



**Automated Wildlife Species Classification Using Deep Neural Networks**

Prepared By

Name: [Ahmad Mohammed Bashatah](#)

ID No.: [243001738](#)

Submitted in fulfillment of the requirements for the degree Master of  
Science in Artificial Intelligence

At

College of Informatics Midocean University

UAE

Supervisor Name: [Dr. Ashraf Said](#)

College of Informatics, Midocean University, Comoros

Email: [243001738@st.midocean.edu.km](mailto:243001738@st.midocean.edu.km)

# Acknowledgements

In the name of Allah, who illuminated my paths with the light of knowledge and aided me in crossing the seas of effort and patience. To my soul, which wandered in the horizon of ambition, stood firm against the storms of challenges, danced with the nights of determination, and completed this journey with a resolve as radiant as dawn.

To my father, the lighthouse that lit my path with lessons of resilience and taught me to persevere on the road to success, and to my mother, the source of my inspiration and strength, the spirit who sang me anthems of hope with a tender voice that soothes the heart. You are my foundation, and this achievement is but a bouquet of roses offered to your flourishing garden—accept it from your son with gratitude and loyalty.

To my brothers and sisters, the branches of my tree shaded me with support and love. You are the stars of my journey, my joy, my melodies, and my happiness. Thank you for every smile that revived me, and for every hand you extended to me; your stars never fade, and I take pride in you.

To my beloved wife, the sea in which I sailed my dreams, and the shore that welcomes me with safety after every storm. I appreciate your patience that eased my fatigue, and for bearing the burdens while I was preoccupied. You are the light of my home, and your support was the wind that propelled me toward success; this achievement is a poem I dedicate to you with endless love.

Finally, to my children, my three roses that bloomed in my small garden. To Muhammad, my eldest son, thank you for your wisdom that guided me like a compass; to Rudayna, my precious daughter, thank you for your smile that shines like the sun in my sky; and to Youssef, my little son, thank you for your energy that flows like a river overcoming rocks. You are my motivation to reach the peaks, and I hope to be a mountain that protects you. This achievement is a shield to guard you and a future I will build for you.

Thank you to all who contributed to this new dawn; without you, I would have been a tree without roots, but today I stand as a fruitful tree, nourished by your love and support.

# Abstract

Biodiversity is declining at an alarming rate, making the accurate identification of wildlife species an essential task for conservation work. Traditional identification methods often depend on human experts, which is slow and not always practical in field conditions. Deep learning offers an alternative that can provide fast and reliable results, even with limited resources.

This thesis develops a reproducible benchmarking framework to evaluate several deep learning models for the classification of four visually similar species: *Plestiodon inexpectatus*, *Agkistrodon contortrix*, *Caretta caretta*, and *Oreaster reticulatus*. The study compares a range of models, including ResNet18, MobileNetV3-Large, EfficientNet-B0, InceptionV3, ViT-B/32, Swin Transformer, YOLOv11n-cls, and a simple CNN baseline.

The results show that the Swin Transformer achieved the best overall performance, with an accuracy of 98.25%, followed closely by InceptionV3 at 97.66%. MobileNetV3-Large and EfficientNet-B0 provided strong results of 95.9% while remaining efficient for deployment. ResNet18 also performed well at 95.32%. YOLO-Classification achieved 94.15% accuracy, while ViT-B/32 and the CNN baseline were weaker due to the limited dataset.

Beyond overall accuracy, the study also reports weighted Precision, Recall, and F1 scores, per-class Precision–Recall curves, macro-mAP values, and confusion matrices. These results show not only which models perform best, but also where errors are concentrated and how they could be reduced. The framework provides conservation researchers with clear evidence and practical recommendations for selecting models under small-data constraints.

# Table of Contents

|                                                      |          |
|------------------------------------------------------|----------|
| Acknowledgements.....                                | 2        |
| Abstract.....                                        | 3        |
| <b>List of Abbreviations.....</b>                    | <b>5</b> |
| <b>List of Figures.....</b>                          | <b>6</b> |
| <b>List of Tables.....</b>                           | <b>7</b> |
| Chapter 1. Introduction .....                        | 8        |
| 1.1 Thesis Objectives.....                           | 10       |
| 1.2 Thesis Contribution.....                         | 11       |
| Chapter 2 Overview .....                             | 15       |
| Chapter 3 Related Work.....                          | 19       |
| 3.1 Introduction .....                               | 19       |
| 3.2 History .....                                    | 22       |
| 3.3 Summary.....                                     | 28       |
| Chapter 4 Proposed Approach .....                    | 30       |
| 4.1 Dataset.....                                     | 30       |
| 4.2 Preprocessing and Augmentation .....             | 31       |
| 4.2.2 Normalization.....                             | 32       |
| 4.2.3 Data Augmentation.....                         | 33       |
| 4.3 Models and Training Settings.....                | 35       |
| 4.3.1 Training and Validation Loop .....             | 35       |
| 4.3.2 Loss Function .....                            | 36       |
| 4.3.3 Optimizer.....                                 | 37       |
| 4.3.4 Learning Rate Scheduler .....                  | 37       |
| 4.3.5 Epochs and Batch Size .....                    | 38       |
| 4.3.6 Model-Specific Adjustments .....               | 38       |
| 4.3.7 Hyperparameter Exploration .....               | 39       |
| 4.4 Evaluation Metrics and Procedure .....           | 39       |
| 4.5 Implementation and Reproducibility.....          | 40       |
| 4.6 Design Rationale and Trade-offs .....            | 40       |
| Chapter 5 Experimental Results and Discussion.....   | 42       |
| 5.1 Overall Comparison .....                         | 43       |
| 5.2 Precision–Recall curves & confusion matrix ..... | 46       |
| 5.3 Per-class metrics analysis .....                 | 52       |
| 5.4 Practical Implications .....                     | 53       |
| Chapter 6 Conclusion and Future Work.....            | 55       |
| References .....                                     | 60       |

# List of Abbreviations

**AI** — Artificial Intelligence  
**AP** — Average Precision  
**AUC-PR** — Area Under the Precision–Recall Curve  
**DL** — Deep Learning  
**DNN** — Deep Neural Network  
**EffNet-B0** — EfficientNet-B0  
**F1** — F1-score (harmonic mean of Precision and Recall)  
**FN** — False Negative  
**FP** — False Positive  
**GFLOPs** — Giga Floating-Point Operations (computational cost)  
**GPU** — Graphics Processing Unit  
**HPO** — Hyperparameter Optimization  
**IncV3** — InceptionV3  
**kappa ( $\kappa$ )** — Cohen’s Kappa Statistic  
**LR** — Learning Rate  
**MCC** — Matthews Correlation Coefficient  
**mAP (macro)** — mean Average Precision (macro-averaged across classes)  
**ML** — Machine Learning  
**MobileNetV3-L** — MobileNetV3-Large  
**PR** — Precision–Recall  
**PRC** — Precision–Recall Curve  
**ResNet18** — Residual Network, 18 layers  
**ROC** — Receiver Operating Characteristic (*mentioned only for context; primary analysis uses PR*)  
**SOTA** — State Of The Art (*used descriptively where needed*)  
**StepLR** — Step Learning-Rate scheduler  
**TL** — Transfer Learning  
**TN** — True Negative  
**TP** — True Positive  
**Top-1 / Top-5** — Top-1/Top-5 Accuracy (classification rank metrics)  
**ViT-B/32** — Vision Transformer, Base, 32-patch  
**WD** — Weight Decay (regularization)  
**YOLO** — You Only Look Once  
**YOLOv11n-cls (Ultralytics)** — YOLO in Classification mode (Ultralytics)

## Confusion-Matrix Shortcuts (used in captions/figures):

**CM** — Confusion Matrix (counts)  
**NCM** — Normalized Confusion Matrix (row-normalized)

## Averaging schemes for metrics (as used in the thesis):

**macro** — Unweighted mean across classes  
**Weighted** — Class-frequency weighted mean across classes

# List of Figures

|                                                                                                                 |    |
|-----------------------------------------------------------------------------------------------------------------|----|
| <b>Figure 4.1.</b> Dataset classes                                                                              | 30 |
| <b>Figure 5.1.</b> Accuracy, Precision, Recall, and F1 (weighted) for all models on the test set ( $n = 171$ ). | 45 |
| <b>Figure 5.2.</b> swin_tiny_patch4_window7_224 — Precision–Recall Curves & Confusion Matrix.                   | 47 |
| <b>Figure 5.3.</b> InceptionV3 — Precision–Recall Curves & Confusion Matrix.                                    | 48 |
| <b>Figure 5.4.</b> EfficientNet-B0— Precision–Recall Curves & Confusion Matrix.                                 | 48 |
| <b>Figure 5.5.</b> MobileNetV3-Large — Precision–Recall Curves & Confusion Matrix.                              | 49 |
| <b>Figure 5.6.</b> ResNet18 — Precision–Recall Curves & Confusion Matrix.                                       | 49 |
| <b>Figure 5.7.</b> YOLOv11n-cls (Ultralytics) — Precision–Recall Curves & Confusion Matrix.                     | 50 |
| <b>Figure 5.8.</b> ViT-B/32 — Precision–Recall Curves & Confusion Matrix.                                       | 50 |
| <b>Figure 5.9.</b> Custom CNN baseline — Precision–Recall Curves & Confusion Matrix.                            | 51 |

# List of Tables

- **Table 4.1.** Dataset split per class and partition (Train/Validation/Test; support on test set). 43
- **Table 5.1.** Overall test performance of all models (Accuracy, Precision (weighted), Recall (weighted), F1 (weighted)). 53

# Chapter 1. Introduction

Biodiversity is the foundation of healthy ecosystems. However, it is now declining at alarming rates due to urban expansion, pollution, climate change, and illegal poaching. Monitoring wildlife populations is therefore essential for guiding conservation strategies. A key part of this monitoring is species identification: determining which animals appear in field-collected imagery and how frequently they occur.

Traditionally, species identification has been carried out manually by trained experts. This process is effective but slow, expensive, and prone to human error, especially when dealing with large-scale camera-trap deployments that can generate hundreds of thousands of images per year. These challenges create a pressing need for automated methods that can reliably classify species without continuous expert intervention.

Deep learning has emerged as a powerful solution for image classification across multiple domains, including healthcare, security, and ecology. Convolutional Neural Networks (CNNs), in particular, have achieved remarkable performance by automatically learning hierarchical features directly from raw pixels, eliminating the need for handcrafted descriptors. In wildlife monitoring, CNNs have already been applied to large-scale datasets with encouraging results. However, several issues remain:

- **Small and imbalanced datasets:** Many ecological studies focus on rare species with limited training examples.
- **Visual similarity:** Species may share color, texture, and shape characteristics, making them difficult to separate.
- **Field variability:** Images often contain poor lighting, occlusions, cluttered backgrounds, and unusual poses, which degrade performance.

To overcome these issues, research has increasingly turned toward advanced architectures. Multi-scale CNNs such as InceptionV3 allow richer feature extraction, while lightweight networks like MobileNetV3 and EfficientNet-B0 balance accuracy with efficiency—making them suitable for deployment on mobile or embedded devices. Recently, transformer-based vision models such as ViT and Swin Transformer have redefined image classification by modeling long-range dependencies across image patches, with Swin introducing hierarchical, window-based self-attention that has proven effective even on smaller datasets. Meanwhile, YOLO, traditionally a detection framework, has expanded to include classification, offering a streamlined alternative with fast training and inference.

Against this backdrop, this thesis investigates how these different model families perform when directly compared under controlled conditions. The study focuses on four target species that are visually similar yet ecologically important: *Plestiodon inexpectatus* (a skink), *Agkistrodon contortrix* (a venomous pit viper), *Caretta caretta* (loggerhead turtle), and *Oreaster reticulatus* (Caribbean starfish). By benchmarking CNNs, efficient/mobile architectures, transformers, and YOLO

classification, the work not only quantifies accuracy but also analyzes model behavior through class-level diagnostics.

## **1.1 Thesis Objectives**

The rapid decline of biodiversity caused by habitat destruction, climate change, and human activity has increased the demand for reliable and accurate methods to identify and monitor wildlife species. In many conservation contexts, researchers and practitioners face the challenge of classifying species that are visually similar, often under difficult capture conditions such as poor lighting, cluttered backgrounds, and partial occlusion. The four species considered in this thesis—*Plestiodon inexpectatus* and *Agkistrodon contortrix* have similar visuals. Also, *Caretta*, and *Oreaster reticulatus* are examples of classes that share visual traits and therefore require robust methods to distinguish them correctly.

The main objective of this thesis is not to propose a new architecture, but rather to establish a reproducible benchmarking pipeline that systematically evaluates several families of deep learning models under a unified standard. By doing so, the study seeks to highlight which models perform best in small, balanced datasets that reflect realistic field conditions. The evaluation includes classical convolutional networks such as ResNet18, efficiency-oriented backbones such as MobileNetV3-Large and EfficientNet-B0, multi-scale designs such as InceptionV3, transformer-

based models including ViT-B/32 and Swin Transformer, and YOLO in classification mode.

The goal is summarized in two points; First, to provide a fair and transparent comparison of these architectures when trained and validated under the same conditions, ensuring that differences in results are due to the models themselves rather than inconsistencies in training. Second, to translate the numerical outcomes—accuracy, weighted precision/recall/F1, macro-mAP, and confusion matrices—into practical guidance for researchers and conservation practitioners who may adopt such models in real-world biodiversity monitoring tasks.

## **1.2 Thesis Contribution**

This thesis contributes to the field of wildlife image classification in three main ways.

First, it offers a compact and reproducible benchmark across multiple architectures, covering CNNs, efficient/mobile models, Inception-style networks, vision transformers, and a Detector-native in Classification mode (YOLO). The experiments were designed to be consistent: the same preprocessing steps, the same augmentation strategies, and the same optimization settings were applied across models to ensure fairness.

Second, the study goes beyond reporting single accuracy values. It integrates deployment-oriented diagnostics, including per-class Precision–Recall curves, macro-mAP, and confusion matrices. These analyses allow not only a ranking of the models but also an understanding of where errors occur, which classes are most challenging, and how different models trade off between accuracy and efficiency. For example, the experiments identify *Agkistrodon contortrix* and *Plestiodon inexpectatus* as the most frequently confused pair, a finding that is consistent across several models. Such insights are directly useful for targeted data augmentation and curation strategies.

Third, the thesis provides practical guidance for application. The results show that Swin Transformer gives the best performance in small datasets, while InceptionV3 is also very stable and reliable. MobileNetV3-Large and EfficientNet-B0 provide a good balance between accuracy and efficiency, and YOLO works well with a simpler pipeline. At the same time, MobileNetV3-Large and EfficientNet-B0 demonstrate an excellent balance between accuracy and computational efficiency, making them suitable for deployment on resource-constrained hardware. YOLO-Cl shows competitive results with a simple pipeline, while ViT-B/32 and the CNN baseline illustrate the limitations of models that lack sufficient inductive bias in this context.

## 1.3 Thesis Organization

The remainder of this thesis is organized into five chapters.

Chapter 2 introduces the research problem in greater detail, providing background on wildlife image classification, its importance in biodiversity monitoring, and the challenges posed by visually similar species. It also outlines the dataset used in this study, the chosen evaluation metrics, and the scope and limitations of the work.

Chapter 3 reviews related work, tracing the development of image classification approaches from early convolutional networks to lightweight mobile models and transformer-based architectures. The chapter highlights prior successes and difficulties, especially in ecological applications, and situates the models used in this thesis within the broader research landscape.

Chapter 4 describes the proposed benchmarking pipeline in detail. It explains the dataset split, preprocessing and augmentation steps, training settings, and evaluation protocol. Each model family is introduced with its configuration, and the design choices—such as input resolutions, loss functions, optimizers, and schedulers—are justified with respect to both reproducibility and fairness.

Chapter 5 presents the experimental results and discussion. The chapter compares overall performance across models, then dives into class-level analysis using Precision–Recall curves and confusion matrices. It interprets the results to identify common patterns of misclassification and offers insights into why certain architectures outperform others under the dataset constraints.

Chapter 6 concludes the thesis by summarizing the key findings, emphasizing their implications for real-world deployment, and outlining directions for future work. These include targeted data collection for difficult species pairs, exploring alternative loss functions, improving augmentation strategies, and evaluating model robustness under domain shifts.

## Chapter 2 Overview

This thesis studies the problem of classifying wildlife species from images in conditions similar to real field deployments. The challenge is not only about building accurate models but also about making sure they work with the kinds of data that conservation researchers usually collect: small datasets, uneven image quality, and species that look very similar to one another. By designing a controlled and reproducible benchmarking pipeline, this study seeks to identify which modern vision architectures perform best in such constrained environments and why.

The dataset used in this thesis includes four species of interest: *Plestiodon inexpectatus*, *Agkistrodon contortrix*, *Caretta caretta*, and *Oreaster reticulatus*. In total, 841 images were initially collected. However, four images (one from each class) were removed because they were extremely difficult to label, even for human experts. Keeping these images could have introduced noise that does not reflect real learning capacity but rather labeling ambiguity. After this cleaning step, the final dataset consisted of 837 images, split into 582 for training, 84 for validation, and 171 for testing. The split was kept consistent across all experiments to ensure fair comparison and reproducibility.

Preprocessing and augmentation steps were deliberately simple to avoid distorting key species characteristics. Images were resized to 224×224 pixels for most models, to 299×299 for InceptionV3, and to 640×640 for YOLOv11n-cls (Ultralytics). Normalization followed ImageNet statistics to align with pre-trained weights. Augmentations were limited to horizontal/vertical flips and small rotations ( $\pm 10^\circ$ ). This simple choice adds some variation to improve robustness, but avoids heavy changes like color jitter that could hide the key visual features of the species.

The benchmark covers a wide range of architectural families. ResNet18 represents classical convolutional backbones, while MobileNetV3-Large and EfficientNet-B0 focus on efficient and lightweight designs. InceptionV3 illustrates the multi-scale approach, and ViT-B/32 and swin\_tiny\_patch4\_window7\_224 represent transformer-based vision models. In addition, YOLOv11n-cls (Ultralytics) is included to test the classification mode of a detection-native ecosystem, and a simple custom CNN baseline serves as a lower bound for comparison. All models were fine-tuned from ImageNet pre-trained weights, using consistent optimization settings (AdamW, Cross-Entropy loss, StepLR scheduler) to ensure fairness.

Evaluation was designed to go beyond a single accuracy score. For each model, this thesis reports Accuracy, weighted Precision, Recall, and F1 scores on the test split.

To provide a threshold-independent view, per-class Precision–Recall (PR) curves were plotted, and macro-averaged Average Precision (mAP) was computed. Confusion matrices (both raw and normalized) were also analyzed to identify which pairs of species are most often confused. By combining these metrics, the evaluation framework not only ranks models but also explains their strengths and weaknesses at the class level.

The findings of this chapter highlight consistent patterns. The `swin_tiny_patch4_window7_224` model achieved the highest performance, with an accuracy of 98.25% and nearly perfect macro-mAP, confirming the value of hierarchical transformers in small-data settings. InceptionV3 closely followed with 97.66%, showing that multi-scale networks remain strong candidates. MobileNetV3-Large and EfficientNet-B0 reached around 95.9% each, offering excellent accuracy-to-efficiency trade-offs. ResNet18 also performed competitively at 95.3%, while YOLOv11n-cls (Ultralytics) achieved 94.15% with a simpler pipeline. As expected, ViT-B/32 and the custom CNN baseline lagged behind at 62% and 58.5%, reflecting their weaker fit for the limited dataset size.

Finally, qualitative analyses support these numbers. PR curves for top models (Swin and InceptionV3) were close to ideal, hugging the top-right corner, which means they maintained high precision even at high recall levels. Confusion matrices revealed a recurring difficulty in distinguishing *Agkistrodon contortrix*

from *Plestiodon inexpectatus*, two reptile species with overlapping textures and poses, and occasional confusion between *Oreaster reticulatus* and *Caretta caretta* in marine contexts. These results suggest practical improvements, such as targeted data collection for the harder pairs and background-aware preprocessing (cropping or segmentation), that could further enhance performance without altering the overall benchmark design.

In summary, this chapter shows that a clear and reproducible benchmark can give useful guidance for wildlife classification. It shows that Swin Transformer and InceptionV3 are the strongest performers when accuracy is paramount, that MobileNetV3-Large and EfficientNet-B0 offer robust efficiency without much compromise, and that YOLOv11n-cls (Ultralytics) remains a solid option when pipeline simplicity is a priority.

# Chapter 3 Related Work

## 3.1 Introduction

Wildlife identification with computer vision has improved in recent years, but field deployment still faces practical issues. Camera-trap and field photos vary in lighting, pose, occlusion, and background clutter. Many species also look similar at a fine scale, which makes recognition harder when data are limited or unevenly sampled [1], [2], [5], [7], [8]. These factors make generalization to new sites or seasons difficult, so evaluations should consider robustness and consistency, not accuracy alone [4], [5], [8]. These problems become worse when datasets are small or unbalanced, and models usually fail when tested on new sites or different seasons, so evaluations must consider not only accuracy but also robustness and consistency under realistic conditions [4], [5], [8].

A key shift in the literature has been the move from hand-crafted features to deep learning with transfer learning. Early work showed that convolutional neural networks (CNNs) can identify animals directly from images at scale, and that pretraining on large natural-image datasets followed by fine-tuning can substantially reduce the amount of labeled data needed in ecological studies [1], [2]. To address annotation bottlenecks, several studies combined deep models with citizen-science pipelines, producing larger labeled sets without sacrificing accuracy and helping stabilize training on difficult classes [9], [10]. Together, these

ideas—transfer learning, careful splits, and disciplined evaluation—set the foundation for modern wildlife classifiers.

Beyond whole-image classification, object detection has been adapted to ecological imagery to localize animals in cluttered backgrounds. This can reduce background interference and improve downstream classification, but dense vegetation, occlusions, small targets, and variable viewpoints remain common failure modes that must be handled through preprocessing and augmentation choices [3]. Another recurring challenge is domain shift: performance often drops when training and testing are performed on different locations or seasons. The “terra incognita” setting made this explicit and encouraged evaluation protocols that test models under site shift rather than only within-site splits [4]. In parallel, survey papers argued for broader metrics—weighted precision/recall/F1, per-class precision–recall (PR) curves, macro-averaged average precision (mAP), and confusion matrices—to reveal class-specific errors that a single accuracy number can hide [5].

Recent work spans several architectural families. Classical residual CNNs remain strong baselines and are effective with modest data and augmentations, including in herpetofauna case studies that highlight both their promise and their sensitivity to class similarity and sample size [6]. Efficiency-oriented backbones (e.g., mobile and scaled CNNs) target restricted compute and memory while keeping accuracy

competitive for field use [2], [6], [8]. Vision transformers have introduced an alternative modeling paradigm; while vanilla variants often benefit from larger datasets or stronger regularization, hierarchical designs and careful fine-tuning have expanded their use in small-to-medium regimes [8], [10]. Alongside architectures, recent studies also examined the impact of hyperparameters on performance in camera-trap classification, reinforcing the value of systematic, fair comparisons rather than ad-hoc settings [11].

Within this context, the present thesis focuses on a compact, balanced four-class problem and builds a reproducible benchmark that compares representative models under a single, unified protocol. The study emphasizes transfer-learned fine-tuning, conservative but meaningful augmentations, and evaluation that goes beyond aggregate metrics to include per-class PR curves, macro-mAP, and confusion-matrix analysis. The goal is not to propose a new architecture, but to provide clear, deployment-oriented evidence about which families perform best under small-data constraints and where the remaining errors concentrate, in line with the methodological direction outlined in prior work , with attention to practical choices and reporting recommended by the community .

## 3.2 History

### *3.2.1 CNN-based classification on camera-trap imagery*

Early progress in wildlife image classification came from the work of Norouzzadeh et al. [1], who showed that convolutional neural networks (CNNs) could be used to identify and count animals directly from large sets of camera-trap images. Their study proved that deep learning can replace hand-crafted features and still achieve strong accuracy when enough labeled data is available.

Later, Tabak et al. [2] tested transfer learning, where CNN models trained first on ImageNet were fine-tuned on smaller ecological datasets. This approach reduced the need for very large training sets and allowed models to perform well even with limited data, which is important in field studies where annotated images are often scarce.

To address the challenge of limited labels, Willi et al. [9] combined CNN pipelines with citizen science projects. Volunteers helped to annotate images, which created larger and more reliable datasets and improved the performance of the models. More recent work, such as Luo et al. [10], suggested combining image recognition with language models and metadata to support more flexible classification tasks.

These studies together created what is now the standard practice in wildlife classification: start with pretrained CNNs, split data carefully into training, validation, and test sets, and evaluate models with multiple metrics rather than accuracy alone. This same strategy is followed in our thesis when benchmarking classical CNNs alongside more advanced architectures.

### ***3.2.2 Detection, cluttered scenes, and ecological context***

While many studies have focused on whole-image classification, others have adapted object detection methods to camera-trap images. The motivation was clear: in natural habitats, animals often appear in only a small part of the frame, surrounded by vegetation, shadows, or other distracting background elements. Object detectors such as Faster R-CNN and YOLO were therefore applied to localize animals before classification, reducing the impact of background noise [3].

These approaches showed promising results in identifying animals in complex ecological settings, especially when multiple species or individuals appeared in the same image. However, they also revealed recurring limitations. Dense vegetation, heavy occlusions, and animals captured at unusual poses or very small sizes often led to missed detections or misclassifications. These conditions are common in field deployments, where researchers cannot control the environment or lighting.

As a result, even when detection models are not directly used, many classification studies—including this thesis—adopt robust preprocessing and augmentation strategies to partially mitigate the same challenges. Simple techniques such as cropping, rotation, or controlled flipping help the models generalize better to cluttered or occluded scenes, even when full detection pipelines are not part of the workflow..

### ***3.2.3 Generalization to new domains and site shift***

One of the hardest problems in ecological machine learning is how models behave when they are moved to new environments. A model that performs well on data from one location may lose much of its accuracy when tested on images from a different site, season, or camera setup. This issue is called domain shift, and it is especially important in wildlife research where conditions vary widely.

The study Recognition in Terra Incognita [4] made this challenge explicit. It tested models by training them on data from certain locations and then evaluating them on completely different sites. The results showed a clear performance drop, even for strong CNN backbones. This highlighted the risk of relying only on within-site evaluations, which can give an overly optimistic impression of model reliability.

To address this, recent research has argued for evaluation protocols that simulate real-world deployment, not just lab conditions. Instead of focusing on accuracy alone, studies have added threshold-free measures such as precision–recall (PR) curves and macro-averaged average precision (mAP). These metrics reveal how models behave across thresholds and help identify whether a drop in performance is uniform or concentrated in specific species.

In this thesis, the lessons from domain-shift studies guide the evaluation strategy. By combining per-class PR curves, mAP, and confusion matrices, the results provide not

only overall accuracy but also insight into how errors distribute across species. This approach gives a clearer picture of model robustness and shows where improvements are most needed when moving to new or unseen data.

### ***3.2.4 Data quality, imbalance, and evaluation practice***

Another important challenge in wildlife image classification is the quality and balance of data. Camera-trap datasets often contain classes with very different numbers of images, as well as variations in lighting, background, or camera placement. This imbalance can bias the model toward the majority classes, while rare species receive less attention and are more often misclassified.

Surveys such as Wäldchen and Mäder [5] show that dataset imbalance and acquisition bias are common problems that directly affect model training and evaluation. They recommend moving beyond accuracy as the only metric, since a model can achieve high accuracy simply by performing well on the dominant classes. To properly understand performance, researchers are encouraged to report precision, recall, and F1-scores (weighted to account for class size), as well as per-class PR curves and confusion matrices. These tools reveal asymmetric errors, such as when one species is often confused with another, even if the overall accuracy looks acceptable.

In addition, program-level studies [8] emphasize that evaluation should reflect practical deployment. This means using transparent reporting, robust validation protocols, and diagnostic metrics that highlight weaknesses rather than hide them. These practices help bridge the gap between laboratory experiments and real-world field applications, where class imbalance and noise are unavoidable.

For this thesis, these insights informed the evaluation design. Weighted metrics, per-class PR curves, macro-mAP, and confusion matrices were all included to give a balanced view of model performance. This makes it possible to identify not only which models score highest overall, but also how they behave under class imbalance and which species pairs are most challenging to separate.

### ***3.2.5 Domain-specific case studies and fine-grained taxa***

Several domain-specific studies highlight both the strengths and the limits of deep learning for wildlife classification. For example, Islam et al. [6] applied CNN backbones such as ResNet to herpetofauna species (reptiles and amphibians) using curated camera-trap datasets. Their results showed that modern CNNs can exceed 85% accuracy with modest augmentation, but performance drops quickly when species are visually similar or when the dataset is small. This underlines the importance of both data scale and class distinctiveness in achieving reliable performance.

Earlier contributions in ecology, such as Weinstein [7], discussed how computer vision could support animal ecology beyond just classification. These studies pointed to applications like population monitoring and behavioral inference, showing the long-term potential of automated image analysis in conservation. They also stressed the need for careful metric selection and error analysis, since ecological questions often depend on detecting rare species or subtle behaviors where simple accuracy is not enough.

Together, these case studies provide useful lessons for this thesis. They show that CNNs are promising but need well-curated data to perform strongly, especially in fine-grained classification tasks where species are hard to distinguish. They also emphasize the value of going beyond “yes/no” accuracy to include detailed diagnostic tools, which is why this work adopts PR curves, confusion matrices, and weighted metrics to better capture strengths and weaknesses at the class level.

### ***3.2.6 Emerging directions: foundation models and zero-shot transfer***

More recent research has started to look beyond supervised training and into the use of foundation models and vision—language approaches. For example, Luo et al. [10] proposed *WildMatch*, which connects visual features from wildlife images with language descriptions. This type of alignment enables few-shot or even zero-shot transfer, meaning that models can recognize new species with very few or no labeled

images. Such methods open the door for handling rare or unseen taxa, which is one of the biggest challenges in biodiversity monitoring.

At the same time, Tuia et al. [8] discussed the broader perspective of combining machine learning with conservation practice. They emphasized that new tools should not only reach high accuracy in lab conditions but also support field applications, including cases where labeled data are scarce or where rapid deployment is needed. Vision transformers and large pre-trained models are part of this trend, as they can bring stronger generalization if adapted carefully to ecological settings.

For this thesis, these developments provide an important future direction. While the current work focuses on supervised fine-tuning in a small-data regime, the success of foundation models suggests that future systems for wildlife monitoring could benefit from transfer across datasets and species. This would make them more practical for conservationists who often deal with new sites, new species, and limited annotation capacity.

### **3.3 Summary**

The body of work reviewed in this chapter shows how the field of automated wildlife classification has developed from early CNN-based image classifiers [1], [2] to more complex pipelines that include object detection [3], domain-shift evaluations [4], and systematic reviews of best practices [5]. Case studies on specific taxa, such as herpetofauna, highlight both the potential and the sensitivity of deep networks to class similarity and dataset size [6]. Broader discussions in

ecology and computer vision emphasize the need for practical and reproducible methods that can bridge the gap between laboratory performance and field deployment [7], [8].

More recent directions move toward large-scale citizen science annotation [9], vision–language alignment [10], and the exploration of hyper-parameter tuning for more reliable results [11]. Collectively, these studies highlight three key challenges: (1) data scarcity and imbalance, (2) generalization to new environments, and (3) the need for evaluation beyond accuracy.

Within this context, the present thesis positions itself as a compact benchmark study. It focuses on four visually similar species and evaluates a range of representative architectures—CNNs, efficient models, Inception-style networks, transformers, and YOLO-Classification—under a single, controlled protocol. By doing so, it directly responds to the gaps identified in prior literature: the lack of reproducible, small-data benchmarks and the need for deployment-oriented evaluation that combines aggregate metrics with class-level diagnostics.

# Chapter 4 Proposed Approach

This chapter explains the design of the pipeline developed for multi-class wildlife species classification. The main aim is to compare a wide range of deep learning architectures under one unified and reproducible protocol. This ensures that differences in performance come from the model architecture themselves, and not from inconsistent preprocessing or training strategies. The approach also reflects real field conditions where datasets are small, balanced, and collected under variable imaging situations.

## 4.1 Dataset



Figure 4.1. Dataset classes

The dataset used in this thesis consists of four visually similar species:

*Plestiodon inexpectatus*, *Agkistrodon contortrix*, *Caretta caretta*, and *Oreaster reticulatus*.

- **The total cleaned dataset:** 837 images (after removing 4 ambiguous samples, one from each class, because even for human eyes they were too difficult to label reliably. Removing these avoided introducing misleading noise into the learning process).

- **Split:**
  - Training: 582 images
  - Validation: 84 images
  - Test: 171 images
- **Test support:**
  - *Plestiodon inexpectatus*: 43
  - *Agkistrodon contortrix*: 43
  - *Caretta*: 43
  - *Oreaster reticulatus*: 42

This split is fixed across all experiments to ensure fairness and reproducibility. Only training and validation sets are used for model selection, while the test set is kept untouched until the final evaluation.

## 4.2 Preprocessing and Augmentation

Preprocessing and data augmentation were crucial parts of this study. Since the dataset was small (837 images in total), the pipeline had to maximize the useful information extracted from each image while avoiding artificial distortions that might confuse the models. The preprocessing pipeline was standardized across all models, with only minor adjustments for architectures that required specific input sizes.

### 4.2.1 Image Resizing

Each image was resized to fit the standard input size expected by the model:

- $224 \times 224$  pixels: for ResNet18, MobileNetV3-Large, EfficientNet-B0, ViT-B/32, and Swin Transformer.
- $299 \times 299$  pixels: for InceptionV3 (required for its multi-scale inception modules).
- $640 \times 640$  pixels: for YOLOv11n-cls (Ultralytics) to match its default classification training pipeline.

Formally, resizing can be written as a linear mapping from original pixel coordinates  $(x,y)$  to new coordinates  $(x',y')$ :

$$y \cdot \frac{newH}{origH} = 'x, \quad y \cdot \frac{newW}{origW} = 'x$$

where  $(origW_{orig}, H)$  are the original width and height, and  $(newW_{new}, H)$  are the target dimensions.

### 4.2.2 Normalization

To stabilize training and leverage ImageNet-pretrained weights, images were normalized using the standard ImageNet channel statistics:

$$\frac{x_c - \mu}{\sigma} = 'x$$

where  $x$  is the pixel value in channel  $c \in \{R,G,B\}$ ,  $c\mu$  is the mean, and  $c\sigma$  is the standard deviation for that channel.

The values used:

- $\mu=[0.485,0.456,0.406]$
- $\sigma=[0.229,0.224,0.225]$

This ensured that the input distribution was consistent with the distribution used during pretraining on ImageNet.

### ***4.2.3 Data Augmentation***

Because the dataset was small and visually similar between classes, augmentation was applied conservatively. The main goal was to improve generalization by exposing models to slight variations, while still preserving fine-grained species-specific cues (color, texture, patterns).

#### **1. Horizontal and Vertical Flips**

- With a probability of 50%, each image was flipped along the horizontal or vertical axis.
- This simulates changes in camera-trap placement or animal pose.

Mathematically, for a horizontal flip:

$$(y, 1 - x - W) \mapsto (x, y)$$

For a vertical flip:

$$(1 - y - H, x) \mapsto (x, y)$$

#### **2. Small In-plane Rotations ( $\pm 10^\circ$ )**

- Images were randomly rotated within a range of  $[-10^\circ, +10^\circ]$

- This accounts for slight tilts in camera placement and animal movement.

The transformation for a rotation by angle  $\theta$ :

$$\begin{bmatrix} x \\ y \end{bmatrix} \cdot \begin{bmatrix} \sin \theta & \cos \theta \\ \cos \theta & \sin \theta \end{bmatrix} = \begin{bmatrix} 'x \\ 'y \end{bmatrix}$$

### 3. Avoided Augmentations

- No color jittering (brightness, contrast, saturation changes) was used, since these species rely heavily on subtle color and texture patterns for identification.
- No heavy policies (like RandAugment or Mixup during preprocessing) were applied, to avoid creating unrealistic samples that might mislead the classifier

In short, the preprocessing pipeline guaranteed that inputs were consistent with ImageNet pretraining, small augmentations improved generalization, and architectural differences (224 vs 299 vs 640 input size) were respected without introducing bias. This design ensured a level playing field across all models and reliable evaluation of their true strengths and weaknesses.

## 4.3 Models and Training Settings

The training setup for this thesis was designed to be simple, reproducible, and fair across all model families. Each model followed the same general loop, with differences only where the architecture itself required them. Below, the training loop, loss functions, optimizers, schedulers, and model-specific details are explained step by step.

### 4.3.1 Training and Validation Loop

Each epoch was divided into two distinct phases:

#### First: Training phase

- The model parameters were set to *train()* mode.
- Each input batch was passed through the network to produce predictions.
- The classification loss was computed using Cross-Entropy Loss.
- Backpropagation was applied: gradients were computed and accumulated.
- The optimizer (AdamW) updated the parameters based on these gradients.
- The learning rate scheduler (StepLR) adjusted the learning rate according to the epoch count.

#### Second: Validation phase

- The model parameters were switched to *eval()* mode.

- Each batch from the validation set was passed through the network without gradient updates.
- Validation loss and accuracy were calculated and recorded.
- The best checkpoint was saved whenever validation accuracy improved.

This two-phase structure ensured that learning occurred only in the training stage, while validation served as an unbiased measure of generalization.

### 4.3.2 Loss Function

For most models, the standard **Cross-Entropy Loss** was used:

$$-\sum_{i=1}^C \hat{p}_i \log y_i = L$$

where  $C$  is the number of classes (4 in this case),  $y_i$  is the one-hot encoded true label, and  $\hat{p}_i$  is the predicted probability for class  $i$ .

**the only special case for InceptionV3** as it has an auxiliary output head in addition to the main output. To make use of this, the total loss was:

$$L_{aux} \times 0.4 + L_{main} = L_{total}$$

The auxiliary branch helped stabilize gradients in the middle layers during training, especially in early epochs. The weight of 0.4 is the standard setting recommended in the original InceptionV3 paper and was applied here.

### 4.3.3 Optimizer

The optimizer across all PyTorch models was **AdamW**, chosen for its stability and ability to handle weight decay more effectively than standard Adam.

Initial learning rate:  $lr = 1 \times 10^{-4}$

Weight decay: default value (to control overfitting).

For YOLOv11n-cls (Ultralytics), the framework applied its internal configuration, which typically corresponds to an AdamW optimizer with an effective learning rate of around  $1.25 \times 10^{-3}$

### 4.3.4 Learning Rate Scheduler

A step decay scheduler (StepLR) was applied to gradually reduce the learning rate over time:

$$\left\lfloor \frac{epoch}{step\_size} \right\rfloor \gamma \times lr = newlr$$

Where  $\gamma = 0.97$  and  $step\_size = 3$

This meant that every 3 epochs, the learning rate was multiplied by 0.97. The gradual decay helped the models converge smoothly and reduced oscillations late in training.

### ***4.3.5 Epochs and Batch Size***

- **Epochs:**
  - PyTorch models (ResNet18, MobileNetV3-Large, EfficientNet-B0, InceptionV3, ViT-B/32, Swin Transformer, Custom CNN): 20 epochs.
  - YOLOv11n-cls (Ultralytics): 50 epochs (default setting in Ultralytics).
- **Batch size:** tuned to fit comfortably on a Tesla T4 GPU (15 GB VRAM). For most models, a batch size of 32 was used. This ensured efficient GPU utilization without running into memory errors.

### ***4.3.6 Model-Specific Adjustments***

- ResNet18: final fully connected layer replaced with a 4-class head.
- MobileNetV3-Large / EfficientNet-B0: last classifier head adjusted to 4 classes.
- InceptionV3: input resized to 299×299; auxiliary loss included.
- ViT-B/32: patch size of 32 pixels, global self-attention across patches.
- swin\_tiny\_patch4\_window7\_224: hierarchical windowed self-attention; input 224×224.
- YOLOv11n-cls (Ultralytics): input resized to 640×640; used Ultralytics classification pipeline.
- Custom CNN baseline: a small, shallow convolutional model built for comparison and to establish a lower bound under the same dataset.

### ***4.3.7 Hyperparameter Exploration***

To confirm robustness, limited hyperparameter tuning was tested with Optuna on ResNet18 as a representative model. The search space included:

- Learning rate ( $lr$ )
- Weight decay
- Step size (*step\_size*)
- Gamma ( $\gamma$ )

Only three trials were run, as the goal was not to optimize a single model but to check whether the baseline settings were reasonable. Results confirmed that the default configuration ( $lr=1e-4$ ,  $\gamma=0.97$ ) was already close to optimal for the dataset size.

#### **Summary**

This unified setup ensured comparability across models. By fixing the data splits, standardizing augmentations, and keeping training hyperparameters controlled, the observed differences in performance can be attributed to the architectures themselves rather than external factors

## **4.4 Evaluation Metrics and Procedure**

Model selection is performed strictly by validation accuracy. The test split is evaluated once using the best checkpoint to avoid optimistic bias and to emulate deployment. Evaluation goes beyond overall accuracy to include weighted precision, weighted recall, and weighted F1, which better reflect per-class balance and asymmetric errors. To obtain threshold-independent insights,

precision–recall (PR) curves are computed for each class, and the macro-averaged average precision (mAP) is reported as a compact summary statistic across classes. Confusion matrices, both raw counts and row-normalized, are analyzed to expose systematic confusions between visually similar taxa that PR curves may not fully capture. Together, these metrics provide a comprehensive picture of ranking quality, decision thresholds, and error modes.

## 4.5 Implementation and Reproducibility

- Frameworks: PyTorch 2.6 for standard models, Ultralytics 8.3.x for YOLOv11n-cls.
- Hardware: Google Colab with Tesla T4 GPU (15 GB).
- Reproducibility steps:
  - Fixed random seeds for PyTorch and NumPy.
  - CuDNN set to deterministic mode.
  - Same preprocessing/augmentation enforced across models.
  - Training logs, figures, and metrics stored for each run.

Additionally, Optuna was tested on ResNet18 as a demonstration of light hyperparameter tuning, exploring lr, weight decay, step\_size, and gamma

## 4.6 Design Rationale and Trade-offs

Each model was selected for a reason:

- Swin Transformer: best suited for small datasets thanks to hierarchical attention.
- InceptionV3: robust, stable, and benefits from auxiliary logits.

- MobileNetV3-Large / EfficientNet-B0: strong balance between accuracy and resource efficiency.
- ResNet18: simple, interpretable, and strong baseline.
- YOLOv11n-cls: easy to use, integrated with Ultralytics tools, though slightly less accurate.
- ViT-B/32: weaker in small datasets, but included as a transformer baseline.
- Custom CNN baseline: anchors the lower bound for performance.

Finally, all setup that mentioned previously in this chapter was designed to produce actionable insights. Instead of focusing on raw accuracy only, the combination of PR curves, macro-mAP, and confusion matrices gave detailed class-level diagnostics.

## Chapter 5 Experimental Results and Discussion

This chapter presents the results of the experiments and provides an interpretation of what these results mean for wildlife image classification in practice. The analysis is based on the held-out test set of 171 images, which includes balanced support across the four target species. The objective here is not only to show which model achieves the highest accuracy, but also to investigate how different architectural families behave when trained and tested under the same conditions on a small and carefully curated dataset.

The results are reported in several complementary ways. First, overall performance is compared using weighted Accuracy, Precision, Recall, and F1 Score. These metrics give a clear picture of the average model behavior across all classes. Second, threshold-independent diagnostics are introduced, namely per-class Precision–Recall (PR) curves and macro-averaged Average Precision (mAP), which capture the quality of class separation beyond a single threshold. Third, confusion matrices are examined to reveal where the models make systematic errors, especially when two species are visually similar and easy to confuse.

By combining these perspectives, the chapter highlights not only which architectures perform best, but also why certain patterns of errors occur and how they could be reduced in future work. The discussion is framed around practical insights: which models are best suited for deployment when accuracy is the priority, which ones are more efficient for resource-constrained environments, and what improvements could be made at the data or training level to further strengthen performance.

## 5.1 Overall Comparison

The overall test performance of all models on the held-out split ( $n = 171$ ) is summarized in Table 5.1. Metrics are reported as weighted averages across the four species (Accuracy, Precision, Recall, and F1 Score), which ensures that the evaluation reflects balanced contributions from all classes rather than being dominated by the easiest or most frequent ones.

| Model                                              | Accuracy       | Precision (w) | Recall (w)   | F1 (w)         |
|----------------------------------------------------|----------------|---------------|--------------|----------------|
| Swin Transformer<br>(swin_tiny_patch4_window7_224) | <b>0.98246</b> | <b>0.983</b>  | <b>0.982</b> | <b>0.98245</b> |
| InceptionV3                                        | 0.97661        | 0.977         | 0.977        | 0.97667        |
| MobileNetV3-Large                                  | 0.95906        | 0.959         | 0.959        | 0.95913        |
| EfficientNet-B0                                    | 0.959          | 0.959         | 0.959        | 0.959          |
| ResNet-18                                          | 0.95322        | 0.954         | 0.953        | 0.95318        |
| YOLOv11n-cls (Ultralytics, yolo11n-cls.pt)         | 0.94152        | 0.943         | 0.942        | 0.94127        |
| Vision Transformer (ViT-B/32)                      | 0.61988        | 0.629         | 0.62085      | 0.61695        |
| Custom Custom CNN baseline                         | 0.5848         | 0.597         | 0.586        | 0.58565        |

Table 5.1. Overall performance on the held-out test split ( $n=171$ ).

As shown in Table 5.1, the Swin Transformer (swin\_tiny\_patch4\_window7\_224) achieves the strongest results, with an Accuracy of 98.25%, weighted Precision of 98.3%, Recall of 98.2%, and F1 Score of 98.3%. This confirms the benefit of hierarchical attention, which captures both local and global patterns effectively in compact datasets.

Close behind is InceptionV3, which reaches 97.66% Accuracy with very similar Precision, Recall, and F1 values. Its stability reflects the advantages of multi-scale convolutional factorization and its canonical  $299 \times 299$  configuration, making it a reliable backbone for transfer learning in ecological tasks.

The efficiency-oriented models—MobileNetV3-Large and EfficientNet-B0—form a solid middle tier, each achieving 95.9% Accuracy. These models are particularly attractive in field deployments where computational and memory resources are limited, since they retain high accuracy while being lightweight.

ResNet-18, representing a classical CNN baseline, also performs competitively with 95.32%, while YOLOv11n-cls (Ultralytics) records 94.15%, slightly lower but notable for its simple, end-to-end pipeline that integrates well with detection-based workflows.

As expected under small-data conditions, ViT-B/32 and the Custom CNN baseline underperform, with Accuracies of 61.9% and 58.5% respectively. These results highlight their limited inductive biases and heavier reliance on larger datasets.

To complement these weighted metrics, the threshold-independent results are visualized in Figure 5.1. The macro-mAP is nearly saturated for the Swin Transformer (0.999) and remains very high for InceptionV3 (0.996), MobileNetV3-Large (0.996), EfficientNet-B0 (0.992), and ResNet-18 (0.988). YOLOv11n-cl is slightly lower at 0.976, while ViT-B/32 (0.700) and the Custom CNN baseline (0.611) show the expected decline.

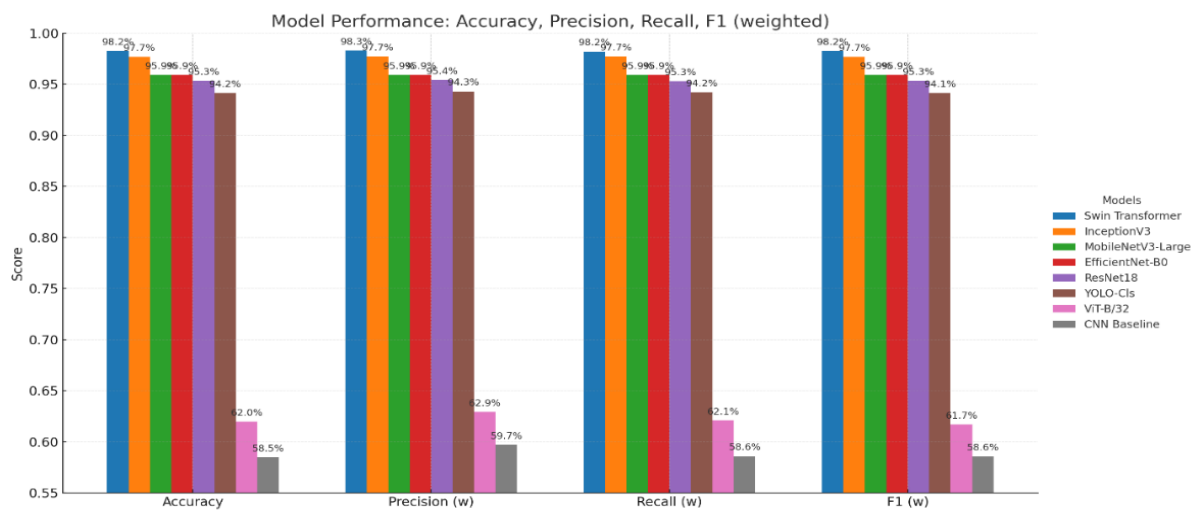


Figure 5.1. Accuracy, Precision, Recall, and F1 (weighted) for all models on the test set (n = 171).

In summary, as is clear from both Table 5.1 and Figure 5.1, hierarchical attention (Swin Transformer) and mature multi-scale CNNs (InceptionV3) stand out as the most effective models for small, balanced ecological datasets, while efficient

models such as MobileNetV3-Large and EfficientNet-B0 offer an attractive trade-off between accuracy and efficiency for deployment in real-world scenarios.

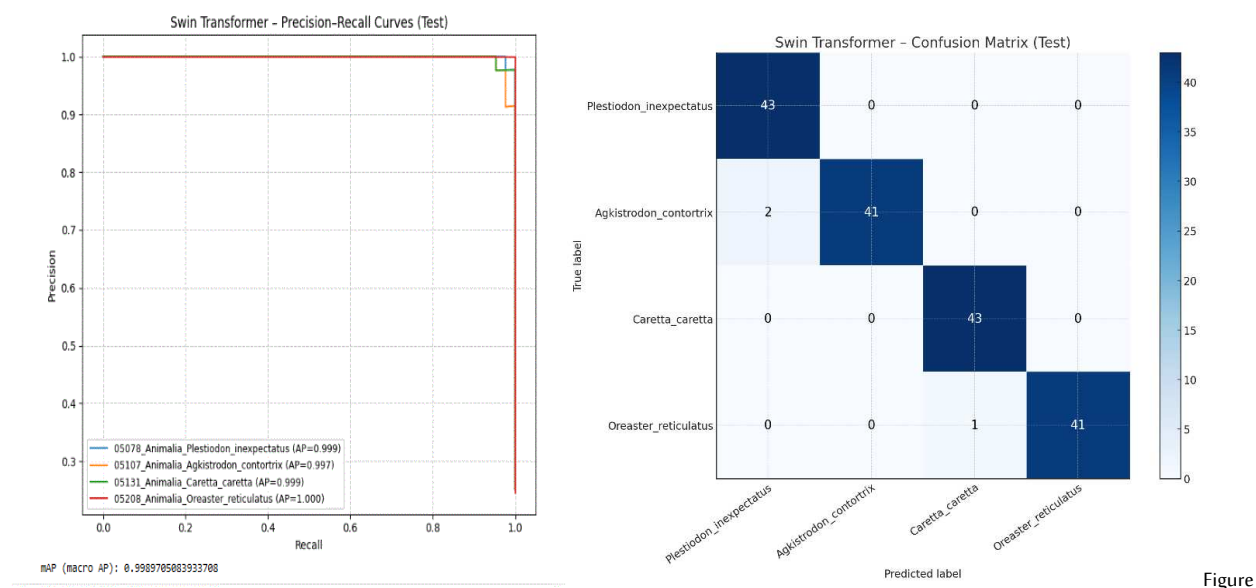
## **5.2 Precision–Recall curves & confusion matrix**

In this section, the results are examined using Precision–Recall (PR) curves and confusion matrices (CM) for each model. Taken together, these two views give a clear picture of model performance. The PR curves show how well each class can be separated across different thresholds, while the CMs reveal where the remaining errors occur at the chosen operating point. By combining them, it becomes possible to identify which classes are the most difficult, how robust the models are across thresholds, and what types of targeted improvements may help.

For the best-performing models—swin\_tiny\_patch4\_window7\_224 (Figure 5.2), InceptionV3 (Figure 5.3), EfficientNet-B0 (Figure 5.4), MobileNetV3-Large (Figure 5.5), and ResNet18 (Figure 5.6)—the PR curves are almost rectangular and close to the top-right corner for all species. This indicates that high recall does not significantly reduce precision. The macro-mAP values confirm this observation: 0.999 for Swin, 0.996 for InceptionV3 and MobileNetV3-Large, 0.992 for EfficientNet-B0, and 0.988 for ResNet18. Their confusion matrices support this result, showing most predictions on the diagonal with only a few, localized errors.

As a common patterns of confusion across these models, the most consistent challenge is the boundary between *Agkistrodon contortrix* and *Plestiodon inexpectatus*. In the PR curves, *Agkistrodon* typically bends earlier at very high recall, which suggests borderline cases with shared textures or poses. In the CMs, this is visible as a small but compact off-diagonal block. A lighter and less frequent confusion appears between *Oreaster reticulatus* and *Caretta*, mostly when background features dominate the image.

Looking at Figure 5.2. The PR curves are nearly perfect for all classes, and the CM shows almost no errors. When errors occur, they are usually 1–2 misclassifications between *Agkistrodon* \ *Plestiodon* also with an occasional *Oreaster* \ *Caretta* case.



Figure

5.2. swin\_tiny\_patch4\_window7\_224 — Per-class Precision—Recall curves on the test set & Confusion matrix.

Figure 5.3 refers that InceptionV3 shows Very similar behavior to Swin; The PR curves are very close to Swin, only slightly less saturated. The CM shows symmetric one-off swaps in the *Agkistrodon*/*Plestiodon* pair and rare cases of *Caretta* \ *Agkistrodon* or *Oreaster* \ *Caretta*..

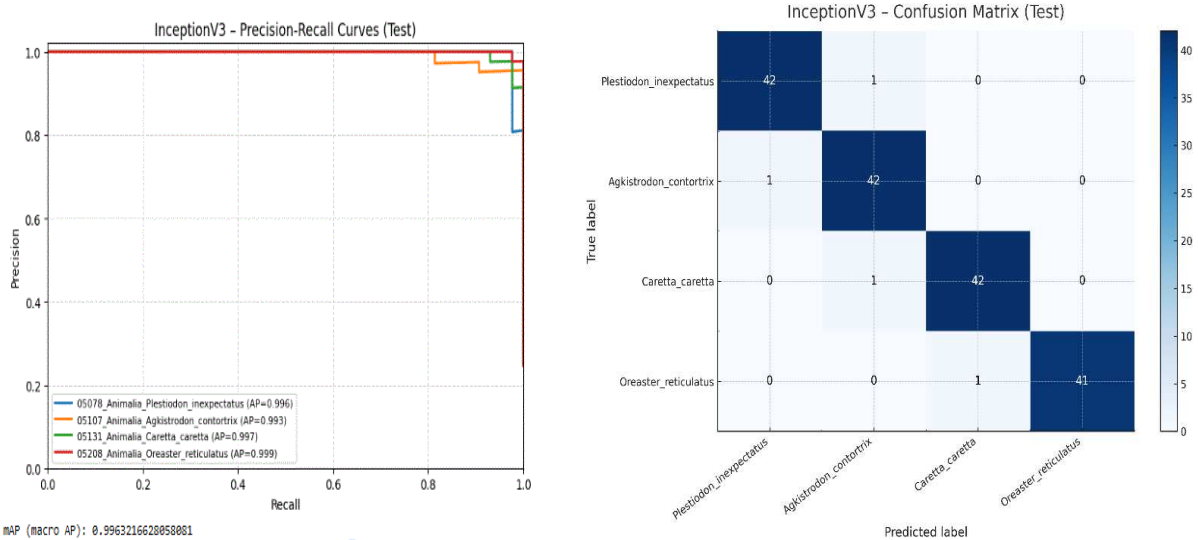


Figure 5.3. InceptionV3 — Per-class Precision–Recall curves & Confusion matrix

For EfficientNet-B0 in Figure 5.4, The PR curves remain very high, but the CM shows a slightly thicker off-diagonal band for Agkistrodon \Plestiodon compared to Swin and Inception.

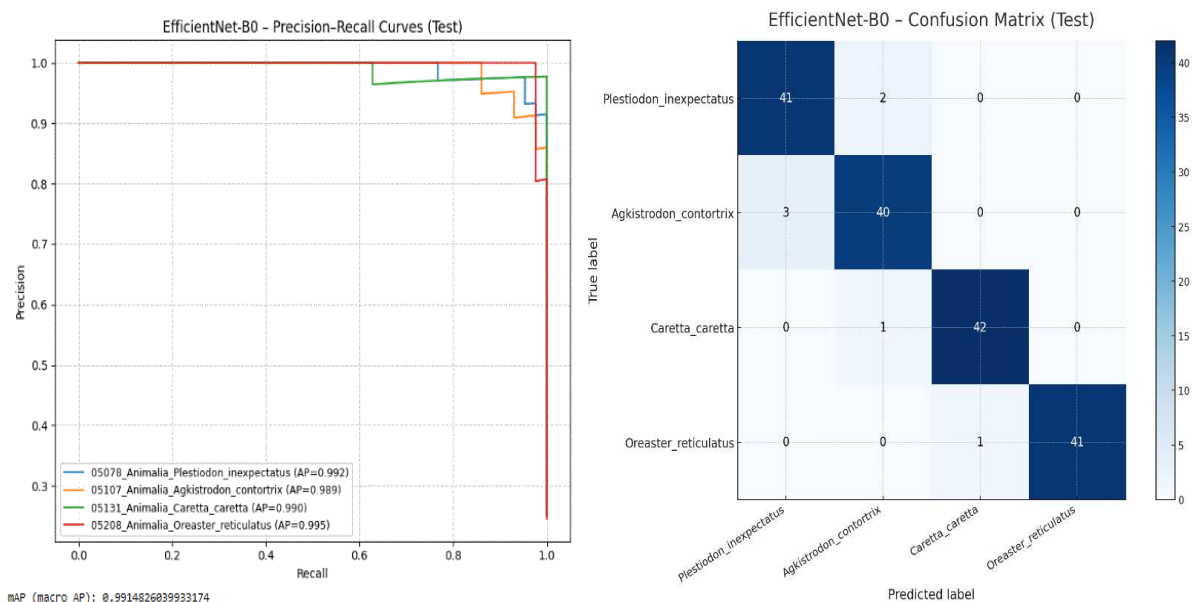


Figure 5.4. EfficientNet-B0 — Per-class Precision–Recall curves & Confusion matrix

PR curves for MobileNetV3-Large in Figure 5.5 is Similar to EfficientNet-B0, with high PR curves and a compact but visible confusion between Agkistrodon \ Plestiodon

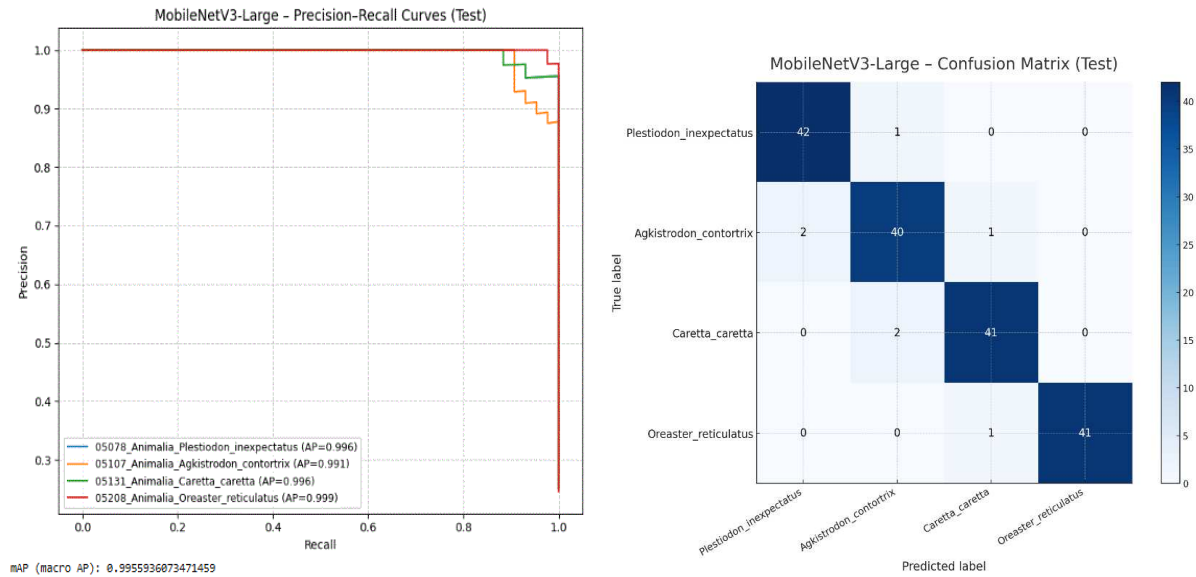


Figure 5.5. MobileNetV3-Large — Per-class Precision–Recall curves & Confusion matrix

ResNet18 in (Figure 5.6): Shows the same structure as the models above, but with a bit more dispersion off the diagonal. Nevertheless, PR curves remain well above operating thresholds.

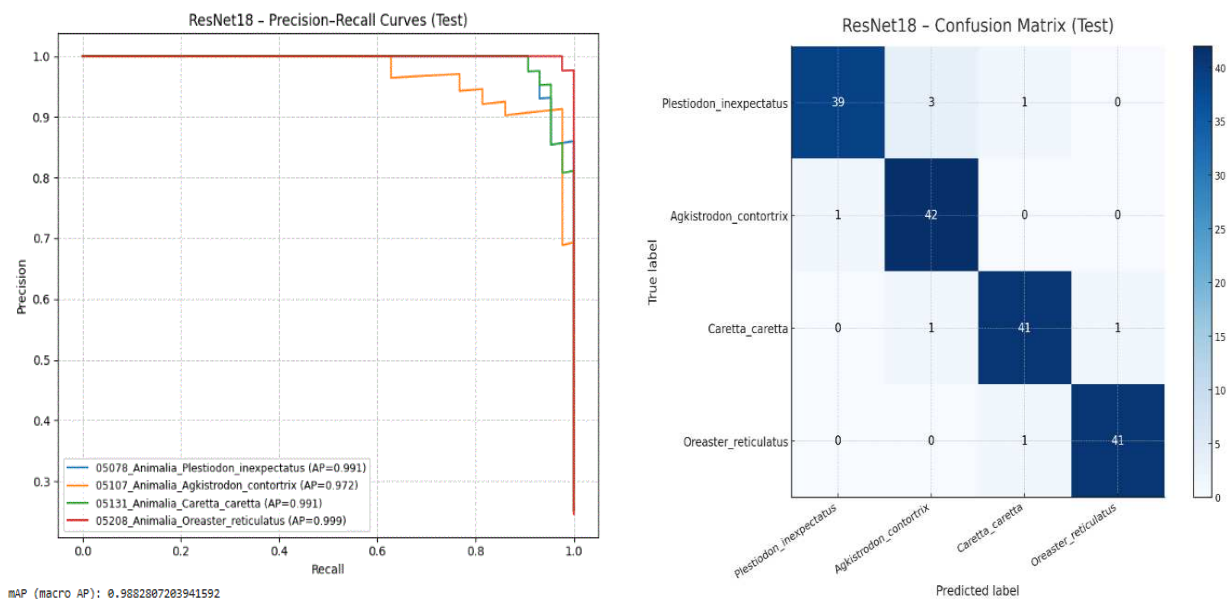


Figure 5.6. ResNet18 — Per-class Precision–Recall curves & Confusion matrix.

PR curves for YOLO in (Figure 5.6) are strong but consistently below the top CNN and Transformer models. The CM shows broader off-diagonal activity, especially for *Agkistrodon* being misclassified as neighboring classes, along with occasional *Caretta* confusions. This matches its slightly lower overall accuracy, even though the training and inference pipeline is simpler. training/inference pipeline.

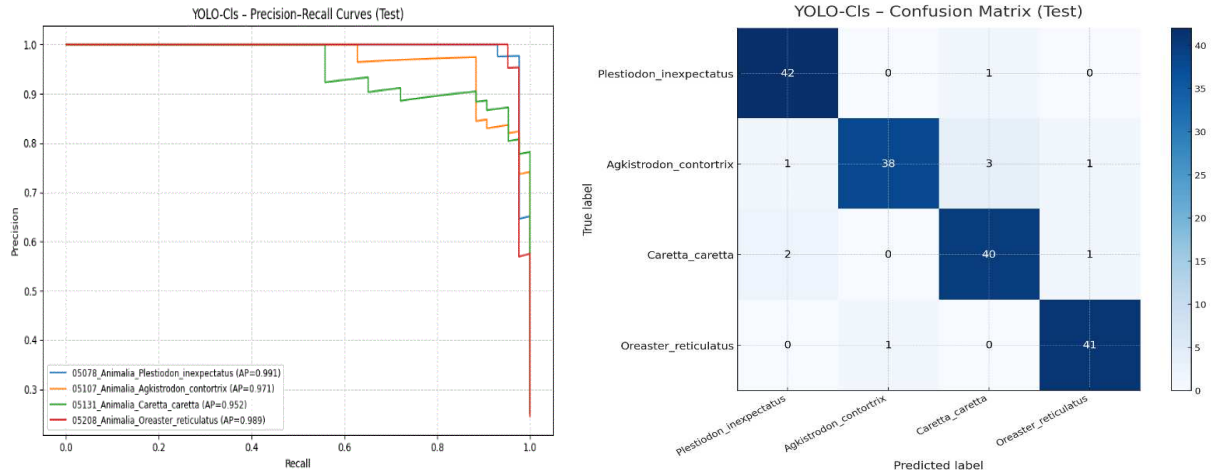


Figure 5.7. YOLO-Classification — Per-class Precision–Recall curves & Confusion matrix.

iT-B/32 (Figure 5.8) shows weaker PR curves, with earlier bending and lower area, alongside broader CM off-diagonals—especially for *Caretta* and *Plestiodon*.

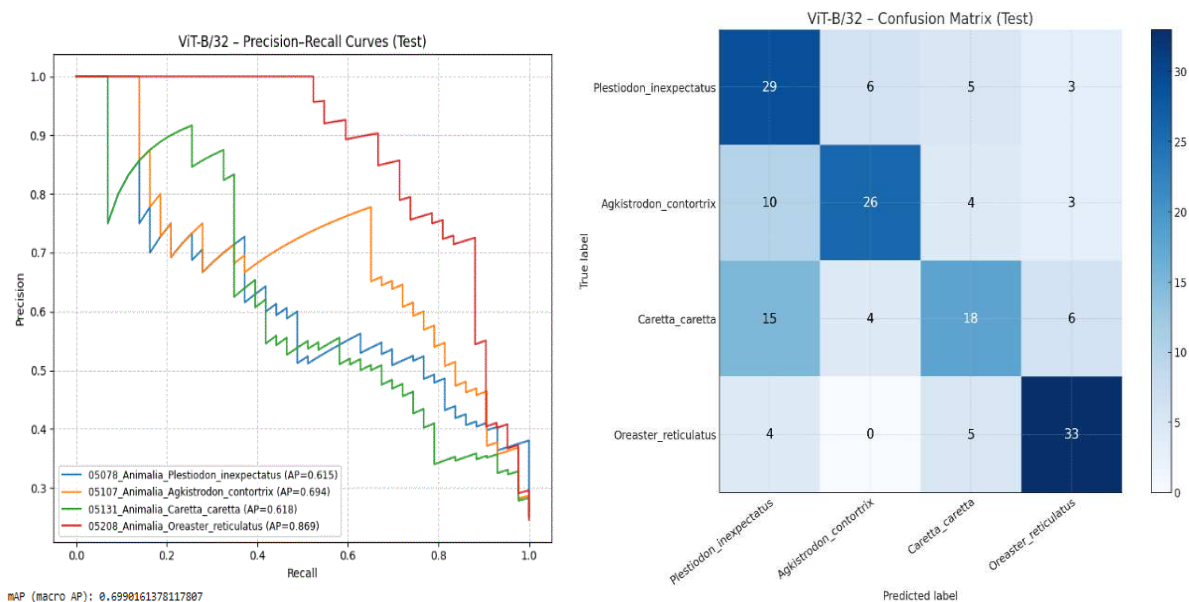


Figure 5.8. ViT-B/32 — Per-class Precision–Recall curves & Confusion matrix.

The CNN Custom baseline (Figure 5.9) performs the worst, with weak PR separability and the broadest CM off-diagonals, confirming the limitations of shallow CNNs in this dataset.

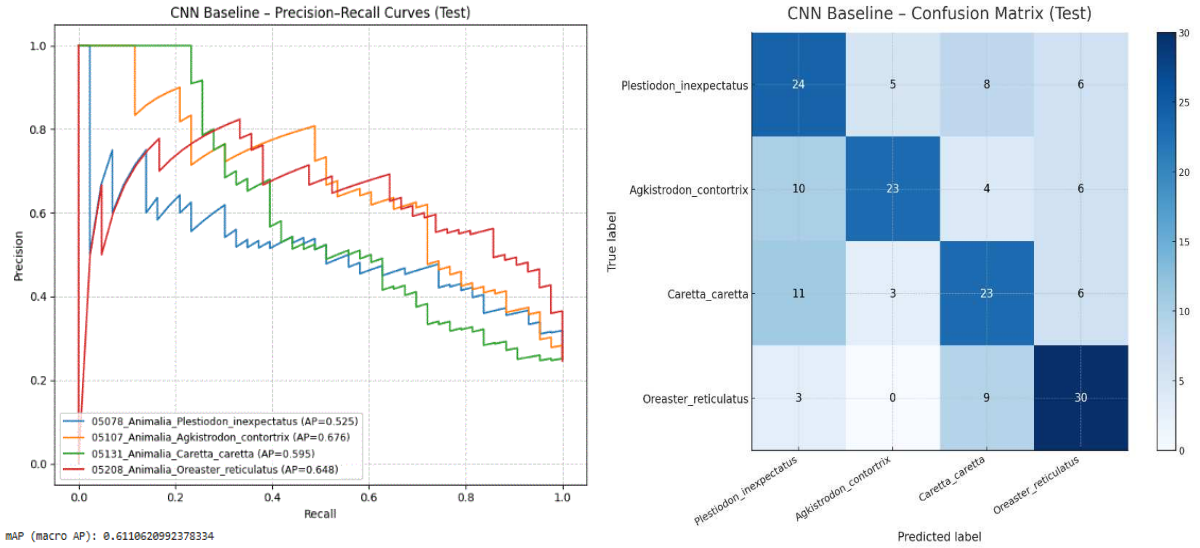


Figure 5.9. Custom CNN baseline — Per-class Precision–Recall curves & Confusion matrix.

As shown in (Figures 5.2–5.9) the PR curves and confusion matrices explain why the ranking seen in the overall metrics (Table 5.1, Figure 5.1) holds true. Swin and Inception lead because they show clean separation with only rare, localized mistakes. MobileNet and EfficientNet are strong middle options, trading a small amount of separability for efficiency. ResNet18 remains competitive as a classical baseline. YOLO offers a practical but slightly less accurate pipeline. Finally, ViT-B/32 and the Custom CNN baseline show the expected drop under small-data constraints. These insights also point to the main improvement area: better handling of the Agkistrodon–Plestiodon pair, through targeted data collection and augmentation.

### 5.3 Per-class metrics analysis

Table 5.2 reports the per-class Precision, Recall, and F1-scores for all models. This view complements the aggregate results in Table 5.1 and the PR/CM figures (Figures 5.2–5.9) by showing explicitly how each architecture performed on individual species. As can be seen, the most consistent difficulty lies in **Agkistrodon contortrix**, where Precision and Recall values are slightly lower compared to the other classes across nearly all models. In contrast, **Oreaster reticulatus** is generally the easiest class, often reaching near-perfect scores. These patterns match the error trends already observed in the confusion matrices and confirm the need for targeted improvements in data and augmentation for the reptile.

| Approach         | Model             | Class                   | Precision | Recall | F1-score |
|------------------|-------------------|-------------------------|-----------|--------|----------|
| Pre-trained CNNs | InceptionV3       | Plestiodon_inexpectatus | 0.98      | 0.98   | 0.98     |
|                  |                   | Agkistrodon_contortrix  | 0.96      | 0.98   | 0.97     |
|                  |                   | Caretta_caretta         | 0.98      | 0.98   | 0.98     |
|                  |                   | Oreaster_reticulatus    | 1         | 0.98   | 0.99     |
|                  | EfficientNet-B0   | Plestiodon_inexpectatus | 0.93      | 0.95   | 0.94     |
|                  |                   | Agkistrodon_contortrix  | 0.93      | 0.93   | 0.93     |
|                  |                   | Caretta_caretta         | 0.98      | 0.98   | 0.98     |
|                  |                   | Oreaster_reticulatus    | 1         | 0.98   | 0.99     |
|                  | MobileNetV3-Large | Plestiodon_inexpectatus | 0.96      | 0.98   | 0.97     |
|                  |                   | Agkistrodon_contortrix  | 0.93      | 0.93   | 0.93     |
|                  |                   | Caretta_caretta         | 0.95      | 0.95   | 0.95     |
|                  |                   | Oreaster_reticulatus    | 1         | 0.98   | 0.99     |
|                  | ResNet18          | Plestiodon_inexpectatus | 0.98      | 0.91   | 0.94     |
|                  |                   | Agkistrodon_contortrix  | 0.91      | 0.98   | 0.94     |
|                  |                   | Caretta_caretta         | 0.95      | 0.95   | 0.95     |
|                  |                   | Oreaster_reticulatus    | 0.98      | 0.98   | 0.98     |
| Transformer      | Swin Transformer  | Plestiodon_inexpectatus | 0.96      | 1      | 0.98     |
|                  |                   | Agkistrodon_contortrix  | 1         | 0.95   | 0.98     |
|                  |                   | Caretta_caretta         | 0.98      | 1      | 0.99     |
|                  |                   | Oreaster_reticulatus    | 1         | 0.98   | 0.99     |
|                  | ViT-B/32          | Plestiodon_inexpectatus | 0.5       | 0.67   | 0.57     |
|                  |                   | Agkistrodon_contortrix  | 0.72      | 0.61   | 0.66     |
|                  |                   | Caretta_caretta         | 0.56      | 0.42   | 0.48     |
|                  |                   | Oreaster_reticulatus    | 0.73      | 0.79   | 0.76     |
| Pre-trained YOLO | YOLO-CIs          | Plestiodon_inexpectatus | 0.93      | 0.98   | 0.96     |
|                  |                   | Agkistrodon_contortrix  | 0.97      | 0.88   | 0.93     |
|                  |                   | Caretta_caretta         | 0.91      | 0.93   | 0.92     |
|                  |                   | Oreaster_reticulatus    | 0.95      | 0.98   | 0.97     |
| Baseline         | CNN Baseline      | Plestiodon_inexpectatus | 0.5       | 0.56   | 0.53     |
|                  |                   | Agkistrodon_contortrix  | 0.74      | 0.54   | 0.62     |
|                  |                   | Caretta_caretta         | 0.52      | 0.54   | 0.53     |
|                  |                   | Oreaster_reticulatus    | 0.63      | 0.71   | 0.67     |

**Table 5.1.** Overall test performance of all models (Accuracy, Precision (weighted), Recall (weighted), F1 (weighted)).

## 5.4 Practical Implications

The results presented in this thesis are not only theoretical but carry direct value for real-world deployment in conservation projects. The ranking observed in Table 5.1 and the per-class analyses (Figures 5.2–5.9) provide clear guidance on which models are most suitable depending on the practical constraints of the field.

For **protected terrestrial areas**, such as reserves that rely on camera traps, the choice of model is crucial. Swin Transformer, with its high accuracy (98.25%) and nearly perfect separability across classes, is the best candidate when the goal is to minimize misclassifications in environments where false alarms can be costly. For instance, distinguishing *Agkistrodon contortrix* from *Plestiodon inexpectatus* with very high reliability allows field biologists to automate much of the daily monitoring workload. In this context, even rare mistakes can lead to unnecessary site visits or missed opportunities to detect a species of interest. InceptionV3, which follows closely with 97.66% accuracy, provides a stable and more lightweight alternative that remains attractive for these camera-trap deployments.

When **hardware and energy are constrained**, such as in remote desert or forest reserves where edge devices or solar-powered units are used, MobileNetV3-Large and EfficientNet-B0 become highly practical. Their performance (~95.9% accuracy) is slightly lower than Swin or Inception, but their efficiency means they can be deployed on smaller embedded devices with limited compute and memory without sacrificing too much reliability. These models can be integrated into on-site camera systems that process data locally before transmitting results, reducing storage and transmission costs.

For **marine monitoring applications**, particularly in environments such as turtle nesting beaches or coral reef surveys, the findings are equally relevant. The confusion matrices showed occasional errors between *Caretta caretta* (sea turtle) and *Oreaster reticulatus* (starfish), often when background context dominates the image. In practice, this suggests that preprocessing steps like background suppression or segmentation should be prioritized in marine deployments to avoid these specific confusions. Swin and Inception again stand out as the best-performing models in this setting, but even the lightweight CNNs can provide robust support when paired with background-aware image pipelines.

From an **operational perspective**, the Precision–Recall (PR) curves demonstrate that the top-performing models are tolerant to threshold adjustments. This is especially useful in conservation workflows where trade-offs between precision and recall must be tailored to context. For example, in high-risk settings (such as monitoring venomous reptiles), one may set the threshold to maximize recall to avoid missing any instance of *Agkistrodon contortrix*, even if it means tolerating a few false positives. Conversely, in low-risk but resource-limited settings, thresholds can be tuned toward higher precision to reduce the burden of false alarms and manual verification.

Finally, the results provide **clear data priorities** for future expansion. The recurring confusion between *Agkistrodon* and *Plestiodon* points to the need for additional and more diverse data for these species, particularly with varied poses, backgrounds, and lighting conditions. In practical terms, conservation teams should focus on collecting more field data for these taxa to strengthen model robustness.

## Chapter 6 Conclusion and Future Work

This thesis set out to evaluate the problem of wildlife species classification in small-data scenarios using a reproducible benchmarking pipeline. The focus was on four species that are visually similar and often confused in practice: *Plestiodon inexpectatus*, *Agkistrodon contortrix*, *Caretta caretta*, and *Oreaster reticulatus*. Unlike studies that introduce new architectures, this work aimed to create a fair and systematic comparison of existing deep learning families under identical preprocessing, training, and evaluation settings. In doing so, the study not only ranked models by performance but also explained how and why certain architectures worked better under the constraints of compact, balanced datasets. The experimental results presented in Chapter 5 provided a clear hierarchy. The Swin Transformer (*swin\_tiny\_patch4\_window7\_224*) achieved the highest accuracy on the held-out test set (98.25%) with equally strong weighted precision, recall, and F1 scores. InceptionV3 came close at 97.66%, confirming that multi-scale feature extraction and auxiliary training remain powerful strategies for image classification. MobileNetV3-Large and EfficientNet-B0 formed a highly competitive tier, offering about 95.9% accuracy while being lightweight and efficient, which makes them suitable for real-time or resource-limited deployments. ResNet18, though older, proved that classical CNNs remain strong baselines with an accuracy of 95.32%. YOLOv11n-cls (Ultralytics) scored 94.15%, showing that even detection-native frameworks can adapt to classification tasks with good results. As expected, ViT-B/32 and the custom shallow CNN baseline

performed significantly lower (62% and 58.5%), demonstrating the importance of inductive biases and sufficient data in small-sample regimes.

Beyond raw numbers, the analysis of Precision–Recall curves and confusion matrices provided deeper insights. The top-performing models produced PR curves that were nearly rectangular, went straight close to the top-right corner and maintaining high precision even at very high recall. This indicates that threshold adjustments, such as prioritizing precision in high stakes monitoring or recall in biodiversity surveys, can be made without sacrificing overall performance. The confusion matrices, on the other hand, revealed exactly where the models struggled. Across nearly all backbones, the boundary between *Agkistrodon contortrix* and *Plestiodon inexpectatus* remained the most problematic, reflecting how visually similar these two reptiles are in shape, color, and background context. A smaller but recurring confusion was observed between *Oreaster reticulatus* and *Caretta caretta* in marine scenes where background cues were dominant. These findings connect directly to practical recommendations: targeted data expansion, especially for the reptile pair, and background-aware preprocessing could yield the most immediate improvements.

From a methodological perspective, this thesis makes two main contributions. First, it demonstrates the value of a unified experimental pipeline, where preprocessing, augmentation, training, and evaluation are standardized to avoid confounding factors. This ensures that differences in performance come from architectural strengths and weaknesses rather than arbitrary settings. Second, it shows that

diagnostic tools beyond accuracy—specifically per-class PR curves, macro-mAP, and confusion matrices—are critical for understanding real-world error modes. These tools move the analysis from abstract percentages to practical, class-specific insights that conservation practitioners can act upon.

**For future work** there are several directions for extension emerge naturally from this study.

The clearest priority is to expand the dataset, particularly for the *Agkistrodon* / *Plestiodon* pair. Collecting images that vary in pose, lighting, angle, and background will reduce confusion between these two reptiles. Another step is to preprocess images in ways that minimize background influence, such as cropping to focus on the animal or applying segmentation masks. Color augmentation (e.g., slight jitter) could also highlight subtle chromatic cues that differentiate species.

Also the exploring class-aware or focal loss functions may reduce the weight of easy examples and focus learning on the hardest boundaries. Techniques such as mixup and cutmix can encourage models to generalize better, especially when the dataset size is limited. Hyper-parameter optimization with tools like Optuna could also provide small but measurable improvements in performance by fine-tuning learning rates, decay, and scheduling in a systematic way.

One limitation of this study is that training and testing were performed on a fixed dataset from a single distribution. In real-world deployments, however, models are exposed to new locations, sensors, and conditions. Testing under domain shift—such as training on one region and testing on another—would provide stronger evidence of generalization. Lightweight domain adaptation methods (e.g., fine-tuning only normalization layers) could be explored to mitigate drops in accuracy.

For field applications, accuracy is not the only concern. Models must be interpretable and trusted by conservation staff who may not have technical expertise. Adding explainability methods such as Grad-CAM would help confirm that models attend to species-diagnostic features (e.g., body shape or skin texture) rather than irrelevant background cues. Deployment could also benefit from lightweight ensembling of Swin and Inception models, which would likely eliminate the few remaining errors without large additional computational cost.

The most important step is to integrate this pipeline into actual conservation workflows. Camera traps in protected areas, aerial drones in terrestrial reserves, and underwater sensors in marine ecosystems all stand to benefit from automated, reliable classification. For example, in Saudi Arabia’s protected regions, real-time monitoring of reptile populations could help measure ecosystem health and detect illegal hunting. Similarly, in marine reserves, automated recognition of turtles and starfish could aid in monitoring coral reef conditions. Adapting the pipeline to these

use cases would involve optimizing for inference speed, energy efficiency, and user-friendly interfaces, but the benefits for biodiversity monitoring are direct and tangible.

# References

1. 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.
2. Tabak, M. A., Norouzzadeh, M. S., Wolfson, D. W., Sweeney, S. J., Vercauteren, K. C., Snow, N. P., Halseth, J. M., 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.
3. Schneider, S., Taylor, G. W., & Kremer, S. (2018). Deep learning object detection methods for ecological camera trap data. In *2018 25th IEEE International Conference on Image Processing (ICIP)* (pp. 1363–1367). IEEE.
4. Beery, S., Horn, G., & Perona, P. (2018). Recognition in terra incognita. In *Proceedings of the European Conference on Computer Vision (ECCV)* (pp. 456–473). Springer.
5. Wäldchen, J., & Mäder, P. (2018). Machine learning for image-based species identification. *Methods in Ecology and Evolution*, 9(11), 2216–2225.
6. Islam, S. B., Valles, D., Hibbitts, T. J., Ryberg, W. A., Walkup, D. K., & Forstner, M. R. J. (2023). Deep learning models for classifying herpetofauna species in camera trap images. *Animals*, 13(5), 789.
7. Weinstein, B. G. (2018). A computer vision for animal ecology. *Journal of Animal Ecology*, 87(3), 533–545.
8. 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., & Stewart, F. E. C. (2022). Perspectives in machine learning for wildlife conservation. *Nature Communications*, 13, 792.
9. Willi, M., Pitman, R. T., Cardoso, A. W., Locke, C., Swanson, A., Boyer, A., Veldthuis, M., & Fortson, L. (2019). Identifying animal species in camera trap images using deep learning and citizen science. *Methods in Ecology and Evolution*, 10(1), 80–91.
10. Luo, S., Zhao, Y., Liu, C., Zhang, Y., Zhu, J., & Tang, K. (2023). WildMatch: Grounding wildlife perception in language models. *arXiv preprint arXiv:2311.01064*.
11. Mohammed, A., Khan, F., Ali, M., & Rahman, S. (2024). Deep learning and model hyperparameters selection for mammal species classification using camera traps. *International Journal of Intelligent Systems and Applications in Engineering*, 12(1), 37–44.