	0.753	0.852	0.788			0.718	0.624	0.704	0.718

2.6.2 Normalized confusion matrix and performance matrix for each model on the validation set

2.6.2.1 YOLOv8x

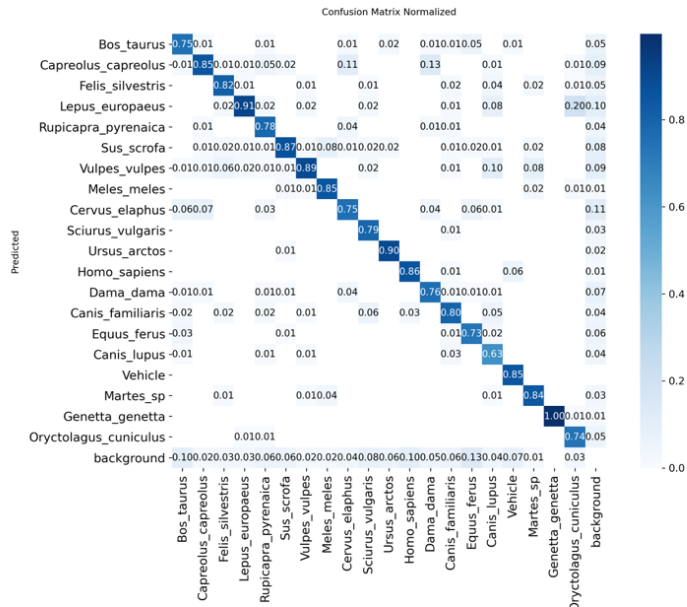


Figure 8. Normalized confusion matrix showing the recall of each class on the validation set for YOLOv8x model.

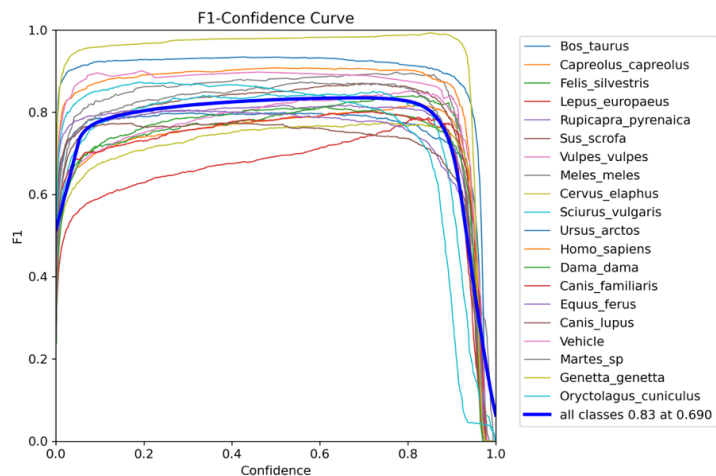


Figure 8. 1 F1-Confidence Curve for each class on the validation set for YOLOv8x model. The average indicates the best trade-off between confidence, recall and precision.

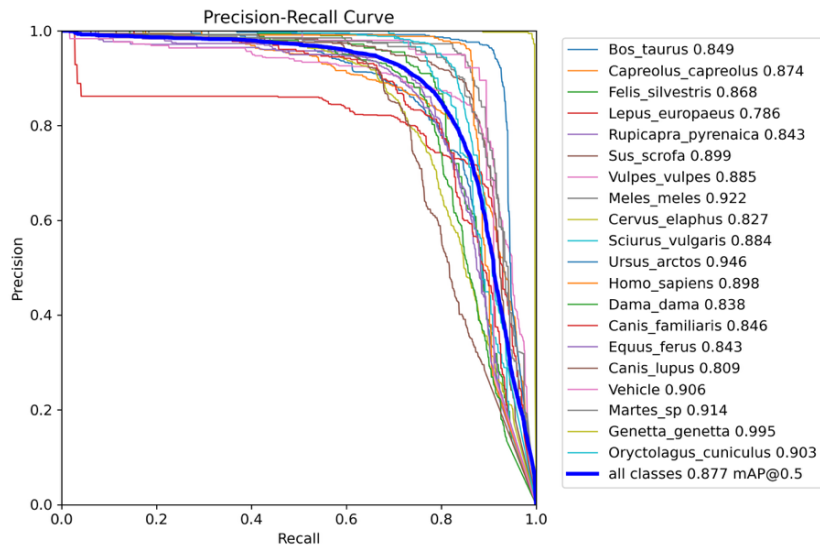


Figure 8.2 Precision-recall curve for each class on the validation set for YOLOv8x model.

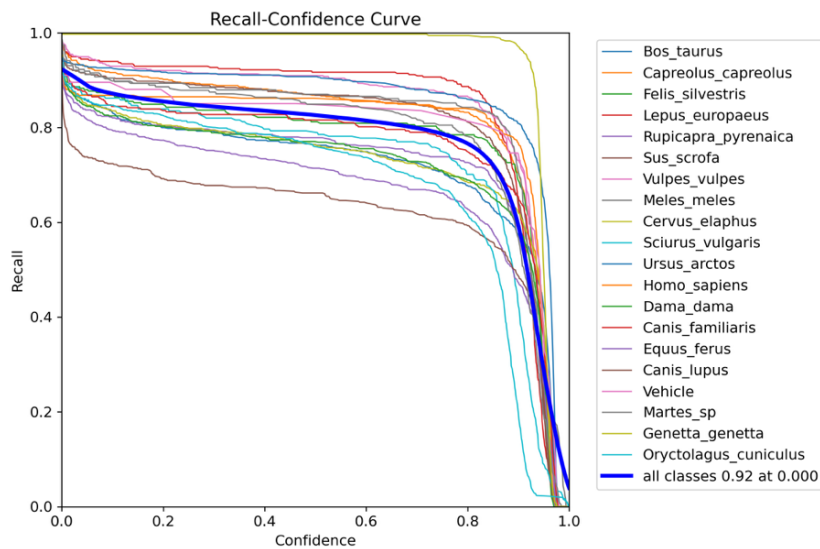


Figure 8.3 Recall-confidence curve for each class on the validation set for YOLOv8x model.

2.6.2.2 YOLOv9e

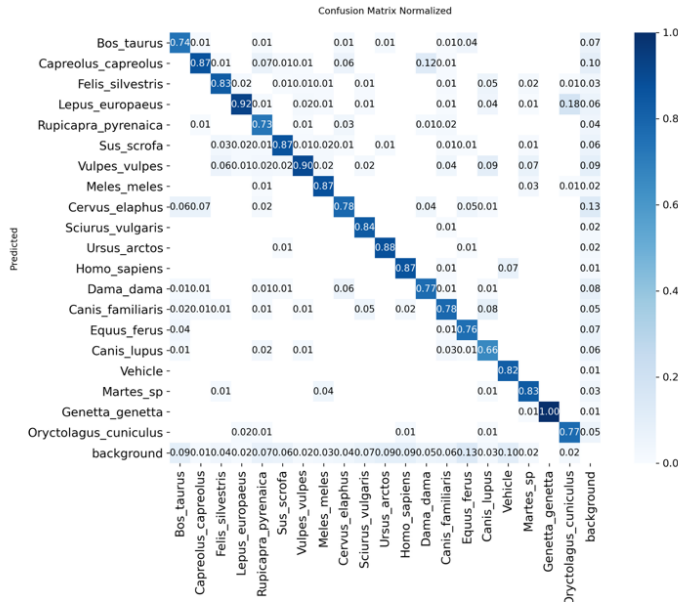


Figure 8. 4 Normalized confusion matrix showing the recall of each class on the validation set for YOLOv9e model.

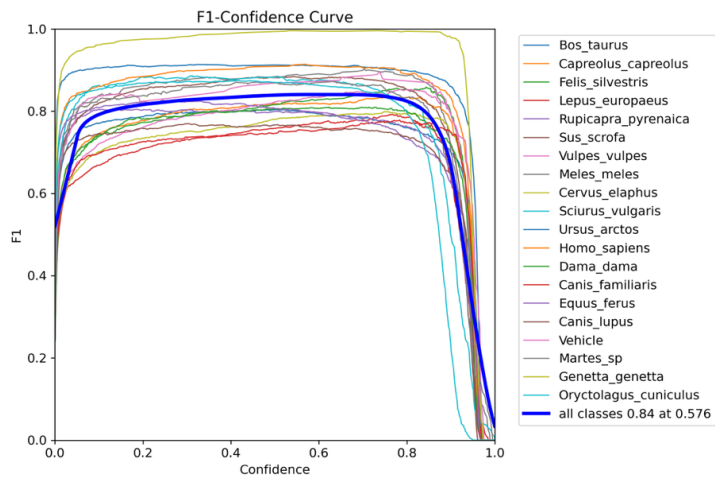


Figure 8. 5 F1-Confidence Curve for each class on the validation set for YOLOv9e model. The average indicates the best trade-off between confidence, recall and precision.

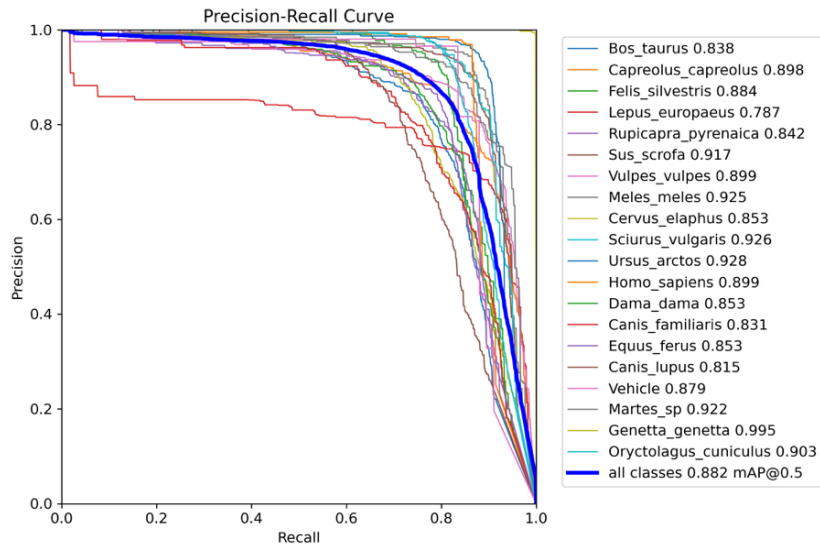


Figure 8. 6 Precision-recall curve for each class on the validation set for YOLOv9e model.

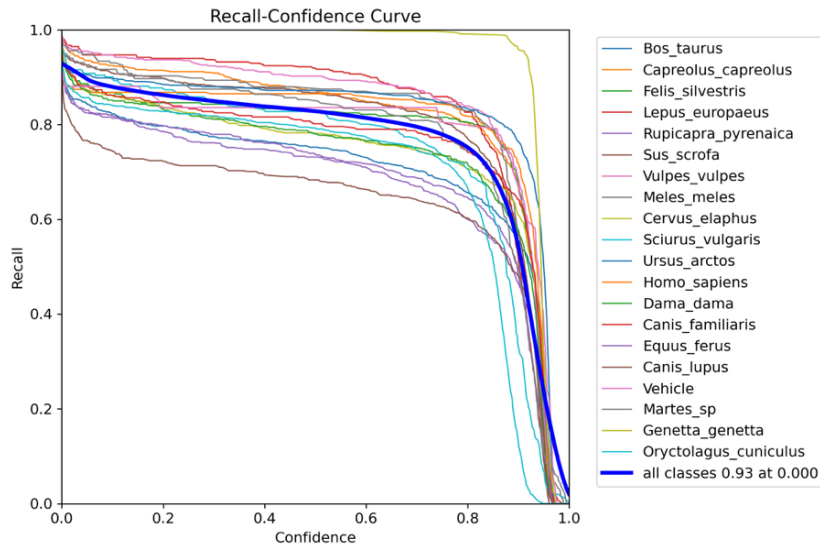


Figure 8. 7 Recall-confidence curve for each class on the validation set for YOLOv9e model.

2.6.2.3 YOLOv10x

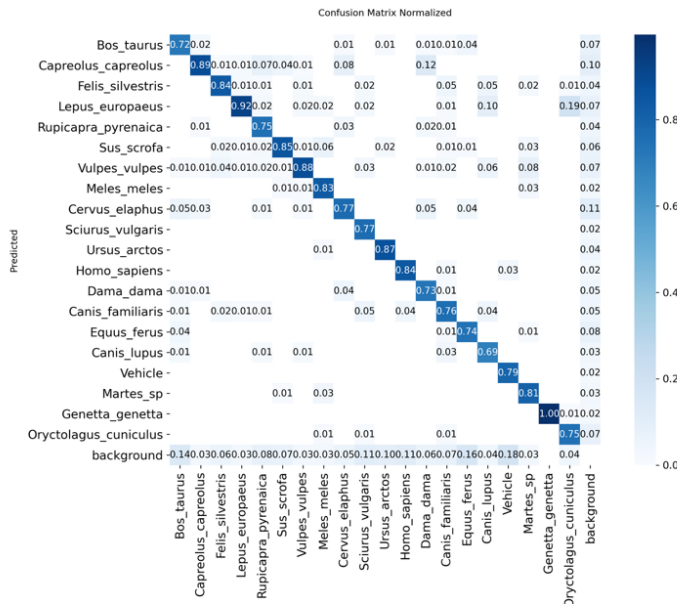


Figure 8. 8 Normalized confusion matrix showing the recall of each class on the validation set for YOLOv10x model.

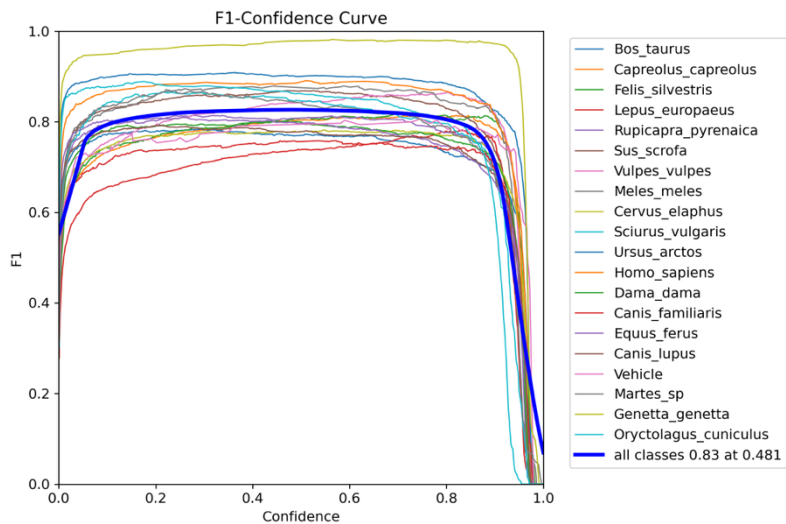


Figure 8. 9 F1-Confidence Curve for each class on the validation set for YOLOv10x model. The average indicates the best trade-off between confidence, recall and precision.

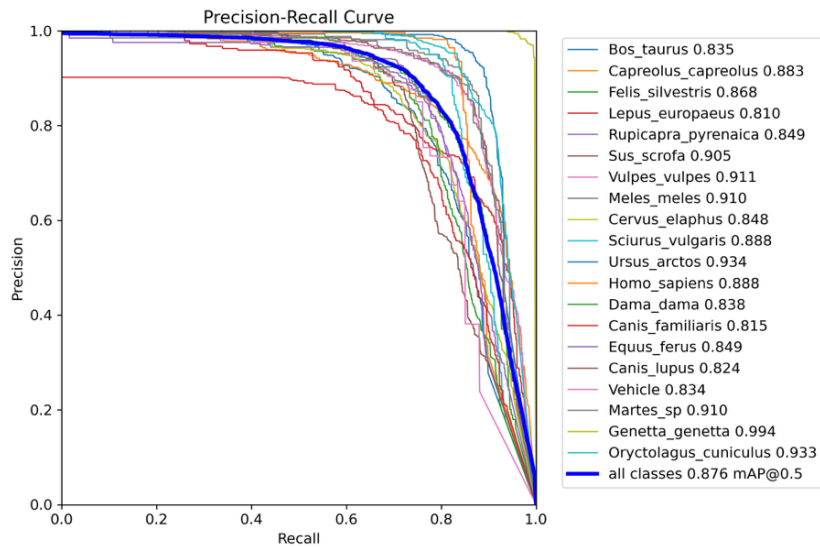


Figure 8. 10 Precision-recall curve for each class on the validation set for YOLOv10x model.

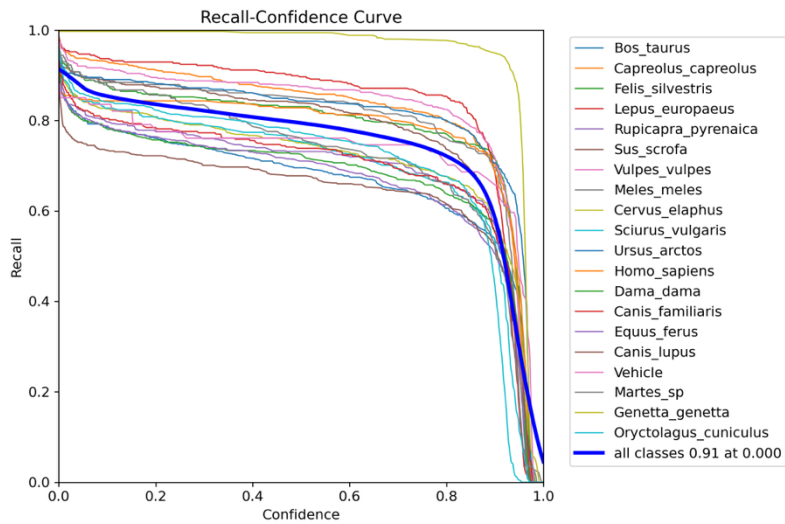


Figure 8. 11 Recall-confidence curve for each class on the validation set for YOLOv10x model.

2.6.2.4 YOLOv11x

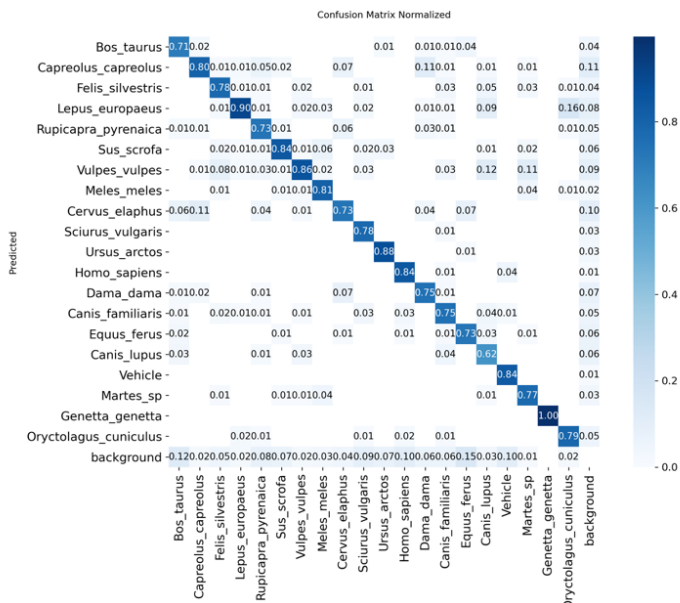


Figure 8. 12 Normalized confusion matrix showing the recall of each class on the validation set for YOLOv11x model.

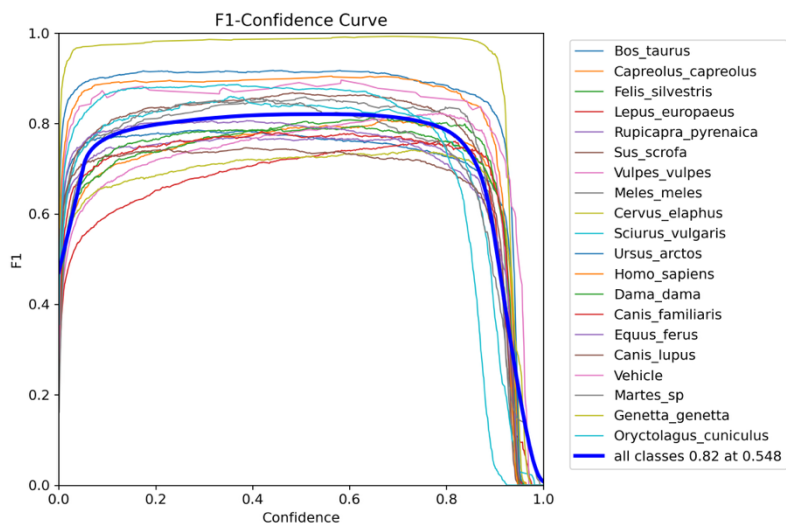


Figure 8. 13 F1-Confidence Curve for each class on the validation set for YOLOv11x model. The average indicates the best trade-off between confidence, recall and precision.

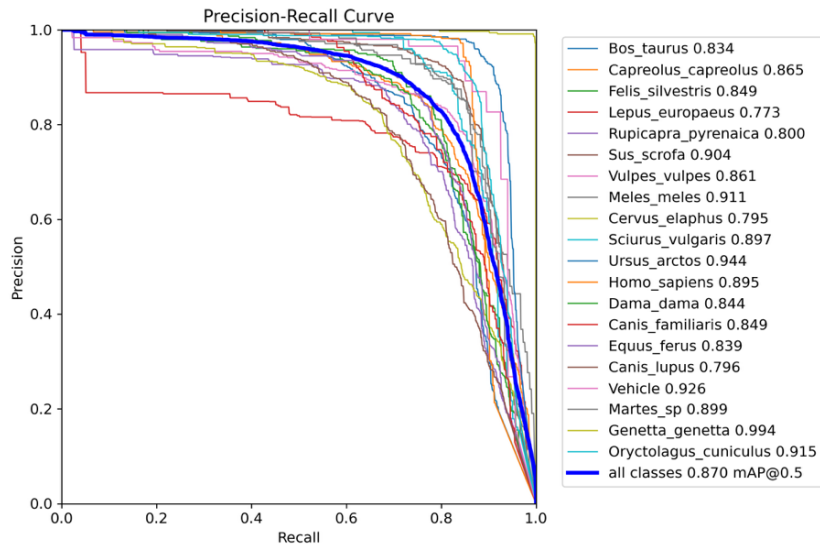


Figure 8. 14 Precision-recall curve for each class on the validation set for YOLOv11x model.

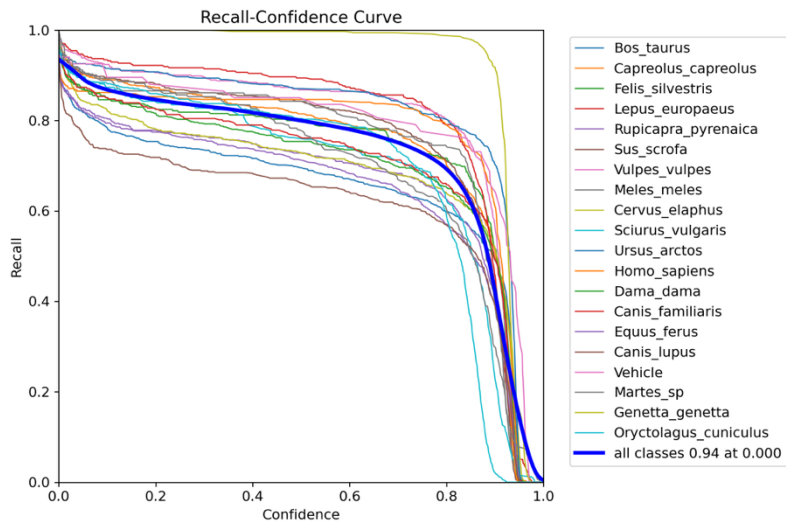


Figure 8. 15 Recall-confidence curve for each class on the validation set for YOLOv11x model.

2.6.2.5 YOLOv12x

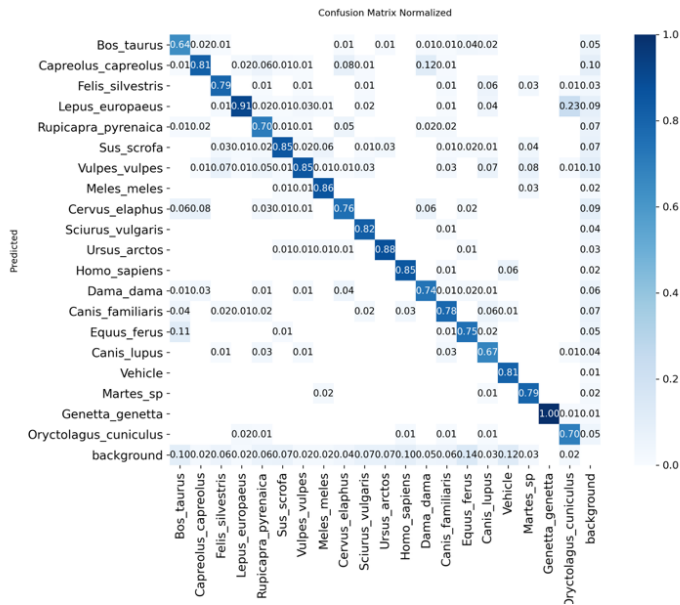


Figure 8. 16 Normalized confusion matrix showing the recall of each class on the validation set for YOLOv12x model.

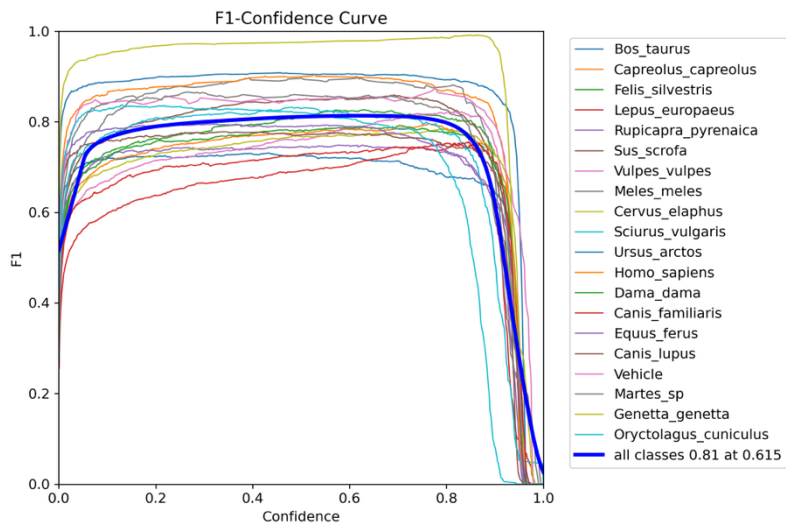


Figure 8. 17 F1-Confidence Curve for each class on the validation set for YOLOv12x model. The average indicates the best trade-off between confidence, recall and precision.

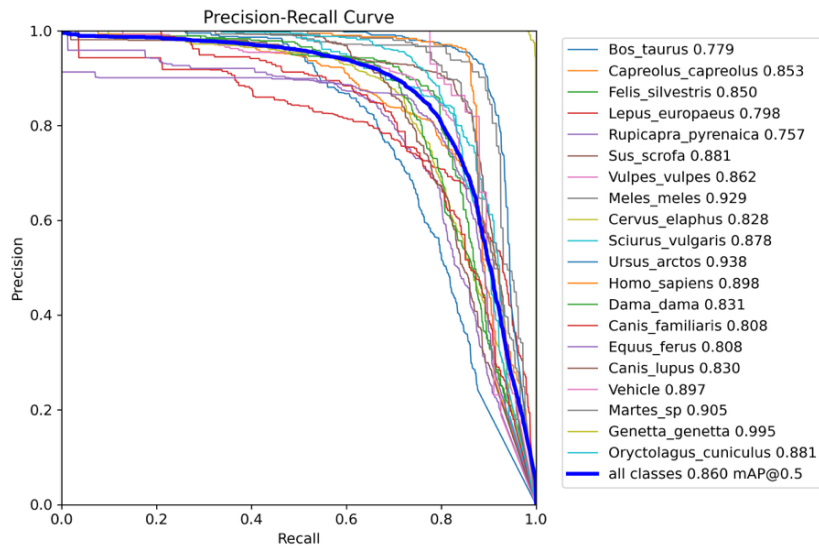


Figure 8. 18 Precision-recall curve for each class on the validation set for YOLOv12x model.

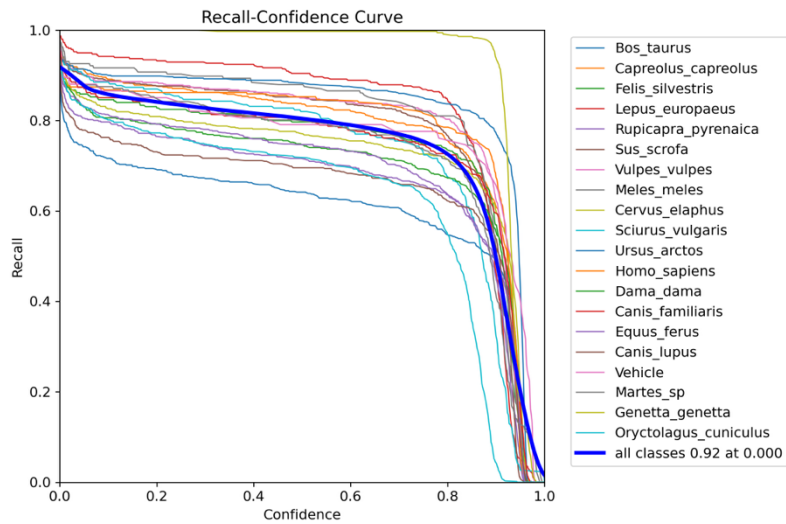


Figure 8. 19 Recall-confidence curve for each class on the validation set for YOLOv12x model.

2.6.3 Normalized confusion matrix and performance metrics for each model on the test set

2.6.3.1 YOLOv8x

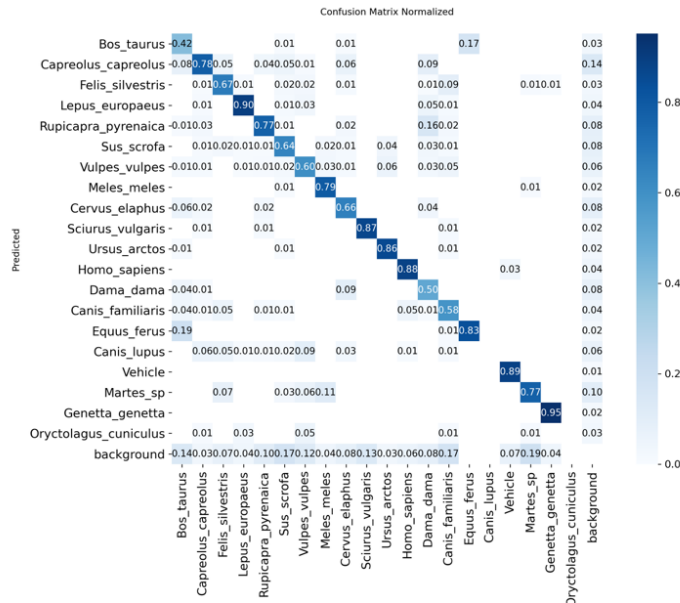


Figure 9. Normalized confusion matrix showing the recall of each class on the test set for YOLOv8x model.

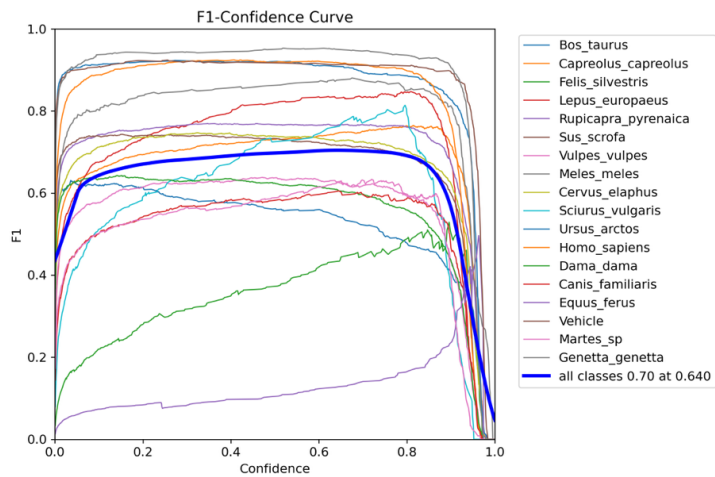


Figure 9. 1 F1-Confidence Curve for each class on the test set for YOLOv8x model. The average indicates the best trade-off between confidence, recall and precision.

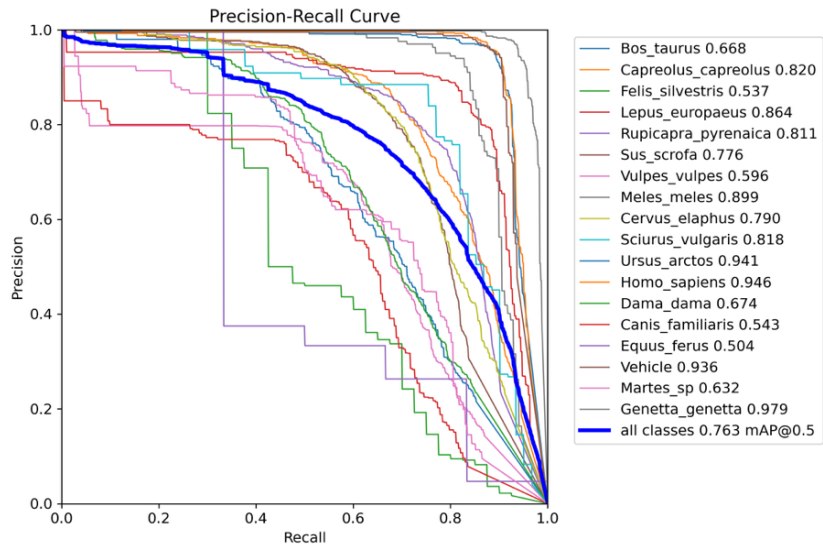


Figure 9.2 Precision-recall curve for each class on the test set for YOLOv8x model.

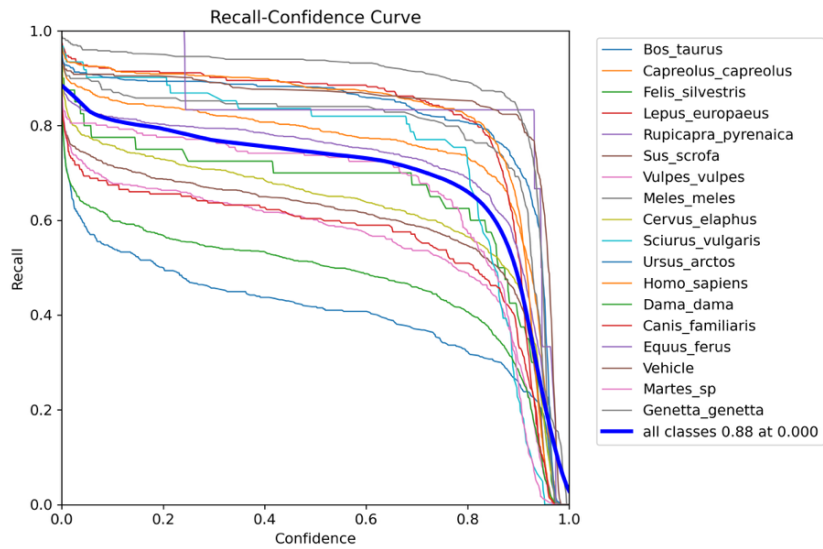


Figure 9.3 Recall-confidence curve for each class on the test set for YOLOv8x model.

2.6.3.2 YOLOv9e

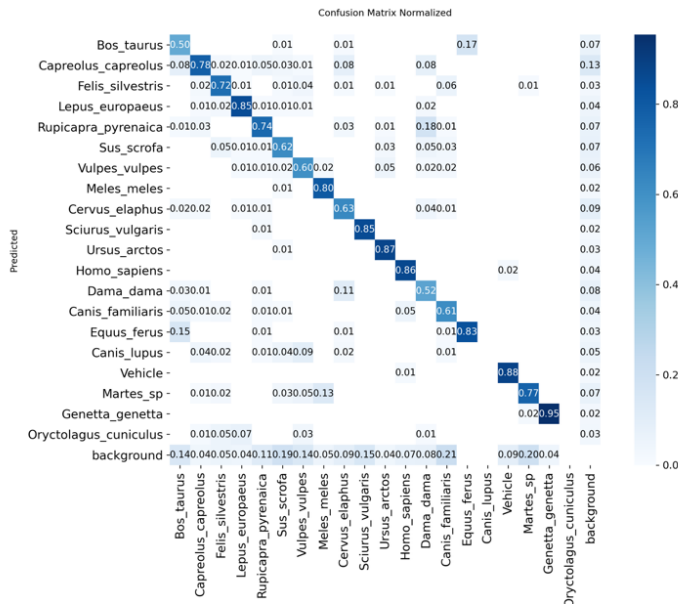


Figure 9. 4 Normalized confusion matrix showing the recall of each class on the test set for YOLOv9e model.

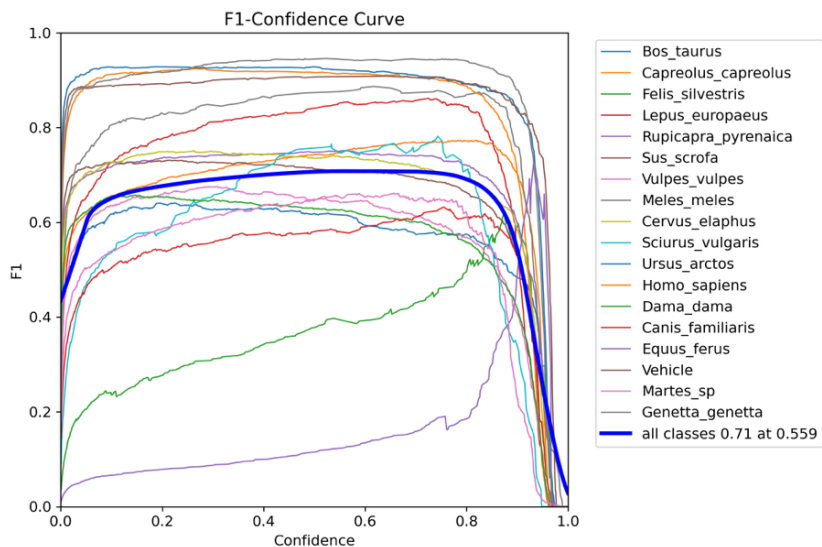


Figure 9. 5 F1-Confidence Curve for each class on the test set for YOLOv9e model. The average indicates the best trade-off between confidence, recall and precision.

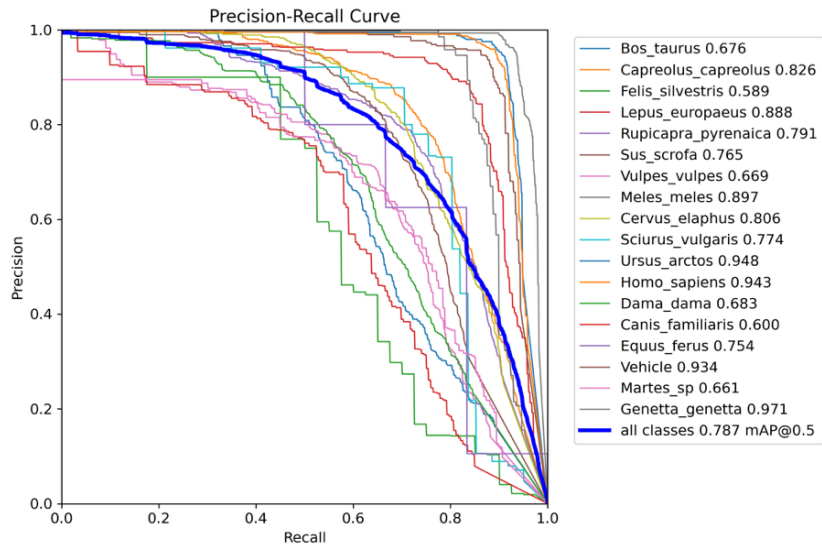


Figure 9. 6 Precision-recall curve for each class on the test set for YOLOv9e model.

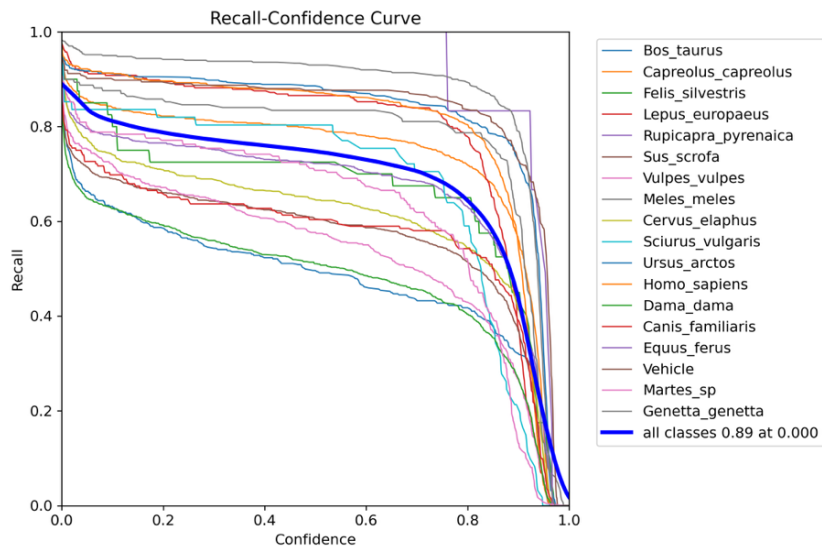


Figure 9. 7 Recall-confidence curve for each class on the test set for YOLOv9e model.

2.6.3.3 YOLOv10x

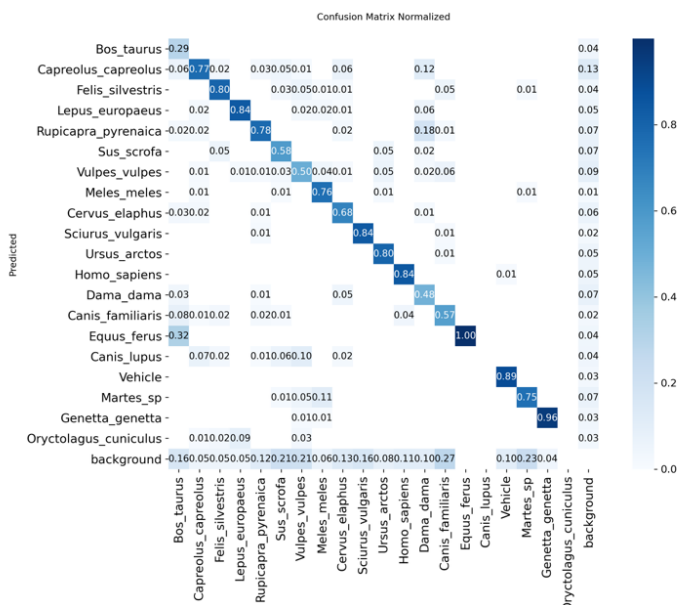


Figure 9. 8 Normalized confusion matrix showing the recall of each class on the test set for YOLOv10x model.

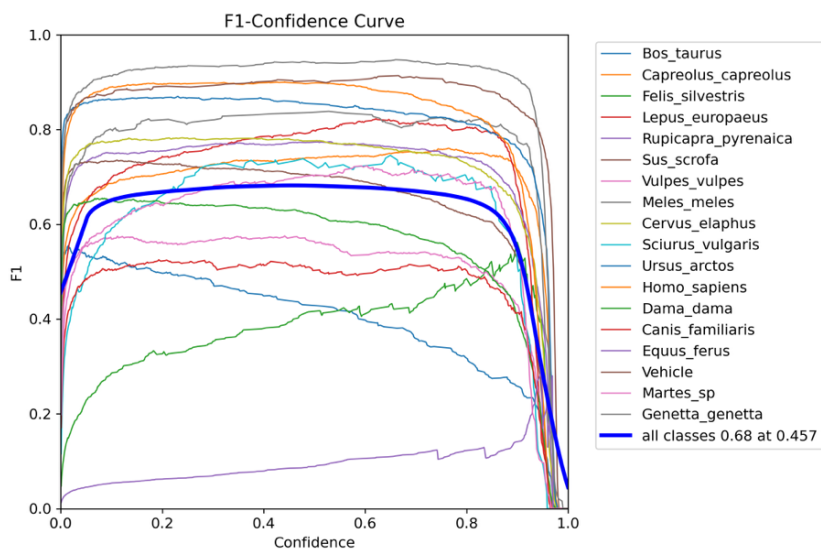


Figure 9. 9 F1-Confidence Curve for each class on the test set for YOLOv10x model. The average indicates the best trade-off between confidence, recall and precision.

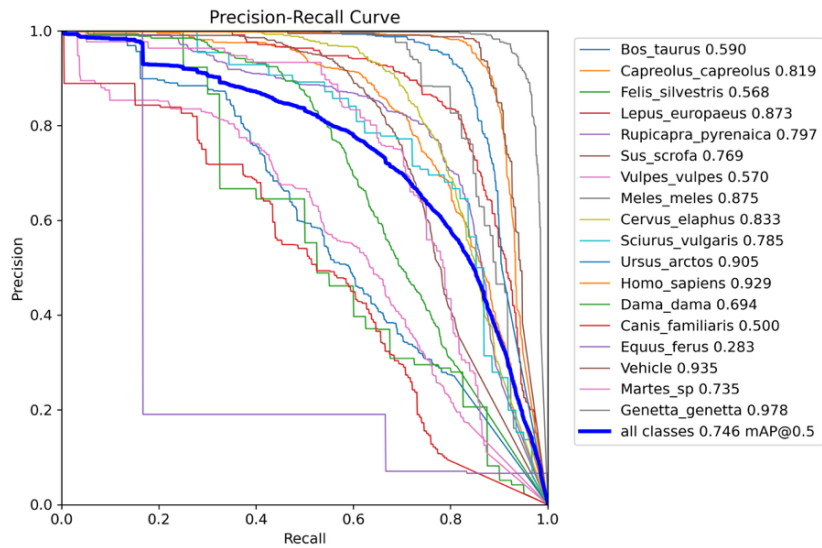


Figure 9. 10 Precision-recall curve for each class on the test set for YOLOv10x model.

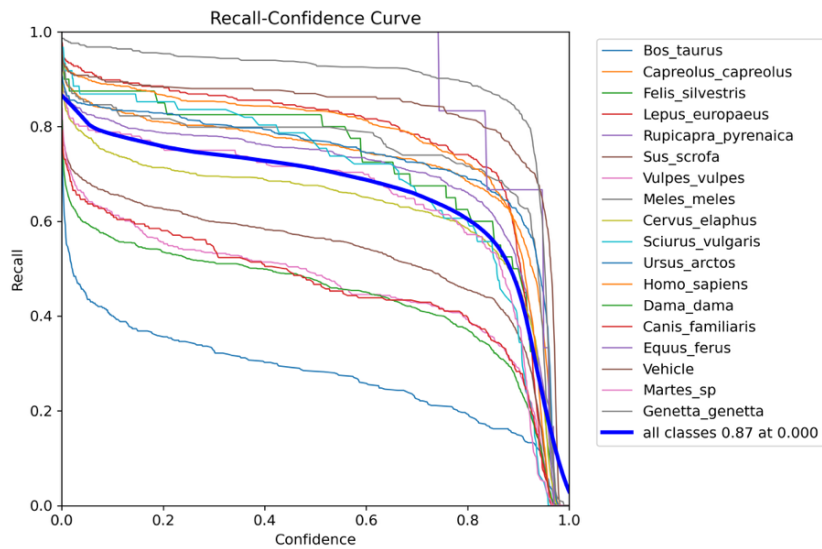


Figure 9. 11 Recall-confidence curve for each class on the test set for YOLOv10x model.

2.6.3.4 YOLOv11x

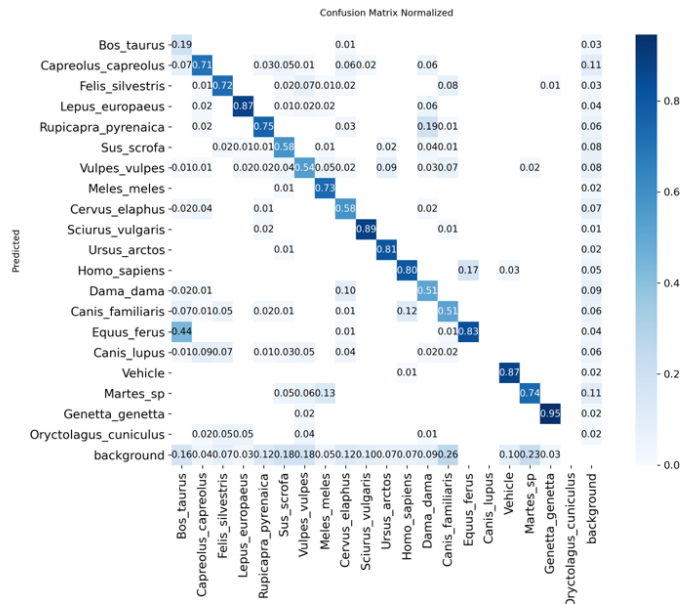


Figure 9. 12 Normalized confusion matrix showing the recall of each class on the test set for YOLOv11x model.

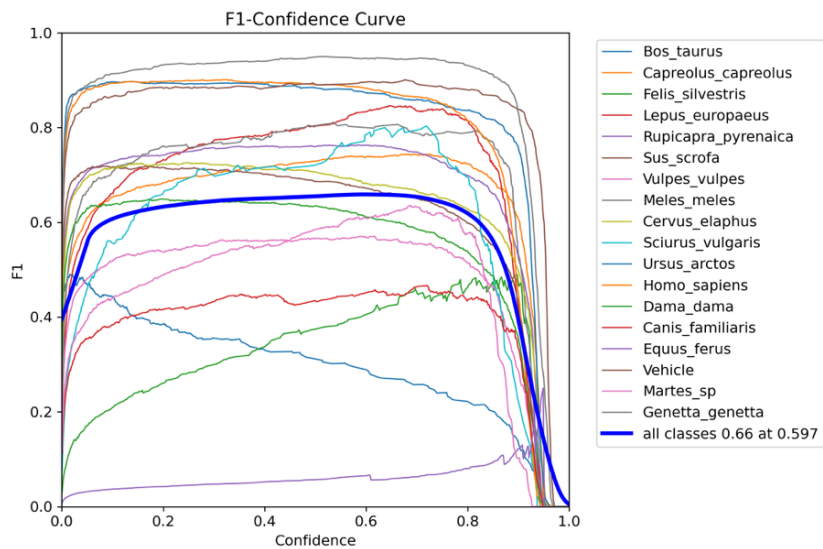


Figure 9. 13 F1-Confidence Curve for each class on the test set for YOLOv11x model. The average indicates the best trade-off between confidence, recall and precision.

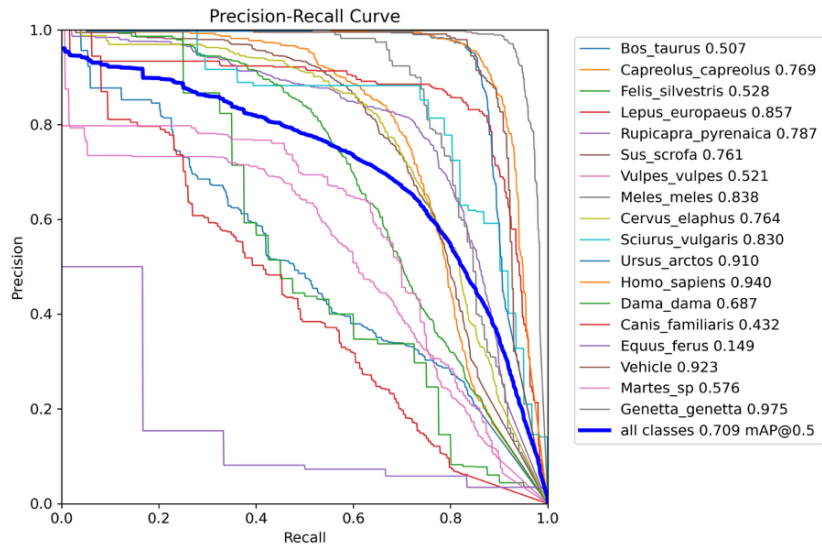


Figure 9. 14 Precision-recall curve for each class on the test set for YOLOv11x model.

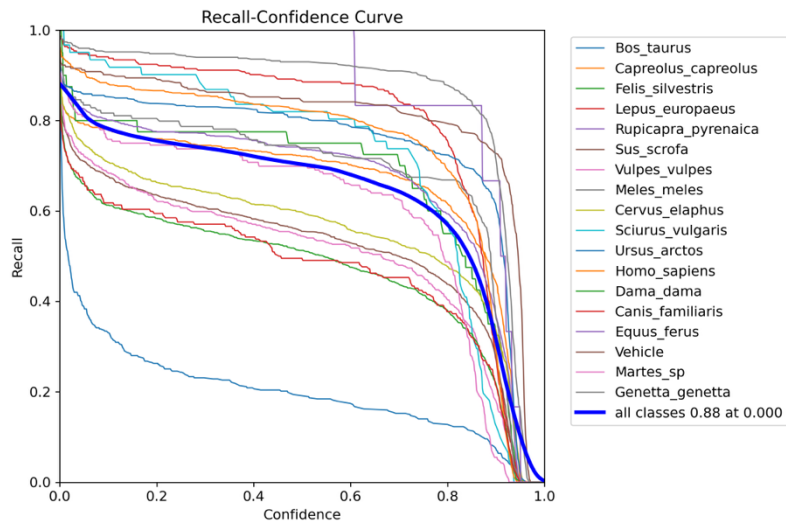


Figure 9. 15 Recall-confidence curve for each class on the test set for YOLOv11x model.

2.6.3.5 YOLOv12x

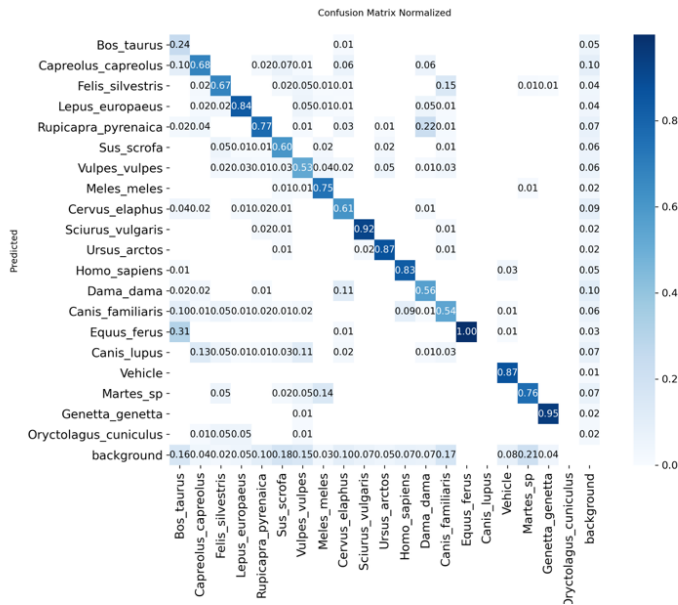


Figure 9. 16 Normalized confusion matrix showing the recall of each class on the test set for YOLOv12x model.

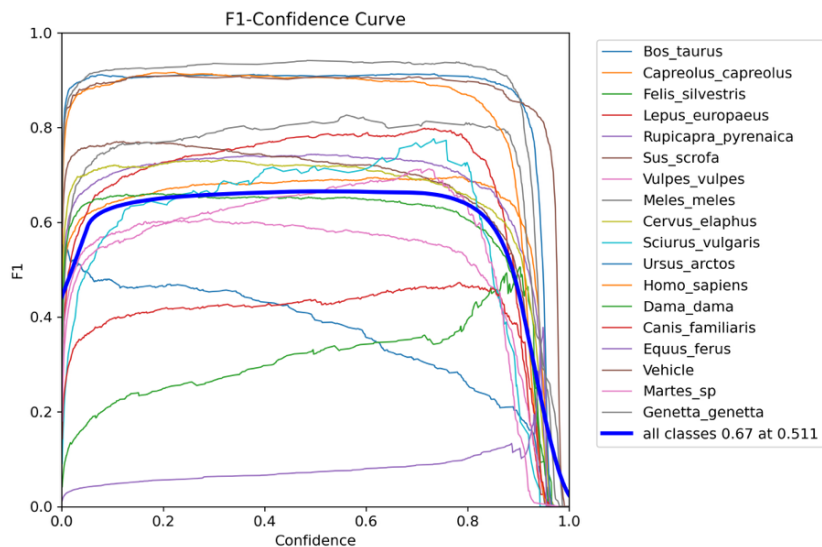


Figure 9. 17 F1-Confidence Curve for each class on the test set for YOLOv11x model. The average indicates the best trade-off between confidence, recall and precision.

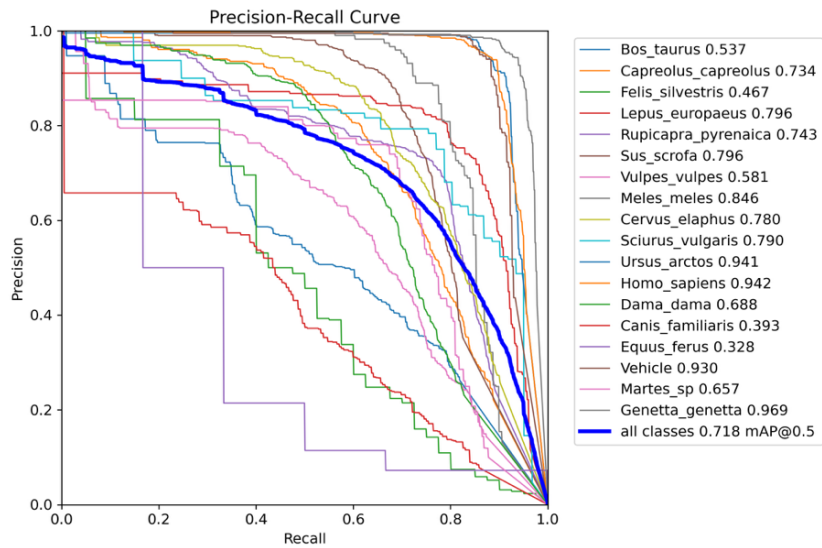


Figure 9. 18 Precision-recall curve for each class on the test set for YOLOv12x model.

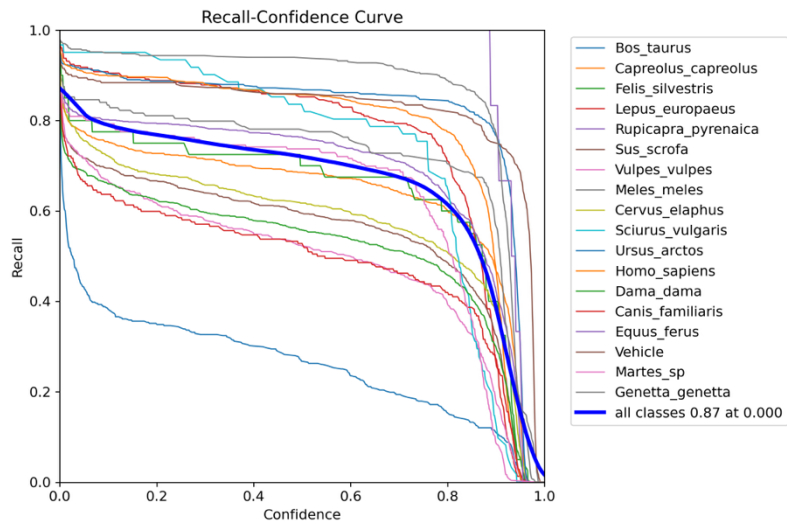
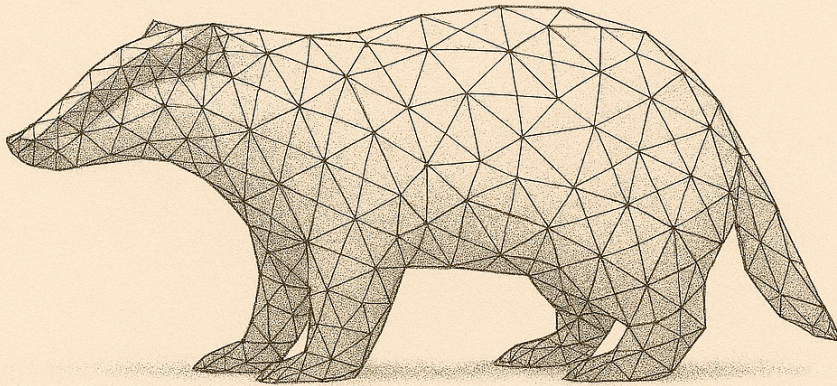


Figure 9. 19 Recall-confidence curve for each class on the test set for YOLOv12x model.

CHAPTER 3



3 Enhancing Wildlife Detection in Small Camera Trap Datasets using YOLOv9e

Arnau Campanera-Moliné, Daniel Ponsa, Víctor Sazatornil, Felipe Lumbreras,
Robert Benavente, Lluís Brotons

ABSTRACT

Computer vision algorithms applied to wildlife monitoring have significantly improved their performance in the last decade. Yet, a key limitation of many computer vision models is their ability to generalize to unseen data, critical for practical applications. High performing models trained on millions of images do not show the same performance on custom datasets. Some practical strategies can enhance model performance without requiring additional data collection or annotation, such as hyperparameter tuning. Adapting various models by training them in a specific visual domain, such as RGB and grayscale has already improved model performance in some studies.

In this study, hyperparameter tuning gains were evaluated in a small dataset context. Furthermore, three separate models were trained on three variants of the same dataset based on mixed, color and grayscale palettes to explore model domain specificity in camera trap datasets. Additionally, the mixed dataset was halved to account for data quantity. Optimizing hyperparameters enhanced model performance across all datasets by 11%, 19% and 10% for mixed, color and grayscale models respectively. Moreover, the model trained on the mixed dataset showed a superior performance over the two models trained on the color and grayscale datasets. Finally, the halved mixed dataset showed similar performance than the whole mixed dataset, only varying by 1%, suggesting that data diversity might be more influential, especially when contributing with feature richness data, than strict domain specialization for object detection tasks.

3.1 Introduction

Since their first use in 1890, camera traps have become an essential tool for wildlife monitoring. They have transformed how scientists estimate species presence, density and abundance (Howe et al., 2017), while causing minimal disturbance and enabling long-term monitoring, even in remote areas. The increasing affordability of modern camera traps now allows for extensive, scalable monitoring while also generating massive volumes of raw imagery. Managing and analyzing this influx of data has become a growing challenge in the ecological community, who are increasingly calling for automated solutions (Hahn et al., 2022).

In parallel, computer vision methods have significantly improved their performance in the past two decades (Perry et al., 2022). This rapid progress have been possible due to the availability of large, annotated datasets (Deng et al., 2009; Lin et al., 2014) and breakthroughs in deep learning. In ecology, deep convolutional neural networks were first applied to camera trap images over a decade ago (G. Chen et al., 2014a). Since then, classification accuracy has steadily improved (Gomez et al., 2016; Norouzzadeh et al., 2018), in some cases even surpassing human expertise in image classification (Van Horn et al., 2018) and audio recognition (Kahl et al., 2021).

Despite the growing need for automated processing of raw camera-trap data, most available solutions still rely on technical codebases, specialized terminology, and complex performance metrics. User-friendly applications remain scarce, often requiring ecologists to engage with computer science methodologies that fall outside their area of expertise. This mismatch limits the adoption and practical applicability of automated computer vision methods in ecological research (Hahn et al., 2022). Moreover, the rapid pace of artificial intelligence development and the challenges of establishing effective interdisciplinary collaborations with machine learning specialists constitute substantial barriers for many practitioners. Consequently, a large proportion of ecologists continue to rely on traditional data-processing approaches (Stephenson, 2020).

A key limitation of many computer vision models is their restricted ability to generalize to unseen data. Deep learning systems typically perform best when training

and prediction conditions are closely aligned (Rigoudy et al., 2023). A common challenge arises when models must make predictions outside the feature space encountered during training (Beery et al., 2018). Many widely used benchmark datasets evaluate models on tests sets drawn from the same visual domain as the training sets (Deng et al., 2009; Everingham et al., 2010; Kuznetsova et al., 2020; Lin et al., 2014). As a consequence, models that perform exceptionally well on large-scale benchmarks often do not maintain comparable performance to custom or previously unseen ecological imagery. Although generalization is a critical requirement for practical applications, it remains a persistent weakness in recognition algorithms (Beery et al., 2018).

Several practical strategies can improve model performance without requiring additional data collection or annotation. One common approach is hyperparameter optimization, whereby key training parameters are systematically tuned. These parameters influence the learning dynamics of machine learning algorithm and can substantially affect predictive accuracy (Bischl et al., 2021). Identifying optimal hyperparameters for a given dataset can lead to significant performance improvements (Feurer & Hutter, 2019).

Another theoretical means of improving performance is to train models on narrower, domain-specific subsets of data. Differences in spectral properties between daytime and nighttime images, for example, may justify training separate models for distinct visual domain. Kristo et al. showed in their experiments that training separate models for RGB and thermal images yielded much better results for human identification in difficult weather conditions. In general, domain specialization can simplify the learning task and lead to better performance within similar contexts, compared to training a single model spanning diverse visual conditions. In the context of species-detection models, such domains might include previously seen backgrounds (coniferous forests, subalpine zone, etc.), similar prominent objects (tree stumps, logs, rocks, etc.) and resembling environmental conditions (rain, snow, shadow casting, etc.). For the present study, this primarily involves specializing models for daytime and nighttime imagery.

Light conditions constitute a major source of variation in camera-trap images, typically manifesting as differences in palette: ‘color’ (RGB) versus ‘grayscale’.

Recognition algorithms extract different features depending on the information stored in available channels. Grayscale images contain a single channel of information (or three identical channels), limiting them to shape and texture cues, whereas RGB images provide three independent channels, enabling models to learn a richer set of discriminative features. In addition, grayscale images often result from low-light conditions, where long exposure times can introduce motion blur. This distortion appears when the image being taken changes during the capturing process as a result of the low shutter speed needed to capture enough light and the rapid movement of the animal. For instance, nocturnal images of the European hare (*Lepus europaeus*) may often appear elongated or smeared. This appearance greatly differs from colored images, where hares are typically displayed clear and proportionated.

The hypothesis of this study is that training domain-specific models for each palette will enable each network to more effectively learn class-specific features under consistent visual conditions.

Accordingly, the aim of this study is twofold: (1) to experimentally evaluate the performance gains attributable to hyperparameter tuning in a small-dataset context, and (2) to assess the effects of domain specialization by training two separate models based on image palette, and comparing their performance to that of a single, generalized model trained on a mixed dataset.

3.2 Materials and Methods

3.2.1 Dataset

For this study, the dataset assembled in Chapter 1 was used.

3.2.1.1 Main Dataset Composition and Characteristics

The original dataset comprised 48,863 images, captured between 2017 to 2024. These included 19 mammal species and “vehicles”: cow (*Bos taurus*), roe deer (*Capreolus capreolus*), wild cat (*Felis silvestris*), European hare (*Lepus europaeus*), Pyrenean chamois (*Rupicapra pyrenaica*), wild boar (*Sus scrofa*), red fox (*Vulpes vulpes*), European badger (*Meles meles*), red deer (*Cervus elaphus*), red squirrel (*Sciurus vulgaris*), bear (*Ursus arctos*), fallow deer (*Dama dama*), horse (*Equus ferus*), dog (*Canis familiaris*), wolf (*Canis lupus*), human (*Homo sapiens*), the Martes genus (*Martes sp.*), common genet (*Genetta genetta*), rabbit (*Oryctolagus cuniculus*) and the class vehicle.

Image resolutions ranged from 0.5 to 36 megapixels, with dimensions between 965 x 576 and 8000 x 4500 pixels. Most images were stored using mainly JPG format, with a minority in PNG. The dataset was partitioned at camera-station level, where all images from a given station were assigned entirely to either training, validation or test subsets, to preventing data leakage. Within each station, images were subdivided by species.

- Training subset: 83 stations, 33,390 images, 36,892 animal instances
- Validation subset: 52 stations, 9,075 images, 9,911 instances
- Test subset: 30 stations, 6,837 images, 8,997 instances.

The training and validation subsets followed an approximate ~79/21% split proportion. No ‘wolf’, ‘rabbit’ or ‘horse’ instances were included in the test set due to lack of data. The dataset also contained background, which accounted for a total of 5.6% of overall images.

Images had been previously annotated by wildlife technicians. Annotations consisted of pixel-tight bounding boxes drawn around all the individuals present in an image with their corresponding class label, as for any object detection dataset.

3.2.2 Dataset Partitioning by Image Palette

To investigate domain-specific model performance, the main dataset was further split into two new datasets based on image palette: *grayscale* (or nighttime) and *color* (or daytime).

Due to the image imbalance of some classes, and the different circadian habits of the species within the *mixed* dataset, a naïve random split into training and validation in the new partitions was avoided. Instead, the following actions were taken:

- Camera stations were kept as intact blocks and assigned exclusively to one subset to prevent data leakage.
- Within each station, images already grouped by species were further separated according to palette and assigned to their corresponding new dataset.
- A thorough attempt to maintain the camera stations' distribution from the original dataset was made. However, some stations required reassignment to achieve an approximate 80-20% split ratio between training and validation subsets.

Similarly, classes that had very few images in the new partitions were excluded, as their low number of instances would not have allowed the model to properly generalize (Figure 10):

- **Grayscale dataset:** the red squirrel (*Sciurus vulgaris*), the human (*Homo sapiens*), the dogs (*Canis familiaris*), the horse (*Equus ferus*) and the vehicle.
- **Color dataset:** the badger (*Meles meles*), the Martes genus (*Martes sp.*) and the common genet (*Genetta genetta*).

To ensure that comparison across datasets were not confounded by differing data volumes, a reduced mixed dataset was generated by halving the number of images (Table 7). This allowed the analysis to isolate whether performance differences were

attributable to dataset size rather than data composition. Halving was performed using a simple Python script that removed every second file. This method was preferred over random removal to preserve class proportions between both datasets (Figure 10).

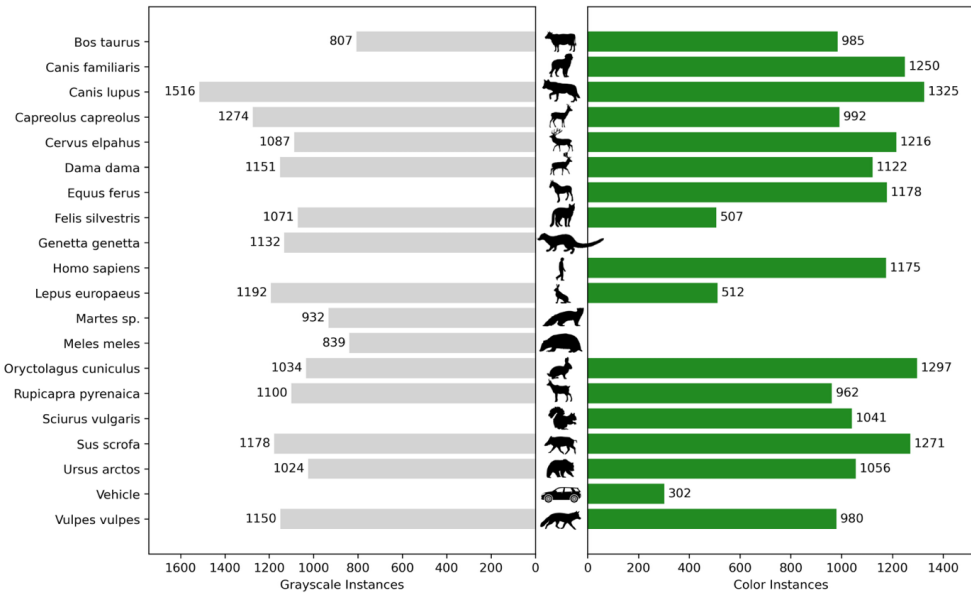


Figure 10. Instances for each newly created partition.

Table 7. Summary of the number of images in each partition and subset.

Datasets		Images			Classes
		Training	Validation	Test	
Mixed	Whole	33,390	9,075	6,837	20
	Halved	16,695	4,537		
Grayscale		16,005	4,080	3,256	16
Color		15,555	4,401	3,581	17

3.2.3 Model architecture: YOLOv9e overview

For this study, the YOLOv9e (C.-Y. Wang et al., 2024) was selected based on the benchmark results presented in Chapter 1, where it outperformed other evaluated variants, including YOLOv8, YOLOv10, YOLOv11 and YOLOv12.

3.2.3.1 Model initialization

In Chapter 1, YOLOv9 was fine-tuned using weights pre-trained on the COCO 2017 Dataset. Because COCO consists exclusively of RGB images, initializing the grayscale model with RGB-domain weights could introduce a domain mismatch. The feature representations learned from color images, particularly color-based edges, textures, and chromatic patterns, are less transferable to grayscale imagery, whose feature space differs substantially.

To address this limitation, a YOLOv9e model was additionally trained from scratch using a modified COCO 2017 Dataset. All original RGB images were converted to grayscale, thereby aligning the pre-training domain with the palette characteristics of the target dataset.

3.2.4 Hyperparameter tuning

Hyperparameters -also often called metaparameters- govern the training dynamics of machine-learning models and can be tuned to optimize performance for a specific dataset. Identifying the optimal values of these meta-parameters requires exploring a multidimensional search space containing all candidate values.

For this study, hyperparameter optimization was conducted using the tuning module integrated into the Ultralytics YOLO framework. This module searches the hyperparameter space using a genetic algorithm, a mechanism inspired by natural selection and gene mutation. This method applies small, random modifications to each hyperparameter value for each iteration, simulating random mutation. Depending on the performance of the training process for each new set of generated hyperparameters, the values are either retained (“survive”) or are mutated again, guiding the search towards high-performing regions of the hyperparameter space.

To track the training performance of the model on each iteration, a fitness function is defined. This fitness function consists of a weighted combination of metrics mAP@50 (10%) and mAP@50-95 (90%), described in section 3.2.7.

To reduce computational demands while preserving class representation, tuning was performed on a 20% stratified subset of each partition (mixed, grayscale, and color). Hyperparameter tuning was conducted independently for each partition.

- For the mixed and color partitions, the tuning processes was initialized with the pretrained weights on the default RGB COCO 2017 Dataset
- For the grayscale partition, the tuning process was initialized from the custom pre-trained COCO 2017 grayscale dataset.

The following hyperparameters were researched for the best combination and performance; initial learning rate (lr0), final learning rate (lrf), SGD momentum (momentum), optimizer weight decay (weight_decay), warmup epochs (warmup_epochs), warmup initial momentum (warmup_momentum), box loss gain (box), class loss gain (cls), distribution focal loss (dfl), pose loss gain (pose), keypoint loss gain (kobj), image HSV-Hue augmentation (hsv_h), image HSV-Saturation augmentation (hsv_s), image HSV-Value augmentation (hsv_v), image translation (translation), image scale (scale), image flip left-right (fliplr) and image mosaic (mosaic).

3.2.5 Model Training and Configuration

3.2.5.1 Training with Optimized Hyperparameters

Upon completion of the three hyperparameter tuning processes, YOLOv9e was trained on each dataset using the respective optimized hyperparameter configurations. For the halved mixed dataset, the hyperparameters obtained from the full mixed dataset were reused to maintain comparability.

Models were initialized with the standard pre-trained weights provided by Ultralytics, trained on the Common Objects in Context (COCO) dataset (Lin et al., 2014). This dataset contains approximately ~118,000 images and ~886,000 annotated, everyday objects across 80 different classes. Although COCO's domain differs substantially

from camera-trap imagery, transfer learning offers two main advantages: (1) reduced computational costs, and (2) accelerated convergence during training, particularly when working with limited data. Pre-trained models are already capable of detecting low-level image features, such as edges, contours, and textures, which can be recomposed into higher-level representations more efficiently than when training from scratch.

To ensure fair comparison across datasets and models, the input image resolution and batch size were kept constant for all training runs. Early stopping was applied with a patience of 100 epochs. This form of regularization prevents overfitting by halting the training if the model shows no improvement on the validation set over a defined number of consecutive epochs. This approach enables the model to generalize more effectively to unseen data.

All training and hyperparameter-tuning processes were conducted on a Supermicro GPU server equipped with 768 GB of RAM and four NVidia L40S GPUs, totaling 184 GB of VRAM. The same software environment was kept across experiments, with CUDA version 12.5 and Ultralytics version 8.3.193.

3.2.5.2 Data augmentation

No custom, user-defined augmentations were applied. Instead, the built-in augmentation mechanisms of the Ultralytics YOLO framework were used. These augmentations expand the effective training set by introducing controlled variations in the input images, improving model robustness and reducing overfitting. The following augmentation types were applied:

- Color space adjustments: hue (hsv_h), saturation (hsv_s), and brightness (hsv_v).
- Geometric transformations: translation (translate) and scaling (scale).
- Mosaic augmentation (mosaic): combining four images into one.
- Flipping images horizontally (fliplr). Vertical flipping was excluded due to its impracticality in real-world scenarios.

3.2.6 Performance evaluation

Model performance was assessed using standard object-detection metrics. In object detection models, two main tasks are considered: the precision of the placement of the predicted bounding box, and the correctness of the assigned class to that box. To quantify these concepts several metrics are computed.

3.2.6.1 Intersection over Union (IoU)

A popular metric to quantify the ability of a model to precisely place a bounding box around an object is *Intersection over Union* (IoU). This metric evaluates the overlap between the ground truth annotation, and the model predicted box. An IoU of 1 indicates a perfect overlap, whereas 0 indicates no overlapping of the boxes. IoU thresholds are used to determine whether a predicted bounding box is considered a correct detection.

3.2.6.2 Classification Metrics: Precision, Recall, and F1 Score

Another crucial required ability is to correctly predict the class of the detected object. To evaluate this, precision, recall and F1-score metrics are computed:

- **Precision:** measures out of all the detections, how many were correct. In other words, the ratio between true positives and all predicted positives.
- **Recall:** measures how many labels were correctly predicted out of all the ground truth labels. Also in other words, the ratio between true positives and all the positives that should have been predicted.
- **F1 score:** It combines precision and recall metrics into a single value balancing the trade-off between both. F1 score represents the harmonic mean between these two metrics, and it is usually given at different confidence values.

3.2.6.3 Mean Average Precision (mAP)

Two metrics can combine in a single score these two abilities; how precise is the placement of the inferred bounding box and the correctness of the predicted class of that box. These two metrics are mAP (mean Average Precision);

- **mAP@50**: Setting an IoU threshold of 0.5 for bounding box placement, the average precision obtained for each class is computed, and the overall mean is calculated.
- **mAP@50-95**: Similar to mAP@50, but uses 10 different thresholds, from 0.5 to 0.95 in 0.05 steps. Then it averages them all in a single score.

While mAP@50 tends to produce higher scores, it may not show model performance very accurately, as it tolerates substantial localization error. In contrast, mAP@50-95 is significantly stricter and measures model quality more realistically. This is why it is widely used as a model performance comparison metric. For this reason, mAP@50-95 was used to compare between model performance across the mixed, grayscale, and color partitions.

3.3 Results

3.3.1 Hyperparameter tuning

For the *mixed* dataset, the hyperparameter tuning process ran for 700 iterations, with each iteration training for 50 epochs. For the *color* and *grayscale* datasets, convergence occurred earlier, requiring only 500 iterations. This difference suggests that domain-specific datasets may converge more quickly than those trained on mixed data, potentially due to reduced variability in visual characteristics (Figure 11).

The fitness values for each partition are shown on Table 8. The optimized hyperparameter configurations are detailed on Table 9.

Table 8. Fitness results of the hyperparameter tuning process.

Dataset	Initial fitness	Best fitness	Best fitness iteration
Mixed	0.3950	0.7023	670
Grayscale	0.3695	0.6237	433
Color	0.1757	0.6413	400

Table 9. Initial and optimized values for all hyperparameters research during the tuning process.

Hyperparameter		Mixed	Grayscale	Color
	Default	Optimized	Optimized	Optimized
lr0	0.01	0.0001	1.7e-04	3.0e-05
lrf	0.01	0.0047	0.0109	0.0036
momentum	0.937	0.9232	0.8948	0.7057
weight decay	0.0005	0.0002	4.0e-05	0.0005
warmup epochs	3.0	4.6886	5.0	3.2814
warmup momentum	0.8	0.6441	0.5816	0.6419
box	7.5	6.3386	10.9992	2.3458
cls	0.5	0.4869	0.9085	0.5053
df1	1.5	0.6042	0.8918	2.3169
hsv_h	0.015	0.0726	0.0087	0.0176
hsv_s	0.7	0.3126	0.4687	0.6821
hsv_v	0.4	0.9	0.4749	0.2371
translate	0.1	0.0444	0.1378	0.0769
scale	0.5	0.6202	0.6820	0.6727
fliplr	0.5	0.8882	0.3982	0.4662
mosaic	1	0.7646	0.7685	0.2162

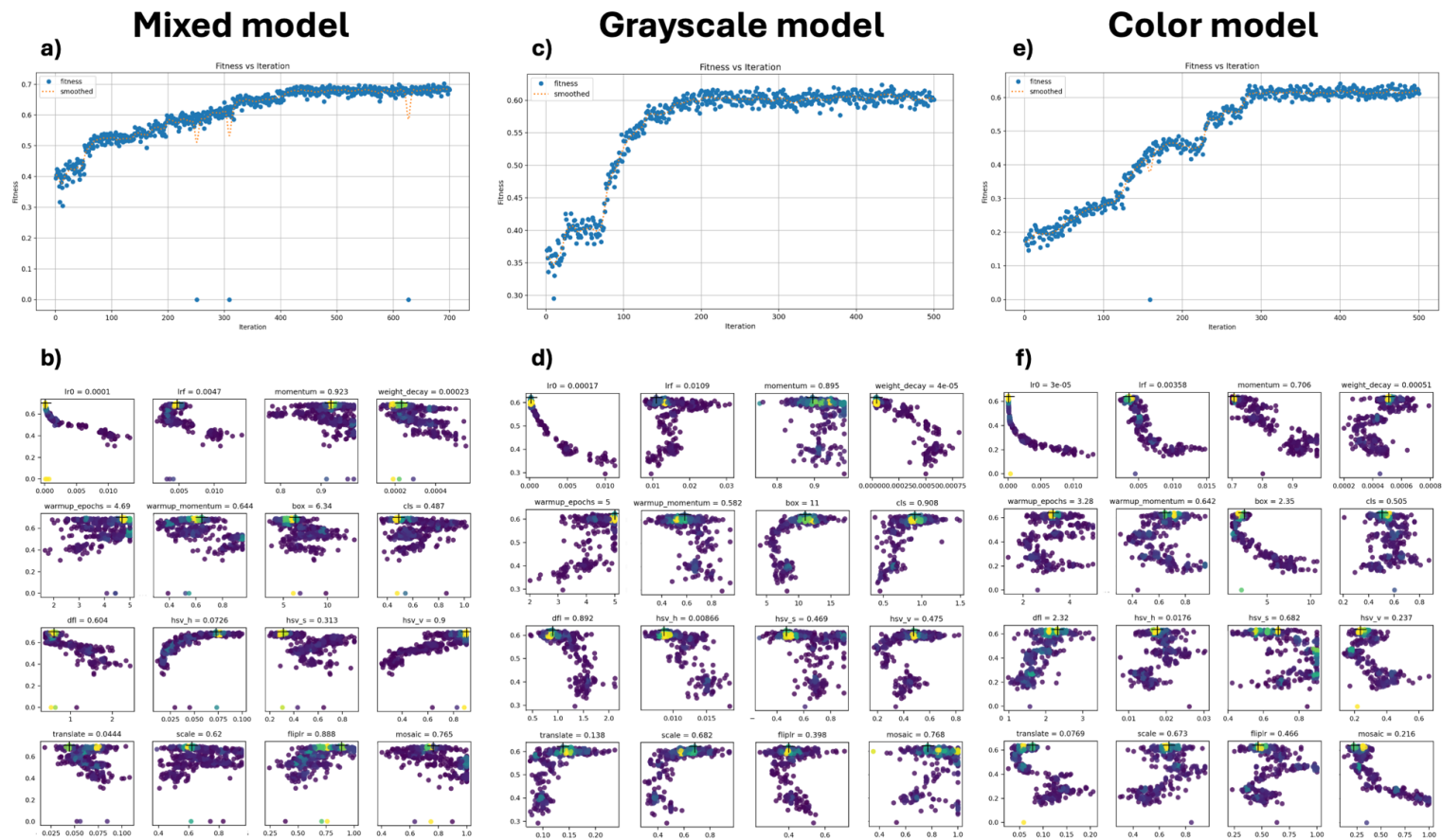


Figure 11. Results from the hyperparameter tuning process performed on the three dataset variants. Fitness evolution vs iteration are shown on a), c) and e), while hyperparameter searched space and their corresponding fitness values for each model variant are shown on b), d) and f).

3.3.1.1 Training performance improvement

Training the models with the optimized hyperparameters led to substantial performance improvements across all three datasets, as summarized in Table 10 (evaluated on the corresponding test subsets).

Table 10. Comparison between metrics obtained before and after hyperparameter tuning.

Metrics	Mixed (whole)		Grayscale		Color	
	Default	Optimized	Default	Optimized	Default	Optimized
mAP@50	0.787	0.86	0.77	0.817	0.727	0.835
mAP@50-95	0.693	0.769	0.67	0.719	0.639	0.761
Precision	0.753	0.844	0.778	0.826	0.74	0.81
Recall	0.738	0.77	0.698	0.73	0.662	0.786
F1-score @ confidence	0.71 @ 0.56	0.78 @ 0.67	0.71 @ 0.53	0.75 @ 0.49	0.67 @ 0.67	0.78 @ 0.62

- **Mixed partition:**

mAP@50-95 increased from 0.693 to 0.769 (11% relative improvement), while precision improved by 12%.

- **Grayscale dataset:**

mAP@50-95 increased from 0.64 to 0.703 (10% relative improvement), while recall improved by 6%.

- **Color dataset:**

mAP@50-95 increased from 0.639 to 0.761, representing the largest relative gain (19%) among the three datasets.

These results indicate that tuning hyperparameters consistently and significantly enhances detection accuracy, with the strongest gains observed for the color dataset.

3.3.2 Data Partitioning Performance Comparison

To evaluate the impact of domain-specific partitioning, models trained on the mixed, grayscale and color datasets were tested on their respective dedicated test subsets. As described in the Methods section, a reduced mixed dataset containing half the image count of the mixed dataset was created to ensure a fairer comparison by controlling for differences in dataset size. This approach allowed to assess whether performance variations arose primarily from data size or domain-specific data quality.

The two mixed dataset variations (full and halved) were evaluated twice: on the grayscale subset, followed by the color subset. Results are shown on Table 11.

Table 11. Per-class recall by model. The classes ‘wolf’ (Canis lupus) and ‘rabbit’ (Oryctolagus cuniculus) were not present in the test set. Similarly, class ‘horse’ (Equus ferus) only had 6 instances in the test set, which made their results not relevant. Blue cells indicate best score on grayscale subset, while green indicates best score on color subset.

Class	Mixed (whole)		Mixed (halved)		Grayscale model	Color model
	Grayscale	Color	Grayscale	Color		
<i>Bos taurus</i>	0.63	0.69	0.61	0.62	0.50	0.71
<i>Canis familiaris</i>		0.69		0.63		0.65
<i>Canis lupus</i>						
<i>Capreolus capreolus</i>	0.77	0.89	0.74	0.89	0.65	0.86
<i>Cervus elpahus</i>	0.80	0.76	0.74	0.72	0.69	0.79
<i>Dama dama</i>	0.47	0.84	0.58	0.81	0.60	0.78
<i>Equus ferus</i>						
<i>Felis silvestris</i>	0.96	0.67	0.96	0.73	0.96	0.67
<i>Genetta genetta</i>	0.94		0.93		0.94	
<i>Homo sapiens</i>		0.88		0.90		0.94
<i>Lepus europaeus</i>	0.90	0.82	0.86	0.83	0.81	0.62
<i>Martes sp.</i>	0.77		0.70		0.78	
<i>Meles meles</i>	0.85		0.80		0.86	
<i>Oryctolagus cuniculus</i>						
<i>Rupicapra pyrenaica</i>	0.75	0.82	0.71	0.82	0.61	0.79
<i>Sciurus vulgaris</i>		0.85		0.77		0.82
<i>Sus scrofa</i>	0.66	0.68	0.72	0.68	0.61	0.65
<i>Ursus arctos</i>	0.94	0.84	0.94	0.93	0.90	0.92
<i>Vehicle</i>		0.90		0.89		0.90
<i>Vulpes vulpes</i>	0.55	0.79	0.57	0.74	0.55	0.79
<i>Average</i>	0.77	0.79	0.77	0.78	0.73	0.78

The two models trained on the mixed dataset, the *whole* and *halved* variants, demonstrated superior performance compared to the two models trained on the palette-specific partitions. When evaluated on grayscale images, the mixed models outperformed the specific *grayscale* model by achieving better scores on 8 out of 13 classes. Similarly, on color images, the *mixed* models achieved higher scores in 9 out of 14, and matched the color-specific model in 2 additional classes. The color specific model showed better performance in only 3 out of 14 classes.

Notably, the halved mixed-dataset model also outperformed overall the grayscale-specific model on grayscale images and achieved comparable performance to the color-specific model on color images.

Finally, performance differences between the *whole* and *halved* mixed models were minimal, differing by only 1%, indicating that reducing the dataset size by half did not substantially impact overall performance.

3.4 Discussion

This study assessed whether model performance could be improved through hyperparameter tuning and domain-specific dataset partitioning based on image palette.

3.4.1 Hyperparameter tuning

Optimizing hyperparameter values led to clear and consistent improvements across all evaluated models, enhancing performance at multiple IoU thresholds and strengthening overall robustness.

- The **color model** benefited the most, with nearly double the improvement compared to the grayscale model, which showed the smallest relative increase.
- The **mixed model** achieved its highest F1-score (0.78 @ 0.67), indicating that the model could afford to be more selective while not losing recall.
- In contrast, both **grayscale** and **color** models improved their F1 scores but reached their optimal values at lower confidence thresholds, suggesting that these models needed to be more permissive in their detections to maintain an adequate balance between precision and recall.

3.4.2 Data partitioning

3.4.2.1 Data quantity

Partitioning the original dataset into separate grayscale and color subsets to increase domain specificity, did not yield superior model performance. This finding may initially appear counterintuitive. Domain specialization is often associated with improved generalization, especially when training and inference data share similar visual characteristics (Beery et al., 2018). Moreover, the palette-specific datasets displayed more balanced class distributions than the mixed datasets, a factor which, according to prior work, should support improved performance (Van Horn & Perona, 2017).

A straightforward explanation could be that the mixed model had access to a larger training set (approximately twice the number of images), thereby benefiting from greater data diversity. However, this hypothesis does not fully explain the results, as the halved mixed model, trained on only half the data, achieved nearly identical performance to the full mixed model, with only a 1% difference, suggesting that data quantity alone was not the driving factor.

More revealing is the comparison between the whole mixed and palette-specific models:

- The **color model** performed nearly as well as the whole mixed model on color images, despite having half the training data. This suggests that adding grayscale images to the mixed dataset did not substantially improve the model's ability to recognize color images.
- Conversely, the mixed model's performance on grayscale images exceeded that of the grayscale-specific model by an overall 5%, indicating that the inclusion of color images did help the model generalize better to grayscale contexts.

A plausible explanation that data augmentation methods, particularly transformations involving hue, saturation or brightness, may have altered RGB images in ways that approximated grayscale characteristics. These augmented variations could have enriched the mixed model's representation of low-light or low-contrast features, thereby improving its performance on the grayscale test (Figure 12).

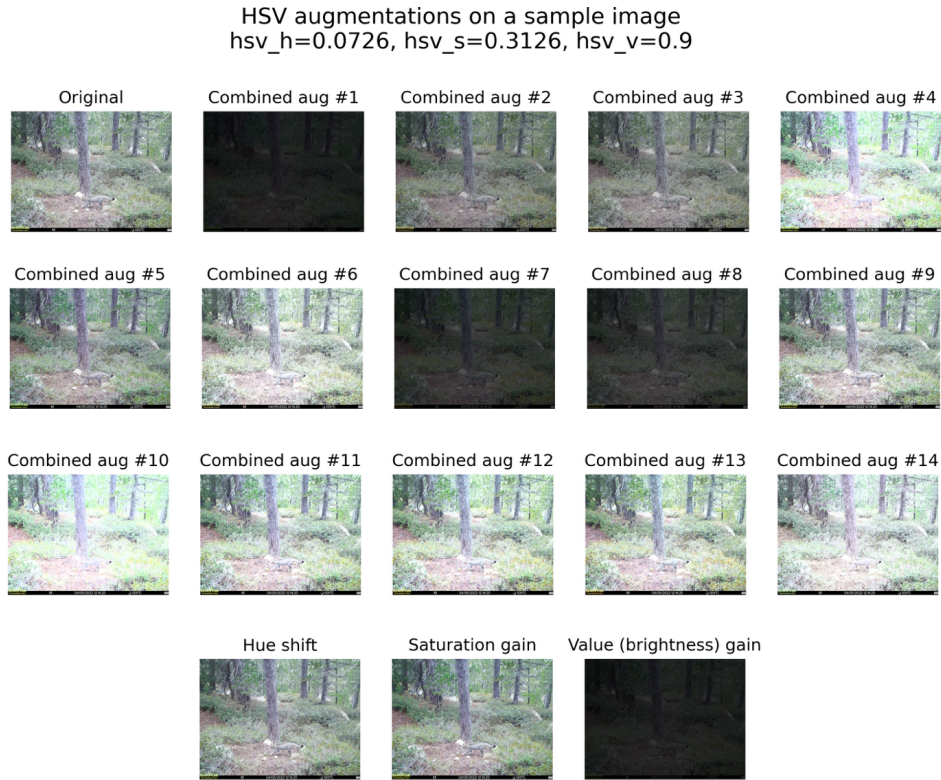


Figure 12. Random HSV augmentations performed on a sample RGB image. In examples #1, #7 and #8, random modifications of the brightness rendered an RGB image to a grayscale-like aspect.

It is important to note that the grayscale model still achieved top performance in **five classes**. Three of these corresponded to species represented exclusively by grayscale images (*Martes sp.*, *Meles meles* and *Genetta genetta*), for which a domain-specific model is expected to perform well. For a fourth species (*Felis silvestris*) the grayscale model matched the performance of the whole mixed model. The fifth species, *Dama dama*, showed the largest margin of improvement relative to any other model, with the grayscale model outperforming the next best variant by +2%.

3.4.2.2 Data quality

When controlling for training size and comparing the palette-specific models with the halved mixed model, the same performance pattern persisted. Despite having

approximately the same number of training images, the halved mixed model achieved equivalent performance to the color model on the color test subset and outperformed the grayscale model by 5% on grayscale images. These results reinforce that the superior performance of the mixed model cannot be attributed solely to data quantity.

The comparison between the full and halved mixed models further supports this interpretation. The two models differed by only ~1%, suggesting that a substantial portion of the full dataset contributes limited additional information. This likely reflects redundancy introduced by repeated captures from the same camera stations across multiple years, where animals are photographed under nearly identical poses and viewpoints. Such redundancy adds volume but not necessarily diversity, implying that the mixed dataset's advantage arises primarily from greater variability in visual cues, not from the sheer number of images.

3.4.3 Further research

Future work could investigate whether increasing the variability of camera-station configurations, such as altering camera heights, angle and perspective improves model generalization while keeping the total number of images constant. Such an experiment would help clarify whether the performance gap observed between the *whole* and *halved* datasets is driven by the diversity of visual contexts rather than the number of training samples.

Enhancing viewpoint variability may reduce dataset redundancy and provide models with more informative representations of species morphology, allowing for clearer assessment of the relative contributions of data quantity and data quality.

3.5 Conclusions

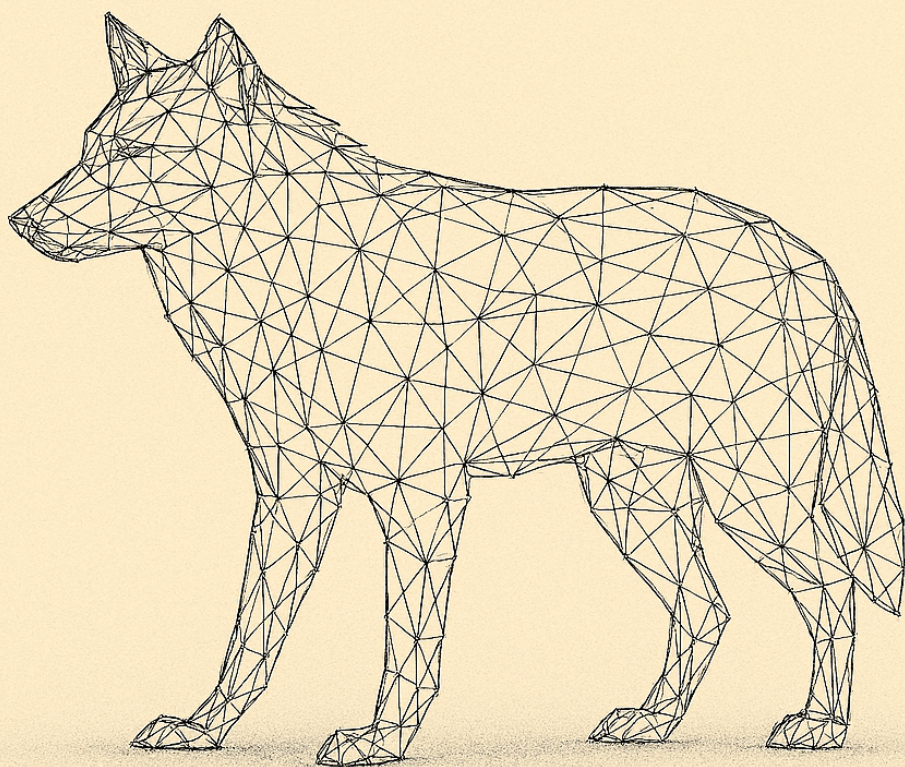
This study evaluated two strategies to improve wildlife object detection in camera-trap imagery: hyperparameter tuning and domain-specific dataset partitioning based on image palette.

Hyperparameter optimization consistently improved model performance across all dataset partitions, with the color model showing the largest relative gain. These results confirm that tuning training parameters is a critical step for achieving robust and accurate models, even when working with relatively small datasets.

In contrast, partitioning the dataset into separate grayscale and color subsets did not yield superior performance in either case. Mixed models trained on diverse data outperformed or matched palette-specific models, even when trained on only half the number of images. These findings suggest that data diversity may play a more influential role than strict domain specialization, particularly when the additional data contribute meaningful variability and feature richness. They also indicate that constructing multiple palette-specific datasets is unnecessary when effective data-augmentation techniques are employed.

Future research should explore pre-training strategies specifically adapted to grayscale imagery and further investigate the role of data variability, such as camera-station diversity, on model generalization. Overall, this study highlights the importance of optimizing training procedures and leveraging heterogeneous datasets to advance automated wildlife monitoring.

CHAPTER 4



4 Real-time wildlife detection using 4G camera traps and Deep Learning

Arnau Campanera-Moliné, Lluís Brotons, Daniel Ponsa, Elena Pi,
Noel Caparroz,, Víctor Sazatornil

ABSTRACT

Timely access to wildlife occurrence data is increasingly important for conservation and management, particularly for rare, elusive, or conflict-prone species in human-dominated landscapes, yet conventional camera-trap workflows often involve substantial delays between detection and data interpretation due to periodic data retrieval and manual image review. This chapter presents the development and evaluation of an automated real-time wildlife detection system that integrates 4G-enabled camera traps with a deep learning-based image recognition pipeline to enable near real-time monitoring. Images captured in the field are transmitted via cellular networks, automatically processed using a YOLO-based wildlife detection model, and delivered as instant notifications to end users through a messaging platform. The system was deployed in a mountainous environment with heterogeneous cellular coverage, providing a realistic assessment of its operational performance under field conditions. System performance was evaluated by quantifying image transmission latency, automated processing time, and the total end-to-end delay between image acquisition and notification delivery, as well as by analyzing the relationship between signal intensity and transmission time and comparing different computing platforms for real-time inference. Results show that transmission time is primarily constrained by network conditions, while processing time represents a minor component of the overall delay on most platforms. Even under suboptimal connectivity and low-power computing configurations, the system consistently delivered notifications within timeframes compatible with near real-time ecological applications. These results demonstrate the feasibility of integrating 4G camera traps and deep learning into a reliable real-time wildlife monitoring system capable of providing timely, actionable information to support conservation, management, and human-wildlife conflict mitigation.

4.1 Introduction

Wildlife monitoring is fundamental to ecological research and conservation. It provides researchers with critical information on species distribution, behavior and population dynamics. Amid the ongoing sixth mass species extinction (Ceballos et al., 2020) accelerating biodiversity loss, and increasing human pressures, accurate and efficient wildlife monitoring has become not only essential (Sutherland et al., 2004), but also an explicit requirement in internationally biodiversity agreements (Pimm et al., 2015; T. Dalton et al., 2022). This need is particularly acute for species that are rare, elusive, or involved in human-wildlife conflicts, such as large carnivores inhabiting human-dominated landscapes. Effective monitoring of such species is crucial not only for informed conservation decision-making but also mitigating conflicts arising from their presence and interactions with people.

Real-time telemetry from GPS collars and other tracking devices has long been used to gather positional data (Petkov et al., 2011; Wall et al., 2014). However, such approaches can be impractical or inefficient for solitary species like bears, for which locating individuals is difficult and for which collared animals may not represent population-level space use, among other impracticalities (Matthews et al., 2013). Furthermore, GPS collars can impose short-term burdens on some species (Stabach et al., 2020), while acting as deterrents to poaching in others (Milleret et al., 2021).

One of the most widely used tools to monitor wildlife are camera traps (Oliver R. Wearn & Paul Glover-Kapfer, 2017). These devices use passive infrared (PIR) sensors to trigger image capture when an animal passes within range. During low-light conditions, an infrared flash enables the acquisition of grayscale images, allowing for 24-hour monitoring. Camera traps are relatively inexpensive, easy to deploy, and resilient under harsh field conditions, making them suitable for large-scale monitoring schemes (Steenweg et al., 2017). They have transformed how presence, density and abundance estimates are produced (Howe et al., 2017), while also providing contextual information, such as time of detection, body condition of the individuals, or the social configuration of gregarious species.

Despite these advantages, camera traps depend on batteries and memory cards, requiring maintenance visits by field workers every three to six months depending on detection rates and storage capacity. Improper deployment, such as failing to clear moving vegetation in front of the sensor or orienting cameras towards the sun, can

rapidly fill memory cards with false triggers (empty frames), rendering cameras temporarily un-operational.

The large volume of imagery produced poses a major challenge for timely data processing. Managing datasets from dozens or hundreds of cameras constitutes a major operational bottleneck (Keitt & Abelson, 2021). Considering the entire workflow, from image retrieval to data analysis, a substantial delay typically occurs between an animal passing in front of a camera and the generation of structured ecological information.

Recent camera trap models incorporate cellular communication technology, transmitting images and videos over LTE/4G/3G or even 2G networks. This capability enables near real-time access to collected data. However, each transmitted image must still be manually reviewed to determine whether an animal is present and, if so, to identify the species. For large camera networks, this interpretation step can quickly become another bottleneck.

Advances in artificial intelligence offers a promising solution. Image recognition algorithms have already demonstrated strong performance in detecting wildlife in camera trap images (Norouzzadeh et al., 2018; Rigoudy et al., 2023), sometimes surpassing human accuracy (Van Horn et al., 2018). Integrating computer vision with real-time data transmission from 4G camera traps can substantially reduce interpretation time while improving cost-effectiveness and accuracy (Kissling, Evans, et al., 2024; Smith et al., 2024). Such systems could also support automated alerts for target species, providing immediate information about conflict-prone wildlife entering sensitive areas, or detecting reproductive females with offspring in order to temporarily restrict human activities.

The aim of this study was to develop a system capable of automatically processing transmitted camera trap images and generate near real-time notifications for selected species to support timely conservation and management decisions. Examples include detecting conflict-related species in sensitive zones, confirming reproduction in threatened species, or documenting illegal activities such as unauthorized vehicle use.

Although some recent studies have explored the efficiency of data transmission in 4G camera traps (Laughlin et al., 2023), no previous work found in the literature has evaluated how cellular signal strength affects transmission times or compared the

viability of different computing devices with different processing power for real-time automated inference.

The present study addresses these gaps by: (1) developing a real-time monitoring system that integrates 4G-enabled camera traps with a computer vision algorithm; (2) evaluating the relationship between cellular signal intensity and image transmission times; and (3) comparing processing times across four computing devices to assess their suitability for deployment within an operational real-time monitoring system.

4.2 Materials and Methods

4.2.1 Study area

The study was conducted in the Catalan Upper Pyrenees region in northeastern Spain, primarily within the county of Pallars Sobirà and more specifically along the Aurati's mountain range (UTM 31T 352387 4712600). The Pyrenees are more than 40% forested, dominated by beech (*Fagus sylvatica*), silver fir (*Abies alba*), mixed forests of both species as well as other dominant tree species like oak (*Quercus robur*, *Quercus pubescens*), chestnut (*Castanea sativa*), hazel (*Corylus avellana*) and gean (*Prunus avium*). With the exception of silver fir, many of these species develop large canopies that reduce incident solar radiation on the understory, potentially lowering the efficiency of solar panels powering camera traps.

Morphologically, the area is characterized by valleys with relatively steep slopes alternated with large massifs, while elevations range from 500 to 3400 m above sea level (Martin et al., 2012). This complex orography hinders uniform cellular network coverage, leaving some without any reliable signal, an important constraint for 4G-enabled monitoring devices.

The Pyrenees range provide a climatic border, between the Atlantic (oceanic) climate prevailing in the northwest and the Mediterranean climate to the southeast. The study area is located on the southern, Mediterranean-facing slope, generally experiencing more sunny days than the northwestern side.

4.2.2 Camera model selection and deployment

Cameras traps are monitoring devices equipped with a passive infrared (PIR) sensors that trigger image capture when motion and heat differences are detected. Two main types of commercial camera trap exist: (1) traditional offline units, widely used for passive monitoring, and (2) network-enabled units capable of transmitting data over 4G/3G/LTE cellular networks

For this study, the SPromise S378 4G/LTE camera trap, which offers data transmission capabilities was selected. This model was preferred over other cellular-enabled devices because it allows captured images and videos to be sent directly to a

customizable mailbox rather than a proprietary cloud platform. The latter has become an increasingly common practice in the past years, mainly targeted to a non-scientific community, such as game or non-professional users. These cloud platforms usually offer compulsory paid subscription plans that provide simple automatic AI tagging performance for a certain number of images. In such systems, false triggers are often counted as “tagged” images, making large-scale monitoring prohibitively expensive. While manageable for a few units, this model becomes economically unfeasible for large monitoring schemes.

The SPromise S378 4G/LTE captures images up to 12 megapixels, has a trigger speed of 0.6s, a 58° field of view, a detection range of up to 25 m, and uses infrared flash under low-light conditions instead of normal flash.

A total of four cameras were deployed, each requiring a SIM card for data transmission. The telecommunications provider with the strongest 4G coverage in the study area was selected, and each camera was assigned an unlimited data plan. Cameras were deployed at altitudes between 1,665 m and 1,838 m above sea level in locations identified by the Catalan Rangers Body (Cos d’Agents Rurals) where fauna was frequently detected and 4G signal was strong most of the time.

The selected model supports external power input through a 6V DC port, enabling near-continuous operation. One camera (tagged as CF71) was connected to a 3W solar panel unit equipped with an internal 8,000 mAh internal battery supplying 6V (Figure 13). The solar battery recharged during daylight hours and partially discharged overnight. During prolonged overcast periods with insufficient sunlight, the solar unit could become depleted, prompting the camera to automatically switch to its internal AA batteries. Once sunlight returned and the solar battery recovered sufficient charge, power supply reverted to the solar unit.

In case of solar panel failure or unavailable external power supply, the camera’s 12 AA batteries could provide up to six months of operation according to manufacturer specifications.

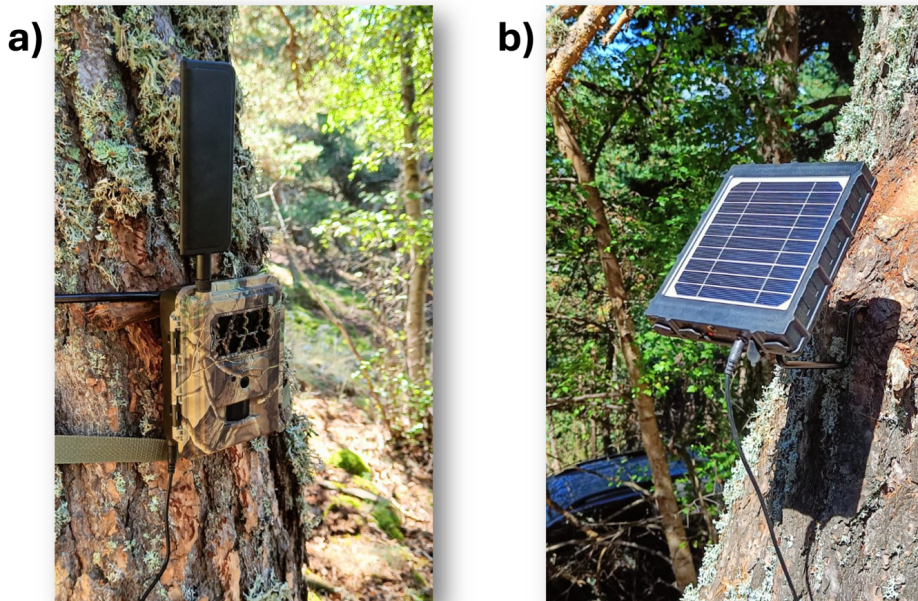


Figure 13. a) 4G camera trap setup with its wired connection to the solar panel. b) The solar panel was installed by screwing the holding arm into the bark of the tree. This method allowed to optimize panel orientation to maximize solar exposure.

4.2.3 Camera configuration

All cameras were configured identically. Upon activation of the passive infrared sensor, each unit captured a burst of three photographs, taken sequentially at one-second intervals. Due to software constraints, only the first image of each burst could be transmitted via the cellular network, while the remaining two were stored locally on the SD card.

Image resolution was set to the maximum available (12 megapixels; 4032 x 3024 px) to optimize remote species recognition by the AI system. When only 2G connectivity was available, the camera automatically downsampled images to 640 x 480 px to ensure successful transmission.

Each camera was set to transmit images immediately upon capture rather than during scheduled upload windows. To prevent cameras from becoming inoperative due to full storage, particularly in cases of repeated false triggers, such as those caused by

wind-moved vegetation, the devices were configured to overwrite the oldest files when memory capacity was reached.

Finally, all camera clocks were synchronized with standard mobile phone time, which itself is aligned with Coordinated Universal Time (UTC) through international time synchronization protocols.

4.2.4 Automatic recognition pipeline using deep learning

A complete automated data-processing pipeline was developed to enable real-time monitoring.

All cameras were configured to forward their images to the same dedicated mailbox. Each email contained the timestamp of image acquisition and the camera identifier in the subject header. The email body included metadata such as battery level, cellular signal strength, total number of photos and videos captured, and available storage capacity (Table 12). The captured image was attached to the message.

Table 12. Example of the camera metadata received with each transmitted image.

Date:20-07-2025 Signal: Very Good(4G) Battery:Full Temperature:25 degree Celsius(C) Total photos:3295, Total videos:0 Total space:28995M, Free space:15869M
--

A custom python script executed a sequence of tasks at 30-second intervals through a polling mechanism:

- **Mailbox polling.**

The script logs into the mailbox and checks for unread messages. When found, the attached images are downloaded to a local *inbox* directory. Checked mails are marked as 'read' to avoid repeated downloads.

- **Inbox directory monitoring.**

A file-watcher function compares the current contents of the inbox with the previous iteration. Newly added images are added to a processing queue. If no new files are detected, the script remains idle until the next polling cycle.

- **Automatic species recognition.**

Images in the queue are passed through a wildlife detection deep learning model. This study uses the enhanced wildlife detection model developed in Chapter 2, based on YOLOv9 (C.-Y. Wang et al., 2024). The model is capable of detecting up to 20 classes present in the Pyrenees with an accuracy of 78%, trained on nearly 50,000 regional images, ensuring high contextual relevance to the study area.

- **File handling.**

Images with no detected wildlife or containing false triggers are moved to a *null* directory. For images with detected animals, the model generates a text file containing the species label and bounding box coordinates. The script then sorts the image from the *inbox* to a species-specific directory.

These steps operated on a continuous loop at 30-second intervals. Shorter intervals were tested but resulted in mail provider access restrictions due to excessive queries. The 30-second interval therefore represented an effective compromise enabling near real-time processing without service interruptions.

4.2.5 Real-time notification system

The aim of the system was to notify the presence of certain species in the cameras. To enable instant reporting of those, a real-time notification system was developed using the Python package **Apprise**, which supports message delivery across numerous communication platforms (e.g., email, SMS, WhatsApp, Telegram). Telegram was chosen as it was one of the few platforms where *Apprise* allowed sending images with accompanying text to group chats, a requirement for validation of detections.

The system permitted notifications for all species or only a subset defined in a configurable “monitored species” list.

In the final stage of the Python pipeline, the script matched each recognized species label against the monitored species list. If a match was found, a Telegram notification was sent to a predefined group through a bot. Each alert included the scientific name of the detected species and the corresponding image, rendered with bounding boxes for verification (Figure 14). For this study, notifications were enabled for all detected species to ensure full monitoring of the system performance.



Figure 14. Four examples of the notifications received on the Telegram platform from different cameras.

The entire Python pipeline was executed on an AI workstation equipped with an NVIDIA A5000 GPU, which handled all image processing, data handling, and notification tasks (Figure 15).

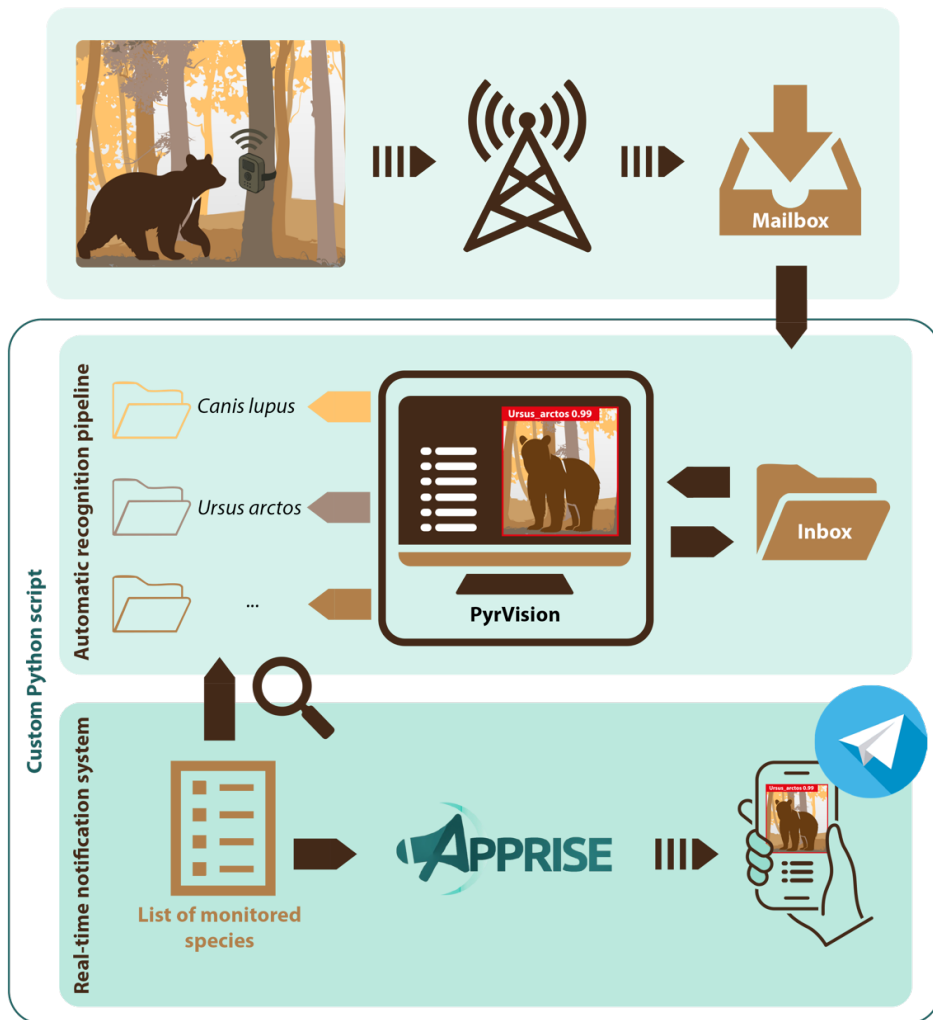


Figure 15. Diagram showing the logic behind the custom real-time Python script described in sections 4.2.4 and 4.2.5.

4.2.6 Signal intensity effect

Cellular signal intensity is known to vary widely depending on topography, distance to antennae, vegetation cover, weather conditions, and interference from other transmitters. Because signal strength directly influences effective bandwidth, it also affects the time required to transmit an image from a camera trap to the receiving mailbox.

To evaluate how fluctuations in cellular signal influence the feasibility of a real-time monitoring system, the relationship between signal intensity and transmission duration for all images sent between camera deployment and retrieval was examined. Each transmitted image included metadata reporting the instantaneous signal strength, enabling a direct comparison between both variables.

4.2.7 Processing time

A key factor shaping the practicality of a real-time monitoring system is the time required to process incoming images, generate species predictions, and, if needed, send notifications. To assess this, four computing platforms with different performance characteristics were benchmarked by processing an identical batch of 50 images.

These platforms included two GPU-based systems with different computational capacities, a standard CPU-based office laptop, and a Raspberry Pi 4B. Although the Raspberry Pi is not intended for large-scale monitoring, it was included to evaluate its potential suitability for low-power edge-based applications. The devices and their specifications are summarized in Table 13.

Table 13. Summary of the four tested systems and their different characteristics.

Device	Computing Unit	Processor Name	Computing Performance (GFlops)
GPU Server	GPU	Nvidia RTX 6000 Ada Gen	91,060
Workstation	GPU	Nvidia RTX A5000	27,770
Office Laptop	CPU	Intel i7-7920HQ	198.4
Raspberry Pi 4B	ARM	Broadcom BCM2711	13.5

Processing time measurements were implemented using Python's 'time' package. The timed workflow included:

- **Image processing** using the deep learning model developed in Chapter 2.
- **Notification dispatch** through the Apprise package to a Telegram group.
- **File management**, including storing each processed image in its corresponding species folder.

4.3 Results

4.3.1 Operational time

All four cameras exhibited different operational periods due to battery depletion or early removal (Table 14). Cameras tagged as CF71, CF73 and CF74 were deployed on the 28th of July 2023, whereas CF75 was installed three days later, on the 31st.

Table 14. Summary of operational duration and transmission statistics for each camera.

Camera Name	Deployment date	Withdraw date	Operational days	Number of transmissions	Average Transmission s/day
CF71	July 28 th , 2023	February 1 st , 2024	189	424	2.24
CF73	July 28 th , 2023	March 18 th , 2024	235	510	2.17
CF74	July 28 th , 2023	September 29 th , 2023	64	34	0.53
CF75	July 31 st , 2023	December 25 th , 2023	151	453	3

Cameras CF71 and CF73 were retrieved before their batteries were depleted because sufficient data had been collected and the units were needed for other projects. At the time of removal, CF71 still reported a ‘full’ battery, demonstrating that the solar panel provided uninterrupted power for more than five months, including during winter when solar radiation is reduced. CF73 reported a ‘median’ battery level upon withdrawal, while cameras CF74 and CF75 reported ‘low’ battery levels in their last transmitted message.

4.3.2 Signal Intensity and transmission time

To evaluate the performance of the camera under varying cellular coverage conditions and assess the system’s feasibility real-time monitoring, transmission time was

compared across five signal categories reported by the devices: Very Good (4G), Good (4G), Weak (4G), Very Weak (4G) and Weak (2G).

Transmission latency was calculated by extracting the exact acquisition timestamp from the image metadata and comparing it to the email receipt timestamp. The resulting difference in seconds, defined as latency, was plotted against signal intensity (Figure 16).

Across most 4G conditions, transmission latency remained relatively consistent. However, under ‘Very Weak (4G)’ conditions, latency increased substantially, reflecting the reduced bandwidth available for transferring full-resolution images.

Although 2G is intrinsically slower than 4G, transmission times under ‘Weak (2G)’ conditions were comparable to those observed under strong 4G coverage. This counterintuitive result is explained by the camera’s automatic downscaling mechanism: when only 2G is available, images are reduced to 640 x 480 px, significantly decreasing file size and compensating for the reduced network speed.

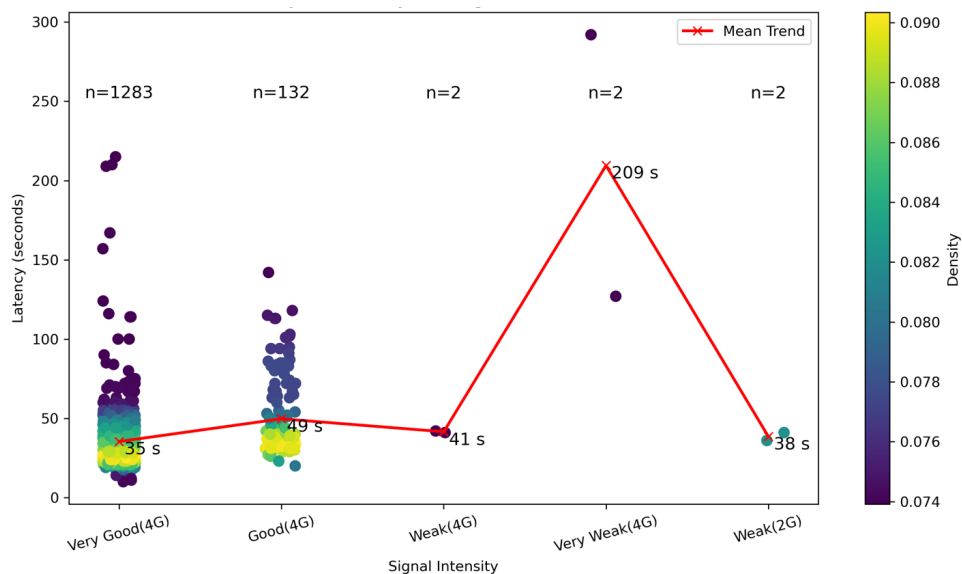


Figure 16. Events from the study grouped by signal intensity and represented in a scatter plot displaying ‘Signal Intensity’ against the latency.

4.3.2.1 Performance per camera

Figure 17 shows the transmission performance of each camera, and Table 15 summarizes transmission statistics. Although all devices were identical, each camera operated under different topographic and cellular-signal conditions, which substantially influenced their performance.

Camera CF71 consistently reported the highest signal intensity for all transmission events, resulting in the best average transmission time (24 s). Camera CF73 showed greater variability in transmission duration under strong signal conditions and recorded two events with ‘Weak (4G)’ and one with ‘Very Weak (4G)’ signal, yielding a higher average transmission time of 46s.

Camera CF74 operated under the poorest signal conditions of all units. Most transmissions occurred under ‘Good (4G)’ rather than ‘Very Good (4G)’ signal strength. This reduced connectivity may also explain the camera’s faster battery depletion, likely caused by additional power expenditure during repeated attempts to secure a stable connection and longer transmission events.

Camera CF75 alternated between ‘Very Good (4G)’ and ‘Good (4G)’ signal intensities, with two events at ‘Weak (4G)’ and one at ‘Very Weak (4G)’. This latter event resulted in the longest individual transmission time recorded in the study (4 min 52 s). CF75 also temporarily lost 4G connectivity during two events, falling back to 2G. Despite these two occurrences, CF75 achieved the second-best median transmission time (33 s).

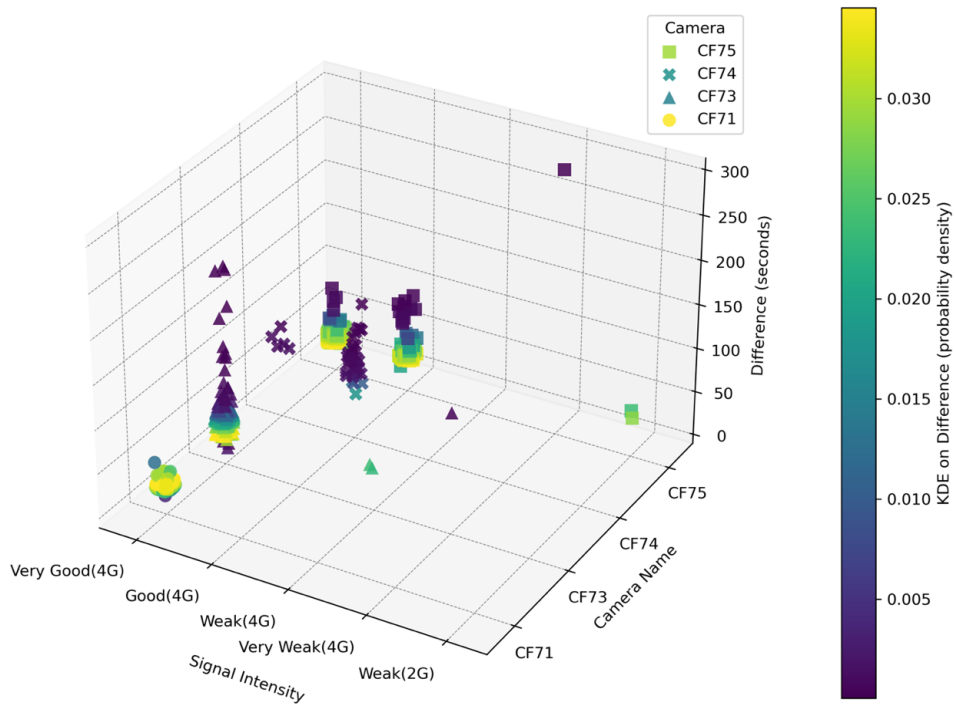


Figure 17. 3D scatter plot showing Signal Intensity against the difference between the time when the picture was taken by the camera trap and the time it was received by the mailbox.

Table 15. Transmission time statistics for each camera.

Camera Name	Transmission Times (s)					
	Images	Minimum	Maximum	Average	Median	Std. Dev
CF71	424	14	51	24	24	3.40
CF73	507	10	215	46	43	18.59
CF74	5	47	142	81	80	13.07
CF75	347	20	292	35	33	6.36
Totals (avg)		22.75	175	46.5	45	10.36

4.3.3 Processing time

Processing time was compared across the four systems described in section 2.7 using the same batch of 50 images. Results are presented in Table 16 and illustrated in Figure 18.

Table 16. Summary of computing metrics.

Device	Computing metrics (s)				
	Average	Min	Max	Median	Std. Dev
GPU Server	1.8152	0.9012	4.8299	1.8724	0.9102
Workstation	1.7614	0.9350	5.1248	1.2991	0.9965
Laptop	6.2812	4.1614	10.9868	5.9259	1.6702
Raspberry Pi	55.4064	52.4565	56.9006	55.6227	0.9607

Despite the GPU unit used by the GPU Server having more than three times the theoretical computational power (GFlops) of the Workstation, both devices exhibited nearly identical average processing times. This suggests that when inference times become extremely short, they contribute minimally to the total execution time, which becomes increasingly dominated by ancillary operations, particularly the notification stage, which involves contacting the Telegram bot and uploading the processed image.

The office laptop, operating purely on CPU, averaged 6.3 s and showed the highest variability in processing time. This may reflect CPU contention from background processes typical in general-purpose operating environments.

The Raspberry Pi 4B was by far the slowest device, averaging 55.4 s per image. Although unsuitable for large-scale, high-frequency monitoring systems, it provides insight into potential performance for low-power edge computing applications.

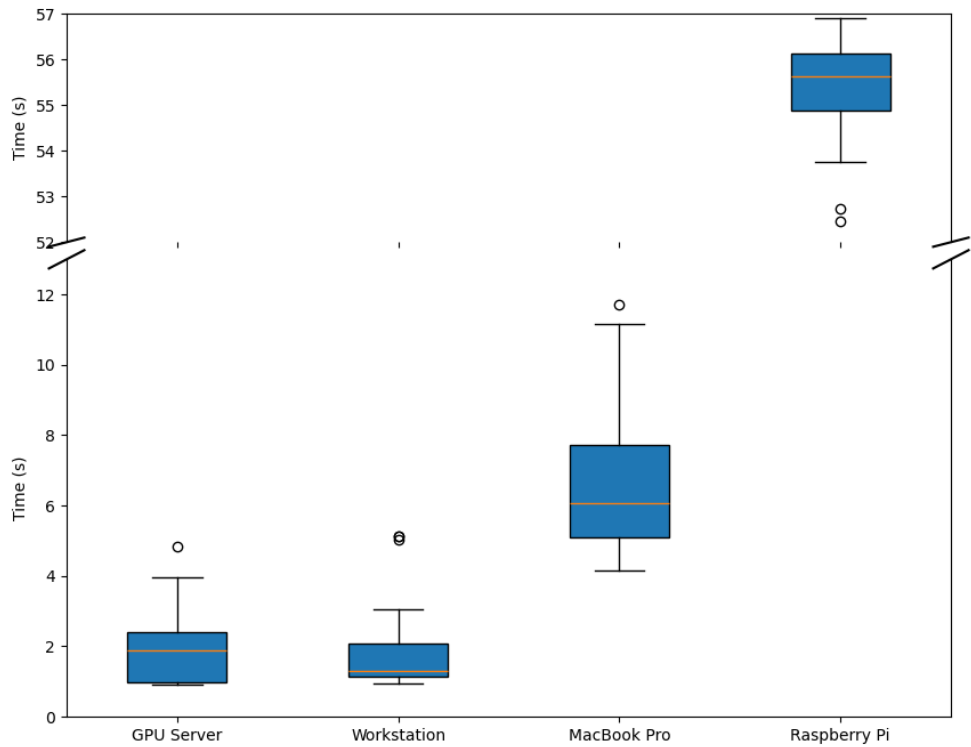


Figure 18. Box plot showing the results for each described system. Raspberry Pi is shown after a broken axis to better depict the results of the faster systems (GPU Server, Workstation and Laptop).

4.3.4 Full pipeline duration

To estimate the total end-to-end time required for the system to process and notify a detection event, three sequential components were considered:

1. **Transmission time**, defined as the interval between image capture and arrival in the mailbox.
2. **Polling time**, corresponding to the delay imposed by the 30-second mailbox-checking interval. An average delay of 15 seconds was assumed.
3. **Processing time**, which includes model inference, file handling, image annotation, and notification dispatch.

Using these components, three operational scenarios were evaluated: a best-case, average-case, and worst-case.

4.3.4.1 Best case scenario

The best-case scenario reflects the fastest possible end-to-end detection cycle. It combines the minimum transmission time recorded for each camera with the minimum computing time achieved by each device. The resulting times are presented in Table 17.

Table 17. Best-case total pipeline duration (s), combining minimum transmission and minimum processing times. Results in bold indicate the best overall times.

Camera	GPU Server	Workstation	Office Laptop	Raspberry Pi 4B
CF71	29.9	29.9	33.2	81.5
CF73	25.9	25.9	29.2	77.5
CF74	62.9	62.9	66.2	114.5
CF75	35.9	35.9	39.2	87.5

4.3.4.2 Average-case scenario

The average-case scenario uses the mean transmission time of each camera combined with the mean processing time of each computing device (Table 18). This scenario represents the most realistic performance estimate under typical field conditions.

Table 18. Average-case total pipeline duration (s), using mean transmission and mean processing times.

Camera	GPU Server	Workstation	Office Laptop	Raspberry Pi 4B
CF71	40.8	40.8	45.3	94.4
CF73	62.8	62.8	67.3	116.4
CF74	97.8	97.8	102.3	151.4
CF75	51.8	51.8	56.3	105.4

4.3.4.3 Worst-case scenario

The worst-case scenario reflects the slowest conditions observed across the entire pipeline. It combines the maximum transmission time registered for each camera with the highest processing time recorded for each device. These values are shown in Table 19.

Table 19. Worst-case total pipeline duration (s), combining maximum transmission and maximum processing times. Results in bold indicate the worst overall time.

Camera	GPU Server	Workstation	Office Laptop	Raspberry Pi 4B
CF71	70.8	71.1	77.0	122.9
CF73	234.8	235.1	241.0	286.9
CF74	161.8	162.1	168.0	213.9
CF75	311.8	312.1	318.0	363.9

4.4 Discussion

4.4.1 Viability of the real-time system

The concept of *real-time* is inherently context-dependent. In wildlife monitoring, the temporal resolution required to consider information “real-time” varies according to the species, movement patterns, and the intended management or conservation action. For instance, hourly GPS positions from a tracking collar may be perceived as real-time compared to retrieving camera trap images every several months. However, for highly mobile species, hourly location updates may still be insufficient for immediate interventions, highlighting the need to define real-time requirements based on operational objectives.

The aim of this study was to develop a system capable of automatically processing transmitted images and generating timely notifications to support conservation and management decisions. Although such actions may be time sensitive, they rarely require immediate (<10 s) warnings. Instead, notification times on the order of one to several minutes are generally adequate for applications such as confirming reproduction, detecting presence of conflict-prone species, monitoring sensitive habitats, or identifying potential illegal activities.

The results demonstrate that all four evaluated computing devices, including the slowest (Raspberry Pi 4B), were capable of processing and notifying detections within acceptable delays. Even under the worst-case conditions, the full pipeline timeframe was still compatible with most ecological and conservation scenarios envisioned for the system. Thus, the proposed architecture is viable for a broad range of near real-time wildlife monitoring applications.

4.4.2 Ecological applications

The proposed automatic 4G camera trap, AI-based recognition, Telegram notification system enables a wide range of ecological and conservation applications by providing rapid, actionable information. The system was originally conceived for the early detection of endangered or conflict-prone species in the Pyrenees, particularly brown bears and potentially recolonizing wolves. However, its utility extends well beyond these target species.

With appropriate spatial deployment, the system could support:

- **Range mapping:** defining the spatial distribution of species with much higher temporal resolution than periodic manual camera trap surveys.
- **Movement and migration monitoring:** documenting short-term spatial dynamics of wildlife populations at fine temporal scales.
- **Den or nest monitoring:** enabling verification of reproductive events, and facilitating management actions (e.g., restricting human activities) depending on animal presence or absence.

The system could also help mitigate human–wildlife conflicts. Real-time notifications sent to shepherds or land managers could reduce predation on livestock by wolves or bears through rapid activation of defensive strategies, such as deploying guardian dogs, relocating herds, reinforcing fencing, or using light- and sound-based deterrents. Similarly, early detection of crop-raiding species such as wild boar or deer could facilitate protective measures in agricultural areas.

Another situation where having a timely response could be crucial is the early warning of potentially dangerous wildlife in populated areas (Saravanan et al., 2024). This could be bears scavenging trash cans or dumpsters inside villages, as well the increasing presence of wild boar sounders wandering on cities (Cahill et al., 2012; Ciach et al., 2023; Marin et al., 2024), entailing potential risks such as wildlife-vehicle collisions. Similar systems have already been deployed to prevent collisions near roads strategically placing cameras along road sections prone to wildlife crossings (Simonthomas et al., 2024; Singh et al., 2025). Yet, most show moderate effectiveness with challenges to solve (Nandutu et al., 2022).

Finally, this real-time system could alert of illegal activities such as detecting hunters and poachers in protected areas or during hunting banned seasons. In these situations, having a timely response would be crucial to allow the arrest of offenders or to avoid losing endangered animals. Furthermore, the presence of unauthorized vehicles in protected areas or restricted zones could also be prosecuted.

4.4.3 Limitations and further research

Despite the versatility of a solar-powered, real-time monitoring system, one major limitation is its dependence on cellular coverage. Rare or elusive species often inhabit remote areas where network connectivity is weak or nonexistent, preventing reliable transmission. One potential solution is to integrate satellite-based communication systems. However, these systems currently offer low bandwidth at high operational costs, making the transmission of full-resolution images impractical.

In such cases, edge-based processing becomes essential. A unified device incorporating a camera trap, a low-power computing module (e.g., Raspberry Pi), and a satellite transmitter could perform AI inference locally and transmit only compact metadata or alerts. This approach would reduce bandwidth requirements but increase power consumption, necessitating larger solar panels or improved energy-management strategies. Future research should therefore explore:

- Algorithmic optimization, such as training smaller sizes of the models or exporting edge-optimized versions for energy-efficient inference and reduced time.
- Deployment of edge-optimized hardware, such as Google Coral or NVIDIA Jetson platforms, which are specifically designed for efficient AI inference in low-power environments (Zuolkernan et al., 2022).

An efficient and reliable edge-computing solution could enable other real-time applications that require to make timely-decisions such as automatic deterrence systems that would, for example, activate lights or sounds in the presence of a predatory species near a fence with livestock in it, or shooring raiding species in crop fields.

A final limitation relates to the detection model itself. The system can only identify species included in the training dataset, typically the most frequently observed species. This restricts early detection of invasive species, rare species, or unexpected events, which may not be recognized by the model. Expanding and diversifying the training datasets, or integrating anomaly detection algorithms, could help mitigate this limitation.

4.5 Conclusions

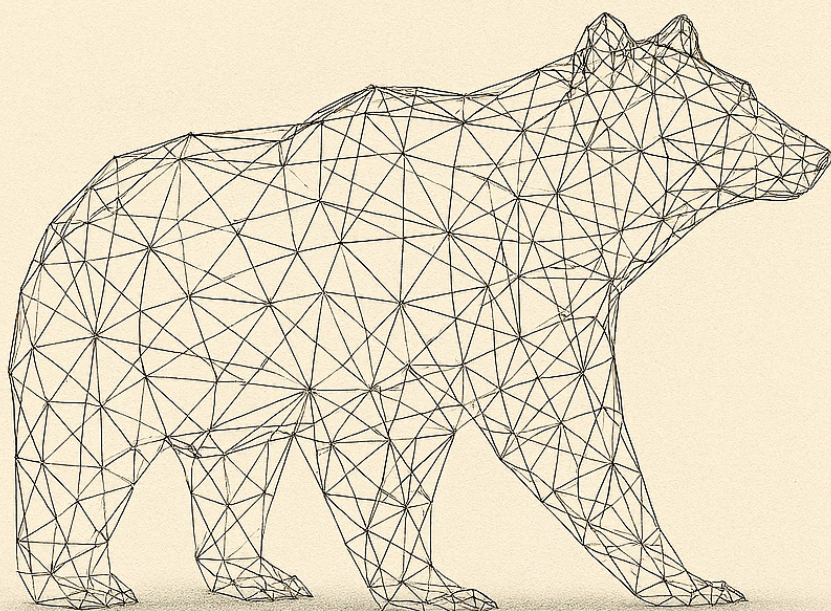
This study demonstrates that a functional real-time wildlife presence notification system can be developed by integrating 4G-enabled camera traps, a deep learning-based wildlife detection model, and an automated processing and notification pipeline. The system reliably transmitted and processed wildlife detections under a range of field conditions, highlighting its applicability for near real-time ecological monitoring.

Image transmission times were strongly influenced by cellular signal intensity; however, even under suboptimal conditions, transmission delays remained compatible with near real-time monitoring, averaging 46.5 seconds across all events. Processing times varied substantially across the four tested computing devices, ranging from 1.7 seconds on the Workstation to 55.4 seconds on the Raspberry Pi. Despite this spread, the total end-to-end delay remained within operationally useful limits. The difference between the best-case (25.9 s) and the worst-case (363.9 s) scenarios was 338 seconds, and even the slowest configuration would be adequate for realistic monitoring loads. For example, the most active camera (CF75) transmitted only 3 images per day on average, far below the processing capacity of even the Raspberry Pi.

Nevertheless, the system's applicability is constrained by the availability and reliability of cellular coverage. Many rare or elusive species inhabit remote or poorly connected areas where 4G connectivity is intermittent or absent. Furthermore, the detection capability of the system depends on the performance and trained examples of the underlying wildlife recognition model, which may overlook species not represented in the training dataset.

Future research should focus on shifting computation to edge-based processing by optimizing wildlife detection models for greater energy efficiency and reduced inference time without compromising accuracy. This would lay the groundwork for integrating satellite communication modules, enabling wildlife monitoring in remote and low-connected areas. Additionally, rapid, on-device inference could support the development of automated deterrence systems that activate in response to the presence of specific species, offering new opportunities for proactive human-wildlife conflict mitigation.

CHAPTER 5



5 Inferring Brown Bear hair snare interactions by automatically detecting bipedalism on camera trap images using Pose Estimation and a Multi-Layer Perceptron

Arnau Campanera-Moliné, Víctor Sazatornil, Lluís Brotons, Felipe Lumbreras, Robert Benavente, Maguiña Ramilo-Henry, Cristina Ramos, Santiago Palazón,

Daniel Ponsa

ABSTRACT

Hair snares, consisting of strands of barbed wire that collect tufts of fur, have long been used as a non-invasive sampling technique for DNA-based monitoring of different species of bears. However, to prevent cross-contamination by other animals or different bear individuals, frequent visits by technicians are required to collect samples and clean snares, a highly demanding process in terms of time and resources that hampers scalability. In the Catalan Pyrenees, only 24.6% of the visiting efforts come of positive, highlighting the need for automated, scalable solutions that can operate independently of cellular networks or human intervention.

This study proposes a novel methodology to automatically infer hair snare interactions by detecting bipedalism in images acquired by camera traps. This is implemented by using a state-of-the-art deep learning model for pose estimation, combined with a Multilayer Perceptron (MLP) to classify bipedalism or quadrupedy based on predicted keypoints. Due to the lack of annotated bear pose datasets, a custom dataset of 2,373 images collected in the Catalan Pyrenees was manually annotated with 15 anatomical keypoints. Using YOLOv11, the trained pose estimation model achieved a keypoint inference precision of 93.2%, while the MLP reached an accuracy of 96.1% in distinguishing bipedalism from quadrupedy.

Finally, to enable inference on the edge, several verification strategies are proposed using only the pose estimation output, including bear geometrical illustrations derived from the inferred keypoints. This innovative lightweight solution, designed for satellite-based transmission, enables scalable deployment in low-connectivity and remote areas, significantly reducing the need for frequent manual inspections.

5.1 Introduction

In the context of rapid biodiversity loss and growing human population, accurate methods for monitoring wildlife are increasingly needed (Sutherland et al., 2004). This is particularly true for rare and/or elusive species, and those involved in human-wildlife conflicts, such as large carnivores persisting in human-dominated landscapes. Monitoring these species is not only essential for making meaningful conservation decisions, manage human-wildlife conflict and policy development, but is also mandated and demanded at different legal and societal levels, respectively (Strampelli et al., 2022; Treves, 2009).

Hair snares have been used for long as a non-invasive sampling technique for DNA-based monitoring of different species of bears (GARDNER et al., 2010). These simple “traps” consist of strands of barbed wire often installed on trees and baited with odorous attractants (Sato et al., 2020), that collect tufts of fur when a bear rubs against them. This technique allows for an individual-scale monitoring approach that has been used to derive bear population estimates, for example through capture-recapture (Mowat & Strobeck, 2000). However, valuable temporal and contextual information is not captured by this method, such as the day and time of the contact, the body condition of the individuals, or the presence of cubs accompanying reproductive females.

To address these limitations, complementary methods such as camera traps pointing to the hair snares have proven effective in filling many of these information gaps, such as in the case of the monitoring of the brown bear (*Ursus arctos*) in the Pyrenees. When used together, DNA samples can be paired to bear images to recognize individuals. Nevertheless, to avoid cross-contamination by other animals or different bear individuals, frequent visits by technicians are required to collect trapped hair and clean snares.

Several factors negatively influence detection rates at hair traps, thus weakening the reliability of population estimates. For instance, bears passing by may not always rub or leave enough hair for DNA amplification. Additionally, their low population density, remote habitats and large home ranges make frequently checking the extensive network of traps highly demanding in terms of time and resources. In the

Catalan Pyrenees, a total of 338 hair traps are blindly checked, ideally every fifteen or thirty days, depending on the located area. However, only 24,26% of the visits result positive (Palazón et al., 2023). Also, long intervals between bear interaction and sample collection often result in DNA degradation or contamination by other individuals.

The ability to send real-time images of some modern camera-traps with mobile connectivity eases visiting pressure of hair-snare stations by technicians. This capability can contribute to more efficient use of resources. However, bear populations often inhabit remote areas with limited or no cellular coverage, making its implementation very limited. Moreover, determining whether a bear physically interacted with a snare often requires analysis of behavioral cues such as bipedal stance. These limitations hamper scalability and are highly demanding in terms of time and resources, if available. At the same time, these challenges highlight the need for automated, scalable solutions that can operate independently of cellular networks or human interpretation.

Recent advances in artificial intelligence, particularly in the field of computer vision, offer promising tools to address these limitations. Among these, deep learning techniques such as Convolutional Neural Networks (CNN), which revolutionized computer vision tasks over a decade ago, offer powerful tools to classify images, detect objects or infer human body positions (Jocher et al., 2023c). Animal ecology has also embraced this machine learning outbreak (Perry et al., 2022), and thanks to the advances in this field, conservationists can use deep learning tools to detect animals in thousands of images (Rigoudy et al., 2023; Roy et al., 2023) or estimate their pose without tracking physical markers (Mathis et al., 2018). Although markerless pose estimation in animals had also been previously done using photogrammetry, lighting and background conditions had to be very specific (Sellers & Hirasaki, 2014). Recently, deep learning-based pose estimation has been used to track behavior in rodents (Maghsoudi et al., 2017) as well as to understand the complex locomotion biomechanics of cheetahs in the wild (Joska et al., 2021), among other cases. Yet, no study has used pose estimation on brown bears to infer behaviors like bipedal standing which could be used as a proxy to hair snare rubbing. Having a reliable tracking methodology could enable the automatization of hair-snare monitoring. Running pose estimation models on the edge and transferring results over

a cheap but reliable satellite connection could enable scalability and a much more efficient use of time and resources.

To assess the capabilities of pose estimation algorithms, a methodology to infer bipedalism in brown bears is proposed. A widely used, state-of-the-art deep learning model capable of pinpointing human body parts is adapted to be capable of recognizing bear body parts. Subsequently, bipedalism or quadrupedy is inferred from predicted keypoints on a Multiple Layer Perceptron model.

The aim of this paper is (1) to develop a methodology to infer bear back-rubbing on hair snares by predicting bipedalism in contrast to quadrupedy. To do so, (2) adapt a human pose estimation model to be able to estimate pose in bears. Due to the scarcity of brown bear images in the growing population in the Pyrenees (3) acquire knowledge from resembling, already annotated, animal datasets by transfer learning. Lastly, (4) find a reliable solution for edge devices to provide inferred pose information to end users in a lightweight form, suitable to be transmitted over satellite connections, thereby enabling scalability in diverse environments.

5.2 Materials and Methods

5.2.1 Study Area and Data Collection

The brown bear (*Ursus arctos*) population in the Pyrenees is currently considered increasing, with an estimated population in 2023 of 86 individuals, distributed across an area of 7,100 km² between France and Spain (Sentilles et al., 2024).

Imagery used in this experiment was provided by the Catalan Government through their Fauna and Flora Service which, in collaboration with the Catalan Rangers body, has been actively monitoring bear populations since its reintroduction in 1996. Monitoring is conducted via an extensive and annually expanding network of camera traps and 223 hair snares designed for individual bear identification through DNA analysis. Each hair trap consists of strands of barbed wire embedded in tree bark (Figure 19b).

The present study analyzed images collected for brown bear population monitoring during 2020 – 2023 in the Catalan Upper Pyrenees, covering an area of 2,336 km², mainly in the counties of Pallars Sobirà, Pallars Jussà and Alta Ribagorça, in northeastern Spain (Palazón et al., 2023)(Figure 19a). Of the 223 hair snares stations, 109 were additionally equipped with camera traps oriented towards the tree with the embedded hair snare.

The monitoring network generally operates from March to November. During bear hibernation period, most cameras are withdrawn.

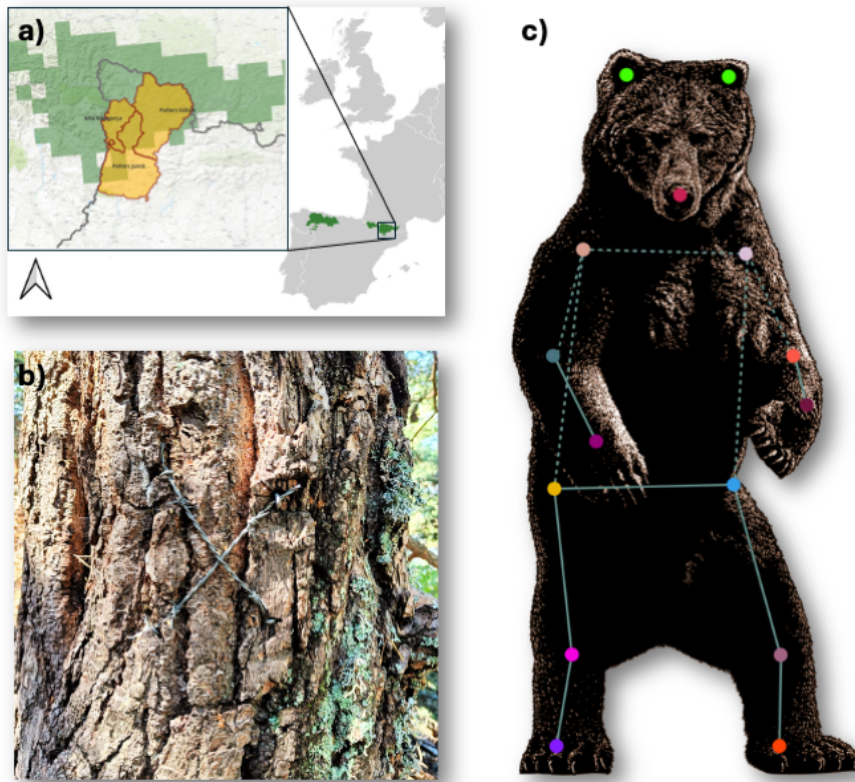


Figure 19. a) Map showing the area from which data was collected. The green grid shows the occurrence of brown bear (Source: (Kaczensky et al., 2024)). Yellow area indicates the counties where images for this study were obtained. b) A hair snare, constructed from strands of barbed wire embedded in a tree's bark, is used to collect bear fur for individual bear identification through DNA analysis. c) Illustration of the 2D stick figure used, with the selected anatomical keypoints to track bear position.

5.2.2 Dataset Composition and Pose Annotation

5.2.2.1 Training and validation sets

Among the entire monitoring network of 109 stations with hair snares and camera traps, 25 stations registered bear individuals during 2020 to 2022. Imagery from this period was used to train and validate the models. This dataset consisted of a total of 2,373 images:

- 1,247 color images captured during daylight.
- 1,126 grayscale images taken mostly at night or twilight, using the camera trap infrared sensor.

This represented a ~53/47% balance between day and night imagery. Brown bears were present in 2,330 images, while 43 were empty background frames. Images were obtained from six different camera trap brands, which allowed to cover different image sizes, lens characteristics, infrared flash systems and color sensors, giving heterogeneity to the dataset.

Image resolutions ranged from 2.1 megapixels (1920x1080) to 14.6 megapixels (4416x3312) and all files were stored using JPG format.

5.2.2.2 Data Splitting Strategy

In machine learning, mathematical models are constructed to make data-driven predictions from input data. These data are usually divided in three subsets; training, validation and test. The training subset is used to fit the model's parameters to the data. This model is successively used to predict responses on the validation set, which provides an unbiased evaluation of the parameters used. These parameters can be changed and the model retrained to search the best predictions on the validation set. Finally, the best fitted model is evaluated on the test set, which consists of a third subset with holdout data, never used for training nor validation.

The 2020-22 dataset was divided into training and validation subsets. To avoid data leakage, the division was made at the camera-station level, ensuring that each station was assigned exclusively to either training or validation.

- Training set. 14 camera stations, providing 1,700 images with 33 empty backgrounds
- Validation set. 11 camera stations, accounting for 673 images with 10 empty backgrounds.

This distribution resulted in a ~72/28% split between training and validation sets.

5.2.2.3 Test set

To further ensure no data contamination, the test set was built using images from new camera trap locations deployed in 2023. These stations were not used in training or validation; therefore, background scenes were always new to the model.

- Test set. 7 camera stations, with 193 images, including 107 grayscale and 86 color images.

5.2.2.4 Pose annotation

To track bear body pose, 15 anatomical keypoints were selected for annotation, corresponding to the nose followed by left and right ears, shoulders, elbows, wrists, hips, knees and feet (Figure 19c). Each keypoint was defined by a triplet of values (X, Y, Z):

- X and Y: normalized image coordinates.
- Z: keypoint visibility
 - o 2 = clearly visible
 - o 1 = located outside the image frame
 - o 0 = occluded by bushes, trees, other body parts or other bear individuals.

Images were manually annotated using Computer Vision Annotation Tool (CVAT) (Sekachev et al., 2020) by two different human experts.

Finally, every image was labeled as either “bipedalism” or “quadrupedy”, depending on whether the bear was standing on two or four legs (Figure 20).

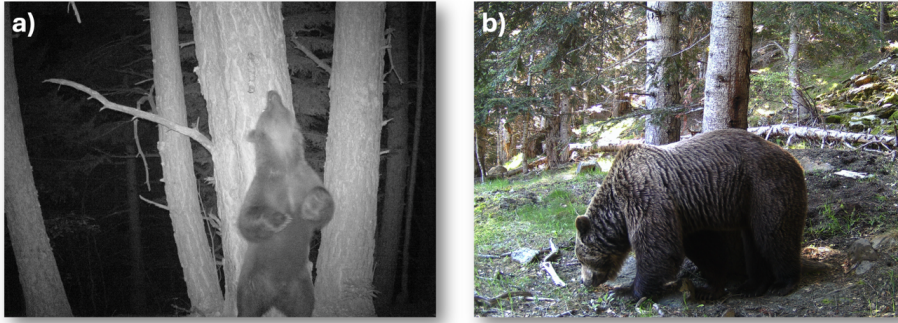


Figure 20. Data examples of a) standing bear in bipedalism and b) a walking bear in quadrupedy.

5.2.3 Model architectures and selection

For this research, the state-of-the-art deep learning architecture for real-time object detection, YOLO (*You Only Look Once*) (Jocher et al., 2023c) was selected due to its superior performance across versions and its capability to perform pose estimation tasks. Alongside pose estimation, the Ultralytics YOLO framework offers multiple versions and capabilities ranging from image classification, object detection (with support for oriented bounding boxes) and semantic segmentation. Ultralytics models use PyTorch framework for training and inference and can be exported to various formats depending on deployment requirements.

Versions supporting pose estimation include YOLOv8 (Jocher et al., 2023b), YOLOv11 (Jocher & Qiu, 2024) and YOLOv12(Tian et al., 2025). However, the latest version was still in early development at the time of writing this manuscript. Therefore, only YOLOv8 and YOLOv11 were chosen for a comparative evaluation to determine which model architecture yielded the best performance on our dataset.

To maximize pose estimation accuracy, the largest model sizes of each architecture were selected. YOLOv8 largest model (YOLOv8x-pose-p6) consists of 294 layers, 99,180,696 parameters, and requires 267.4 GFLOPs. In contrast, YOLOv11 largest model (YOLOv11x-pose) consists of 372 layers, 58,889,881 parameters, and requires 204.3 GFLOPs. It is important to note that newer models do not necessarily correspond to better performance as improvements may be focused on speed of inference, efficiency or portability in detriment of accuracy.

For this study, precision and accuracy in pose estimation were prioritized over speed of inference, as the model was not intended to have a time-sensitive decision-making task.

5.2.4 Model training

The Ultralytics YOLO framework maintains a repository of pre-trained weights for various model architectures and tasks. These weights are trained using the Common Objects in Context (COCO) dataset (Lin et al., 2014) and its variants. For this study, a model trained on the COCO 2017 Keypoints dataset, containing person keypoints in challenging, uncontrolled conditions was used as initialization weights. Using a pre-trained model can significantly reduce time and computational resources as the neural network is already capable of detecting fundamental image features such as edges, shapes and textures that later lead to higher, more complex structures. Indeed, human and bear bodies have high structural similarity, thus, adapting a pose estimation model of the human body to bears generates a better model than just training a bear pose model estimator from scratch, using only bear examples.

Training was conducted sequentially for both YOLOv8 and YOLOv11 variants, using their default hyperparameters (Figure 22a) with the exception of the input image size, which was rescaled from the default 640 to 1024 pixels. This change was made to help the network improve the detection of smaller parts like the nose. To prevent overfitting, an early stopping function was applied with a patience of 100 epochs. This form of regularization halts the training if the model shows no improvement on the validation set over a defined number of consecutive epochs. This method helps the model to better generalize on unseen data.

All computations were performed on a Supermicro GPU server equipped with 768 GB of RAM and four NVidia L40S GPUs, totaling 184 GB of VRAM. The same software environment was kept across experiments, with CUDA version 12.5 and Ultralytics version 8.3.193.

No data preprocessing or manual data augmentation was done by the annotators. However, data augmentation was automatically applied during training by the pipeline of Ultralytics YOLO implementation. Data augmentation consists of a series of transformations applied to the original data to generate more varied training images, thereby improving model generalization and robustness (Figure 21). Specifically, the following set of data augmentation techniques were applied by default to the dataset:

- Color space adjustments: including hue (hsv_h), saturation (hsv_s), and brightness (hsv_v).
- Geometric transformations: such as translation (translate) and scaling (scale).
- Mosaic augmentation: combining four images into one (mosaic).
- Horizontal flipping: (fliplr). To enable this data augmentation technique, a mirror map of keypoints must be provided to preserve anatomical laterality.

These transformations were seamlessly integrated into the training process by Ultralytics YOLO code (Jocher et al., 2013) and could be customized or disabled as needed.

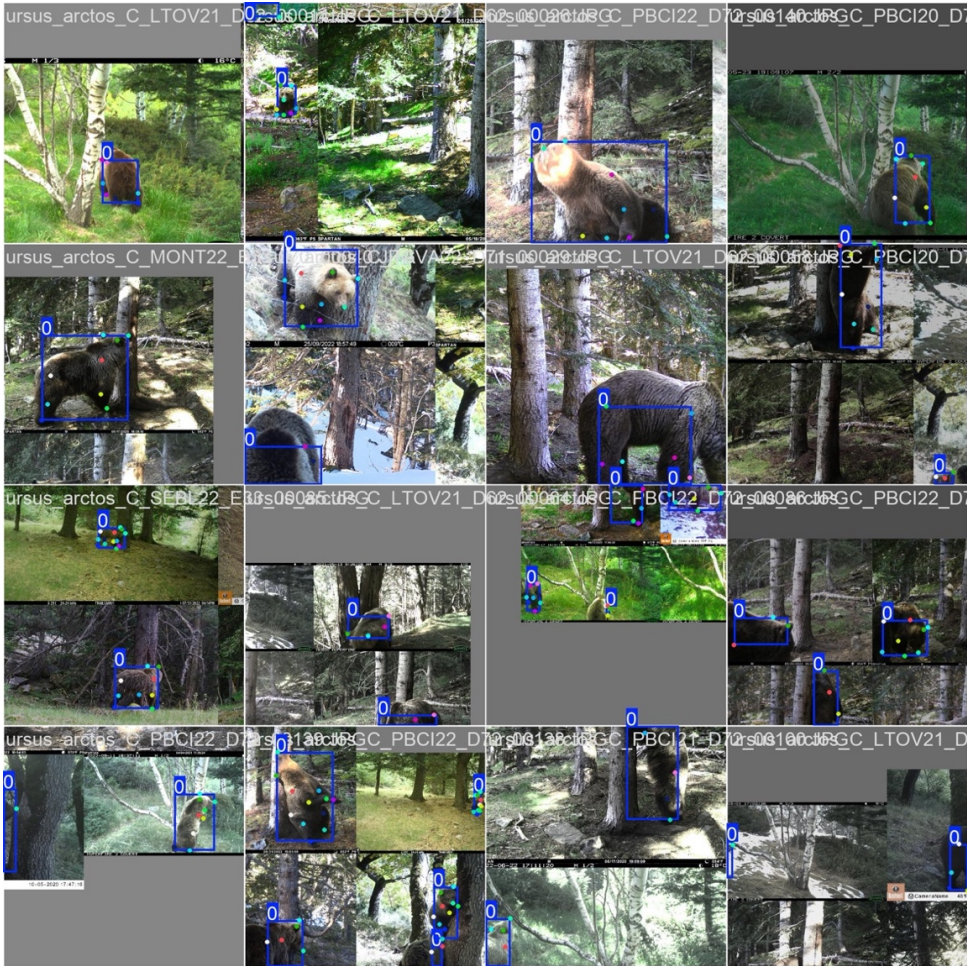


Figure 21. Examples of data augmentation performed by the data preparation process during training. Different applications of color adjustment techniques can be seen, as well as different image scaling and mosaics made by up to four images.

5.2.5 Transfer learning

Transfer learning is a widely adopted technique in computer vision, particularly when working with deep neural networks (Nimma et al., 2024; Sushila Devi & Sowmya, 2025). It enables the transfer of knowledge already acquired by a model on a larger, similar dataset, which is then fine-tuned with the distinctive features of another, often smaller dataset. With this approach, the final trained network often yields superior performance, rarely achievable if the network is trained from scratch with randomly initialized weights.

As previously mentioned, the COCO 2017 Keypoint dataset served as the starting point to train the model, providing the neural network with foundational knowledge of keypoint detection. However, the main limitation of the COCO and most pose estimation datasets is their focus on human subjects and their activities (Zheng et al., 2023). While body joints between humans and bears are mostly shared, body shapes, clothes and fur differ significantly.

According to Mathis et al., 2020, an intermediate fine tuning could improve final performance, thus a second experiment was conducted. A subset of the Stanford Dog Dataset (Khosla et al., 2011) with pose annotations was used for intermediate training, prior to fine-tuning on the bear dataset. This dog-pose dataset contains 6,773 training images and 1,703 validation images, specifically curated for dog keypoint estimation (Biggs et al., 2021). Each image is annotated with 24 keypoints, including visibility and can be used as a solid foundation for training robust pose estimation models.

The aim of this intermediate step was to facilitate a better transfer of knowledge from the human pre-trained model to the bear-specific dataset. Although both classes are very different, training on a dataset of a quadrupedal animal with similar visual features, may help the network to better learn the nuances of bear anatomy. Additionally, the dog dataset contained roughly triple the number of images than the bear dataset.

Following the initial training, a second round of training was performed. This time fine-tuning the pre-trained COCO Keypoint human model on the dog-pose dataset (Figure 22c). The same process was done for both v8 and v11 model variants.

5.2.6 Evaluation metrics

To evaluate the models' ability to accurately predict bear anatomical keypoints and compare the performance between them, several metrics were employed.

5.2.6.1 Object Keypoint Similarity (OKS)

OKS is a widely used metric for pose estimation tasks. It follows a concept similar to *Intersection over Union* (IoU) in traditional object detection tasks to assess the correctness of the inferred keypoints. While IoU is an area-based overlap measure, OKS uses a distance-based approach:

$$OKS = \frac{\sum_i \exp\left(\frac{-d_i^2}{2s^2\left(\frac{1}{N_{kpt}}\right)^2}\right) \delta(v_i > 0)}{\sum_i \delta(v_i > 0)}$$

Where:

- d_i : Euclidean distance between the predicted keypoint coordinates and the corresponding ground truth.
- v_i : visibility flag indicating whether each ground truth keypoint is labeled and visible.
- s : object scale (square root of the bounding box area containing the bear)
- $\sigma = \frac{1}{N_{kpt}}$: constant value controlling sensitivity or “falloff” rate

To compute OKS, d_i ’s are compared to an unnormalized Gaussian curve with a uniform standard deviation defined as $s\sigma$. This yields a per-keypoint similarity value that ranges from 0 to 1. Averaging the values over all visible keypoints results in an OKS score. Perfect matching predictions would yield $OKS = 1$ while predictions deviating by several multiples of $s\sigma$ would yield $OKS \sim 0$.

Following the same logic, other popular metrics for object detection are adapted to pose estimation.

5.2.6.2 Mean Average Precision

One of the most common metrics to assess object detection performance is the mean Average Precision (mAP) in its two variants: mAP@50 and mAP@50-95. These two metrics consider predicted keypoint similarity, as well as correctness of the keypoint label, at different thresholds.

- mAP@50: Predicted keypoints are counted as correct if OKS > 0.5. Then, results are averaged into a single score.
- mAP@50-95: Stricter variant following the same logic as mAP@50. In this case, it computes mAP for 10 different thresholds, from OKS = 0.5 to OKS = 0.95 using increments of 0.5. Then, they are all average to a single score.

In this study, mAP@50-95 using OKS was used as the primary metric to determine the best performing model for bear pose estimation.

5.2.6.3 Precision and Recall

Similar to how they are computed in object detection tasks, precision and recall metrics are defined as follows:

$$Precision = \frac{TP}{(TP + FP)} \qquad Recall = \frac{TP}{(TP + FN)}$$

- Precision: measures out of all the detections, how many were correct. In other words, the ratio between true positives and all predicted positives.
- Recall: measures how many labels were correctly predicted out of all the ground truth labels. Also, in other words, the ratio between true positives and all the positives that should have been predicted.

Where:

- *TP*: True positives. That is, the number of bears detected at OKS ≥ 0.5 .
- *FP*: False positives. That is, the number of inexistent bears detected by the model.
- *FN*: False negatives. That is, the number of bears missed by the model.

5.2.7 Hyperparameter tuning and final training

Most machine learning systems come with different hyperparameters that can be modified (Bischl et al., 2021). Hyperparameters are high-level, structural settings, like the learning rate, momentum or the weight decay, that significantly influence training dynamics and final performance. Recent DL models, as the one used in this experiment, crucially depend on a wide range of hyperparameter choices that should be tailored to the specific problem and dataset to optimize performance (Feurer & Hutter, 2019).

To this end, hyperparameter tuning was performed using the built-in hyperparameter evolution module within the Ultralytics YOLO framework. This module uses genetic algorithms which are inspired by the mechanisms of natural selection and gene mutation. The tuning process runs for a pre-defined number of iterations. In each iteration, new hyperparameter values are generated by applying small, random perturbations to existing values within the local hyperparameter space. Then, the model is trained using the generated hyperparameters for a small number of epochs. A fitness function evaluates the performance of the model with mutated hyperparameters on the dataset, guiding the evolution towards optimal performance.

Tuning hyperparameters is a very computational demanding process, which calls for a lot of iterations to find the best values, thereby requiring a lot of time and energy. The starting weights for this process were taken from the previously fine-tuned dog pose model. Due to the smallness of the bear dataset, the entirety of the training and validation subsets were used to search for the optimal hyperparameters for both YOLOv8 and YOLOv11.

The following hyperparameters were researched for the best combination and performance; initial learning rate (lr0), final learning rate (lrf), SGD momentum (momentum), optimizer weight decay (weight_decay), warmup epochs (warmup_epochs), warmup initial momentum (warmup_momentum), box loss gain (box), class loss gain (cls), distribution focal loss (dfl), pose loss gain (pose), keypoint loss gain (kobj), image HSV-Hue augmentation (hsv_h), image HSV-Saturation augmentation (hsv_s), image HSV-Value augmentation (hsv_v), image translation (translation), image scale (scale), image flip left-right (fliplr) and image mosaic (mosaic).

Upon the completion of the hyperparameter tuning process, two different trainings were performed using the optimal hyperparameters found for both variants:

- COCO → BEAR with optimized Hyperparameters

To assess the gains in performance achieved only by tuning the hyperparameters, the model was retrained on the custom dataset from the COCO pre-trained weights using the optimized hyperparameters (Figure 22b).

- COCO → DOG → BEAR with optimized Hyperparameters

Similarly, to assess the benefits of transfer learning from a more biologically similar species, models were initialized with weights resulting from fine-tuning the COCO pretrained model on the dog dataset, and then fine-tuned on the full bear dataset using the optimized hyperparameters (Figure 22c).

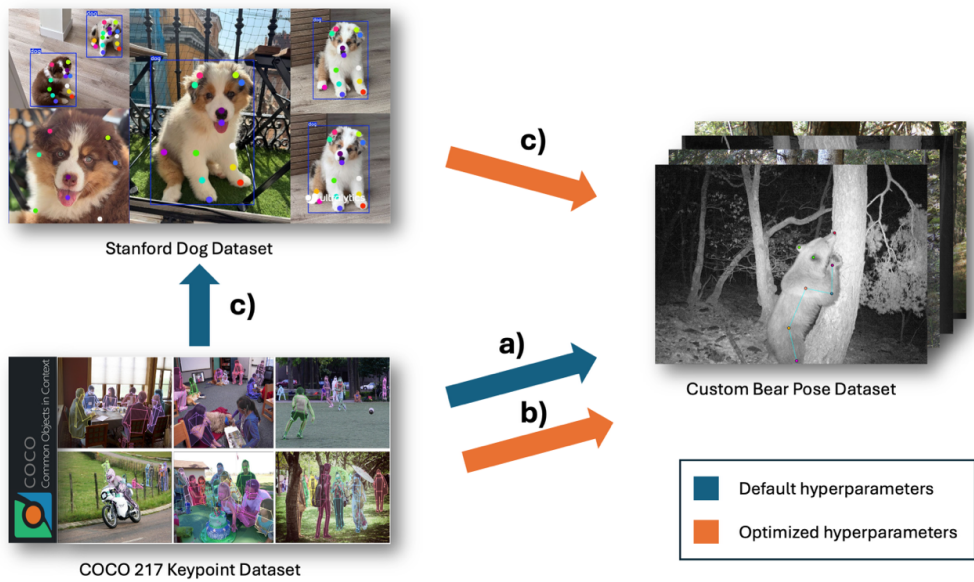


Figure 22. Summary of the different training paths used to evaluate the performance gains obtained from different strategies. a) Initial training from the COCO human dataset pre-trained model to the custom Bear Pose dataset using the default hyperparameters. b) Same training as a), but using the optimized hyperparameters found. c) Training path where an Intermediate training from COCO human dataset to the Stanford Dog dataset was performed, followed by a final fine-tuning from the Stanford Dog dataset trained model to the custom Bear Pose dataset using the optimized hyperparameters.

Following training, pose models were evaluated on the test set.

5.2.8 Bipedalism detection

To classify bipedalism versus quadrupedal posture, different strategies were considered. An initial proposed solution was to consider the angle formed by the two vectors connecting the shoulder to the hip, and the hip to the knee. However, terrains with pronounced slope caused false positives. Additionally, incorrect adjustment of one of the three keypoints already led to misclassifications. Therefore, a more robust solution utilizing the whole keypoint population was considered. For this, a Multi-Layer Perceptron (MLP) was trained.

MLP are simple neural networks consisting of fully connected layers of neurons (perceptrons) organized into three or more layers with different number of neurons in them. They are well-suited for distinguishing patterns in data that are not linearly separable and are trained using backpropagation.

The ground-truth keypoint annotations were reused to train the MLP classifier (Figure 23). To enhance representation, data was combined to define new features in order to determine which combination yielded the best classification performance:

- **angles.** Six different angles were defined (6 values):
 - Right and left back leg (foot, knee, hip)
 - Right and left front leg (hand, elbow, shoulder)
 - Right and left hip angle (knee, hip, shoulder)
- **bbox.** Bounding box coordinates outputted by YOLO (4 values).
- **bbox_keypoints.** Bounding box coordinates computed from the outermost keypoints (4 values).
- **pose.** Normalized keypoint position within the image (30 values).

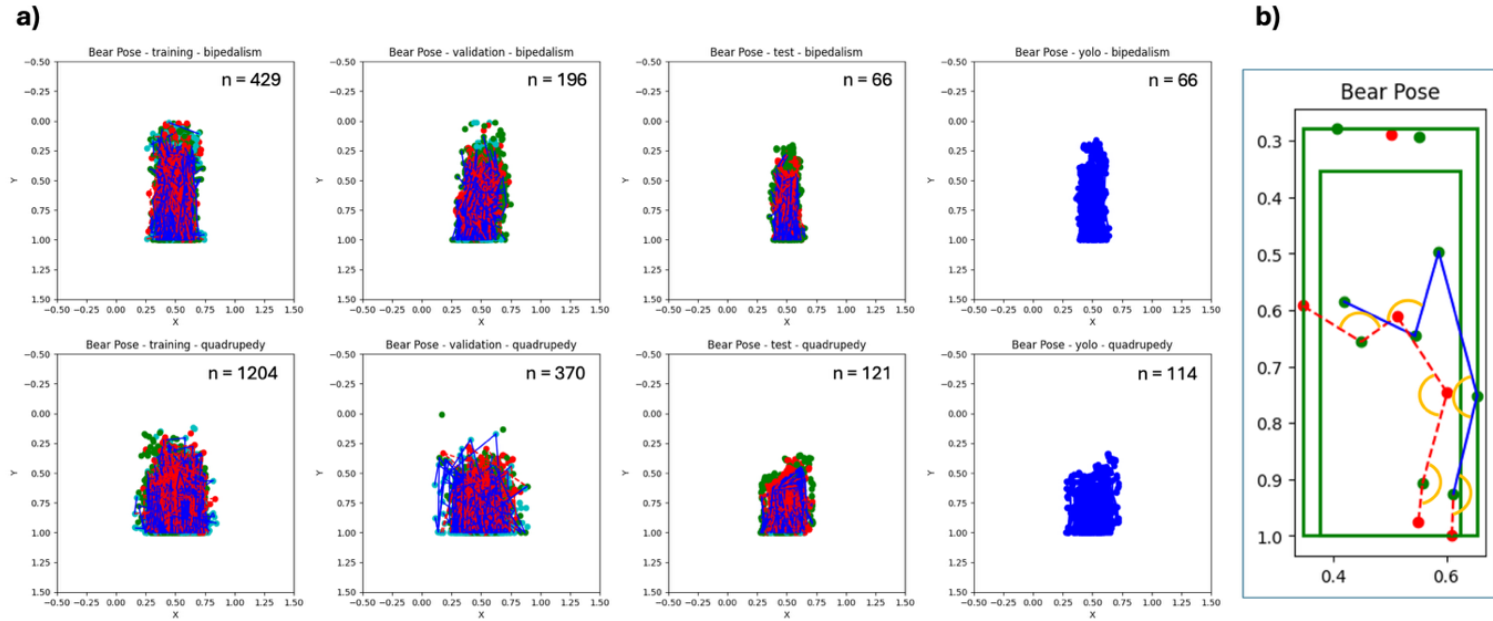


Figure 23. Data used to train and evaluate the MLP. a) Ground-truth annotated keypoints from training, validation and test set classified into bipedalism or quadrupedy. Predicted keypoints by YOLO on the test set were also used to evaluate MLP performance on a real-case scenario. ‘n’ indicates number of examples in each set. b) Example of the difference between the predicted YOLO bounding box and the bounding box computed considering the out-most keypoints. Additionally, the angles considered to classify bipedalism or quadrupedy are also shown.

For this task, an MLP with three layers (an input layer, a hidden layer and an output layer) was trained. The input layer varied depending on the size of the features used. The hidden layer size was established by multiplying the input layer size by a factor of 1, 2 or 3 to evaluate different model complexities. Finally, the output layer had a size of 2, which were either '0' for bipedalism, or '1' for quadrupedy.

Batch sizes of 8, 16 and 32 were also evaluated, as well as different loss functions (cross entropy, weighted cross entropy, focal loss and dice loss). The MLP trained for a maximum of 10,000 epochs, with an early stopping function with a patience of 300 epochs. The starting learning rate was set to 0.001 and was reduced by a factor of 0.5 when the validation loss reached the plateau.

5.2.9 End-user verification solution

To enable scalability in diverse environments, satellite connections provide a global and reliable solution as several satellite constellations now offer near-continuous worldwide coverage. However, data bandwidths are extremely limited or expensive, and they cannot be used as a cost-effective solution to transmit images, preventing any visual verification of inferred keypoints and poses. To address this, several end-user verification solutions are proposed. These approaches enable efficient remote monitoring of bear posture without the need to transmit full-resolution images over costly satellite connections.

5.2.9.1 ASCII Visualizations

ASCII, American Standard Code for Information Interchange, is a character encoding standard for representing 128 characters. Text-based visualizations can be created using combinations of the different characters available.

A custom Python script was developed to convert the normalized relative positions of predicted keypoints into a 100x40 character grid. Keypoints were labeled with their names, and skeleton connections were represented for enhanced interpretability (Figure 24a).

5.2.9.2 Stick figures and geometric illustrations

Another lightweight approach using only the information provided by predicted keypoint positions is to render stick figures recreating the bear's skeleton structure (Figure 24b & 24f). This simple form of pose reconstruction is already very intuitive and computationally efficient.

Finally, a more intuitive and interpretable solution could be to draw bear bodies using geometrical shapes such as ellipses and circles. Using predefined rules, body parts can be drawn by pairing different anatomical keypoints. For example, an ellipse linking elbows and hands to build armpits, or another containing both shoulders and hips to draw the core (Figure 24d & 24h).

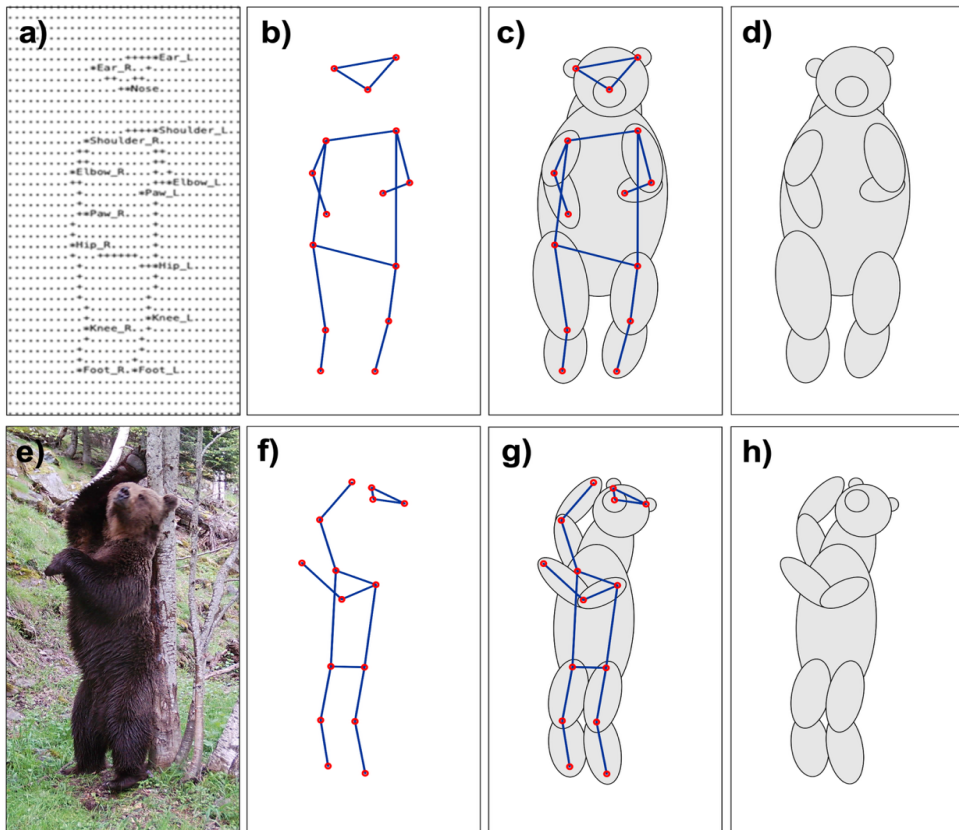


Figure 24. Different examples depicting how verification representations could be presented. a) ASCII representation of a bear with labeled keypoints. b) & f) Different examples of stick figures. c) & g) representation using both stick figures and ellipse-based bear body reconstruction. d) & h) Bear reconstruction examples using only ellipses from keypoints. e) Base image for figures 'f', 'g' and 'h'.

These solutions provide a clear and understandable representation of bear postures without requiring high-bandwidth image transmission. This enables a verification step for end-users in situations where pose models may struggle to make accurate predictions.

When used, the final requirements of data transmission decrease, going from a full image (~3 MB), down to sending only coordinates (~100 B). This huge reduction in weight can enable scalability by bringing down information transmission costs. Additionally, these representations can be drawn on the edge device side or on the client side, depending on data transmission bandwidth or margin, bringing more flexibility to the end user.

5.3 Results

5.3.1 COCO to BEAR Fine tuning with default hyperparameters

The initial experiment involved adjusting the YOLOv8 and YOLOv11 model variants pre-trained from the COCO 2017 Keypoint dataset, by fine-tuning them on the BEAR dataset to establish baseline performance metrics. Training lasted for approximately 8.3 hours for YOLOv8 and 5.3 hours for YOLOv11.

When faced to the test set, YOLOv8 achieved a very poor performance, scoring a $mAP@50$ of 0.125, a $mAP@50-95$ of 0.0532, a precision of 0.265 and a recall of 0.265. Similarly, YOLOv11 training yielded a $mAP@50$ of 0.179, a $mAP@50-95$ of 0.0889, a precision of 0.302 and a recall of 0.277 (Table 3). In a scale of 0 – 1, these are very low performance results, indicating that the model was struggling to correctly learn the new data.

5.3.2 Hyperparameter tuning

Given the very poor performance achieved by both YOLO variants with the default hyperparameters, a tuning process, detailed in section 2.7 was conducted. It ran for 50 epochs per iteration, across a total number of 400 iterations for both models (Figure 25a and 25b).

Tuning results for both models are summarized on Tables 20 and 21 and Figure 25c and 25d.

Table 20. Fitness results of the hyperparameter tuning process.

Model	Initial fitness	Best fitness	Best fitness iteration
YOLOv8x-pose	0.72309	1.01661	186
YOLOv11x-pose	0.70027	0.98493	132

Table 21. Default and optimized values for all hyperparameters researched during the tuning process.

Hyperparameters		YOLOv8x-pose	YOLOv11x-pose
	Default	Optimized	Optimized
lr0	0.01	0.01587	0.00646
lrf	0.01	0.01022	0.01093
momentum	0.937	0.91946	0.96331
weight decay	0.0005	0.00039	0.00054
warmup epochs	3.0	3.77514	3.59038
warmup momentum	0.8	0.92757	0.94474
box	0.2	0.2	0.2
cls	0.5	0.50566	0.48503
dfl	1.5	1.98741	1.2499
pose	12.0	13.10857	19.98896
kobj	1.0	0.57627	0.8385
hsv_h	0.015	0.01522	0.00695
hsv_s	0.7	0.3501	0.23855
hsv_v	0.4	0.30315	0.28526
translate	0.1	0.11464	0.12243
scale	0.5	0.60262	0.50872
fliplr	0.5	0.20968	0.15244
mosaic	1.0	0.99473	0.97912

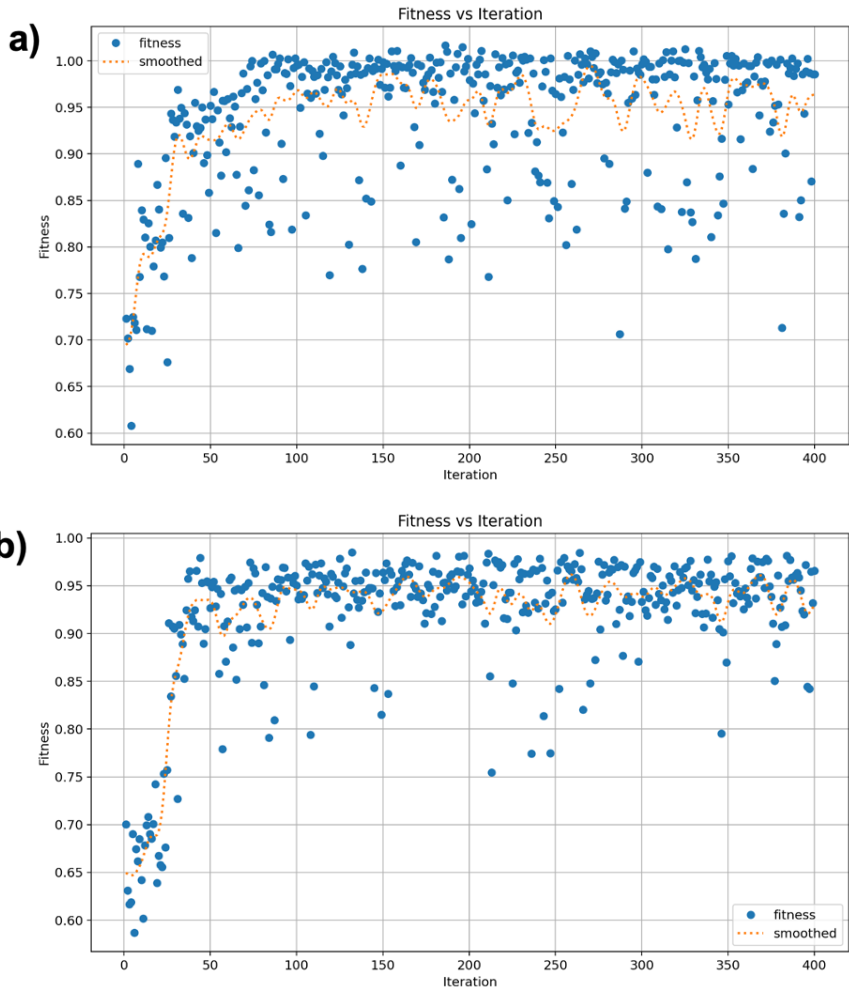


Figure 25. Hyperparameter tuning results. Fitness vs iteration charts for a) YOLOv8 and b) YOLOv11.

Inferring Brown Bear hair snare interactions by automatically detecting bipedalism on camera trap images using Pose Estimation and a Multi-Layer Perceptron

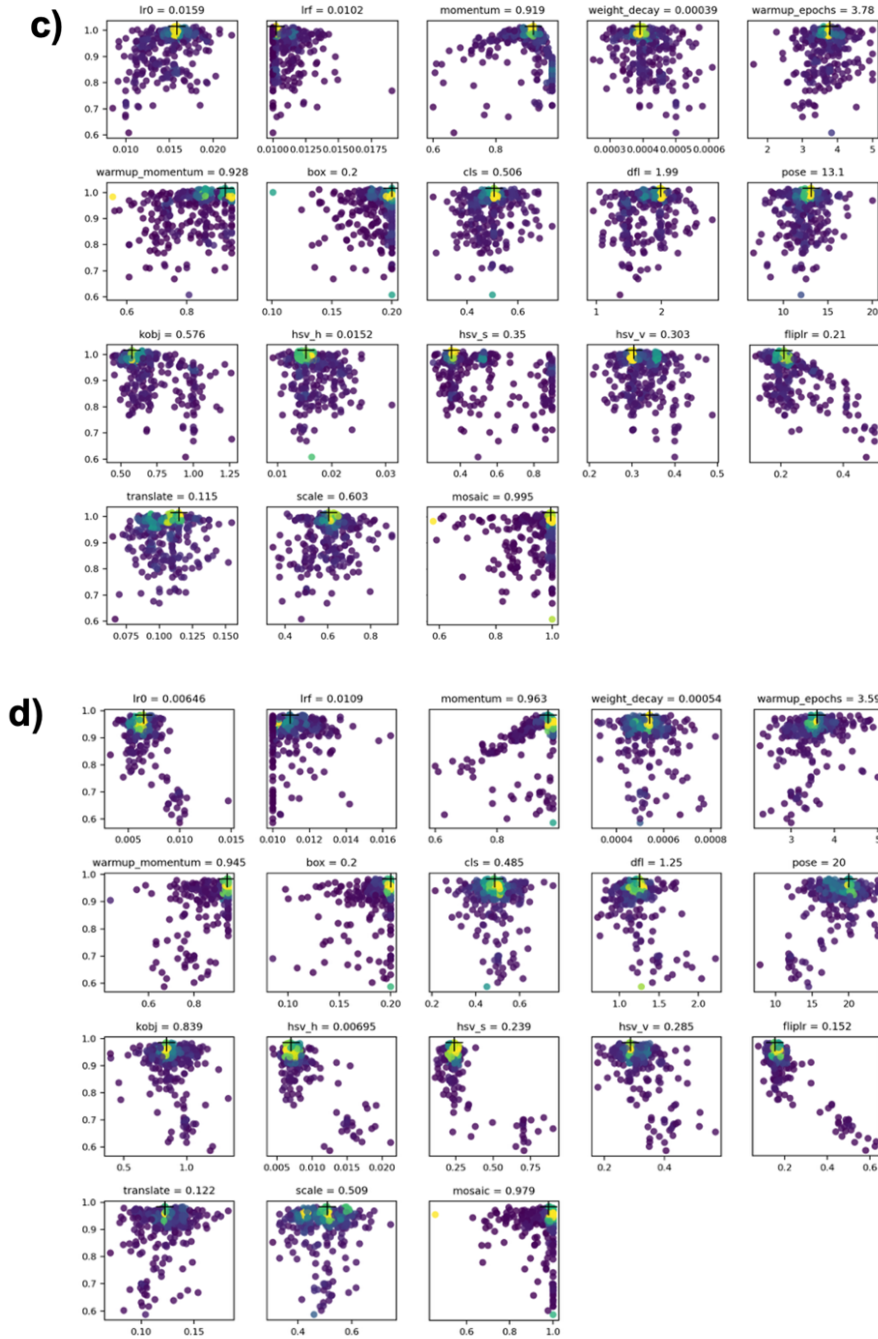


Figure 25. 2 Researched hyperparameters history are shown for c) YOLOv8 and d) YOLOv11 processes. Lighter colors represent more point density. Best found value is shown on top of each graph.

5.3.3 Pose Estimation Models Summary

A total of six different experiments were performed to assess the contributions of the different techniques used to enhance the final models, including the hyperparameter tuning described in section 5.2.7 and the fine-tuning process described in the 5.2.6 section.

Table 22 summarizes final metrics for both models according to each training strategy.

Table 22. Test results for the different training strategies for YOLOv8x-pose-p6 and YOLOv11x-pose. Results account for different intermediate fine-tuning and different hyperparameters choice. Bolted results show the best metric.

Metrics	YOLOv8	YOLOv11	YOLOv8	YOLOv11	YOLOv8	YOLOv11
	COCO	COCO	COCO	COCO	COCO	COCO
	→	→	→	→	→ DOG	→ DOG
	BEAR	BEAR	BEAR	BEAR	→ BEAR	→ BEAR
	Default HP	Default HP	Optimized HP	Optimized HP	Optimized HP	Optimized HP
<i>mAP@50</i>	0.125	0.179	0.838	0.79	0.748	0.889
<i>mAP@50-95</i>	0.0532	0.0889	0.375	0.381	0.451	0.435
<i>Precision</i>	0.265	0.302	0.898	0.878	0.765	0.932
<i>Recall</i>	0.265	0.277	0.815	0.779	0.734	0.851

These results demonstrate a substantial improvement of both models' performance over the baseline, confirming the effectiveness of both intermediate fine-tuning and hyperparameter tuning.

YOLOv11 detected 182 bears on the test set, while missing 13 and wrongly detected 6 inexistent bears. This translates to a precision of 96,8% and a recall of 93,3% for bear detection.

Similarly, YOLOv8 detected 181 bears, missed 14 and wrongly detected 10 bears, meaning a precision score of 94,8% and a recall of 92,8%.

5.3.4 Bipedalism detection

MLP training considered all possible combinations described in section 2.8. A summary of the best results for each feature and their combinations on the test set with ground truth annotations, is shown on Table 23.

Due to the low variability of the testing data, several combinations yielded the same exact results. For example, the best performing model which uniquely used the ‘pose’ feature, yielded the same exact results with hidden layer sized 30, 60 and 90, as well as using cross entropy, weighted cross entropy, focal loss or dice loss.

Table 23. Best results for each feature using Batch size = 8.

Feature	Hidden size	Loss function	Precision	Recall
pose	60	CE	0.9634	0.9626
bbox + pose	34	WCE	0.9572	0.9572
bbox_keypoints + pose	68	WCE	0.9521	0.9519
bbox_keypoints + pose + angles	80	WCE	0.9465	0.9465
bbox + pose + angles	40	WCE	0.9467	0.9465
pose + angles	36	DL	0.9411	0.9412
bbox_keypoints + angles	20	DL	0.9248	0.9251
bbox + angles	10	FL	0.9144	0.9144
angles	12	WCE	0.8953	0.8930
bbox_keypoints	4	FL	0.8967	0.8770
bbox	4	FL	0.8892	0.8663

To further discriminate the best performing combination, all variants of the model using the ‘pose’ feature were tested on the predicted YOLO output on the test set (Table 24).

Table 24. Results for each variant of the MLP model using only ‘pose’ feature information.

Feature	Hidden size	Loss function	Precision	Recall
pose	90	FL	0.9634	0.9611
pose	90	DL	0.9501	0.95
pose	90	CE	0.9444	0.9444
pose	30	WCE	0.9443	0.9444
pose	30	FL	0.9388	0.9389
pose	60	FL	0.9333	0.9333
pose	60	WCE	0.929	0.9278
pose	60	CE	0.929	0.9278
pose	90	WCE	0.9193	0.9167
pose	60	DL	0.9147	0.9111
pose	30	CE	0.9101	0.9056
pose	30	DL	0.8847	0.8667

According to these results, the best performing model and combination was an MLP using the ‘pose’ feature with a hidden layer containing 90 neurons and using focal loss as the loss function.

The chosen MLP results on the test set and on the predicted YOLO output are shown on Figure 26. In both cases, the system fails in just 7 images.

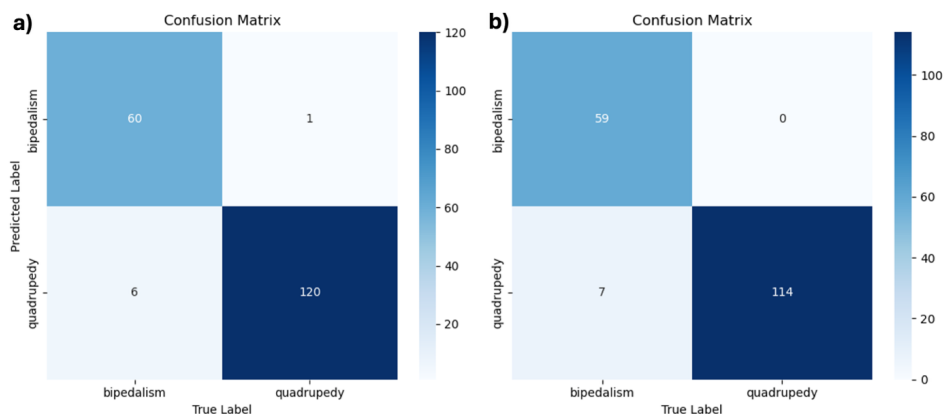


Figure 26. Confusion matrix showing classification results for MLP on the a) ground truth annotations of the test set and the b) predicted output of the trained YOLOv11 pose estimation model.

5.4 Discussion

This study aimed to develop an automated methodology to detect bear back-rubbing on hair snares by predicting bipedalism posture in contrast to quadrupedal posture.

5.4.1 Model performance and Transfer learning

Since no annotated bear pose dataset was available, a new dataset had to be built from scratch. Despite having an extensive monitoring network available, bear images were scarce and keypoint annotation was a slow and labor-intensive process. With only 2,373 images, neither YOLOv8 nor YOLOv11 could achieve an acceptable performance on the first round of trainings. By comparison, COCO Keypoint dataset contains more than 200,000 labeled images, while the Stanford Dog Dataset is made of 8,476 images.

After hyperparameter optimization, YOLOv8 and YOLOv11 models' performance were boost significantly. Based on $mAP@50-95$, YOLOv8 model performance showed a 605% relative improvement, while YOLOv11 also showed a substantial 329%. These were substantial performance leaps, moving the models from a low-performance regime to a more competitive range. Although $mAP@50-95$ scores remained modest, these were moderate-to-good performance as pose estimation tasks are more complex than common object detection, so scores tend to be lower.

Aiding the model shifting its domain from human-centric to more visually similar dog features improved overall performance for both models, resulting in a 20% relative increase in $mAP@50-95$ performance for YOLOv8, and a more discreet 14% gain for YOLOv11. This shows that, in a wildlife data-scarce landscape where no bear annotated datasets could be found, taking advantage of a bigger, visually similar dataset should be regarded as a valid approach.

Both models showed similar scores on $mAP@50-95$ after hyperparameter optimization and fine-tuning, with only a 0.016 score difference between both models. Given that circumstance, the rest of the metrics were also considered in order

to choose the most appropriate model. YOLOv11 outperformed YOLOv8 in all other metrics; mAP@50 (+19% relative performance difference), precision (+22%) and recall (+16%). Consequently, YOLOv11 was chosen as the best performing model.

5.4.2 Bipedalism detection and Ecological relevance

The primary objective of this study was to evaluate the feasibility of using machine learning to infer specific bear behavior, such as bipedalism or quadrupedy position.

The trained MLP showed a very strong performance, with an accuracy of 97%, indicating that further improvement should focus on enhancing keypoint inference by the pose estimation model, rather than the classification stage

However, seven misclassifications occurred that were not due to incorrect keypoint inference (Figure 27). Most involved bears transitioning between quadrupedal and bipedal positions, while one case involved a bear sitting in front of a tree, a posture that was neither bipedalism (despite having the forelimbs being lifted) nor quadrupedy. These misclassifications were not straightforward, but rather edge cases. Yet, only receiving a binary response (either bipedalism or quadrupedy) or the raw predicted keypoint coordinates make it impossible to know how confident the model was in its prediction, and if a particular image falls within the “edge” cases category or not.

Therefore, visual representation through the proposed verification solutions (ASCII and geometric illustrations) provided clarity to each case, while avoiding the need to transmit full images, making them highly practical for remote ecological monitoring.

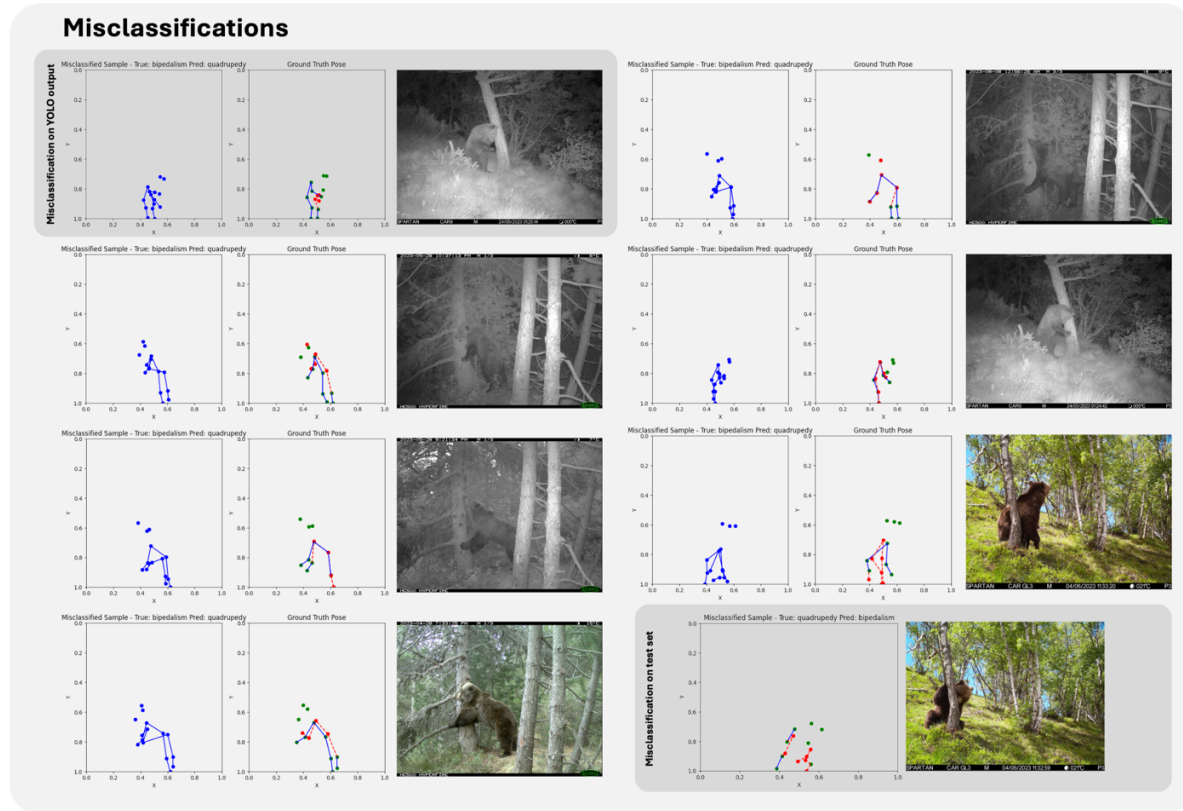


Figure 27. Misclassification examples. In each case, pose estimated by the model is shown on the left, ground truth annotations are shown in the middle, while original corresponding image is shown on the right. Model predictions on test and YOLO output had both 7 misclassifications. The first example was correctly classified by the model on the test set with ground truth annotations, while being misclassified on the pose estimation model predicted keypoints. Conversely, the last example was misclassified on the test set but correctly classified on the YOLO output. The six other examples were misclassified on both test and YOLO output.

To improve efficiency, this solution could also be integrated with wildlife recognition systems such as PyrVision (Campanera-Moliné, 2025) or DeepFaune (Rigoudy et al., 2023). In this configuration, the bipedalism detection pipeline would only be initialized in cases where a Brown Bear was detected.

Overall, this solution has practical implications to make hair snare monitoring much more scalable in remote or low-connectivity environments. Bipedalism detection can serve as a proxy for identifying potential hair snare interactions, under the assumption that a bear standing near a snare embedded in a tree is likely rubbing its back against the barbed wire. Furthermore, the proposed visualization options allow users to verify these interactions without needing to transmit the full-resolution image, saving costly bandwidth and reducing the need for frequent manual trap inspections. This could significantly optimize field operations by focusing efforts only on confirmed visited snares.

5.4.3 Limitations

While the final results show substantial improvement thanks to the intermediate training and hyperparameter tuning, the limited size of the dataset remains the main hindrance to achieving more reliable and generalizable inference. Deep learning neural networks typically require large and diverse data to achieve trustworthy results. The intrinsic elusiveness of the brown bear and the relatively small population in the Pyrenees made building a bigger dataset difficult and tedious.

Expanding the data set with camera trap images from similar biomes, such as the Alps, could help mitigate this limitation. However, it would require inter-regional collaboration between monitoring programs of both regions and a significant effort to manually label the new imagery. This limitation can also affect generalization to other regions or bear species outside the Pyrenees, as they may exhibit different behavior, have different body size, fur patterns, etc.

5.5 Conclusions

This study demonstrates that deep learning pose estimation models can be effectively adapted to wildlife monitoring tasks, specifically for inferring brown bear postures in the Pyrenees. This study contributed (1) a new dataset of bear images, each with 15 annotated anatomical keypoints designed specifically for bear pose analysis. (2) A YOLO pose estimation model capable of competently infer bear anatomical keypoints with 96% precision. (3) An MLP classifier for bipedalism and quadrupedy with a 97% accuracy. Finally, (4) three different visual solutions built from YOLO keypoint predictions for visual verification of the predicted posture.

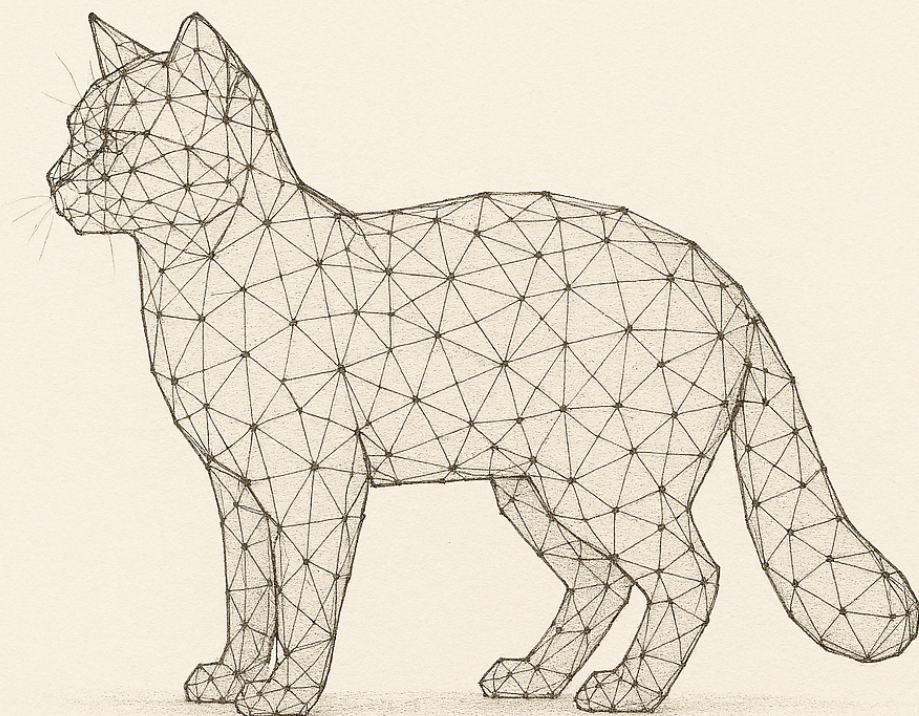
Hyperparameters optimization significantly improved model performance, leading to a 605% relative increase in mAP@50-95 score for YOLOv8 and 329% for YOLOv11, highlighting the importance of tailoring hyperparameters to the target dataset. Additionally, intermediate training on the Stanford Dog dataset provided additional performance improvements by another 20% for YOLOv8 and 14% for YOLOv11. These results highlight the value of transfer learning and domain-shifting for ecological applications, where a visually similar class can serve as a bridge between generalist and specialized models in small datasets contexts.

A custom MLP classifier achieved 97,35% accuracy distinguishing bipedalism from quadrupedy by using the relative keypoint positions, enabling automated inference of behaviors such as back-rubbing on hair snares.

Furthermore, a practical solution enabling verification of remote posture inference was introduced, using YOLO output and lightweight visualizations without the need to transmit full-resolution images. This innovative approach enables scalable deployment in low-bandwidth environments and significantly reduces the need for frequent manual inspections.

However, the limited dataset size remains a key constrain, affecting pose-estimation model performance which in turn hampers MLP capabilities. Future work should focus on expanding the dataset with similar biomes through inter-regional collaboration, as well as handle edge cases by adding more pose situations such as sitting or pose transitioning.

CHAPTER 6



6 General Discussion

6.1 General Discussion

Humanity is living through an ongoing sixth mass extinction, driven primarily by anthropogenic pressures. Dirzo et al., 2022 highlights that public and scientific perceptions often focus on species extinctions as the central problem. However, the most immediate threat lies in the rapid widespread collapse of local populations, even among common or “low concern” species, and the cascading ecological consequences of such declines (Ceballos et al., 2017). Detecting and understanding these processes requires precise and timely monitoring of wildlife populations, especially because shifts in abundance, behavior, or distribution may serve as early warning indicators of ecosystem degradation.

Fortunately, advances in sensor technology have dramatically improved our ability to record wildlife presence, movement, and behavior. Modern monitoring tools such as camera traps, acoustic recorders, and GPS devices are becoming increasingly accurate, affordable, and scalable (Pimm et al., 2015; T. Dalton et al., 2022). Yet, this technological progress has introduced a new challenge: the unprecedented volume of data produced. In particular, the rapid expansion of camera traps, favored for their low invasiveness, long deployment periods, and relative affordability, has made them one of the primary contributors to data overload in ecology.

The overarching aim of this thesis was to explore the potential of automated data processing by developing and enhancing computer-vision tools to address the growing burden of wildlife monitoring data. A secondary objective was to design methods that leverage emerging hardware, including consumer-grade 4G camera traps, to enable real-time wildlife monitoring. Understanding the strengths and limitations of both software and hardware solutions is essential for ecologists seeking to integrate machine learning effectively into monitoring workflows.

Chapter 1 laid the foundation by developing a custom camera-trap dataset of approximately 50,000 annotated images representing 20 classes found in the Pyrenees. Constructing this dataset was a slow but crucial step, enabling control over class balance, annotation accuracy, and subsequent field validation. Object detection has become a central challenge in computer vision, and YOLO. Currently one of the

most popular model families, was selected for systematic evaluation. Chapter 1 benchmarked several state-of-the-art YOLO variants on camera-trap imagery, establishing a detailed performance baseline. The results revealed that YOLO models, despite being designed for general-purpose computer vision tasks, perform remarkably well on ecological data, even under conditions of low resolution, motion blur, infrared nighttime images, and heterogeneous backgrounds. Interestingly, YOLOv9 emerged as the best-performing model, despite not being the newest iteration. More recent YOLO versions appear increasingly optimized for edge devices, prioritizing efficiency and speed over maximum accuracy. This trade-off may become important as edge computing becomes more widespread in field monitoring. For applications requiring the highest possible accuracy, older, more computationally expensive models still provide superior performance, reducing manual verification workloads. The identification of clear trade-offs among accuracy, speed, and computational cost provided the conceptual basis for subsequent chapters.

Building on this, Chapter 2 addressed one of the most persistent constraints in ecological machine learning: data scarcity. Although the dataset developed contains nearly 50,000 images, it is still considered small in the context of deep learning. Chapter 2 therefore evaluated strategies commonly used in computer vision, such as hyperparameter optimization and training domain-specific models, yet often unfamiliar to ecologists. The intention was to boost performance without collecting additional data. Surprisingly, the model trained on the complete mixed dataset outperformed models trained separately on day (color) or night (infrared-grayscale) subsets, even when evaluated on each domain's respective test sets. These unexpected results likely reflect the contribution of data augmentation, which may have supplied sufficient variability to offset the benefits of domain-specific training. The study also provided insights into the degree of redundancy present within the dataset. Overall, Chapter 2 demonstrated that with careful model design and parameter optimization, high-performing detection systems can be developed even under the data limitations typical of ecological studies.

With optimized models in place, Chapter 3 transitioned from offline analysis to real-time wildlife monitoring. By pairing the detection model with 4G-enabled camera

traps, a real-time alerting system was created, capable of issuing automatic notifications for target species. Chapter 3 evaluated how cellular signal intensity influenced transmission speed across four camera traps deployed in areas with differing signal conditions and assessed the suitability of four computing devices with vastly different processing capabilities. The results showed that object-detection models can be effectively deployed in operational field conditions, enabling near-real-time monitoring. This opens new opportunities for conservation practitioners, including early-warning systems for human–wildlife conflict and rapid detection of rare or invasive species. At the same time, Chapter 3 identified important practical constraints relating to network connectivity, bandwidth, and inference speeds on embedded devices. These findings point toward future research directions, including further optimization of detection models for edge devices and the integration of satellite communication technologies for fully remote deployments.

Finally, Chapter 4 extended the thesis beyond species detection to explore behavioral inference using pose estimation and multilayer perceptrons. The specific goal was to detect brown bear interactions with hair snares by identifying bipedalism postures associated with back-rubbing behavior. Working with a very small dataset of roughly 2,200 images collected in the Catalan Pyrenees, the study adapted a human pose-estimation model to track anatomical keypoints on bears and demonstrated the feasibility of fine-tuning pose models for wildlife applications. This work highlights how combining machine learning and deep learning approaches can extract complex behavioral signals from camera trap imagery. Importantly, the proposed method was designed with edge-device deployment in mind, enabling behavior inference in locations where image transmission is impractical or prohibitively expensive. The chapter also proposes practical solutions for validating model outputs in low-bandwidth contexts, underscoring the potential for AI to support behavioral ecology even in remote or data-limited environments.

Together, the chapters of this thesis illustrate a complete operational workflow for integrating artificial intelligence into wildlife monitoring, from dataset creation and model benchmarking to real-time field deployment and behavioral inference. The results highlight both the opportunities and the constraints associated with applying

computer vision to ecological data. More broadly, this work contributes to the rapidly growing field of ecological AI by demonstrating how carefully adapted deep learning tools can expand our capacity to monitor wildlife, interpret behavior, and support timely, data-driven conservation decision-making.

6.2 Contributions to Ecology and Conservation

The work carried out in this thesis contributes to ecology and conservation by demonstrating how the integration of machine learning into monitoring workflows can substantially reduce the burden of manually processing the vast quantities of data generated by camera traps and emerging ecological sensors. Beyond automation, this thesis introduces methods for acquiring and processing real-time ecological information and for extracting complex behavioral signals automatically, capabilities increasingly essential in modern biodiversity monitoring.

The benchmarking and optimization efforts presented in Chapters 1 and 2 provides practical guidance for ecologists interested in adopting AI-based detection systems. In particular, these chapters show that high-performing models can be trained even on relatively small, region-specific datasets, an important insight for studies of rare, cryptic or protected species where data scarcity is pervasive. As an additional contribution, Chapters 1 and 2 introduce PyrVision (Campanera-Moliné, 2025), a YOLO-based object-detection model capable of recognizing 20 classes, primarily the most commonly detected mammal species of the Catalan Pyrenees. This model constitutes a new resource for practitioners working in this region and a foundation for further methodological developments.

Building on PyrVision, Chapter 3 demonstrates the development of a real-time monitoring system with direct conservation relevance. By pairing 4G camera traps with the optimized detection model, the proposed pipeline can process incoming images instantaneously, tackling data-overload challenges while opening the door to a wide range of ecological applications. Chapter 3 also provides empirical guidance on the relationship between cellular signal intensity and image-transmission time, information of practical value for field practitioners planning remote deployments. Importantly, the real-time workflow presented in this chapter supports automated early-warning systems, enabling rapid responses such as alerting livestock owners to

predator presence or notifying authorities of potential poaching activity. The ability to stream near-instantaneous ecological information represents a shift from retrospective analysis toward proactive management and intervention.

Finally, Chapter 4 expands the contributions of AI beyond species detection into behavioral inference. Although the study focuses on distinguishing two postures, bipedalism versus quadrupedy, as a proxy for brown bear back-rubbing behavior at hair snare, the underlying methodological advances have broader relevance. Accurate prediction of anatomical keypoints and general body pose constitutes an important step for behavioral ecology, as behavior can provide essential insights into animal health, assessing stressors, evaluating the success of mitigation measures, and identifying interactions with human infrastructures. Demonstrating that pose estimation and behavioral inference can be achieved using standard camera-trap imagery, processed on edge devices in remote regions, highlights new possibilities for conservation monitoring, especially where bandwidth limitations prevent image transfer for human verification. The results show that ML systems can extract subtle behavioral indicators at scale, supporting more nuanced ecological inference.

6.3 Contributions to Computer Vision

Although the primary focus of this thesis has been the development of operational ecological applications, several chapters also contribute valuable insights to the field of computer vision. Chapter 2, in particular, sheds light on the nature of feature extraction in YOLO backbones and the type of information models rely on when trained on camera-trap imagery. The results suggest that color information provides more discriminative power than grayscale, even in cross-domain evaluations, highlighting the importance of spectral information for object detection in natural environments. These findings are relevant not only for ecologists seeking to maximize performance from limited datasets but also for broader computer-vision applications involving heterogeneous lighting or multi-domain data.

Across the thesis, the results also highlight model behavior under data-scarcity conditions, identifying factors that influence domain generalization, including dataset variability, redundancy, and differences in camera-trap placement (such as sensor

height, which modifies the perspective from which animals are captured). These insights help clarify why models trained on conventional benchmarks often fail when transferred to ecological datasets and illustrate how targeted design choices can mitigate these issues.

Chapter 4 contributes further by demonstrating the feasibility of applying pose-estimation models outside the human-centric datasets for which they were originally developed. Despite using a very small dataset of approximately 2,200 brown bear images, the YOLO pose-estimation model successfully learned the anatomical structure of a non-human species. Importantly, the chapter introduces an intermediate fine-tuning strategy: first adapting a human pose-estimation model to a keypoint-annotated dog dataset, and only then fine-tuning from human data to the limited bear dataset. This incremental transfer-learning strategy is broadly applicable and offers a promising solution for cases in which extremely small datasets limit model performance.

Overall, the contributions of this thesis to computer vision include (1) empirical evidence on how YOLO models generalise under real ecological constraints, (2) insights into the role of colour, redundancy, and perspective in feature extraction, and (3) a demonstration of how pose estimation architectures can be adapted to nonhuman species through biologically guided transfer learning. These advancements have relevance beyond ecology, offering methodological pathways for improving model performance in any domain where annotated data are scarce and domain shifts are substantial.

6.4 Strengths and limitations

The results of this thesis provide clear evidence of the potential for machine-learning techniques to address the substantial data-processing challenges created by modern, high-resolution ecological sensors. In particular, this work demonstrates the adaptability and performance of YOLO-based models in contexts characterized by data scarcity, a common constraint in ecological research. Their versatility across object detection, pose estimation, and other computer-vision tasks highlights YOLO as a strong candidate for a wide range of ecological applications.

Overall, the results show that AI models can achieve moderate to excellent performance even when trained on relatively small datasets, provided that appropriate optimization strategies are applied. The successful development of a real-time wildlife-detection and notification system further validates the methods developed in Chapters 1 and 2 under real-world conditions. Importantly, Chapter 3 demonstrates that even across sites with varying cellular signal intensities and computing capacities, a functional real-time monitoring system can be deployed effectively.

Chapter 4 extends these contributions by illustrating an innovative ecological application of computer vision: using pose estimation to infer bear-hair-snare interactions through the detection of bipedal postures. Automatically extracting behavioral indicators from anatomical keypoints enables behavioral monitoring to be scaled over larger areas, while significantly reducing the costs associated with manual inspection and field sampling.

Despite these strengths, several limitations emerged throughout the development of the thesis. Foremost among them is data scarcity, an inherent challenge in ecological research. However, this data scarcity was not shared across species. The Catalan forests have an important population of game species such as deer, wild boar or roe deer. Data availability for these species was significant. Yet, maintaining class balance requires limiting the growth of the dataset. Rarely detected species (e.g., badger, red squirrel, *Martes* genus) restricted the ability to expand the dataset without introducing strong imbalances. Data scarcity was also a significant constraint in Chapter 4, where only ~2,200 brown-bear images were available and needed to be distributed among training, validation and test subsets. Strategies such as transfer learning from a biologically similar species were necessary to mitigate these limitations.

Another challenge was the rapid pace of development in machine learning. Throughout the progression of this thesis, new YOLO versions and alternative architectures emerged quickly, occasionally rendering earlier analysis outdated. For context, the initial experiments for Chapter 1, conducted only four years ago, relied on YOLOv4 and YOLOv5 and included just five species. Keeping pace with evolving architectures required substantial re-analysis and repeated benchmarking.

Finally, a key limitation was the author's initial lack of formal training in computer science. The complexity of computer-vision concepts, coding practices, and

evaluation metrics required extensive effort to master, and undoubtedly extended the duration of research. Nonetheless, this steep learning curve ultimately contributed to a deeper interdisciplinary understanding.

6.5 Future Research directions

Several opportunities for future research emerge from this dissertation. Although the custom dataset developed here provided essential species context and contributed valuable annotated data to the local research community, it nonetheless suffers from data scarcity and limited variability, issues repeatedly highlighted across chapters.

One avenue for future work is to the dataset by increasing the number of images for under-represented species. For data-hungry deep-learning models, additional training samples typically lead to improved performance. However, as shown in Chapter 2, model improvement is influenced not only by dataset size but also by dataset variability. For species with high intra-class variability (e.g., domestic dogs, vehicles), increasing the diversity of backgrounds, seasons, weather conditions, and camera placements, rather than simply adding more redundant images, may better enhance generalization. Similarly, deploying camera traps at different heights or orientations could introduce valuable variability in perspective.

Another promising direction involves pre-training detection models on large, multi-regional camera-trap datasets rather than on COCO, which consists primarily of human-centric objects. Features learned from ecologically relevant datasets (e.g., other mammals, vegetation, and natural backgrounds) may transfer more effectively to new environments. With a sufficiently large and well-balanced dataset, pre-training on COCO or other generic datasets may no longer be necessary.

Further research is also needed to evaluate the viability of deploying edge-computing devices for real-time monitoring, especially in areas where cellular coverage is limited or absent. Technologies such as satellite communication modules or Low-Power Wide-Area Network (e.g., LoRaWAN) could provide alternatives to 4G-based systems. On the software side, this would require exploring more compact and energy-efficient model architectures and carefully managing the trade-offs among accuracy, inference speed, and power consumption. Advancing reliable edge-based inference could unlock numerous applications outlined in Chapter 3, including

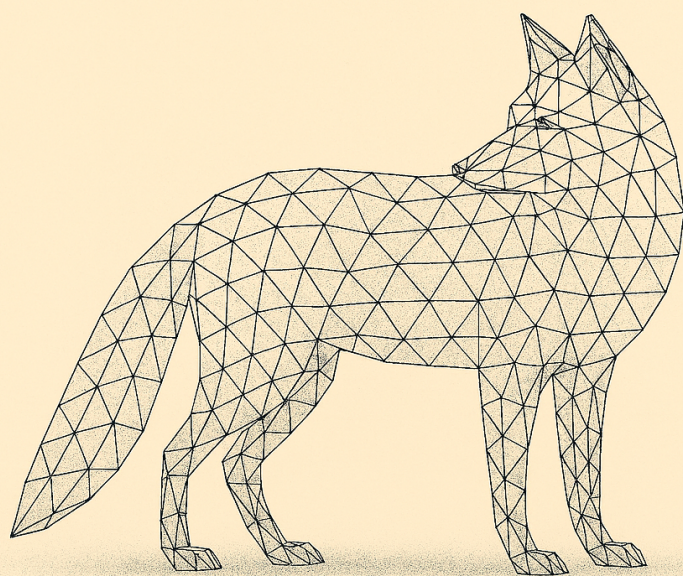
autonomous deterrence systems, automated alerts, and rapid-response wildlife monitoring.

Additionally, expanding the number of deployed 4G camera traps would allow for more robust analysis of the relationship between cellular signal strength and transmission times, producing more generalizable insights about performance in heterogeneous landscapes.

The methodology developed in Chapter 4 relies on the assumption that a bear standing bipedally near a hair snare is interacting with it. While this assumption aligns with field protocols, further research is needed to quantify the true correlation between posture and hair-snare contact. More dynamic behavioral assessment could be achieved by using image sequences or video data, although this would significantly increase computational requirements and energy use. As in earlier chapters, exploring lightweight versions of the pose-estimation models would be essential for enabling their deployment on edge devices.

Finally, pose-estimation methods could be expanded to additional species, including wolves or large carnivores, to classify a wider range of behaviors such as feeding, aggression, vigilance, or territorial marking. Reliable pose-based behavioral inference could deepen our understanding of ecological processes such as habitat use, reproductive cycles, seasonal movements, or predator-prey interactions.

CHAPTER 7



7 Conclusions

Conclusions

1. **Among the YOLO variants evaluated (YOLOv8–YOLOv12), the YOLOv9 “e” model consistently provided the highest detection performance on camera trap imagery.**

This finding indicates that newer architectures do not always outperform previous versions and underscores the importance of empirical benchmarking before selecting a model for ecological applications.

2. **Hyperparameter tuning proved to be an effective-though computationally demanding—strategy for improving model performance without collecting additional data.**

This demonstrates that careful optimisation of training procedures can substantially enhance accuracy even in data limited ecological contexts.

3. **Partitioning the dataset by image type (colour vs. infrared/grayscale) did not improve performance of domain specific models; instead, data variability emerged as a more influential factor than dataset size.**

These results suggest that ecological computer vision models benefit more from diverse visual environments than from strictly segregated domain specific training.

4. **A functional real time wildlife monitoring system can be built by integrating 4G wireless camera traps with deep learning detection models, which can run efficiently on edge computing devices.**

The system developed here achieved acceptable processing and reporting times, demonstrating the feasibility of near–real time ecological monitoring in operational field conditions.

5. **Cellular signal strength and network band type were found to significantly influence image transmission times in wireless 4G camera traps.**

This highlights the need to consider local connectivity conditions when deploying real time monitoring systems, particularly in remote landscapes.

6. **Human pose estimation models can be successfully adapted to nonhuman species such as brown bears, even under severe data scarcity constraints.**

Intermediate finetuning using a larger, visually similar dataset (e.g., dog key points) substantially improved model performance, offering a practical approach for other taxa with limited annotated data.

7. **Multilayer Perceptrons (MLPs) can reliably classify bipedal posture from detected key points, providing a viable proxy for identifying bear–hair snare interactions.**

This demonstrates that combining pose estimation outputs with lightweight machine learning classifiers can yield actionable behavioural insights, even from modest datasets.

REFERENCES

- Barnosky, A. D., Matzke, N., Tomiya, S., Wogan, G. O. U., Swartz, B., Quental, T. B., Marshall, C., McGuire, J. L., Lindsey, E. L., Maguire, K. C., Mersey, B., & Ferrer, E. A. (2011). Has the Earth's sixth mass extinction already arrived? In *Nature* (Vol. 471, Issue 7336, pp. 51–57). <https://doi.org/10.1038/nature09678>
- Beery, S., Morris, D., & Yang, S. (2019). *Efficient Pipeline for Camera Trap Image Review*.
- Beery, S., van Horn, G., & Perona, P. (2018). *Recognition in Terra Incognita*. <http://arxiv.org/abs/1807.04975>
- Benton, M. J. (1995). Diversification and Extinction in the History of Life. *Science*, 268. <https://www.science.org>
- Besson, M., Alison, J., Bjerge, K., Gorochowski, T. E., Høye, T. T., Jucker, T., Mann, H. M. R., & Clements, C. F. (2022). Towards the fully automated monitoring of ecological communities. *Ecology Letters*, 25(12), 2753–2775. <https://doi.org/10.1111/ele.14123>
- Biggs, B., Boyne, O., Charles, J., Fitzgibbon, A., & Cipolla, R. (2021). *Who Left the Dogs Out? 3D Animal Reconstruction with Expectation Maximization in the Loop*. <https://doi.org/10.1007/978-3-030-58621-8>
- Bischl, B., Binder, M., Lang, M., Pielok, T., Richter, J., Coors, S., Thomas, J., Ullmann, T., Becker, M., Boulesteix, A.-L., Deng, D., & Lindauer, M. (2021). *Hyperparameter Optimization: Foundations, Algorithms, Best Practices and Open Challenges*. <http://arxiv.org/abs/2107.05847>
- Bottou, L. (2010). Large-Scale Machine Learning with Stochastic Gradient Descent. In Y. Lechevallier & G. Saporta (Eds.), *Proceedings of COMPSTAT'2010* (pp. 177–186). Physica-Verlag HD.
- Brondizio, E. Sonnewend., Settele, Josef., Díaz, Sandra., & Ngo, H. Thu. (2019). *The global assessment report of the intergovernmental science-policy platform on biodiversity and ecosystem services*. Intergovernmental Science-Policy Platform on Biodiversity and Ecosystem Services (IPBES).

- Cahill, S., Llimona, F., Cabañeros, L., & Calomardo, F. (2012). Characteristics of wild boar (*Sus scrofa*) habituation to urban areas in the Collserola Natural Park (Barcelona) and comparison with other locations. *Animal Biodiversity and Conservation*, 35(2), 221–233. <https://doi.org/10.32800/abc.2012.35.0221>
- Campanera-Moliné, A. (2025). *PyrVision*. <https://github.com/ArnauCampanera/PyrVision>
- Ceballos, G., Ehrlich, P. R., & Dirzo, R. (2017). Biological annihilation via the ongoing sixth mass extinction signaled by vertebrate population losses and declines. *Proceedings of the National Academy of Sciences of the United States of America*, 114(30), E6089–E6096. <https://doi.org/10.1073/pnas.1704949114>
- Ceballos, G., Ehrlich, P. R., & Raven, P. H. (2020). Vertebrates on the brink as indicators of biological annihilation and the sixth mass extinction. *Proceedings of the National Academy of Sciences*, 117(24), 13596–13602. <https://doi.org/10.1073/pnas.1922686117>
- Chen, G., Han, T. X., He, Z., Kays, R., & Forrester, T. (2014a). Deep convolutional neural network based species recognition for wild animal monitoring. *2014 IEEE International Conference on Image Processing (ICIP)*, 858–862. <https://doi.org/10.1109/ICIP.2014.7025172>
- Chen, G., Han, T. X., He, Z., Kays, R., & Forrester, T. (2014b). Deep convolutional neural network based species recognition for wild animal monitoring. *2014 IEEE International Conference on Image Processing (ICIP)*, 858–862. <https://doi.org/10.1109/ICIP.2014.7025172>
- Chen, S., Sun, P., Song, Y., & Luo, P. (2023). *DiffusionDet: Diffusion Model for Object Detection*. <http://arxiv.org/abs/2211.09788>
- Choiński, M., Rogowski, M., Tynecki, P., Kuijper, D. P. J., Churski, M., & Bubnicki, J. W. (2021). A First Step Towards Automated Species Recognition from Camera Trap Images of Mammals Using AI in a European Temperate Forest. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 12883 LNCS, 299–310. https://doi.org/10.1007/978-3-030-84340-3_24

- Ciach, M., Tetkowski, P., & Fedyń, I. (2023). Local-scale habitat configuration makes a niche for wildlife encroaching into an urban landscape: grubbing sites of wild boar *Sus scrofa* in a city matrix. *Urban Ecosystems*, 26(3), 629–639. <https://doi.org/10.1007/s11252-022-01310-y>
- Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009). ImageNet: A Large-Scale Hierarchical Image Database. *2009 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2009*, 248–255. <https://doi.org/10.1109/CVPR.2009.5206848>
- Dirzo, R., Ceballos, G., & Ehrlich, P. R. (2022). Circling the drain: the extinction crisis and the future of humanity. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 377(1857). <https://doi.org/10.1098/rstb.2021.0378>
- Dornelas, M., Chow, C., Patchett, R., Breeze, T., Brotons, L., Beja, P., Carvalho, L., Jandt, U., Junker, J., Kissling, W. D., Kühn, I., Lumbierres, M., Lyche Solheim, A., Mjelde, M., Moreira, F., Musche, M., Pereira, H., Sandin, L., & Van Grunsven, R. (2023). *Deliverable 4.2 Novel technologies for biodiversity monitoring - Final Report*. <https://doi.org/10.3897/arphapreprints.e105600>
- Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., & Zisserman, A. (2010). The pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 88(2), 303–338. <https://doi.org/10.1007/s11263-009-0275-4>
- Fennell, M., Beirne, C., & Burton, A. C. (2022). Use of object detection in camera trap image identification: Assessing a method to rapidly and accurately classify human and animal detections for research and application in recreation ecology. *Global Ecology and Conservation*, 35. <https://doi.org/10.1016/j.gecco.2022.e02104>
- Fernández-Cabello, I., Franch, M., Vilella, M., Fernandez-Arrieta, N., Rota, M., Sanglas, A., Baqué-Díaz, E., Gallardet, M., Federico, P., Peris, A., Serratosa, E., Real, J., Sayol, F., & Puig-Gironès, R. (2025). Assessing the role of habitat, climate, and anthropization gradients on terrestrial mammal diversity in the western Mediterranean basin. *Integrative Zoology*, 20(3), 485–503. <https://doi.org/10.1111/1749-4877.12866>

- Feurer, M., & Hutter, F. (2019). *Hyperparameter Optimization* (pp. 3–33). https://doi.org/10.1007/978-3-030-05318-5_1
- GARDNER, B., ROYLE, J. A., WEGAN, M. T., RAINBOLT, R. E., & CURTIS, P. D. (2010). Estimating Black Bear Density Using DNA Data From Hair Snares. *The Journal of Wildlife Management*, 74(2), 318–325. <https://doi.org/10.2193/2009-101>
- Gomez, A., Salazar, A., & Vargas, F. (2016). *Towards Automatic Wild Animal Monitoring: Identification of Animal Species in Camera-trap Images using Very Deep Convolutional Neural Networks*. <http://arxiv.org/abs/1603.06169>
- Hahn, N. R., Bombaci, S. P., & Wittemyer, G. (2022). Identifying conservation technology needs, barriers, and opportunities. *Scientific Reports*, 12(1). <https://doi.org/10.1038/s41598-022-08330-w>
- Hampton, S. E., Strasser, C. A., Tewksbury, J. J., Gram, W. K., Budden, A. E., Batcheller, A. L., Duke, C. S., & Porter, J. H. (2013). Big data and the future of ecology. In *Frontiers in Ecology and the Environment* (Vol. 11, Issue 3, pp. 156–162). <https://doi.org/10.1890/120103>
- He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2018). *Mask R-CNN*. <http://arxiv.org/abs/1703.06870>
- Howe, E. J., Buckland, S. T., Després-Einspenner, M. L., & Köhl, H. S. (2017). Distance sampling with camera traps. *Methods in Ecology and Evolution*, 8(11), 1558–1565. <https://doi.org/10.1111/2041-210X.12790>
- Jocher, G., Chaurasia, A., & Qiu, J. (2023a). *Ultralytics YOLOv8*. <https://github.com/ultralytics/ultralytics>
- Jocher, G., Chaurasia, A., & Qiu, J. (2023b). *Ultralytics YOLOv8* (8.0.0). <https://github.com/ultralytics/ultralytics>
- Jocher, G., Chaurasia, A., & Qiu, J. (2023c). *YOLO by Ultralytics* (8.0.227).
- Jocher, G., Jing, Q., & Chaurasia, A. (2013). *Ultralytics YOLO* (8.0.0). <https://github.com/ultralytics/ultralytics>
- Jocher, G., & Qiu, J. (2024). *Ultralytics YOLO11* (11.0.0). <https://github.com/ultralytics/ultralytics>

- Joska, D., Clark, L., Muramatsu, N., Jericevich, R., Nicolls, F., Mathis, A., Mathis, M. W., & Patel, A. (2021). *AcinoSet: A 3D Pose Estimation Dataset and Baseline Models for Cheetahs in the Wild*. <https://doi.org/10.1109/ICRA48506.2021.9561338>
- Kahl, S., Wood, C. M., Eibl, M., & Klinck, H. (2021). BirdNET: A deep learning solution for avian diversity monitoring. *Ecological Informatics*, 61. <https://doi.org/10.1016/j.ecoinf.2021.101236>
- Keitt, T. H., & Abelson, E. S. (2021). Ecology in the age of automation. *Science*, 373(6557), 858–859. <https://doi.org/10.1126/science.abi4692>
- Khosla, A., Jayadevaprakash, N., Yao, B., & Li, F.-F. (2011, June). Novel Dataset for Fine-Grained Image Categorization: Stanford Dogs. *IEEE Conference on Computer Vision and Pattern Recognition*. <http://vision.stanford.edu/aditya86/StanfordDogs/>
- Kissling, W. D., Breeze, T. D., Liqueste, C., Lyche Solheim, A., McCallum, I., Maes, J., Hirsch, T., Lumbierres, M., van Grunsven, R. H. A., Beja, P., Smets, B., Capinha, C., Ceia-Hasse, A., Fernández, N., Moreira, F., Junker, J., Leese, F., Hammond, E., Brotons, L., ... Pereira, H. (2024). *Towards a modern and efficient European biodiversity observation network fit for multiple policies* (p. 16). <https://doi.org/10.32942/X2K34F>
- Kissling, W. D., Evans, J. C., Zilber, R., Breeze, T. D., Shinneman, S., Schneider, L. C., Chalmers, C., Fergus, P., Wich, S., & Geelen, L. H. W. T. (2024). Development of a cost-efficient automated wildlife camera network in a European Natura 2000 site. *Basic and Applied Ecology*, 79, 141–152. <https://doi.org/10.1016/j.baae.2024.06.006>
- Kolesnikov, A., Beyer, L., Zhai, X., Puigcerver, J., Yung, J., Gelly, S., & Houlsby, N. (2020). *Big Transfer (BiT): General Visual Representation Learning*. <http://arxiv.org/abs/1912.11370>
- Kristo, M., Ivacic-Kos, M., & Pobar, M. (2020). Thermal Object Detection in Difficult Weather Conditions Using YOLO. *IEEE Access*, 8, 125459–125476. <https://doi.org/10.1109/ACCESS.2020.3007481>

- Kuznetsova, A., Rom, H., Alldrin, N., Uijlings, J., Krasin, I., Pont-Tuset, J., Kamali, S., Popov, S., Mallocci, M., Kolesnikov, A., Duerig, T., & Ferrari, V. (2020). The Open Images Dataset V4: Unified Image Classification, Object Detection, and Visual Relationship Detection at Scale. *International Journal of Computer Vision*, 128(7), 1956–1981. <https://doi.org/10.1007/s11263-020-01316-z>
- Larsen, B., Costa, A., Hall, M., McGinnis-Brown, L., & Fry, V. (2020). *Human Wildlife Conflict Monitoring: Understanding Human Human Wildlife Conflict Through Big Data*. https://scholarworks.boisestate.edu/ipi_reports/30
- Laughlin, L. A., Freeman, H. M., Blevins, C. A., Depuy, V. E., Gatewood, A., MacKenzie, B., Ranallo, N., McQuaig, J., Davis, C. A., Goodman, L. E., Fuhlendorf, S. D., Andersson, K., Duchardt, C. J., & Chitwood, M. C. (2023). Assessing efficacy of cellular transmission technology in camera trapping for wildlife research. *Wildlife Society Bulletin*, 47(4). <https://doi.org/10.1002/wsb.1491>
- Lei, M., Li, S., Wu, Y., Hu, H., Zhou, Y., Zheng, X., Ding, G., Du, S., Wu, Z., & Gao, Y. (2025). *YOLOv13: Real-Time Object Detection with Hypergraph-Enhanced Adaptive Visual Perception*. <http://arxiv.org/abs/2506.17733>
- Lei, X., Wu, S., Wu, W., & Jiang, Z. (2025). *MambaNeXt-YOLO: A Hybrid State Space Model for Real-time Object Detection*. <http://arxiv.org/abs/2506.03654>
- Leorna, S., & Brinkman, T. (2022). Human vs. machine: Detecting wildlife in camera trap images. *Ecological Informatics*, 72. <https://doi.org/10.1016/j.ecoinf.2022.101876>
- Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2018). *Focal Loss for Dense Object Detection*. <http://arxiv.org/abs/1708.02002>
- Lin, T.-Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C. L., & Dollár, P. (2014). *Microsoft COCO: Common Objects in Context*. <http://arxiv.org/abs/1405.0312>
- Loshchilov, I., & Hutter, F. (2019). *Decoupled Weight Decay Regularization*. <http://arxiv.org/abs/1711.05101>

- Lv, W., Zhao, Y., Chang, Q., Huang, K., Wang, G., & Liu, Y. (2024). *RT-DETRv2: Improved Baseline with Bag-of-Freebies for Real-Time Detection Transformer*. <http://arxiv.org/abs/2407.17140>
- MacKenzie, D. I. . (2018). *Occupancy estimation and modeling : inferring patterns and dynamics of species occurrence*. Academic Press.
- Maghsoudi, O. H., Tabrizi, A. V., Robertson, B., & Spence, A. (2017). Superpixels based marker tracking vs. hue thresholding in rodent biomechanics application. *Conference Record of 51st Asilomar Conference on Signals, Systems and Computers, ACSSC 2017, 2017-October*, 209–213. <https://doi.org/10.1109/ACSSC.2017.8335168>
- Marin, C., Couderchet, L., Le Champion, G., & Werno, J. (2024). Wildlife and the city. Modelling wild boar use of urban nature: Empirical contribution, methodological proposal: Bordeaux (France) as an example. *Urban Ecosystems*, 27(4), 1291–1312. <https://doi.org/10.1007/s11252-024-01510-8>
- Martin, J., Revilla, E., Quenette, P. Y., Naves, J., Allainé, D., & Swenson, J. E. (2012). Brown bear habitat suitability in the Pyrenees: Transferability across sites and linking scales to make the most of scarce data. *Journal of Applied Ecology*, 49(3), 621–631. <https://doi.org/10.1111/j.1365-2664.2012.02139.x>
- Mathis, A., Biasi, T., Schneider, S., Yüsekşönül, M., Rogers, B., Bethge, M., & Mathis, M. W. (2020). *Pretraining boosts out-of-domain robustness for pose estimation*. <http://arxiv.org/abs/1909.11229>
- Mathis, A., Mamidanna, P., Cury, K. M., Abe, T., Murthy, V. N., Mathis, M. W., & Bethge, M. (2018). DeepLabCut: markerless pose estimation of user-defined body parts with deep learning. *Nature Neuroscience*, 21(9), 1281–1289. <https://doi.org/10.1038/s41593-018-0209-y>
- Matthews, A., Ruykys, L., Ellis, B., Fitzgibbon, S., Lunney, D., Crowther, M. S., Glen, A. S., Purcell, B., Moseby, K., Stott, J., Fletcher, D., Wimpenny, C., Allen, B. L., Van Bommel, L., Roberts, M., Davies, N., Green, K., Newsome, T., Ballard, G., ... Wiggins, N. (2013). The success of GPS collar deployments on mammals in Australia. *Australian Mammalogy*, 35(1), 65–83. <https://doi.org/10.1071/AM12021>

- McIvor, A. J., Williams, C. T., Rich, W. A., Knochel, A. M., Burns, N. M., & Berumen, M. L. (2024). Mark-recapture validates the use of photo-identification for the widely distributed blue-spotted ribbontail ray, *Taeniura lymma*. *Scientific Reports*, 14(1). <https://doi.org/10.1038/s41598-024-68302-0>
- Merkel, D. (2014). Docker: Lightweight Linux Containers for Consistent Development and Deployment. *Linux J.*, 2014(239).
- Milleret, C., Bischof, R., Dupont, P., Brøseth, H., Odden, J., & Mattisson, J. (2021). GPS collars have an apparent positive effect on the survival of a large carnivore. *Biology Letters*, 17(6). <https://doi.org/10.1098/rsbl.2021.0128>
- Mowat, G., & Strobeck, C. (2000). Estimating Population Size of Grizzly Bears Using Hair Capture, DNA Profiling, and Mark-Recapture Analysis. In *Source: The Journal of Wildlife Management* (Vol. 64, Issue 1). <https://www.jstor.org/stable/3802989>
- Nandutu, I., Atemkeng, M., & Okouma, P. (2022). Intelligent Systems Using Sensors and/or Machine Learning to Mitigate Wildlife–Vehicle Collisions: A Review, Challenges, and New Perspectives. In *Sensors* (Vol. 22, Issue 7). MDPI. <https://doi.org/10.3390/s22072478>
- Niedballa, J., Sollmann, R., Mohamed, A. Bin, Bender, J., & Wilting, A. (2015). Defining habitat covariates in camera-trap based occupancy studies. *Scientific Reports*, 5. <https://doi.org/10.1038/srep17041>
- Nimma, D., Dandu, V., Bisht, A. S., Rao, V. S., Kumar, P. S., & Muthuperumal, S. (2024). High-Accuracy Animal Species Classification Using Transfer Learning with ResNet50 on Wildlife Camera Trap Images for Enhanced Conservation Efforts. *2nd IEEE International Conference on Innovations in High-Speed Communication and Signal Processing, IHCSPP 2024*. <https://doi.org/10.1109/IHCSPP63227.2024.10960173>
- Norouzzadeh, M. S., Nguyen, A., Kosmala, M., Swanson, A., Palmer, M. S., Packer, C., & Clune, J. (2018). Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning. *Proceedings of the National Academy of Sciences of the United States of America*, 115(25), E5716–E5725. <https://doi.org/10.1073/pnas.1719367115>

- Oliver R. Wearn, & Paul Glover-Kapfer. (2017). Camera-trapping for conservation: a guide to best-practices. In *WWF Conservation Technology Series 1* (Vol. 1).
- Palazón, S., Espinós, N., & Casanovas, R. (2023). *OS BRU 2022 Informe anual del seguiment i conservació de l'os bru i el llop a Catalunya*. https://mediambient.gencat.cat/web/.content/home/ambits_dactuacio/patrimon_i_natural/fauna_salvatge_autoctona/gestio-especies/mamifers/ursus_arctos_os_bru/Report-2022-os-bru_llop.pdf
- Pardo, L. E., Bombaci, S., Huebner, S. E., Somers, M. J., Fritz, H., Downs, C., Guthmann, A., Hetem, R. S., Keith, M., le Roux, A., Mqatsa, N., Packer, C., Palmer, M. S., Parker, D. M., Peel, M., Slotow, R., Maartin Strauss, W., Swanepoel, L., Tambling, C., ... Venter, J. A. (2021). Snapshot Safari: A large-scale collaborative to monitor Africa's remarkable biodiversity. *South African Journal of Science*, 117(1–2). <https://doi.org/10.17159/SAJS.2021/8134>
- Perry, G. L. W., Seidl, R., Bellvé, A. M., & Rammer, W. (2022). An Outlook for Deep Learning in Ecosystem Science. *Ecosystems*, 25(8), 1700–1718. <https://doi.org/10.1007/s10021-022-00789-y>
- Petkov, V., Rutishauser, M., Boice, J., Obraczka, K., Mantey, P., Williams, T. M., & Wilmers, C. C. (2011). CARNIVORE: A disruption-tolerant system for studying wildlife. *Eurasip Journal on Wireless Communications and Networking*, 2011. <https://doi.org/10.1155/2011/968046>
- Pichler, M., & Hartig, F. (2023). Machine learning and deep learning—A review for ecologists. In *Methods in Ecology and Evolution*. British Ecological Society. <https://doi.org/10.1111/2041-210X.14061>
- Pimm, S. L., Alibhai, S., Bergl, R., Dehgan, A., Giri, C., Jewell, Z., Joppa, L., Kays, R., & Loarie, S. (2015). Emerging Technologies to Conserve Biodiversity. In *Trends in Ecology and Evolution* (Vol. 30, Issue 11, pp. 685–696). Elsevier Ltd. <https://doi.org/10.1016/j.tree.2015.08.008>
- Pimm, S. L., Jenkins, C. N., Abell, R., Brooks, T. M., Gittleman, J. L., Joppa, L. N., Raven, P. H., Roberts, C. M., & Sexton, J. O. (2014). The biodiversity of species and their rates of extinction, distribution, and protection. In *Science* (Vol. 344, Issue 6187). American Association for the Advancement of Science. <https://doi.org/10.1126/science.1246752>

- Plank, B. (2022). *The “Problem” of Human Label Variation: On Ground Truth in Data, Modeling and Evaluation*. <http://arxiv.org/abs/2211.02570>
- Proença, V., Martin, L. J., Pereira, H. M., Fernandez, M., McRae, L., Belnap, J., Böhm, M., Brummitt, N., García-Moreno, J., Gregory, R. D., Honrado, J. P., Jürgens, N., Opige, M., Schmeller, D. S., Tiago, P., & van Swaay, C. A. M. (2017). Global biodiversity monitoring: From data sources to Essential Biodiversity Variables. *Biological Conservation*, 213, 256–263. <https://doi.org/10.1016/j.biocon.2016.07.014>
- Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). *You Only Look Once: Unified, Real-Time Object Detection*. <http://arxiv.org/abs/1506.02640>
- Ren, S., He, K., Girshick, R., & Sun, J. (2016). *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*. <http://arxiv.org/abs/1506.01497>
- Rigoudy, N., Dussert, G., Benyoub, A., Besnard, A., Birck, C., Boyer, J., Bollet, Y., Bunz, Y., Caussimont, G., Chetouane, E., Carriburu, J. C., Cornette, P., Delestrade, A., De Backer, N., Dispan, L., Le Barh, M., Duhayer, J., Elder, J. F., Fanjul, J. B., ... Chamaillé-Jammes, S. (2023). The DeepFaune initiative: a collaborative effort towards the automatic identification of European fauna in camera trap images. *European Journal of Wildlife Research*, 69(6). <https://doi.org/10.1007/s10344-023-01742-7>
- Roy, A. M., Bhaduri, J., Kumar, T., & Raj, K. (2023). WilDect-YOLO: An efficient and robust computer vision-based accurate object localization model for automated endangered wildlife detection. *Ecological Informatics*, 75, 101919. <https://doi.org/10.1016/j.ecoinf.2022.101919>
- Saravanan, S., Raj, J. R. F., Rajkumar, G. V., Kumar, C. A. V., Soundiraraj, N., & Bharathi, K. (2024). Deploying Faster R-CNN for Real-Time Tiger Monitoring in Wildlife Conservation. *3rd International Conference on Automation, Computing and Renewable Systems, ICACRS 2024 - Proceedings*, 1713–1720. <https://doi.org/10.1109/ICACRS62842.2024.10841726>
- Sato, Y., Nakamura, H., Kyoko, K., Sekiguchi, M., Ishibashi, Y., & Itoh, T. (2020). Evaluation of the Effectiveness of Scented Wooden Posts for DNA Hair

- Snagging of Brown Bears. *Mammal Study*, 45(3), 213–218.
<https://doi.org/10.3106/ms2018-0045>
- Schneider, S., Greenberg, S., Taylor, G. W., & Kremer, S. C. (2020). Three critical factors affecting automated image species recognition performance for camera traps. *Ecology and Evolution*, 10(7), 3503–3517.
<https://doi.org/10.1002/ece3.6147>
- Sekachev, B., Manovich, N., Zhiltsov, M., Zhavoronkov, A., Kalinin, D., Hoff, B., TOsmanov, Kruchinin, D., Zankevich, A., DmitriySidnev, Markelov, M., Johannes222, Chenuet, M., a-andre, telenachos, Melnikov, A., Kim, J., Ilouz, L., Glazov, N., ... Truong, T. (2020). *Computer Vision Annotation Tool (CVAT)*. Zenodo. <https://doi.org/10.5281/zenodo.4009388>
- Sellers, W. I., & Hirasaki, E. (2014). Markerless 3D motion capture for animal locomotion studies. *Biology Open*, 3(7), 656–668.
<https://doi.org/10.1242/bio.20148086>
- Sentilles, J., Lemaitre, P.-L., Vanpe, C., & Quenette, P.-Y. (2024). *Rapport annuel du Réseau Ours Brun 2023*.
- Simonthomas, S., Rohith, K., & Shalini, K. S. (2024). Machine Learning based Wild-Animal Detection near Roads using IoT Sensor and Optimization. *2024 International Conference on Intelligent Systems and Advanced Applications, ICISAA 2024*. <https://doi.org/10.1109/ICISAA62385.2024.10828816>
- Singh, M., Negi, S. K., Chandra, P. K., & Sharma, S. (2025). Innovative AI Solutions with Animal Detection for Road Safety and Preventing Wildlife Collisions. *ESIC 2025 - 5th International Conference on Emerging Systems and Intelligent Computing, Proceedings*, 888–893.
<https://doi.org/10.1109/ESIC64052.2025.10962657>
- Smith, J., Wycherley, A., Mulvaney, J., Lennane, N., Reynolds, E., Monks, C. A., Evans, T., Mooney, T., & Fancourt, B. (2024). Man versus machine: cost and carbon emission savings of 4G-connected Artificial Intelligence technology for classifying species in camera trap images. *Scientific Reports*, 14(1).
<https://doi.org/10.1038/s41598-024-65179-x>

- Stabach, J. A., Cunningham, S. A., Connette, G., Mota, J. L., Reed, D., Byron, M., Songer, M., Wachter, T., Mertes, K., Brown, J. L., Comizzoli, P., Newby, J., Monfort, S., & Leimgruber, P. (2020). Short-term effects of GPS collars on the activity, behavior, and adrenal response of scimitar-horned oryx (*Oryx dammah*). *PLoS ONE*, *15*(2). <https://doi.org/10.1371/journal.pone.0221843>
- Steenweg, R., Hebblewhite, M., Kays, R., Ahumada, J., Fisher, J. T., Burton, C., Townsend, S. E., Carbone, C., Rowcliffe, J. M., Whittington, J., Brodie, J., Royle, J. A., Switalski, A., Clevenger, A. P., Heim, N., & Rich, L. N. (2017). Scaling-up camera traps: monitoring the planet's biodiversity with networks of remote sensors. In *Frontiers in Ecology and the Environment* (Vol. 15, Issue 1, pp. 26–34). Wiley Blackwell. <https://doi.org/10.1002/fee.1448>
- Stephenson, P. J. (2020). Technological advances in biodiversity monitoring: applicability, opportunities and challenges. In *Current Opinion in Environmental Sustainability* (Vol. 45, pp. 36–41). Elsevier B.V. <https://doi.org/10.1016/j.cosust.2020.08.005>
- Strampelli, P., Campbell, L. A. D., Henschel, P., Nicholson, S. K., Macdonald, D. W., & Dickman, A. J. (2022). Trends and biases in African large carnivore population assessments: identifying priorities and opportunities from a systematic review of two decades of research. *PeerJ*, *10*. <https://doi.org/10.7717/PEERJ.14354>
- Sushila Devi, S. V., & Sowmya, V. J. (2025). Automated Wildlife Identification Using CNNs and Transfer Learning Techniques. *2025 6th International Conference on Data Intelligence and Cognitive Informatics, ICDICI 2025*, 758–763. <https://doi.org/10.1109/ICDICI66477.2025.11134976>
- Sutherland, W. J., Pullin, A. S., Dolman, P. M., & Knight, T. M. (2004). The need for evidence-based conservation. *Trends in Ecology and Evolution*, *19*(6), 305–308. <https://doi.org/10.1016/j.tree.2004.03.018>
- Swanson, A., Kosmala, M., Lintott, C., Simpson, R., Smith, A., & Packer, C. (2015). Snapshot Serengeti, high-frequency annotated camera trap images of 40 mammalian species in an African savanna. *Scientific Data*, *2*(1), 150026. <https://doi.org/10.1038/sdata.2015.26>

- T. Dalton, D., Pascher, K., Berger, V., Steinbauer, K., & Jungmeier, M. (2022). Novel Technologies and Their Application for Protected Area Management: A Supporting Approach in Biodiversity Monitoring. In *Protected Area Management - Recent Advances*. IntechOpen. <https://doi.org/10.5772/intechopen.99889>
- Tabak, M. A., Falbel, D., Hamzeh, T., Brook, R. K., Goolsby, J. A., Zoromski, L. D., Boughton, R. K., Snow, N. P., Vercauteren, K. C., & Miller, R. S. (2022). *CameraTrapDetector: Automatically detect, classify, and count animals in camera trap images using artificial intelligence*. <https://doi.org/10.1101/2022.02.07.479461>
- Tabak, M. A., Norouzzadeh, M. S., Wolfson, D. W., Sweeney, S. J., Vercauteren, K. C., Snow, N. P., Halseth, J. M., Di Salvo, P. A., Lewis, J. S., White, M. D., Teton, B., Beasley, J. C., Schlichting, P. E., Boughton, R. K., Wight, B., Newkirk, E. S., Ivan, J. S., Odell, E. A., Brook, R. K., ... Miller, R. S. (2019). Machine learning to classify animal species in camera trap images: Applications in ecology. *Methods in Ecology and Evolution*, 10(4), 585–590. <https://doi.org/10.1111/2041-210X.13120>
- Thompson, T. (2023). How AI can help to save endangered species. *Nature*, 623(7986), 232–233. <https://doi.org/10.1038/d41586-023-03328-4>
- Tian, Y., Ye, Q., & Doermann, D. (2025). *YOLOv12: Attention-Centric Real-Time Object Detectors Latency (ms) MS COCO mAP (%)*. <https://doi.org/10.0>
- Tishby, N., & Zaslavsky, N. (2015). Deep learning and the information bottleneck principle. *2015 IEEE Information Theory Workshop (ITW)*, 1–5. <https://doi.org/10.1109/ITW.2015.7133169>
- Treves, A. (2009). Hunting for large carnivore conservation. In *Journal of Applied Ecology* (Vol. 46, Issue 6, pp. 1350–1356). <https://doi.org/10.1111/j.1365-2664.2009.01729.x>
- Tuia, D., Kellenberger, B., Beery, S., Costelloe, B. R., Zuffi, S., Risse, B., Mathis, A., Mathis, M. W., van Langevelde, F., Burghardt, T., Kays, R., Klinck, H., Wikelski, M., Couzin, I. D., van Horn, G., Crofoot, M. C., Stewart, C. V., & Berger-Wolf, T. (2022). Perspectives in machine learning for wildlife

- conservation. In *Nature Communications* (Vol. 13, Issue 1). Nature Research. <https://doi.org/10.1038/s41467-022-27980-y>
- Van Horn, G., Mac Aodha, O., Song, Y., Cui, Y., Sun, C., Shepard, A., Adam, H., Perona, P., & Belongie, S. (2018). *The iNaturalist Species Classification and Detection Dataset*. <http://arxiv.org/abs/1707.06642>
- Van Horn, G., & Perona, P. (2017). *The Devil is in the Tails: Fine-grained Classification in the Wild*. <http://arxiv.org/abs/1709.01450>
- Wall, J., Wittemyer, G., Klinkenberg, B., & Douglas-Hamilton, I. (2014). Novel opportunities for wildlife conservation and research with real-time monitoring. *Ecological Applications*, 24(4), 593–601. <https://doi.org/10.1890/13-1971.1>
- Wang, A., Chen, H., Liu, L., Chen, K., Lin, Z., Han, J., & Ding, G. (2024). *YOLOv10: Real-Time End-to-End Object Detection*. <http://arxiv.org/abs/2405.14458>
- Wang, C.-Y., Yeh, I.-H., & Liao, H.-Y. M. (2024). *YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information*. <http://arxiv.org/abs/2402.13616>
- Whytock, R. C., Suijten, T., van Deursen, T., Świeżewski, J., Mermiaghe, H., Madamba, N., Mouckoumou, N., Zwerts, J. A., Pambo, A. F. K., Bahaa-el-din, L., Brittain, S., Cardoso, A. W., Henschel, P., Lehmann, D., Momboua, B. R., Makaga, L., Orbell, C., White, L. J. T., Iponga, D. M., & Abernethy, K. A. (2023). 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*, 14(3), 867–874. <https://doi.org/10.1111/2041-210X.14036>
- Yang, D. Q., Meng, D. Y., Li, H. X., Li, M. T., Jiang, H. L., Tan, K., Huang, Z. P., Li, N., Wu, R. H., Li, X. W., Chen, B. H., Zhang, M., Ren, G. P., & Xiao, W. (2024). A systematic study on transfer learning: Automatically identifying empty camera trap images using deep convolutional neural networks. *Ecological Informatics*, 80. <https://doi.org/10.1016/j.ecoinf.2024.102527>
- Zheng, C., Wu, W., Chen, C., Yang, T., Zhu, S., Shen, J., Kehtarnavaz, N., & Shah, M. (2023). *Deep Learning-Based Human Pose Estimation: A Survey*.

Zuolkernan, I., Dhou, S., Judas, J., Sajun, A. R., Gomez, B. R., & Hussain, L. A. (2022). An IoT System Using Deep Learning to Classify Camera Trap Images on the Edge. *Computers, 11*(1). <https://doi.org/10.3390/computers11010013>

