

School of Engineering and Computer Science
Te Kura Mātai Pūkaha, Pūrorohiko

PO Box 600
Wellington
New Zealand

Tel: +64 4 463 5341
Internet: office@ecs.vuw.ac.nz

Integrating Computer Vision, Explainable AI, and MLOps for Predator Detection in Conservation Contexts

Qaswar Almousa

Supervisor: Andrew Lensen

Submitted in partial fulfilment of the requirements for
Master of Artificial Intelligence.

Abstract

Invasive predators threaten native wildlife in Aotearoa New Zealand. These predators are monitored using camera traps, which produce too many images for a human to review manually. In machine learning literature, models are often evaluated mainly on accuracy, which is not enough for models used in conservation. This is because a model can be right for the wrong reasons, or get worse once deployed.

I built an end-to-end, deployment-aware pipeline classifying mammals vs birds in New Zealand camera-trap images, using a public dataset of over 2.4 million images, from raw data to a deployed, monitored system. This system's goal was to test five hypotheses related to seasonal drift, background reliance, box-cropping, drift detection and retraining, and human oversight.

Contrary to my expectation, seasonal change did not reduce performance, with the model trained on warm-season images reaching 97.3% accuracy on cool-season images, although part of this came from familiarity with camera sites. Explainability showed that the model relied much more on the background for birds than for mammals, with a median of 0.32 of its attention on correctly classified birds compared with 0.73 for mammals, although part of this gap comes from bird boxes being smaller, and Grad-CAM++ showed its false positives were mostly driven by the background. Cropping each image to the animal using a pretrained camera-trap object detector (MegaDetector) cut false positives from 736 to 154 and raised the attention on birds from 0.32 to 0.88, or from 0.27 to 0.69 when compared with attention spread evenly. When tested on images from new reserves, drift was detected as accuracy dropped from 98.1% to 81.3%, and retraining did not harm performance on the original data while improving accuracy on the new data from 85.5% to 94.7%, although from only 5 sites. Finally, flagging the most uncertain 20% of predictions for human review caught 91.1% of errors, although the model's own confidence did just as well, and a person using the dashboard found a problem that offline evaluation had missed.

Overall, this shows that accuracy alone is not enough to trust a model used in conservation. A model also needs to be checked for why it makes its predictions, monitored once it is deployed, and kept under the control of a person, so that it helps protect native species rather than harming them.

Acknowledgments

This work would not be what it is without the support of many amazing people in my life.

Andrew, thank you for being a better supervisor than I could ever wish for. Thank you for trusting me to get the work done as well as being supportive throughout our whole time together. Thank you for being caring when I voiced feeling burnt out. I really appreciate the time you have invested in me and I have had the privilege of learning a lot from you.

I would also like to thank Ryan Jaggers for providing the external dataset used to test drift and retraining.

To my wonderful family, my father, my mother, and my siblings Jaafar and Durar, thank you for setting the example and standard for me. Mum, Dad, without the sacrifices you both made I would not have even been able to attend this university and for that I am forever grateful. Thank you for your unconditional love and support, it means the world to me.

To my amazing friends that I have had the privilege of being around throughout my life at university, to Chansy, Kylie, Aiden, Connor and Yasmin. This group has got me through some of the lowest of lows. Thank you for celebrating my highs and supporting me through the rough patches. Without you all I would not have been able to finish this project and I am lucky to have all of you in my life.

And to my late friends, Muhenda (Kevin) and Joshua, I made it.

Contents

1 Introduction **1**
- 1.1 Problem Statement . . . 1
- 1.2 Motivation . . . 2
- 1.3 Goals and Hypotheses . . . 3

2 Background **4**
- 2.1 Invasive Predator Species in New Zealand . . . 4
  - 2.1.1 Wildlife Monitoring Practices . . . 4
  - 2.1.2 Camera Trap Technology . . . 5
- 2.2 Computer Vision and Machine Learning . . . 6
  - 2.2.1 CNN . . . 6
  - 2.2.2 ViT . . . 6
  - 2.2.3 Pre-Trained Models . . . 7
  - 2.2.4 Evaluating classifiers . . . 7
- 2.3 Explainable AI (XAI) . . . 8
- 2.4 Uncertainty Estimation . . . 9
- 2.5 MLOps . . . 10
- 2.6 Research Gap . . . 11

3 Data **12**
- 3.1 Dataset . . . 12
- 3.2 MegaDetector . . . 13
- 3.3 The Data Pipeline . . . 14
- 3.4 Data Preprocessing . . . 16

4 Machine Learning & Experiments **17**
- 4.1 The Models Used . . . 17
- 4.2 Training Setup . . . 17
- 4.3 Baseline Model Comparisons & Imbalance Correction . . . 19
- 4.4 Season-Based Generalisation . . . 19
- 4.5 XAI . . . 20
- 4.6 Box-cropping . . . 23
- 4.7 MC Dropout Uncertainty . . . 26

5 MLOps **28**
5.1 System Overview 28
5.2 Model Deployment 29
5.3 Drift Detection 33
5.4 Model Retraining 34
5.5 Human Oversight/Governance 36

6 Conclusion **39**
6.1 Findings 39
6.2 Limitations 40
6.3 Ethical and Practical Considerations 41
6.3.1 Ethical Considerations 41
6.3.2 Practical Considerations 41
6.4 Future Work 41
6.5 Conclusions 42

Chapter 1

Introduction

1.1 Problem Statement

New Zealand's native biodiversity is uniquely vulnerable to invasive predator species, many of which have contributed to significant population declines among endemic birds and reptiles [14]. Ongoing conservation efforts increasingly rely on large-scale monitoring to inform intervention strategies and assess ecological outcomes. As part of these efforts, camera traps have become a widely adopted tool for collecting visual data across diverse and often remote environments [7].

The scale at which camera traps are deployed across Aotearoa New Zealand produces vast quantities of visual data, making manual analysis labour-intensive, time-consuming, and costly. As a result, automated approaches based on computer vision have been increasingly adopted to support the detection of pest and predator species within camera trap imagery [29, 24]. These systems offer the potential to significantly reduce human workload while enabling more timely conservation responses.

However, many existing image-based classification systems prioritise overall classification accuracy as the primary evaluation metric. In camera trap datasets, this can be misleading due to class imbalance, where images of one category may be substantially more frequent than the other [38]. In such cases, a model may achieve high accuracy while still performing poorly on ecologically important classifications.

In this work, the positive class is defined as mammal detections, while the negative class corresponds to bird detections. Under this framing, false positive errors arise when birds are incorrectly classified as mammals, potentially leading to inappropriate conservation responses, while false negatives correspond to missed mammal detections that reduce monitoring effectiveness (Figure 1.1). These error types carry different ecological and operational consequences and are not captured by accuracy alone. As a result, additional performance measures such as precision and recall are required to evaluate system reliability, particularly when considering deployment in dynamic conservation environments where class distributions and error characteristics may change over time.

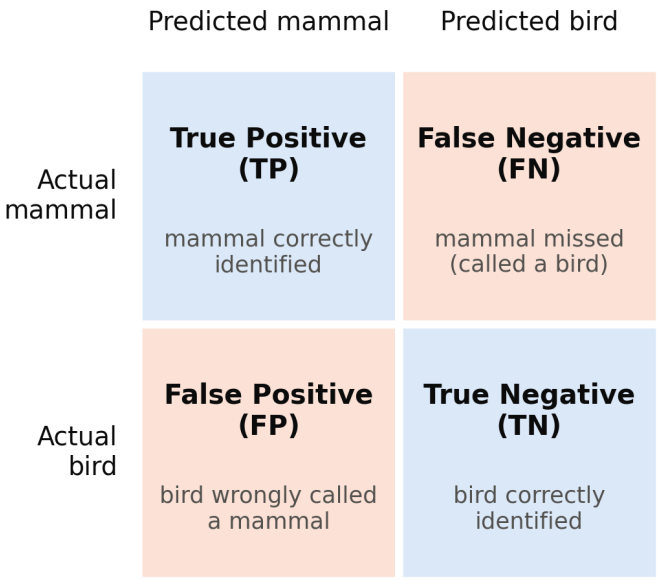

Figure 1.1: Confusion matrix illustrating classification outcomes for mammal and bird detections.

1.2 Motivation

National initiatives such as Predator Free 2050 highlight the importance of effective monitoring and management of invasive species across New Zealand. Tompkins [39] argues that achieving these goals requires scalable and reliable monitoring approaches capable of operating across diverse ecological environments.

Meeting the demands of such initiatives requires more than a single machine learning model trained on a fixed dataset. Conservation monitoring systems must operate at scale, adapt to changing environmental conditions, and maintain reliable performance over time. In particular, while high classification accuracy is important, minimising false positive and false negative detections is critical to avoid unnecessary or harmful interventions affecting native species. This places additional constraints on the design and evaluation of automated wildlife monitoring systems.

However, many applications of machine learning in wildlife monitoring focus primarily on model development and offline evaluation, with limited consideration of how these models are deployed, monitored, and maintained over time [40]. In real conservation settings, models are exposed to changing environmental conditions, evolving data distributions, and operational constraints that can degrade performance after deployment. Addressing these challenges requires a lifecycle-oriented perspective that extends beyond model accuracy to include data management, continuous monitoring, retraining, and system governance.

Furthermore, conservation practitioners require confidence in the outputs produced by automated systems, especially when these outputs inform management decisions. Black-box models that provide predictions without meaningful explanations can hinder trust and limit effective error diagnosis [28]. This lack of transparency also reduces the practical usability of machine learning tools in conservation settings. As a result, there is growing mo-

tivation to incorporate explainability and lifecycle-aware practices into wildlife monitoring systems, ensuring that automated approaches remain transparent, robust, and suitable for long-term deployment.

1.3 Goals and Hypotheses

The goal of this research project is to build an end-to-end machine learning pipeline for classification of mammals versus birds using New Zealand trail camera imagery from raw data through to a served, monitored system. This system tests whether the deployment-aware design principles argued in my AIML501 report hold up against real data.

To do this, the project tests the following five hypotheses:

H1 A model trained on warm-season imagery will lose performance when tested on cool-season imagery, due to drift from seasonal change.

H2 A high-accuracy model can still be untrustworthy because of how it makes its predictions, such as not looking at the animal.

H3 Box-cropping, chosen because of what the explainability analysis shows, can reduce the model's reliance on the background and its false positives.

H4 Monitoring signals can detect drift on new data, and retraining on the combined data can recover performance without hurting the model's performance on the original data.

H5 Model uncertainty can pick out wrong predictions well enough to decide which ones a person should review, and keeping a person in the loop catches failures that offline evaluation misses.

Together, testing these hypotheses examines whether a deployment-aware, lifecycle-managed system, one that accounts for distribution shift, explains its own reasoning, and keeps a human in control of consequential decisions, is a meaningfully different and more trustworthy proposition than a black-box classifier optimised for accuracy alone.

Chapter 2

Background

Aotearoa New Zealand has a predator problem which is monitored using camera traps that produce too many images for humans to monitor manually. The solution is to build a model using computer vision to automate the classification process, but accurate models are not automatically trustworthy as they may be correct for the wrong reasons and deployed models degrade over time due to the change in data.

This chapter reviews literature behind each step of the argument. It looks into the predator problem and traditional methods of dealing with it, how computer vision and machine learning can automate this process, how XAI can help lead us towards a more trustworthy model, and finally, how the MLOps process is able to maintain a deployed model's performance.

2.1 Invasive Predator Species in New Zealand

New Zealand's biodiversity is characterised by a high proportion of endemic species that evolved in the absence of terrestrial mammalian predators. As a result, the introduction of invasive predator species such as rats, stoats, and possums has had a significant and ongoing impact on native bird, reptile, and invertebrate populations. These predators are widely recognised as one of the primary drivers of biodiversity loss across Aotearoa New Zealand [39]. National conservation initiatives have been established in response to this challenge, most notably the Predator Free 2050 programme, which aims to eliminate key invasive predator species across large regions of the country. Tompkins [39] emphasises that achieving such goals requires sustained, large-scale monitoring to inform management strategies, evaluate intervention effectiveness, and adapt conservation efforts over time. Accurate detection and identification of predator species is therefore a critical component of modern conservation practice in New Zealand. However, the spatial scale and environmental diversity of conservation areas present substantial challenges for traditional monitoring approaches. These challenges have driven increasing interest in automated monitoring technologies capable of supporting long-term, large-scale ecological surveillance.

2.1.1 Wildlife Monitoring Practices

Traditional wildlife monitoring has historically relied on human-led methods such as field surveys, direct visual observation, physical trapping, and acoustic identification. These ap-

proaches have been widely used in ecological research and conservation management to estimate species presence, distribution, and abundance. However, their effectiveness is often constrained by practical limitations related to scale, cost, and consistency. Zwerts et al. [46] provide a comprehensive comparison of wildlife monitoring techniques, including human observation, camera trapping, and passive acoustic monitoring. Their review highlights that manual monitoring methods are typically labour-intensive, require specialised expertise, and are limited in both spatial and temporal coverage. In remote or densely vegetated environments, detection probability may also be affected by observer bias and environmental conditions, leading to variability in data quality across sites. As conservation initiatives increasingly operate at national or regional scales, these limitations pose significant challenges for maintaining continuous and standardised monitoring programmes. Zwerts et al. [46] note that automated approaches, such as camera traps, offer advantages in terms of scalability and long-term deployment, motivating a shift toward sensor-based monitoring systems in modern conservation practice. This transition has generated large volumes of ecological data, creating new opportunities and challenges for automated analysis using artificial intelligence techniques.

2.1.2 Camera Trap Technology

Camera traps have become a widely adopted tool in wildlife monitoring due to their ability to collect large volumes of visual data with minimal human intervention. These devices are typically motion or heat triggered and can be deployed for extended periods across diverse environments, making them particularly suitable for monitoring elusive or nocturnal species.

Recent datasets, such as the Trail Camera Images of New Zealand Animals collection [23], demonstrate both the scale and complexity of camera trap data used in conservation contexts. These datasets typically comprise large volumes of images captured under varying lighting conditions, weather patterns, and seasonal changes, often in remote outdoor environments.

In contrast to many wildlife monitoring datasets where target species are rare, camera trap data collected as part of predator control and monitoring programmes in New Zealand often exhibits class imbalance skewed toward animals of management interest. This reflects the targeted placement of camera traps in areas of known predator activity, as well as the operational goals of pest detection and management. Depending on the level of label granularity, this imbalance may manifest at the species level or at broader categorical groupings, such as mammals versus birds, with non-target or native classes underrepresented relative to animals of management concern.

Camera trap imagery also frequently suffers from low image quality and environmental noise. Images may be affected by motion blur, low resolution, poor illumination (particularly in nocturnal settings), partial occlusion of animals, and false triggers caused by vegetation movement or weather conditions [35]. These characteristics complicate both manual annotation and automated analysis, and they place specific demands on machine learning systems designed for reliable predator detection.

2.2 Computer Vision and Machine Learning

2.2.1 CNN

The emergence of deep learning significantly transformed computer vision research, particularly through the development of convolutional neural networks (CNNs). CNNs are a class of neural networks specifically designed to process grid-structured data such as images. Rather than relying on manually engineered features, CNNs learn hierarchical feature representations directly from raw pixel values through a sequence of convolutional and non-linear operations. Lower layers typically capture simple visual patterns such as edges and textures, while deeper layers learn increasingly abstract and semantically meaningful features related to shapes and object structures [45].

A key advantage of CNN-based approaches is their ability to exploit the spatial structure of images through shared weights and local receptive fields, enabling efficient learning while maintaining robustness to small translations and distortions in the input data. As a result, CNNs have become the dominant paradigm for image classification tasks and form the foundation of most modern computer vision systems [45]. Figure 2.1 shows a CNN's architecture.

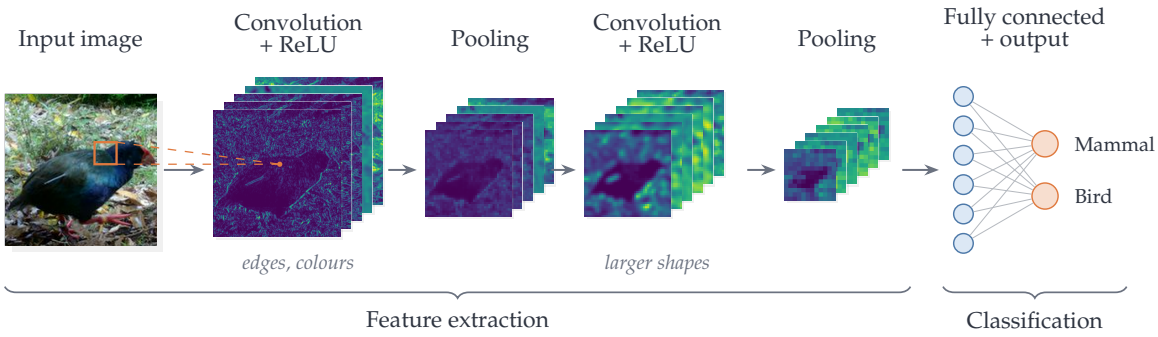

Figure 2.1: Architecture of a convolutional neural network (CNN), illustrating convolutional feature extraction, pooling operations, and fully connected classification layers. The feature maps are illustrative.

2.2.2 ViT

More recently, transformer-based architectures have been introduced for visual recognition and object detection tasks. Vision Transformers (ViTs) [8] apply self-attention mechanisms to image representations, enabling models to capture long-range dependencies and global contextual information within visual scenes [17]. This capability can be advantageous in complex environments where relevant visual cues may be spatially distributed across an image. While Vision Transformers often require substantial training data and computational resources [33], they represent an increasingly important direction in modern computer vision research and are beginning to be explored in conservation and ecological applications. Figure 2.2 shows a ViT's architecture.

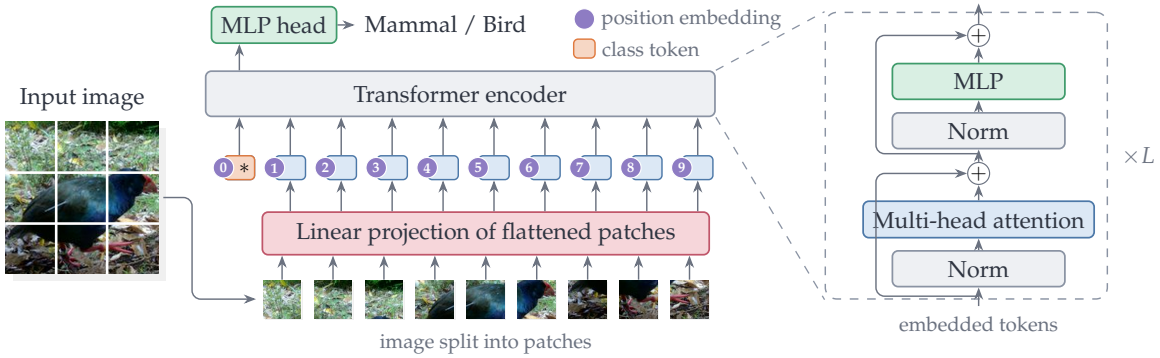

Figure 2.2: Vision Transformer (ViT) architecture, showing image patch embedding, positional encoding, stacked self-attention blocks, and the final classification head. Adapted from Dosovitskiy et al. [8].

2.2.3 Pre-Trained Models

In practical wildlife monitoring applications, computer vision systems frequently leverage pre-trained models that have been trained on large-scale datasets and subsequently fine-tuned for domain-specific tasks. This project makes direct use of that approach at two separate stages. The primary classifier, EfficientNet-B3 [36], and a comparison model, ResNet-50 [13], are both initialised from ImageNet-pretrained weights rather than trained from scratch. Separately, MegaDetector [2], a pre-trained camera-trap object detector built on the YOLO family's unified feature-extraction-and-localisation framework [26, 16], is used throughout this project to locate the animal within each image, rather than training a detector from scratch. Using pre-trained models for both classification and detection reduces data requirements, improves convergence, and improves generalisation, which is particularly valuable in a conservation setting where labelled data is limited and expensive to collect. Within conservation and ecological research more broadly, deep learning-based computer vision models have been widely adopted to automate the analysis of camera trap imagery [24].

2.2.4 Evaluating classifiers

Another important consideration in wildlife image classification is the relationship between evaluation metrics and conservation objectives. While overall classification accuracy is commonly reported in the literature, it may provide limited insight in settings characterised by class imbalance or asymmetric error costs [38]. In predator monitoring contexts, recall is often of particular importance, as failing to detect an invasive species may have greater ecological consequences than incorrectly flagging a non-target image. However, in this project, precision is the most important measurement while recall is maintained above a 0.90 minimum floor. This is due to the fact that any false positives could trigger an unnecessary and potentially harmful intervention to birds. These considerations highlight the need for careful interpretation of computer vision model performance within applied conservation settings.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2.1}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2.2}$$

Overall, computer vision provides a powerful set of tools for automating wildlife image analysis, but its effective application depends on both a sound understanding of underlying techniques and an awareness of domain-specific challenges. These factors motivate further consideration of complementary approaches, such as explainable artificial intelligence and machine learning lifecycle management.

2.3 Explainable AI (XAI)

As machine learning models, particularly deep neural networks, have become increasingly complex, concerns have emerged regarding their transparency and interpretability. In wildlife surveillance specifically, convolutional neural networks are commonly treated as "black box" models: feature extraction and classification occur automatically within the network, and even when a model achieves high classification accuracy, its learned representations and decision process remain difficult to interpret [5]. This lack of transparency is a particular problem in conservation settings, where automated predictions may inform management actions such as pest-control interventions, and an incorrect classification, especially a false positive, can lead to unnecessary disturbance or harm to a non-target native species. A lack of transparency more generally can reduce user trust, hinder error diagnosis, and make it difficult to assess whether a model relies on meaningful visual cues or spurious correlations [44]. When a model relies on cues such as the background instead of the object itself, this is known as shortcut learning [11]. Explainable Artificial Intelligence (XAI) is a response to this opacity: a collection of methods aimed at making a model's decision-making process understandable to a human, so that a prediction can be inspected rather than simply trusted [6]. Explainability methods are commonly distinguished as either intrinsic or post-hoc [15].

Intrinsic explainability refers to models that are interpretable by construction [20], such as linear models or decision trees, but these lack the expressive capacity required for a task like this one. Post-hoc methods instead explain a complex, already-trained model after the fact, without altering it, and represent a pragmatic compromise given that the models used for this task, EfficientNet-B3 and ResNet-50, are chosen for their predictive performance, not their interpretability. This project relies exclusively on post-hoc methods for that reason.

Within computer vision, post-hoc methods are commonly grouped into attribution-based, activation-based, perturbation-based, and transformer-specific approaches [41, 6]. This project uses two attribution-based methods specifically: Grad-CAM++ [4], an extension of the original Grad-CAM method [30] that produces a class-specific heatmap by weighting a convolutional layer's activations by their gradient with respect to the target class, and Integrated Gradients [34], which instead attributes a prediction to individual input pixels by integrating the model's gradient along a straight-line path from a blank baseline to the actual image. Perturbation-based methods, which explain a prediction by observing how the output changes as parts of the input are deliberately masked, are often considered more faithful

to the underlying model but are computationally expensive [6], which is part of why this project does not rely on them. Transformer-specific methods do not apply here, since this project does not use a transformer-based vision model.

A caution raised repeatedly in the XAI literature is directly relevant to how this project uses explainability: post-hoc methods do not necessarily provide a faithful representation of a model's internal decision-making process, only an approximation intended to support human understanding [27]. Grad-CAM++ and Integrated Gradients differ substantially in mechanism and theoretical grounding: Grad-CAM++ approximates importance from a single convolutional layer's gradients, while Integrated Gradients satisfies a completeness axiom that Grad-CAM++ does not: its attributions sum exactly to the difference between the model's output on the real input and on the baseline [34]. Using both, and treating agreement between them as stronger evidence than either alone, is this project's direct response to that caution, rather than trusting a single method's heatmap at face value.

In predator detection specifically, explainability is most valuable for diagnosing false positive classifications: assessing whether a prediction is driven by ecologically meaningful animal features, such as shape or texture, or by spurious cues such as background motion, lighting artifacts, or vegetation patterns [6, 5]. This project uses Grad-CAM++ first in exactly this way, scoring the model's attention against MegaDetector's bounding box for every false positive, then extends the same approach into a full audit across every prediction outcome, not just the errors, since a false-positive-only analysis cannot say whether background-reliance is a property of the model's mistakes specifically or of how it solves the task in general.

Explainability techniques are not without trade-offs. They add computational overhead, and a heatmap can be difficult to interpret reliably or actively misleading if taken at face value, for instance by highlighting a visually prominent background region rather than the animal itself. The literature is consistent that explanations should be treated as a supportive diagnostic tool, not a definitive account of a model's reasoning [6], which is the same reasoning behind this project's decision to cross-check one method against a second rather than reporting either in isolation. In conservation-oriented predator detection systems, explainability plays a critical role in bridging the gap between technical performance and practical usability. This role extends beyond a one-off diagnostic in this project: the same heatmaps are shown to the person reviewing uncertain predictions in the deployed system, so they can see what the model was looking at before deciding whether to trust a prediction. This motivates the discussion of explainability's place within a wider machine learning lifecycle, addressed in Section 2.5.

2.4 Uncertainty Estimation

Explainability methods show which parts of an image a prediction was based on, but they do not show how certain the model is about that prediction. A second way of judging whether a prediction can be trusted is to estimate the model's uncertainty. A model's confidence score is not a reliable measure of how sure it is, because modern neural networks are often overconfident, providing high probabilities despite being incorrect [12].

One way to estimate uncertainty is Monte Carlo (MC) Dropout, which is built on dropout. Dropout randomly switches off parts of the neural network while training to prevent over-

fitting, and dropout is normally turned off at prediction time [32]. MC Dropout is when dropout is kept on during prediction time and the same image is run through the model several times, with each pass going through a slightly different network. The spread of the predictions produces the uncertainty score [9]. This score allows us to decide what to do with uncertain predictions, whether to act on them automatically or pass them on to a human. How uncertain predictions are handled once a model is deployed, and who acts on them, is part of the wider machine learning lifecycle discussed in the following section.

2.5 MLOps

Machine learning models developed for wildlife monitoring must operate reliably beyond the training environment, adapting to changing data distributions and maintaining performance over time. While advances in computer vision have improved species classification accuracy, real-world deployment introduces additional challenges related to monitoring, retraining, and governance. Machine Learning Operations (MLOps) has emerged as a discipline that extends traditional software engineering and DevOps practices to machine learning systems, providing structured methodologies for managing the full lifecycle of a deployed model [37]. Unlike workflows that treat model training as the final stage of development, MLOps emphasises continuous processes that keep a system effective after deployment [19]. A structured MLOps pipeline has been proposed consisting of ten interconnected stages, including data acquisition, model training and testing, continuous integration and delivery, continuous monitoring, explainability, and sustainability [37]. This lifecycle-oriented view is directly reflected in this project's own implementation: a data pipeline and model artifacts versioned through DVC [18], every training run and evaluation tracked through MLflow [43], a continuous-integration gate running on every code change, and a dedicated monitoring and retraining stage discussed below. Evaluating a deployed model solely on laboratory metrics such as accuracy or F1-score gives limited insight into its real-world behaviour: a model can perform well on a static test set while degrading under deployment conditions shaped by data drift, sensor noise, or a changing species mix [37].

Recent work has characterised MLOps platforms by the specific operational functions they provide, including data versioning, pipeline versioning, model deployment, CI/CD availability, performance monitoring, and experiment versioning, comparing tools such as Kubeflow, MLflow, ClearML, and AWS SageMaker along these dimensions [42]. This project's own tooling sits within that same landscape, though it is a smaller subset: DVC [18] provides data and pipeline versioning, and MLflow [43] provides experiment tracking, while heavier managed platforms were judged unnecessary at this project's scale. A key component of MLOps is continuous monitoring, which detects performance degradation after deployment. Models are susceptible to decay as input distributions shift over time, commonly termed data or concept drift, and monitoring allows practitioners to identify when predictions become unreliable and to trigger retraining [37].

Two main types of drift are commonly described. Data drift, also called covariate shift, is when the input data changes while what each label means stays the same, for example when a camera trap model sees new reserves, cameras, seasons or lighting conditions. Concept drift is when the relationship between the input and its label changes, so the same image would now need a different answer [10]. This project implements two deliberately differ-

ent drift signals. The first is confidence drift, which measures whether the distribution of predicted confidence shifts without requiring any ground truth [25]. The second is concept drift, which measures whether labelled correctness actually degrades [10]. This would catch concept drift, but also data drift that makes the model less accurate.

Explainability is increasingly recognised as a core operational requirement within MLOps, not merely a post-hoc analytical tool, and is explicitly included as its own stage within some proposed MLOps lifecycles [37]. This project treats explainability the same way: the same attribution methods are reused directly in the deployed system in Chapter 5, where the heatmaps are shown to the person reviewing uncertain predictions, rather than being confined to a one-off offline analysis. Adopting MLOps practices introduces operational costs beyond standalone model development, arising from data storage, experiment tracking, model versioning, continuous integration, and deployment monitoring [1]. In a conservation setting, where funding is often limited, these costs influence real architectural decisions, such as how much of a system runs on managed cloud infrastructure versus free or already-available compute, a consideration that shaped several of this project's own tooling choices. A further recurring principle in production machine learning, and one this project adopts throughout its own MLOps design, is that consequential automated decisions should remain visible to, and ultimately actioned by, a human rather than an autonomous system [22, 31].

Overall, MLOps provides a structured framework for addressing the limitations of a standalone computer vision model when applied to long-term wildlife monitoring. By combining lifecycle management, continuous monitoring, explainability, and reproducibility, it supports a system that is not only accurate but robust, transparent, and suitable for deployment in a real conservation setting, the requirements this project sets out to meet in Chapters 3 to 5.

2.6 Research Gap

Existing work shows that deep learning can classify camera trap images with high accuracy [24, 35], but much of this work focuses on building the model and testing it on a fixed test set, with less attention on how it is deployed and maintained [40]. Other work shows that these models can do much worse at new sites [3] and can rely on shortcuts such as the background [11], and XAI has been used to make ecological models easier to understand [28]. MLOps work describes how to keep models working after they are deployed [37], but it is usually written for machine learning in general and not for conservation. These areas are mostly studied separately. There is little work that takes a camera trap model through its whole lifecycle, using explainability to find a problem, fixing it, then monitoring the deployed model for drift, retraining it and keeping a person in control, while testing each of these steps on real data. This project does this for mammal and bird classification on New Zealand camera trap images, through the five hypotheses set out in Section 1.3.

Chapter 3

Data

3.1 Dataset

The dataset used for this project is LILA's "Trail Camera Images of New Zealand Animals" [23]. The data is stored and accessed through an S3 bucket which contains over 2.4 million labelled images from 57 contributing projects and 3,127 camera locations. Each image record carries the location, species, project and datetime. They provided a JSON file that had images, annotations and categories, with the annotations being image-level only. There were no bounding boxes provided by humans anywhere in this dataset. This is shown in Table 3.1.

	Value
Images	2,453,840
Species categories	97
Contributing projects	57
Camera locations	3,127
Date range	2010–2023
Images missing a timestamp	7.1%
Mouse images	1,223,161 (49.8%)
Images from the largest single project	1,226,829 (50.0%)

Table 3.1: Summary of the raw LILA "Trail Camera Images of New Zealand Animals" dataset.

When performing my initial exploratory data analysis on the dataset, I found that there were zero images with an empty label, which meant I could cleanly make it into a two-class problem, mammal or bird. The datetime showed that these images were collected from 2010–2023 with only 7.1% of the raw dataset missing timestamps. This means most images can be assigned a season from their timestamp.

The dataset is also heavily imbalanced at the species level. Mice are by far the most common class, accounting for approximately 1.22 million images, around half of the entire dataset as shown in Figure 3.1. This dominance is largely the product of a single project: one of the 57 contributing projects provides roughly 1.23 million images, 78% of which are mice, and accounts for 78% of all mouse images in the dataset. This is an indication that the prevalence of mice is a reflection of a single monitoring deployment. With the data being

left as it is, the model has a high chance to grow a bias towards mice as well as the camera site of that single project.

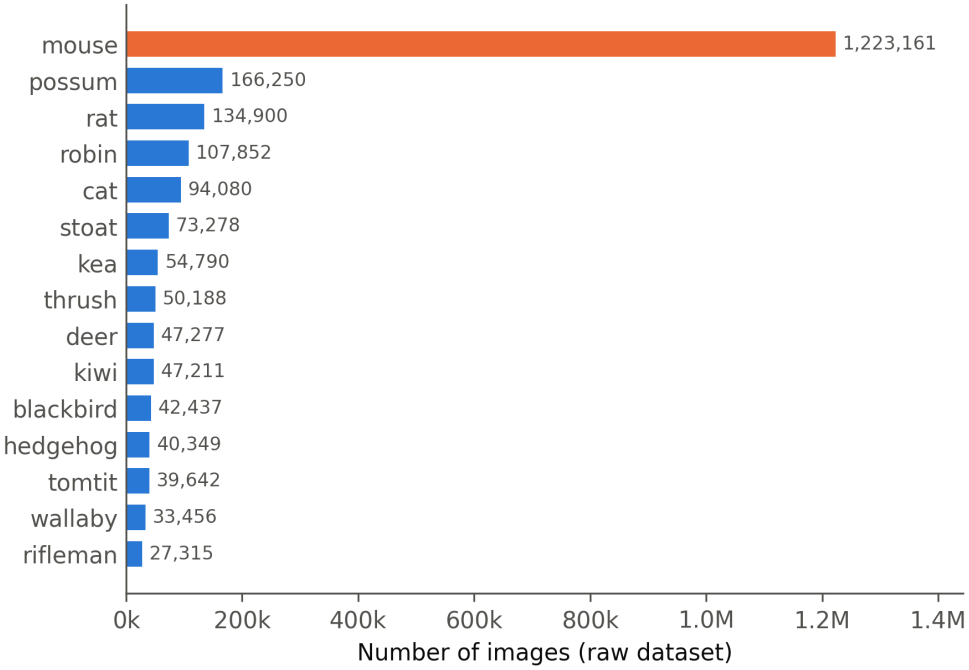

Figure 3.1: The 15 most common species in the raw dataset. Mice account for about half of all images.

3.2 MegaDetector

LILA ran their dataset through a MegaDetector and at a standard 0.2 confidence threshold, 89.8% of images carry a bounding box to the animal. In this project, detection coverage means the proportion of images where there is at least 1 detection at or above the 0.2 confidence threshold. The detection coverage varies by species shown in Table 3.2:

Species	Images	Detection coverage
Cat	94,080	99.4%
Kiwi	47,211	98.9%
Possum	166,250	97.3%
Robin	107,852	97.3%
Rat	134,900	96.4%
Kea	54,790	96.2%
Tomtit	39,642	95.6%
Stoat	73,278	95.1%
Mouse	1,223,161	84.4%
All species	2,453,840	89.8%
All species except mice	1,230,679	95.2%

Table 3.2: MegaDetector detection coverage: the share of images with at least one detection at or above a confidence of 0.2.

Despite mice being the largest class, they have the lowest coverage as shown in Table 3.2. It is important to note that these are model predictions and are not ground truth. I also checked LILA's separate 1.1 million human-annotated MegaDetector training boxes for any overlap with the images used in this project and there was very close to nothing overlapping, therefore anything built on MegaDetector boxes must be described as evaluated against pseudo-ground-truth, not the real annotations.

Due to LILA running their images through MegaDetector, I was able to have bounding boxes for most images in the dataset. In the data pipeline, these boxes are used to drop any images the detector failed to process and to choose the highest-confidence frame from each event (Section 3.3), and they can also be used to crop an image to the detected animal.

3.3 The Data Pipeline

Before the pipeline is run, the raw metadata file is parsed and flattened into a single file. This is so that every later step reads from the same fixed snapshot of the dataset rather than re-parsing the raw JSON as reparsing would be computationally expensive and wasteful. The data pipeline itself has 4 main stages, shown in Figure 3.2, where the first three run within a single DVC stage that builds the data manifest, and the final step is a separate DVC stage that ingests the images.

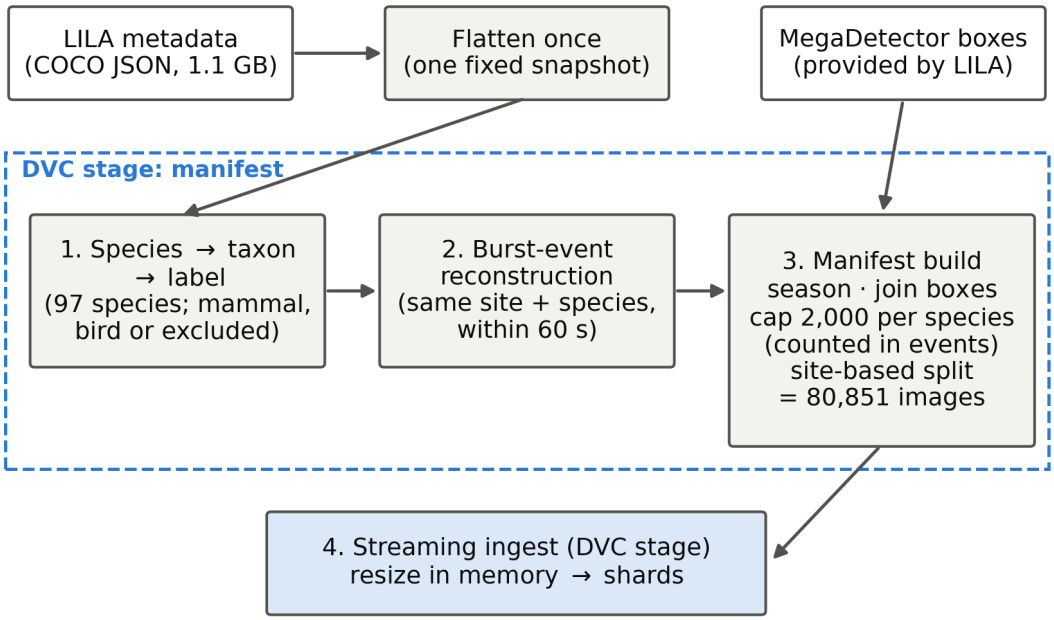

Figure 3.2: The data pipeline, from the raw LILA metadata to image shards. Steps 1–3 run in a single DVC stage, and step 4 is a separate DVC stage.

1. Species - Taxon - Label: The species labels used in this project are human annotations supplied with the dataset, and class labels are derived from them by a fixed mapping. Therefore, they are ground truths unlike the MegaDetector boxes described in the section above. This part of the pipeline is a two-layer separation. The first layer is a taxonomy layer

which is a fixed taxonomic mapping, all 97 species are mapped to either Mammal, Bird or Other; as well as native/wild flags. The second layer is a separate class-mapping layer. This classifies which taxon is the positive class, which is the negative class and which is excluded. The 817 images in the human category are excluded here, so the model is never trained on photos of people. This two-layer split was created to stop unmapped species from defaulting to birds. In this part of the data pipeline, I decided to frame the problem as mammal vs bird instead of native animal vs predator. A native vs predator framing would have to exclude any image that is neither native nor a predator, such as non-native birds (17.0% of the final dataset, and about 30% of all bird images) and mammals that are not predators, such as deer, livestock, rabbits and wallabies (25.6%). Together this would have removed around 43% of the dataset, while the mammal vs bird framing keeps every image obtained from LILA that is a mammal or a bird, maintaining the dataset size.

2. Burst-Event reconstruction: This dataset has no sequence identifier, therefore consecutive frames of the same animal at the same site would otherwise be counted as separate images when each species is capped, allowing a single encounter to fill the cap with near-duplicate frames. Events are reconstructed from what is provided by LILA for their dataset, so the same location, same species, frames within 60 seconds of each other collapse into one event, and only the highest-confidence frame from each event is kept. The train/validation/test split itself is made per site rather than per event (step 3), and since every event belongs to a single site, no event can span two splits. As for the time-stampless rows, they are never merged into one event and are never dropped; each is explicitly flagged and carried forward as its own singleton event.

3. Manifest build: This is the step that ties everything together in my data pipeline, in a deliberate order: Firstly, it resolves species to a label. Secondly it assigns a season from the timestamp. Then it groups images into burst events (step 2), joins in the detection boxes and drops any images that the detector failed to process, so that a failed image can never be chosen to represent an event. Afterwards it caps each species at 2,000 images, counted in events rather than individual frames as described in step 2, and the same rule applies to every species symmetrically. This addresses the mouse dominance described in Section 3.1. Finally, this layer splits the data into train/validation/test at the site level, so no camera location appears in more than one split. This means test performance measures how well the model generalises to cameras it has never seen, rather than rewarding it for memorising the backgrounds of familiar sites. This matters because Beery et al. [3] showed that camera trap models that do well at the sites they were trained on often do much worse at new sites. Images with no recorded location are grouped into one single site for the split, which guarantees no leakage but comes at a cost of losing site diversity for that slice. The resulting manifest contains 80,851 images, one per event, split into 57,761 training, 8,012 validation, and 15,078 test images.

4. Streaming ingests into shards: This final layer converts manifest rows into resized image archives called shards without ever writing a full-resolution image to local disk. Each image is streamed, decoded and resized in memory, then written straight into the archive. This is done because full-resolution images run several megabytes each and with the manifest being 80,851 rows, storing all images at full resolution would cost hundreds of gigabytes which is too expensive.

3.4 Data Preprocessing

Depending on the experiment taking place, there are two sets of preprocessing steps that happen.

Every image, when loaded, goes through a transform: it is resized to 300×300, converted to a tensor and then normalised using the standard ImageNet mean and standard deviation. This happens to each image as it is loaded.

Images can also be cropped to their detected animal's bounding box, with a padding margin of 15% of the box's width and height, before the resize and normalisation steps mentioned above. If no confident detection exists for an image, the full frame is used instead. This cropping is optional and is not applied by default.

Chapter 4

Machine Learning & Experiments

4.1 The Models Used

Throughout the experiments in this chapter, two models were compared against a baseline. The baseline is a majority-class classifier, which always predicts the most common class in the training data regardless of the input. It provides a lower bound that any useful model must exceed, and is used to emphasise my argument in AIML501 that accuracy can be misleading and should not be the driving force behind a model used in conservation contexts. The two models used are a ResNet-50 and an EfficientNet-B3, both fine-tuned from ImageNet-pretrained weights, with EfficientNet-B3 as the primary classifier and ResNet-50 as a widely used architecture for comparison. The EfficientNet-B3 trained on the standard site split, using full images, is referred to as the standard model throughout this project, and is the model that the later experiments are compared against. It is called standard because each later experiment changes one thing about it, such as the data split, the input images or the training data, and is then compared against it to see the effect of that change.

4.2 Training Setup

All the models used throughout all the experiments use the same training setup; the only difference would be the data they receive.

Setting	Value
Initialisation	ImageNet-pretrained weights
Optimiser	AdamW
Learning rate	1×10^{-4}
Weight decay	1×10^{-2}
Learning rate schedule	Cosine annealing
Epochs	30 (final-epoch model used)
Batch size	32
Input size	300×300
Loss	Binary cross-entropy with positive-class weight
Dropout rate	0.3
Data augmentation	None

Table 4.1: Training settings shared by all models in this chapter.

Table 4.1 shows training parameters used for the models. For both architectures, the original classification layer is replaced with a dropout layer and a single output, which gives the probability that an image contains a mammal. The dropout layer is also what allows MC Dropout. To correct for class imbalance, the loss uses a positive-class weight calculated from the ratio of birds to mammals in the training split, so that errors on mammals count proportionally more during training. The validation set is used to monitor training, and the model after the final epoch is used for all results. All models were trained on a GPU on the ECS servers provided by Te Herenga Waka, Victoria University.

Setting	Value
Data split	By camera site, seed 42
Training seed	Not fixed
Positive-class weight	1.265 (32,255 birds / 25,506 mammals in training)
Seasons	Spring Sep–Nov, Summer Dec–Feb, Autumn Mar–May, Winter Jun–Aug
MegaDetector	Version 5a, boxes used at confidence ≥ 0.2, 15% padding on each side
Grad-CAM++	Last convolutional block of EfficientNet-B3 (10×10 map), explaining the predicted class
Integrated Gradients	All-zero baseline after normalisation, 20 steps
MC Dropout	20 passes, only the dropout layer in the classification head ($p = 0.3$) is switched on, all other layers stay in evaluation mode
Review cutoff	Top 20% most uncertain predictions on the validation set
Temperature scaling	$T = 2.84$, fitted on the validation set
Software	PyTorch 2.12, torchvision 0.27, PyTorch Lightning 2.6.4, grad-cam 1.5.5, Captum 0.9.0, Evidently 0.7.21

Table 4.2: Settings needed to reproduce the experiments.

The settings needed to reproduce the experiments are shown in Table 4.2. All models were trained on the site-based split, with 57,761 training, 8,012 validation and 15,078 test

images, and every result in this chapter is on the test set unless stated otherwise. No early stopping was used; the validation set was only used to monitor training, and the model after the final epoch was kept. Each result comes from a single training run, so the variation between runs was not measured, which is discussed in the limitations.

4.3 Baseline Model Comparisons & Imbalance Correction

The majority class baseline achieved an accuracy of 61% while catching zero of the minority class. Its macro F1 collapsed to 38%. This gap is the entire point of this simple baseline model I have built. These numbers point out that despite accuracy looking respectable, the model fails completely to do what is most important.

Given the point made above, the metric reporting convention used here and everywhere after reports accuracy, precision/recall/F1 (mammal-class), plus macro-averaged and micro-averaged versions, plus per-class breakdown. Micro-average is mathematically identical to accuracy in binary classification; it is reported for transparency more than information; macro average weights both classes equally regardless of test size, which is very informative given the class imbalance.

$$\mathrm{F1}_{\mathrm{macro}} = \frac{\mathrm{F1}_{\mathrm{mammal}} + \mathrm{F1}_{\mathrm{bird}}}{2} \tag{4.1}$$

$$\mathrm{F1}_{\mathrm{micro}} = \frac{\sum_c \mathrm{TP}_c}{\sum_c \mathrm{TP}_c + \frac{1}{2}\sum_c (\mathrm{FP}_c + \mathrm{FN}_c)} \tag{4.2}$$

On top of all the metrics reported, I also built a class imbalance correction mechanism. This is a loss function positive-class weight computed automatically from the training split's real bird/mammal ratio.

The two main models, the ResNet-50 and EfficientNet-B3, are both ImageNet-pretrained models. They are trained through identical pipelines and training loops, only differing in architecture.

Model	Accuracy	Precision	Recall	F1	Macro F1	ROC-AUC	PR-AUC
Majority class (baseline)	61.0%	–	0.0%	0.0%	37.9%	0.500	–
ResNet-50	93.6%	87.9%	96.8%	92.1%	93.3%	0.969	0.925
EfficientNet-B3	94.1%	88.6%	97.3%	92.8%	93.9%	0.991	0.987

Table 4.3: Results on the site-split test set (15,078 images) at a threshold of 0.5. Precision, recall and F1 are for the mammal class. The majority-class baseline always predicts "bird".

Table 4.3 shows performance of both models. On every metric reported, EfficientNet outperformed ResNet-50, with the most dominant win on ROC-AUC/PR-AUC. Nevertheless, ResNet-50 remains a credible model comparison as intended.

4.4 Season-Based Generalisation

This was the first major experiment I conducted using the stronger model, EfficientNet. I wanted to see how the models handled different seasons and if they remained trustworthy enough to run throughout the whole year. The experiment is simple: train the models on the

warm-season (Spring and Summer) imagery and test entirely on the cool-season (Autumn and Winter) imagery. This experiment is to test covariate shift. As the relationship between the image and its label does not change over seasons, a mammal stays a mammal and vice versa for birds, but the images themselves do with vegetation growing, landscape changing, weather and lighting varying all year round, as shorter winter days mean more images are taken at night using infrared.

There is also a shift in class imbalance with the data used for this experiment, with mammals making up 46.5% of warm-season images and 40.8% of cool-season images. This experiment tests my hypothesis in AIML501 that seasonal change would degrade model performance. A drop in performance from the model on cool data compared to the normal test set for a train/test split would support my hypothesis while the opposite would disprove it.

The setup for this was that rather than splitting the images by site, I split them by season. The existing site-based split is untouched; this is a separate split dimension. Images with no usable timestamp are excluded from both sides and 15% of the warm season data is held out to monitor overfitting. The remaining 23,562 training images are meaningfully fewer than the 57,761 used to train the standard model, due to the fact it restricts two out of the four seasons from training the model. This means that this experiment can show the effects of seasonal changes but it cannot be put down to purely the effects of seasonal changes.

The results of this were interesting: on the cool-season test set, the model reached 97.3% accuracy (ROC-AUC 0.995), performing better than the standard model did (94.1%). This was counter to my expectation, as in AIML501 I theorised that seasonal change would cause model degradation. After the results were returned, I checked whether this was due to seasonal effects or site familiarity, because "no seasonal drop" could just mean the model memorised each camera's background. To check this, the results are split by whether the test image's site was seen during training.

Test subset	n	Accuracy	Precision	Recall	F1
Seen sites	23,739	97.8%	97.4%	97.1%	97.3%
Novel sites	6,300	95.7%	91.2%	98.6%	94.7%
Total Seasonal Testset (Seen+Novel)	30,039	97.3%	96.1%	97.4%	96.7%
Standard model (site split, all novel)	15,078	94.1%	88.6%	97.3%	92.8%

Table 4.4: Seasonal experiment results split by whether the test image's camera site appeared in training, compared with the standard model.

Table 4.4 shows that despite there being a modest site familiarity effect, the novel site performance still holds up well relative to the standard model. The performance gain on the seen sites raised a question: Is the model learning from the animal? Or is it just learning from the vegetation and background?

4.5 XAI

In AIML501, I claimed that high-performing models can still be black-box models that hide whether its predictions come from meaningful visual cues or spurious correlations. Without the knowledge of the above, a model is not able to be truly trusted. This has now directly affected the models in this project, as it is unclear where the model is picking patterns up

with a nudge towards potentially picking up background site patterns. Hence XAI was added to the project after the seasonal experiment.

Initially I implemented Grad-CAM++ only over false positives, where the model wrongly calls a bird a mammal, measured via a score of what fraction of the heatmap's attention falls inside the detector's bounding box. As expected, on the standard model's false positives it obtained a mean 0.14 and median 0.03. This tells us that a large majority of these errors are background-driven. An example is shown in Figure 4.1.

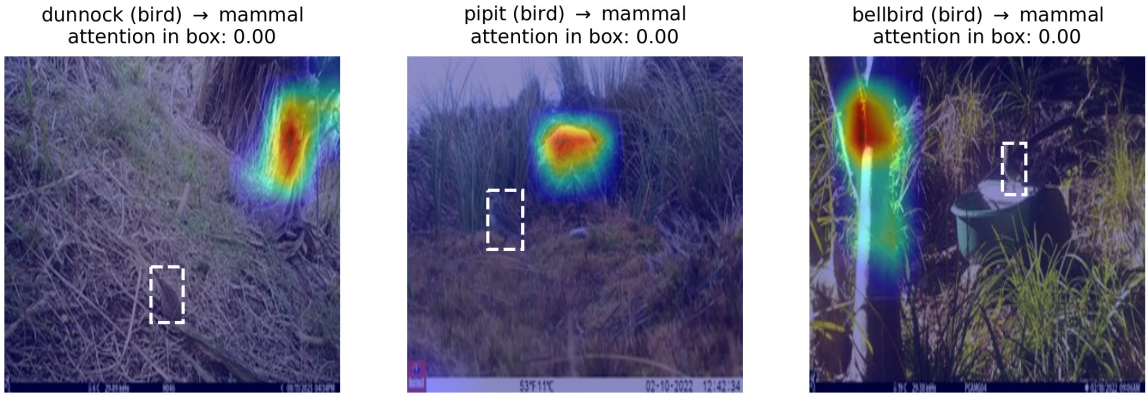

Figure 4.1: Examples of false positives from the standard model: birds classified as mammals while the model's attention falls on the background. The dashed box is the MegaDetector box.

Afterwards, I decided to see if this issue is only tied to the model's mistakes or everything the model predicts, right or wrong, therefore a full attention audit was done. The attention then covered the entire test set, with each heatmap explaining the class the model predicted, and the results were: mammals average 0.67 attention on the animal (median 0.73), but correctly-classified birds only average 0.36 (median 0.32) as shown in Table 4.5. The 13 test images without a MegaDetector box, one of which is a false positive, could not be scored, so the audit covers 15,065 images. This means that the model looks at the animal far less when it predicts a bird, and about a quarter of all correct bird predictions have very little attention on the bird at all (below 0.1). This further reinforces the idea that XAI is essential when working with black box models, as this tells us that the model relies much more on the background for birds than it does for mammals. This is one of the most important findings throughout the project, as it shows the model can be right for partly the wrong reasons, which makes it hard to fully trust.

Outcome	Median	Mean	n
Correct birds	0.32	0.36	8,444
Correct mammals	0.73	0.67	5,727
False positives	0.03	0.14	735
False negatives	0.12	0.21	159

Table 4.5: Attention audit for the standard model: the fraction of Grad-CAM++ attention that falls inside the MegaDetector box, by prediction outcome, on the site-split test set. Each heatmap explains the class the model predicted.

To check that this finding was not specific to Grad-CAM++, I repeated the analysis with a second method, Integrated Gradients (IG), on a sample of 759 images and not the whole test set as this method is computationally expensive. The two methods correlate strongly (Spearman's $\rho = 0.75$) which means both methods largely agree on which images the model focuses on the animal in frame and which it does not. Both methods agree on the same pattern which scores the attention on the animal lower when correctly classifying birds than when correctly classifying the mammals, with IG medians of 0.15 and 0.35 respectively (Table 4.6). The difference in numbers is due to the fact IG is less extreme but nevertheless they agree on pattern which supports the finding of background reliance in the model. However, the two methods do not agree on false positives. Grad-CAM++ gives them the lowest attention (0.07), but IG gives them about the same as correct birds (0.16 compared with 0.15), so the finding that false positives are driven by the background only comes from Grad-CAM++.

Outcome	n	Grad-CAM++ median	Integrated Gradients median
Correct birds	200	0.28	0.15
Correct mammals	200	0.69	0.35
False positives	200	0.07	0.16
False negatives	159	0.12	0.17

Table 4.6: Grad-CAM++ and Integrated Gradients on the same sample of 759 test images from the standard model, both explaining the predicted class. Per-image scores correlate with Spearman's $\rho = 0.75$.

After seeing such a large attention difference between mammals and birds, I decided to test if the MegaDetector box size difference affects the values seen in Table 4.5. This is because a bigger box will catch more of the heatmap's attention just by being bigger, even if the model is not looking at the animal.

Class	Images	Median box size	Mean box size
Birds	9,179	5.3%	9.0%
Mammals	5,886	15.8%	22.8%

Table 4.7: Size of the MegaDetector box as a share of the image, for the 15,065 site-split test images that have a box.

The median bird box is about three times smaller than the median mammal box as shown in Table 4.7. To check this, I grouped the images by how much of the image the box covers and compared the attention within each group (Table 4.8).

Box size (share of image)	Correct birds	Correct mammals	False positives
Under 1%	0.01 (1,260)	0.01 (47)	0.00 (119)
1–2.5%	0.09 (1,583)	0.17 (282)	0.00 (145)
2.5–5%	0.26 (1,117)	0.40 (906)	0.08 (240)
5–10%	0.40 (1,502)	0.56 (809)	0.14 (159)
10–20%	0.59 (1,940)	0.76 (1,167)	0.39 (59)
20–40%	0.77 (863)	0.86 (1,408)	–
Over 40%	0.89 (179)	0.92 (1,108)	–

Table 4.8: Median Grad-CAM++ attention inside the MegaDetector box for the standard model, grouped by how much of the image the box covers. The number of images is shown in brackets. False positives are left out where there are fewer than 20 images.

In almost every group, correctly classified birds still get less attention than correctly classified mammals, so the model does rely more on the background for birds, but the difference is smaller than Table 4.5 shows. False positives have the lowest attention in every group shown, so their low attention is not caused by box size. Box size does explain most of the correct birds with very little attention on the bird (below 0.1), as 90% of them have a box covering less than 2.5% of the image. The Grad-CAM++ heatmap is only a 10 by 10 grid, so when the box is this small, the attention spills outside of it even if the model is looking at the bird.

4.6 Box-cropping

As the aim is a model that is trustworthy enough, it should make its predictions based on the animals. So if the model treats "bird" as a background default, does forcing it to mostly see the detected animal fix that? The model in this case now gets an image that is cropped to the detector's box, with 15% of the box's height and width as padding and falls back to the full frame when there is no confident detection. I decided to go with box-cropping over segmentation because the bounding boxes already existed and were provided with the dataset, while masks were not and would require another model and would be unvalidated pseudo-labels. With MegaDetector already being a part of the data pipeline, the implementation costs are much lower for box-cropping compared to segmentation.

Model	Accuracy	Precision	Recall	F1	FP	FN
Standard model	94.1%	88.6%	97.3%	92.8%	736	159
Box-crop model	98.1%	97.4%	97.9%	97.6%	154	125
Seasonal model	97.3%	96.1%	97.4%	96.7%	485	319
Seasonal box-crop model	97.8%	96.9%	97.7%	97.3%	379	287

Table 4.9: The effect of box-cropping at a threshold of 0.5. The first two rows use the site-split test set; the last two use the cool-season test set.

The results of this experiment on the standard site-split test set showed a large improvement when box-cropping was used (Table 4.9). Accuracy rose from 94.1% to 98.1% and precision from 88.6% to 97.4%, as false positives fell from 736 to 154, a reduction of almost

80%. Recall stayed almost the same, rising slightly from 97.3% to 97.9%.

Afterwards I re-scored the attention audit on the cropped images. The audit was adapted to score the model against the same cropped input it trained on, then remap the heatmap back into the original frame for the sake of comparability. The results show that correctly classified birds' median attention on the bird goes from 0.32 to 0.88, a large improvement, as well as the 0.73 to 0.94 increase mammals get (Table 4.10). The share of correct bird predictions with very little attention on the bird (below 0.1) also drops from about a quarter to 0.3%. This shows that box-cropping largely solves the issue where the model does not look at the animal, as shown in the examples in Figure 4.2.

	Median		Mean	
Outcome	Standard	Box-crop	Standard	Box-crop
Correct birds	0.32	0.88	0.36	0.87
Correct mammals	0.73	0.94	0.67	0.93
False positives	0.03	0.96	0.14	0.86
False negatives	0.12	0.93	0.21	0.90

Table 4.10: Attention inside the MegaDetector box before and after box-cropping, by prediction outcome, on the site-split test set. Each heatmap explains the class the model predicted.

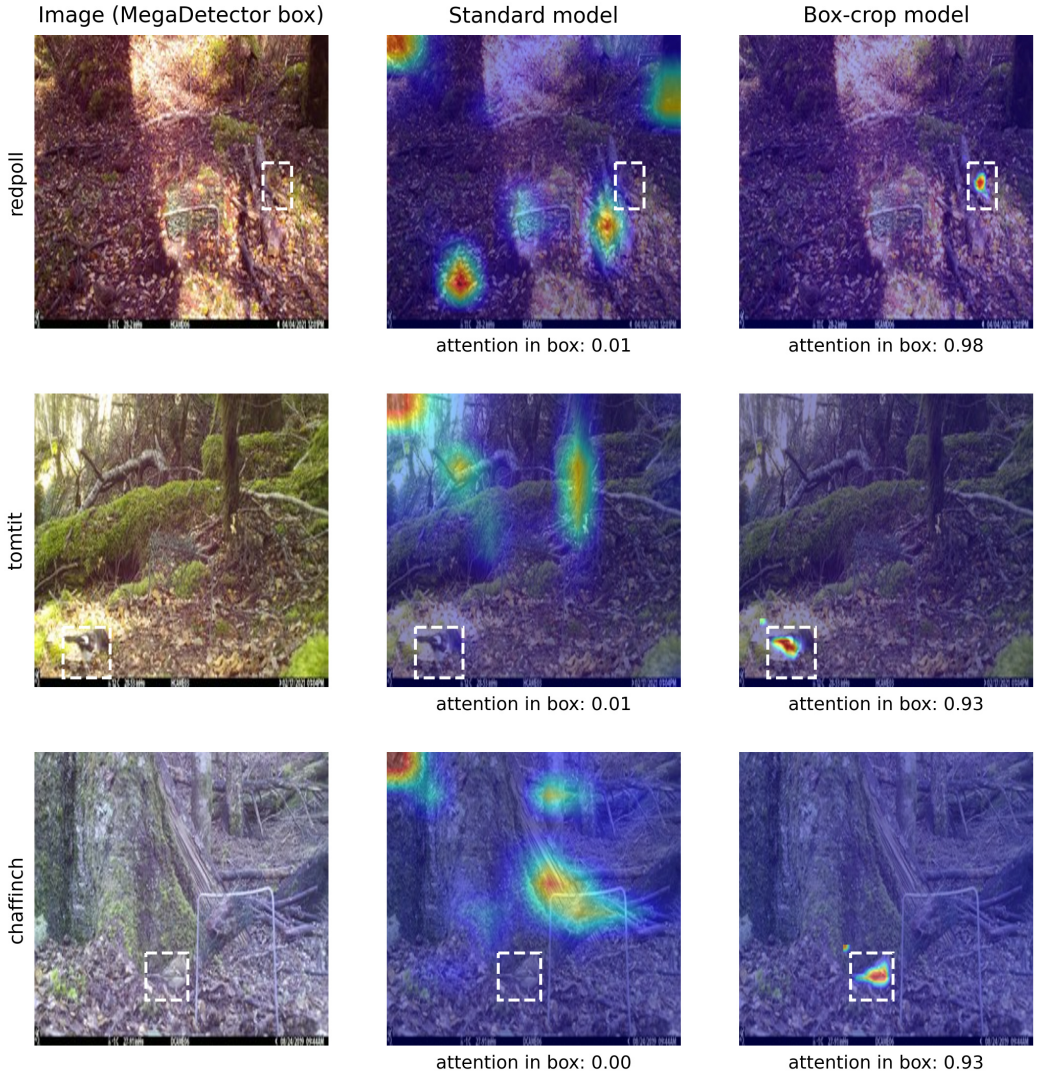

Figure 4.2: Example birds correctly classified by both models. The standard model attends to the background, while the box-crop model attends to the bird. The examples were chosen to show the effect clearly. The dashed box is the MegaDetector box.

Box-cropping does not remove the whole background, as the box itself and its padding still contain some of it. Due to the box now taking up about 60% of the cropped image, part of the rise in attention comes from the crop itself. To check that the model was actually looking at the animal more, I compared each score with what it would be if the attention was spread evenly over the image, where 0 means no better than even and 1 means all of the attention is inside the box, and the median for correctly classified birds still rose from 0.27 to 0.69. On top of this, removing most of the background cut false positives from 736 to 154 which indicates that there was a large reliance on the background which has been reduced. Box-cropping is therefore kept as the main method to reduce background reliance for the rest of this project. To remove more of the background, segmentation could be used, for example by swapping or randomising backgrounds rather than blacking them out, and it would allow a stricter attention audit. This makes segmentation a potential future step.

The same treatment is applied to the seasonal experiment. This resulted in seasonal

accuracy improving slightly (97.3% to 97.8%), and the seen/novel-site gap narrows (2.04 to 1.77 accuracy points; F1 gap 2.54 to 2.27) but does not close. Both box-crop models looked at the animal to the same degree, with a bird-correct median attention of 0.89 for the seasonal model and 0.88 for the standard-split model, despite similar overall accuracy. This informs us that box-cropping made little difference to the seasonal experiment, as the changes in accuracy and the site gap are small enough to be down to chance. However, as both box-crop models now look at the animal, the remaining site-familiarity gap is not caused by where the model looks, and is more likely due to things that cropping does not remove, such as the background inside the box or differences between cameras.

4.7 MC Dropout Uncertainty

Even after box cropping, errors remained in the model's predictions, with 154 false positives and 125 false negatives on the test set. This means the system needs to know which predictions to trust and which not to. MC Dropout uncertainty was introduced to answer the question: is the model actually unsure? MC Dropout runs each image through the model 20 times with dropout left on, and uses the spread of the predictions as the uncertainty score. This flags predictions for human review regardless of which side of the threshold they land on. This was run on the standard model, using the validation set (8,012 images, 234 errors). I chose MC Dropout over other methods such as deep ensembles because it needs no extra training, as EfficientNet-B3 already has a dropout layer in its classification head.

The uncertainty score is a signal for errors. It separates wrong predictions from correct ones with an AUC of 0.92, and the average uncertainty on wrong predictions is about 50 times higher than on correct ones. In practice this becomes a trade-off between catching errors and the amount of work given to a human reviewer. Flagging the most uncertain 10% of predictions catches 76.5% of errors, while flagging 20% catches 87% (Figure 4.3). However, most flagged predictions are still correct: at 10%, only about 1 in 5 flagged images is actually wrong, so the reviewer checks far more images than there are errors. The deployed system flags the most uncertain 20% of predictions.

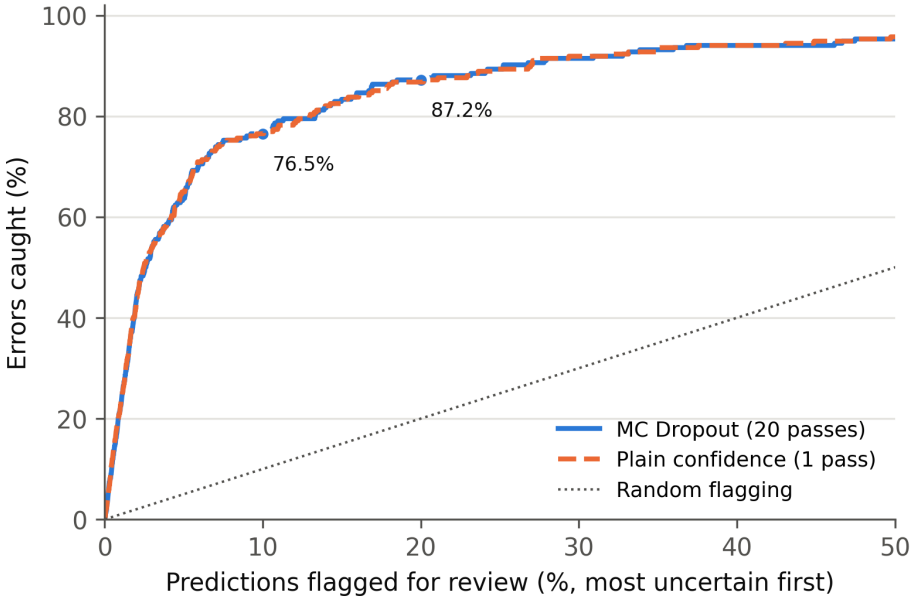

Figure 4.3: Share of errors caught when the most uncertain predictions are flagged for review, for the standard model on the validation set, using MC Dropout and using the model's plain confidence.

I also compared MC Dropout with a simpler score, which is how close the model's single prediction is to 50/50. This caught errors just as well, with an AUC of 0.92 for both, and flagging the most uncertain 20% caught about 87% of errors either way. This means that in this setup, MC Dropout did not add anything over the model's own confidence. This is likely because only the dropout layer in the classification head is switched on, so the spread of the 20 passes mostly reflects how confident the prediction already is.

Chapter 5

MLOps

Chapter 4 showed how the models performed with a fixed test set. In this chapter, I apply MLOps practices to test whether the model is able to maintain its performance over time when it is deployed. I first describe how the system is built and deployed, then specifically test if drift detection is able to detect the signals and if retraining is able to improve the model without hurting its performance on the original data. Finally, I cover the importance of the human in the loop and what decisions they are expected to make for the system.

5.1 System Overview

When a model is trained and evaluated offline, its results reflect only how well it performs on data similar to what it has been trained on. Once this model is deployed, it needs data versioning, experiment tracking, continuous integration and model monitoring as data drift and concept drift occur over time. Therefore this requires an entire MLOps pipeline around the model and not just a model that has been trained once.

The main parts of the system are shown in Table 5.1:

Part	Tool	Why it is needed
Data versioning	DVC	Versions the data pipeline stages, so the dataset can be rebuilt the same way again.
Experiment tracking	MLflow	Logs every training and evaluation run, so all results can be compared side by side.
Testing	GitHub Actions, Codecov	Runs the tests on every push and pull request to main, and reports how much of the code the tests cover.
Serving	FastAPI, Streamlit, Docker Compose	Serves predictions through an API and shows them in a dashboard, with each part running as its own service.
Monitoring	Datadog	Receives the drift results and emails the person maintaining the system when retraining is recommended.

Table 5.1: The main parts of the system built around the model.

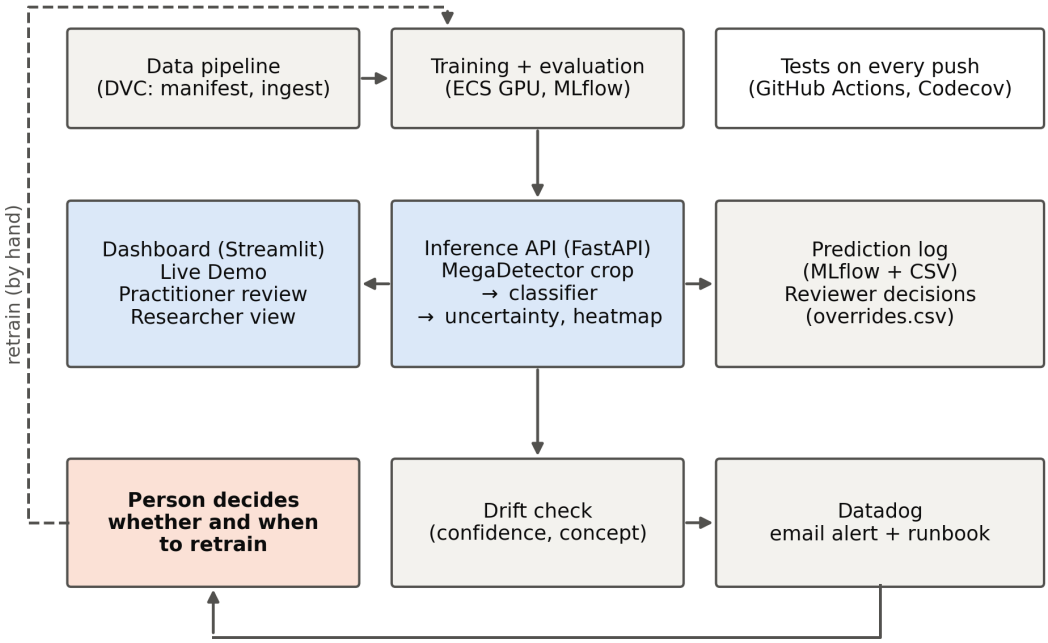

Figure 5.1: The system built around the model. Drift alerts go to a person, who decides whether and when to retrain.

All the tools shown in Table 5.1 play their important role in this system built around the model shown in Figure 5.1, to ensure that the model is logged, monitored and served. The next section looks more closely at the serving part of the system, which is how a single image goes from being uploaded to being given a prediction.

5.2 Model Deployment

To be able to test whether the model could be used in conservation contexts and not just evaluated offline, I deployed the model. This means that users are able to upload one image and receive a logged prediction with a heatmap and an uncertainty score. Deployment contains two main parts, firstly the API that makes the predictions happen and secondly the dashboard displaying all the information for the user. The deployed model is a box-crop EfficientNet-B3. I chose to deploy the box-crop model to keep the background reliance fix talked about in Chapter 4.

There are three FastAPI endpoints that allow this system to function shown in Table 5.2:

Endpoint	Method	Purpose	Returns
`/predict`	POST	Classifies one or more images.	For each image: the label (mammal or bird), confidence, MC Dropout uncertainty, whether it needs human review, and optionally a Grad-CAM++ heatmap.
`/health`	GET	Lets Docker check that the service is running.	A simple "ok" status.
`/model-info`	GET	Shows which model and settings are currently active.	The model name, architecture, checkpoint, decision thresholds, uncertainty review cutoff, and number of MC Dropout passes.

Table 5.2: The endpoints provided by the inference API.

The model is loaded only once when the service starts. Loading the model takes time, so loading it on every request would make the service slow. For each image uploaded, `/predict` is called. Each image is then cropped to the animal using MegaDetector, reverting to the full frame if it is not confident that there is an animal in the frame. Then it classifies it with the deployed threshold of 0.95, applies temperature scaling to the confidence and computes the MC Dropout uncertainty score. A high threshold was chosen because precision is the priority in this project, so the model has to be very sure before it calls an animal a mammal. On the validation set, moving the threshold from 0.05 to 0.95 cut the birds wrongly classified as mammals from 59 to 18, while recall only dropped from 98.4% to 96.9%, staying above the 0.90 floor. After this is complete, the system flags any prediction if uncertainty is above the cutoff regardless of what the prediction was. After flagging uncertain predictions, every prediction is saved in MLflow and a CSV log which the review queue reads from.

The dashboard is built using Streamlit and has two main views and one smaller view, tailored for three different users. The Live Demo view is for anyone looking to try the model. It allows the user to upload an image and press classify to trigger the process talked about above. The view then shows the predicted label, the Grad-CAM++ heatmap and the confidence and uncertainty (Figure 5.2).

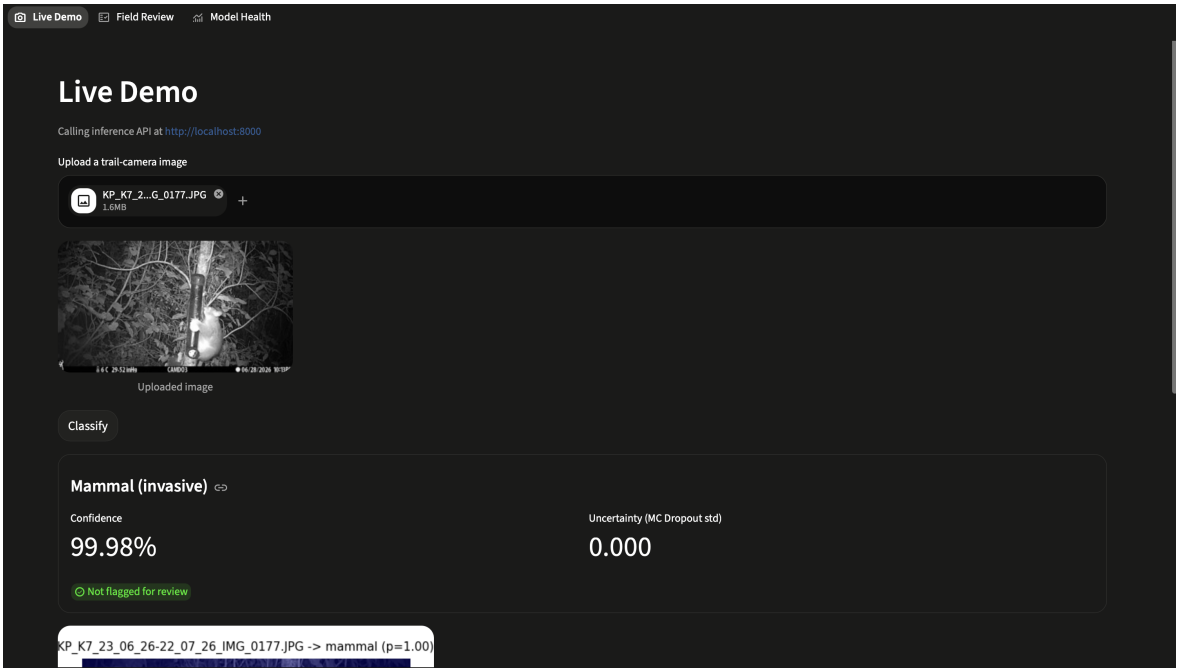

Figure 5.2: The Live Demo view. An uploaded camera-trap image of a possum is classified as a mammal, with its confidence, MC Dropout uncertainty and whether it has been flagged for review. The Grad-CAM++ heatmap is shown below these results.

The Practitioner view is built for the human in the loop of this system, whether that be a ranger or a field worker checking the predictions the model was unsure about. This view shows all flagged predictions the model has made including their respective heatmaps to allow the user to see what the model was looking at when making a prediction. The flagged predictions are listed from the most recent one first, with summary counts at the top to show the number of flagged mammal/bird calls. The user then has two options, either override when the model is incorrectly classifying the animal or confirm if the model got it correct. Every decision is saved in a CSV so it's kept after the session. The saved corrections can be used to retrain the model in the future. This is shown in Figure 5.3. The uncertainty shows as 0.0000 in Figure 5.3 because the cutoff is very small (about 0.0000002), as most predictions have almost no spread, so these predictions are still above the cutoff and get flagged.

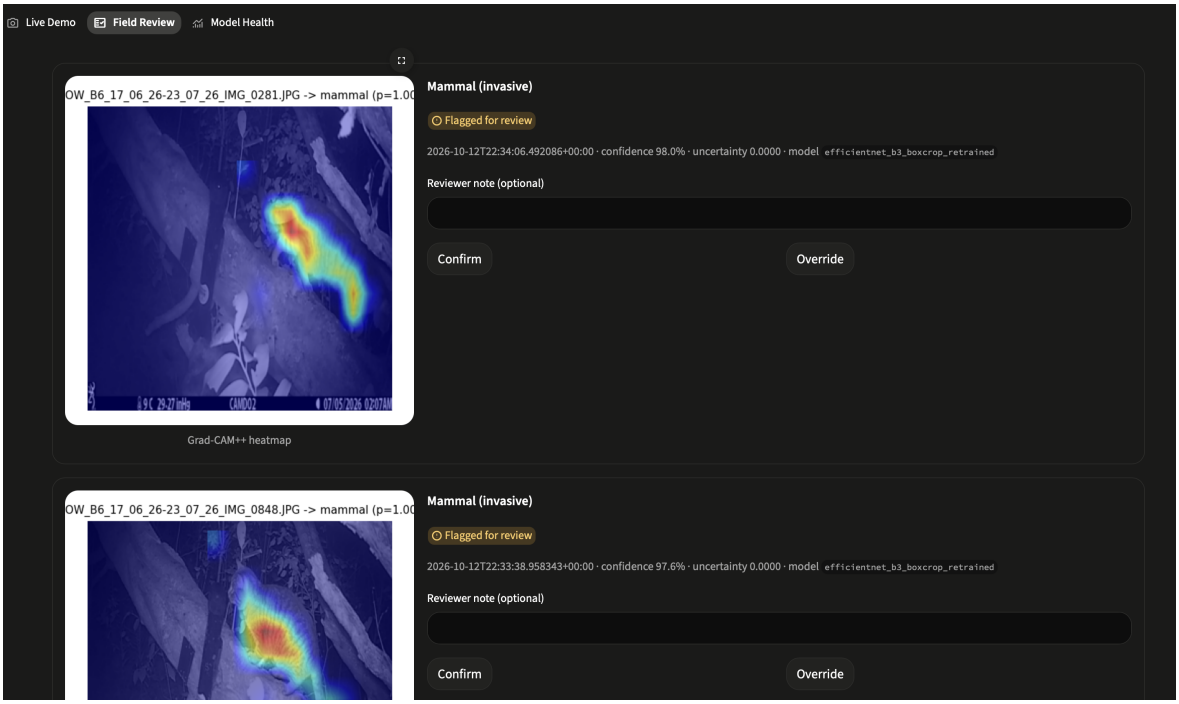

Figure 5.3: The Practitioner view (field review queue). Flagged predictions are listed most recent first, each with its Grad-CAM++ heatmap, and a reviewer can confirm or override each prediction and add an optional note.

The Researcher view is built mainly for maintaining the system, bringing together all the model comparisons, drift status and retraining history in one place shown in Figure 5.4.

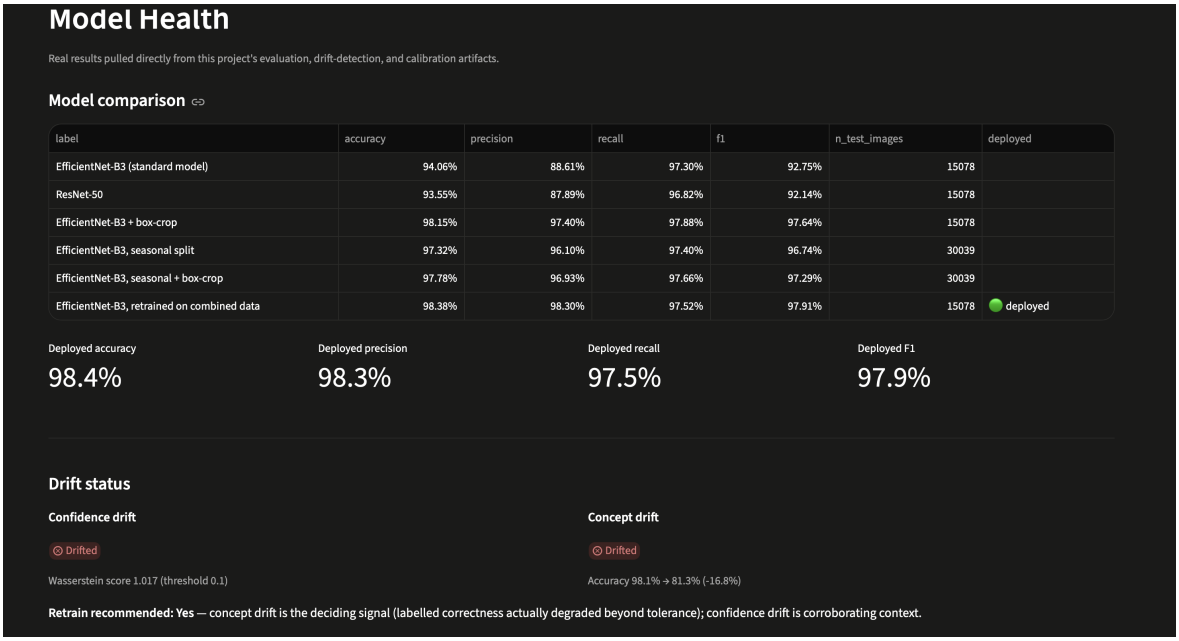

Figure 5.4: The Researcher view (model health). It compares every evaluated model, shows the deployed model's results, and shows the current status of the two drift signals.

The API, dashboard and MLflow server run as three separate services in Docker Com-

pose. The dashboard waits until the API passes its health checks before starting. A limitation of this dashboard is that when an image is uploaded in the Live Demo view, it doesn't send the image itself to the API, it saves the image to a temporary folder and sends the API the file path. This means that the Live Demo only works when the dashboard and API are run directly on the computer, because then they share the same folders. Under Docker Compose, each service runs in its own container with its own files, so the API cannot find the uploaded image. This could be fixed by adding a shared folder between the two containers, or by sending the image itself to the API.

5.3 Drift Detection

Once a model is deployed, it needs to be continuously monitored. As the model sees new places and times, the system needs to notice when the new data no longer looks like the training data.

To test whether the system can detect drift, I used an external dataset given to me by my supervisor, Andrew Lensen, and his PhD student, Ryan Jaggers. This external dataset contained images from different reserves and cameras, with 1,865 new images and 444 events. The dataset contained 1,300 mammals and 565 birds, so around 70% of the images are mammals, compared with around 39% in the original test set. Using this new dataset gives a real shift in the data to see whether the system is able to detect it.

The external images did not come with MegaDetector boxes, so I ran MegaDetector on them the same way the deployed system does, cropping each image to the animal before it is classified. MegaDetector found an animal in 83.9% of the images (1,565 of 1,865), and the rest were classified using the full frame.

The system is built to detect two signals that indicate drift:

- Confidence Drift: Checks if the distribution of the model's confidence scores has changed. This doesn't need labels, so it can run before anyone checks the images. It is measured using the Wasserstein distance, which shows how different the two sets of confidence scores are, where 0 means they are identical. The distance is divided by the spread of the original confidence scores, so it can be above 1. Drift is flagged if the distance is above 0.1. The distance here was 1.02, so drift was detected, meaning the confidence scores moved by about as much as their normal spread. This was not caused by the external data having more mammals, as changing the original test set to the same class mix only moved the distance by 0.02, while birds and mammals on their own drifted by 0.97 and 1.22.

- Concept Drift: Checks if the model has become less accurate, which can be caused by concept drift or by data drift. This requires labels and it is measured at a fixed 0.5 threshold, so that the deployed threshold does not distort the comparison. This flags drift when accuracy drops by more than 3 points. When running the external dataset on the box-crop model, accuracy dropped from 98.1% to 81.3%, a 16.8 point drop, so drift was detected.

Accuracy, precision, recall and F1 all dropped on the external data, as shown in Table 5.3.

Metric	Original test set	External data	Change (points)
Accuracy	98.1%	81.3%	-16.8
Precision	97.4%	90.1%	-7.3
Recall	97.9%	82.2%	-15.6
F1	97.6%	86.0%	-11.6

Table 5.3: Box-crop model on the original test set and on the 1,865 MegaDetector-cropped external images, at a fixed 0.5 threshold.

As a mammal is still a mammal on the new reserves, this drop is most likely caused by data drift from the new places and cameras. The external data also has more mammals (70% compared with 39%), but this does not explain the drop, as both classes got worse by a similar amount, with mammals correctly classified dropping from 97.9% to 82.2% and birds from 98.3% to 79.3%.

Concept drift has been made the deciding signal on whether drift has occurred or not with confidence drift being supporting context. This is because it is the signal that directly measures whether the model is getting predictions wrong. Confidence is able to shift without the performance degrading. The results are sent to Datadog, which then emails a human that drift has been detected and a model retrain is recommended, with a ready-to-run list of commands to retrain the model on the ECS cluster.

Drift is detected automatically, but nothing happens automatically afterwards. It is up to the person maintaining the system to decide whether and when to retrain the model.

5.4 Model Retraining

In the section above, model drift was detected. This essentially means the model is not performing as well on the new data as it did on the original test set, which matters when it is deployed in conservation contexts. Hence the decision was made to retrain the model, to recover its performance on the new data and keep the chance of harm low.

The box-crop model that has been deployed was trained on its original data plus a part of the external dataset with the same training setup as mentioned in Chapter 4. The external dataset was split into train/validation/test sets by site, the same as the original data. The train set included 791 new images from this external dataset, 922 images for validation and 152 images for testing. The external data makes up about 1.4% of the data the model was trained with. The validation images were only used to monitor training, the same as Chapter 4, not to choose the model, so they are evaluated together with the test images as 1,074 held-out images from 5 sites. In this case the validation set was bigger than the train set due to the fact the split is made by site, and with only 14 sites, the two largest sites both ended up in the validation set.

After this retrain I checked for two things. Firstly, a regression check to see if the retrain broke anything. I tested the retrained model on the old test set and compared it to the previous (un-retrained) box-crop model, at the same threshold. At the deployed threshold of 0.95, accuracy went from 98.3% to 98.4%, and at 0.5, accuracy did not change and remained 98.1%. However, at the deployed threshold, false positives went up from 86 to 99 while false negatives went down from 170 to 146, so the retrained model calls slightly more birds mammals than before. Overall, the performance on the original data is mostly unchanged,

which is a positive indication that the retraining did not harm the model's performance on the original data.

Secondly, I tested the retrained box-crop model on the 1,074 held-out images from the external dataset, with the images cropped by MegaDetector in the same way as in the previous section. The results for both models at three thresholds are shown in Table 5.4.

Threshold	Model	Accuracy	False positives	False negatives
0.05	Before retraining	86.0%	129	21
	After retraining	90.7%	97	3
0.5 (drift check)	Before retraining	84.6%	97	68
	After retraining	93.2%	68	5
0.95 (deployed)	Before retraining	85.5%	68	88
	After retraining	94.7%	47	10

Table 5.4: The box-crop model before and after retraining on the 1,074 held-out external images (463 birds, 611 mammals) from 5 sites.

These numbers are higher than the drift check in Table 5.3, as that used all 1,865 external images, while this comparison can only use the 1,074 held-out images, since the other 791 were used to retrain the model. The model happens to perform better on these held-out sites than on the external data overall (84.6% compared with 81.3% at the 0.5 threshold), so the two tables cannot be directly compared.

At the deployed threshold of 0.95, retraining improved the model, cutting false positives from 68 to 47, so fewer birds were wrongly classified as mammals, and accuracy went from 85.5% to 94.7%. On the 152 test images alone, the accuracy at 0.95 did not change, as retraining fixed seven mistakes but made seven new ones. As the test set only contains 152 images and 26 birds, a few images can change the result, which is why all 1,074 held-out images were used.

To check that this improvement was not down to chance, I used a McNemar test [21], which compares two models tested on the same images by looking only at the images they disagree on. On the 1,074 held-out images at the deployed threshold of 0.95, retraining fixed 129 of the old model's mistakes and made 30 new ones, which is very unlikely to be down to chance ($p < 0.001$).

However, even after retraining, the model still classified 47 of the 463 held-out birds (10.2%) as mammals at the deployed threshold, compared with only 1.1% (99 of 9,192) on the original test set. At the threshold of 0.05 this would have been 97 birds (21%), so the precision-first threshold halved these mistakes but could not remove them. Almost all of these were kākā (43 of the 47), and 40 of the 47 came from a single camera site, so this may partly be caused by that site and not by kākā in general. Kākā were rare in the original training data, with only 30 images, so the model had very little chance to learn what they look like. This is the type of mistake this project is trying to avoid, as kākā are a native bird. It shows that a model should be checked for each species before it is used at a new site, and that rare native birds need more training data.

Overall, retraining did not harm the model's performance on the original data and in fact improved the model on the external (new) dataset at the deployed threshold, although more sites are needed to be certain. Every step in this process, from deciding to retrain after

the drift alert to checking the results before deploying the new model, was carried out by a person, emphasising the importance of the human in the loop which is discussed in the following section.

5.5 Human Oversight/Governance

As mentioned in previous sections in this chapter, humans are the key to acting on important decisions, and these are purposely not done automatically. When working in conservation contexts, the animals can be native and endangered, so decisions about them should not be made automatically. This section tests whether keeping a person in the loop helps catch the errors that the model makes, and problems that offline testing alone would miss. During this project, I acted as the person in the loop, while in a real deployment this role would be filled by rangers and the person maintaining the system.

The model flags predictions it is unsure about using MC Dropout. The uncertainty score separates wrong from right predictions with AUC 0.95, and flagging the most uncertain 20% catches 91.1% of errors. As mentioned in Section 4.7, the model's own confidence did just as well, with the same AUC and 91.9% of errors caught at 20%. These flagged predictions are sent to a review queue which allowed me to override the model's incorrect decisions and confirm correct predictions.

I also checked this on the 1,074 held-out external images from Section 5.4, using the same uncertainty cutoff as the deployed system, which was set so that 20% of the validation predictions are flagged. On these new reserves the model was less sure, so about half of the images were flagged for review instead of 20%. However, 53 of the 57 errors were flagged, including 43 of the 47 birds wrongly classified as mammals. This means most of the mistakes that could harm native birds would still reach a person, but at the cost of a much larger review workload.

Initially, the API was giving the model full-frame images. Despite the model being trained on box-cropped images, it was looking elsewhere, such as at the trees in the frame, to make its predictions. This meant that recall was 69.6% at the 0.95 threshold and never above 86.2% at any threshold, so always below the 0.90 floor. This is shown in Figure 5.6. Despite offline test results showing no concern, this was found by a human using the dashboard and was the main motivating factor for adding the MegaDetector to the API calls so any image uploaded live would be box-cropped before the model makes its predictions. An example is shown in Figure 5.5.

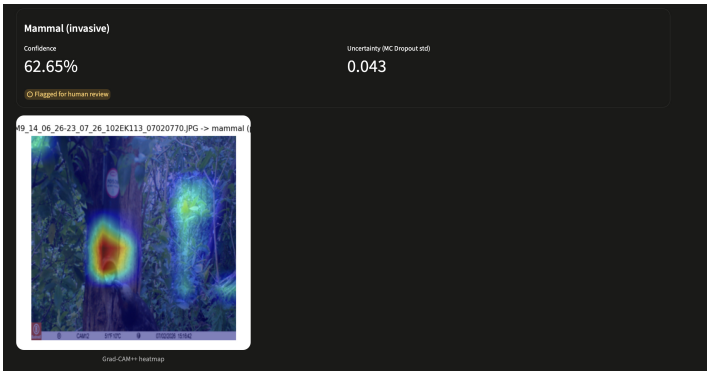

Figure 5.5: A kākā (bird) wrongly classified as a mammal by the deployed model when given the full frame, with the Grad-CAM++ heatmap on the tree trunk instead of the bird. Recreated by temporarily disabling MegaDetector cropping.

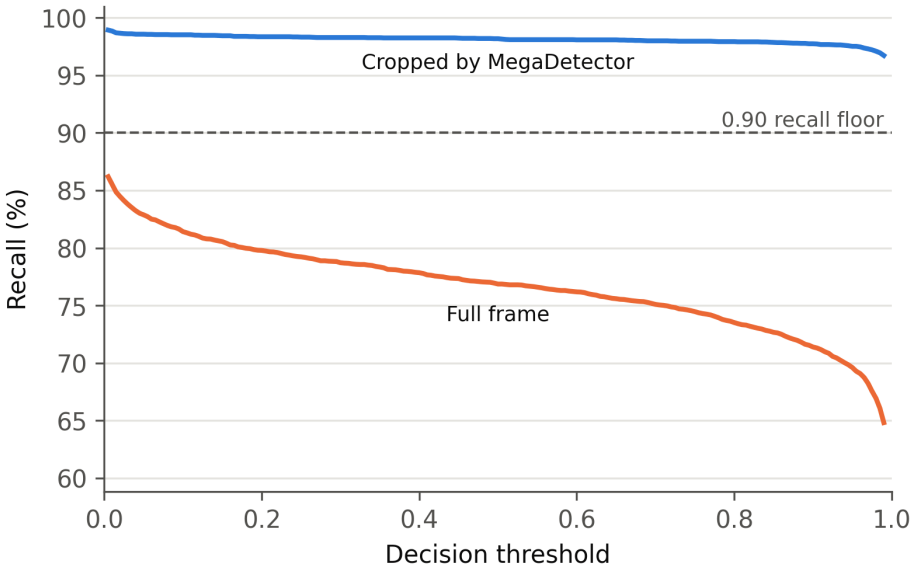

Figure 5.6: Recall of the deployed model across decision thresholds on the original test set. On full frames it never reaches the 0.90 recall floor; with MegaDetector cropping it stays above it at every threshold.

Another key point of the human in the loop is that they decide when the model should be retrained. As mentioned above in the Drift Detection section, an email from Datadog is sent with the steps on how to retrain but it is entirely up to the human governing this system to make the call to retrain the model. The dashboard shows the model's drift status, and evaluation/retraining history in the Researcher view so their decision would be informed, but still their own.

Overall, keeping a person in the loop caught errors at both the prediction level and the system level. However, this would need testing with real users to know how many errors are caught in practice, and the system-level evidence is based on only one case. In this project, I was the human in the loop, but in a real deployment this would be rangers reviewing the predictions and the person maintaining the system. Together with their expertise, the aim of

this system is to minimise harm to native animals while monitoring for invasive predators.

Chapter 6

Conclusion

6.1 Findings

In this project, multiple experiments were carried out. Firstly, in AIML501 I hypothesised that seasons would degrade model performance (H1). The results of this experiment did not support this. The model that trained on warm seasons and tested on cool seasons scored 97.3% accuracy compared to the standard model with 94.1% when trained on the site split.

This led me to investigate site familiarity and how that affects the model's performance, and I found that on sites the seasonal model had seen it performed at 97.8%, and on novel sites at 95.7%, so even with unseen sites it performed better than the standard model. This means H1 was not supported, and seasons affect the model less than expected. However, the seasonal model was trained on fewer images, and part of its result comes from familiar sites, so this is not a pure test of seasons alone.

XAI helped me see where the model was looking for its predictions. Through this I showed that a high-accuracy model can rely on the background for its predictions, so it can be right for partly the wrong reasons. For correctly classified birds, the median attention on the bird was 0.32, compared with 0.73 for mammals. False positives had a median of 0.03. Part of the gap between birds and mammals comes from bird boxes being smaller, but birds still got less attention at the same box size. This was cross-checked with Integrated Gradients, which showed the same pattern (Spearman's $\rho = 0.75$) for correctly classified birds and mammals. This supports H2.

XAI directly led me to perform box-cropping due to the findings. This greatly improved the model by reducing background reliance and false positives. Accuracy rose to 98.1% from 94.1%, with precision rising to 97.4% from 88.6%, and false positives dropped from 736 to 154. Where the model looks also improved, with the median attention on correctly classified birds rising from 0.32 to 0.88, or from 0.27 to 0.69 when compared with attention spread evenly. Box-cropping made little difference to the seasonal experiment. Overall, this supports H3, as box-cropping cut false positives by almost 80% and was kept as the main method in the deployed system.

Then I deployed the box-crop model and tested if drift can be detected on new data (H4). I also wanted to see if retraining can be done to not only improve the model but also to not harm its performance on the data it was already tested on. On the external data, the model's accuracy dropped from 98.1% to 81.3%, so concept drift was detected. Confidence drift sat at 1.02, so it was also detected.

After retraining the model, its performance on the original test set was mostly unchanged, and it improved the model's performance on the 1,074 held-out external images, with accuracy at the deployed threshold rising from 85.5% to 94.7% and false positives falling from 68 to 47, which a McNemar test showed is very unlikely to be down to chance. As these held-out images only came from 5 sites, this is supporting evidence rather than proof, but overall it supports the idea that deployed models should be retrained over time to adjust for change, and the retrained model is now the one deployed. However, it still classified 10.2% of the held-out birds as mammals, mostly kākā, compared with 1.1% on the original test set, so the model should be checked at each new site before it is trusted.

Finally, I tested whether keeping a person in the loop catches errors that the model and offline testing miss (H5). At the prediction level, flagging the most uncertain 20% of predictions caught 91.1% of the deployed model's errors, which are then sent to a person to confirm or override. The model's own confidence did this just as well as MC Dropout. On the new reserves, the same cutoff flagged about half of the images, but caught 43 of the 47 birds wrongly classified as mammals. At the system level, a person using the dashboard noticed the full-frame problem, where recall was never above 86.2%, even though the offline results looked fine, which led to MegaDetector being added to the API. This was only tested with me as the person in the loop and with one system-level case, but it supports H5 and shows the system sends the right predictions to a person to check.

6.2 Limitations

This project has a number of limitations that should be kept in mind when reading the results.

The first limitation is the data. The external images used to check retraining came from only 5 sites, so the result may not hold for other new reserves. Many native birds were also rare in the training data, with 27 of the 49 native bird species having fewer than 100 training images, such as mohua and wrybill with only 4 each. On the new reserves, 15% of kākā, one of these rare species, were still classified as mammals. The MegaDetector boxes used throughout this project are model predictions and were never checked by a human, so they are pseudo-ground truth. Not every image has a box either, with 89.8% of the original images and 83.9% of the external images having one, so the rest were classified using the full frame. Finally, because images without a timestamp could not be grouped into events, they made up a larger part of the data after capping, which meant a few species were mostly missing from the seasonal experiment.

The second limitation is how the models were trained and tested. Each model was trained multiple times during development, but only the final run was used and the runs were never compared, so the variance was never measured and I do not know how much the results would change between runs. This means small differences, such as EfficientNet-B3 against ResNet-50, could be down to chance. No early stopping or data augmentation was used, and the model after the final epoch was used for all results. The seasonal experiment was also not a pure test of seasons, as it trained on fewer images and most of its test images came from sites the model had already seen.

The third limitation is the explainability analysis. The attention score counts any attention inside the bounding box as being on the animal, even if it lands on background inside

the box, so the scores are likely higher than they should be. Integrated Gradients was only run on the standard model, on a sample of 759 images and with 20 steps, and not on the box-crop model. Both methods are also only approximations of what the model is actually doing, which is why two were used.

The last limitation is the human in the loop. No user study was done, and I was the person in the loop for this project, so it is not known how well rangers would catch errors in practice. The system-level evidence is also based on only one case, the full-frame problem. The reviewer corrections are saved, but they are not yet used to retrain the model. MC Dropout was also only applied to the classification head, so it did not flag errors any better than the model's own confidence.

6.3 Ethical and Practical Considerations

Because this system is built for conservation, its predictions could affect vulnerable animals and ecosystems, so there are some ethical and practical points to consider.

6.3.1 Ethical Considerations

The biggest ethical concern is the cost of the model's mistakes. A false positive means a native bird is classified as a mammal, which could lead to an unnecessary and harmful intervention, while a false negative means a predator is missed. This is why this project focuses on precision while keeping recall above a minimum floor of 0.90.

For this reason, the model should not be making decisions on its own. In this system, uncertain predictions are sent to a person to confirm or override, and the heatmaps show them where the model was looking, so they can judge whether the prediction makes sense. Decisions about retraining and deploying a new model are also made by a person, so the responsibility for any action stays with humans in the loop and not the model.

Camera traps can also take photos of people. The dataset includes a human category, and these images were removed before training, so the model was never trained on photos of people.

6.3.2 Practical Considerations

This system was built using free tools and university computing instead of paid cloud services. The university GPU cluster has no job scheduler, so a person has to check which GPU is free before retraining, which is one reason retraining is not automated. This is a privilege of being a student, so in practice training would have to take place elsewhere, making this system more expensive.

The model also cannot be trusted straight away in a new place. Section 5.3 showed that its accuracy dropped from 98.1% to 81.3% on images from new reserves, so before using it somewhere new, it should be checked and retrained if needed.

6.4 Future Work

The limitations explained in the section above directly motivate future work for this project.

The most important next step would be testing on more new sites. The retraining results were based on 1,074 images from only 5 sites, so collecting more data from new reserves would show whether retraining really improves the model everywhere, or only on these sites. More images of rare native birds should also be collected, so the model can learn to recognise them, which would minimise harm to native wildlife.

Models would also be trained multiple times with different seeds so that the variance could be measured and taken into account. It would also be worth testing if adding data augmentation and early stopping further assists the model to achieve better performance.

For explainability, segmentation would be added to further remove background reliance by separating the animal from the background more closely than a bounding box. This could be used to swap or randomise the background during training, so the model cannot rely on it, and it would also allow a stricter attention audit.

For uncertainty, MC Dropout could be applied across more of the network, or replaced with deep ensembles, so that the uncertainty score gives information beyond how confident the model already is.

Finally, the system should be tested with real users. A study with rangers would show how well people catch the model's errors using the review queue and heatmaps. The corrections they make are already saved, so a future version of the system could use them to retrain the model.

6.5 Conclusions

This project set out to test whether a deployment-aware system, one that explains its predictions, notices when the data changes, and keeps a person in control, is more trustworthy than a model that is only judged on its accuracy.

The results support this. Explainability showed that a model with high accuracy was often right for partly the wrong reasons, as it relied much more on the background for birds than for mammals. Box-cropping, added because of this, cut false positives from 736 to 154. The drift signals detected a real drop in performance on new reserves, and retraining improved the model while leaving it mostly unchanged on the original data. Keeping a person in the loop caught errors that the model and offline testing missed, including a problem that would have reached the live system. Not every expectation held, as seasonal change did not reduce performance as I expected in AIML501.

Overall, this project shows that accuracy alone is not enough to trust a model in conservation. A model also needs to be checked for why it makes its predictions, watched after it is deployed, and kept under the control of a person, so that it can help protect native species instead of harming them.

Bibliography

[1] ALLA, S., AND ADARI, S. K. What is MLOps? In *Beginning MLOps with MLflow*. Apress, Berkeley, CA, USA, 2021, ch. 3, pp. 79–124.

[2] BEERY, S., MORRIS, D., AND YANG, S. Efficient pipeline for camera trap image review. *arXiv preprint arXiv:1907.06772* (2019).

[3] BEERY, S., VAN HORN, G., AND PERONA, P. Recognition in Terra Incognita. In *Computer Vision – ECCV 2018* (2018), Lecture Notes in Computer Science, Springer, pp. 472–489.

[4] CHATTOPADHAY, A., SARKAR, A., HOWLADER, P., AND BALASUBRAMANIAN, V. N. Grad-CAM++: Generalized gradient-based visual explanations for deep convolutional networks. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)* (2018), pp. 839–847.

[5] CHEN, R., LITTLE, R., MIHAYLOVA, L., DELAHAY, R., AND COX, R. Wildlife surveillance using deep learning methods. *Ecology and Evolution 9*, 17 (2019), 9453–9466.

[6] CHENG, Z., WU, Y., LI, Y., CAI, L., AND IHNAINI, B. A comprehensive review of explainable artificial intelligence (XAI) in computer vision. *Sensors 25*, 13 (2025), 4166.

[7] DELISLE, Z. J., FLAHERTY, E. A., NOBBE, M. R., WZIENTEK, C. M., AND SWIHART, R. K. Next-generation camera trapping: Systematic review of historic trends suggests keys to expanded research applications in ecology and conservation. *Frontiers in Ecology and Evolution 9* (2021), 617996.

[8] DOSOVITSKIY, A., BEYER, L., KOLESNIKOV, A., WEISSENBORN, D., ZHAI, X., UNTERTHINER, T., DEHGHANI, M., MINDERER, M., HEIGOLD, G., GELLY, S., USZKOREIT, J., AND HOULSBY, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)* (2021).

[9] GAL, Y., AND GHAHRAMANI, Z. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of The 33rd International Conference on Machine Learning* (New York, New York, USA, 20–22 Jun 2016), M. F. Balcan and K. Q. Weinberger, Eds., vol. 48 of *Proceedings of Machine Learning Research*, PMLR, pp. 1050–1059.

[10] GAMA, J., ŽLIOBAITĖ, I., BIFET, A., PECHENIZKIY, M., AND BOUCHACHIA, A. A survey on concept drift adaptation. *ACM Computing Surveys 46*, 4 (2014), 44:1–44:37.

[11] GEIRHOS, R., JACOBSEN, J.-H., MICHAELIS, C., ZEMEL, R., BRENDEL, W., BETHGE, M., AND WICHMANN, F. A. Shortcut learning in deep neural networks. *Nature Machine Intelligence 2*, 11 (2020), 665–673.

[12] GUO, C., PLEISS, G., SUN, Y., AND WEINBERGER, K. Q. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning* (06–11 Aug 2017), D. Precup and Y. W. Teh, Eds., vol. 70 of *Proceedings of Machine Learning Research*, PMLR, pp. 1321–1330.

[13] HE, K., ZHANG, X., REN, S., AND SUN, J. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 770–778.

[14] INNES, J., KELLY, D., OVERTON, J. M., AND GILLIES, C. Predation and other factors currently limiting New Zealand forest birds. *New Zealand Journal of Ecology 34*, 1 (2010), 86–114.

[15] ISLAM, S. R., EBERLE, W., GHAFOOR, S. K., AND AHMED, M. Explainable artificial intelligence approaches: A survey. *arXiv preprint arXiv:2101.09429* (2021).

[16] JOCHER, G. YOLOv5 by Ultralytics. `https://github.com/ultralytics/yolov5`, 2020.

[17] KHAN, S., NASEER, M., HAYAT, M., ZAMIR, S. W., KHAN, F. S., AND SHAH, M. Transformers in vision: A survey. *ACM Computing Surveys 54*, 10s (2022), 1–41.

[18] KUPRIEIEV, R., ET AL. DVC: Data version control – Git for data & models (v3.67.1). `https://dvc.org`, 2026.

[19] LIANG, P., SONG, B., ZHAN, X., CHEN, Z., AND YUAN, J. Automating the training and deployment of models in MLOps by integrating systems with machine learning. *arXiv preprint arXiv:2405.09819* (2024).

[20] MARCINKEVIČS, R., AND VOGT, J. E. Interpretable and explainable machine learning: A methods-centric overview with concrete examples. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 13*, 3 (2023), e1493.

[21] MCNEMAR, Q. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika 12*, 2 (1947), 153–157.

[22] MOSQUEIRA-REY, E., HERNÁNDEZ-PEREIRA, E., ALONSO-RÍOS, D., BOBES-BASCARÁN, J., AND FERNÁNDEZ-LEAL, Á. Human-in-the-loop machine learning: a state of the art. *Artificial Intelligence Review 56*, 4 (2023), 3005–3054.

[23] NEW ZEALAND DEPARTMENT OF CONSERVATION. Trail camera images of New Zealand animals. LILA BC, `https://lila.science/datasets/trail-camera-images-of-new-zealand-animals/`, 2023. Online dataset. Accessed: 14 July 2026.

[24] NOROUZZADEH, M. S., NGUYEN, A., KOSMALA, M., SWANSON, A., PALMER, M. S., PACKER, C., AND CLUNE, J. Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning. *Proceedings of the National Academy of Sciences 115*, 25 (2018), E5716–E5725.

[25] RABANSER, S., GÜNNEMANN, S., AND LIPTON, Z. C. Failing loudly: An empirical study of methods for detecting dataset shift. In *Advances in Neural Information Processing Systems* (2019), vol. 32.

[26] REDMON, J., DIVVALA, S., GIRSHICK, R., AND FARHADI, A. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 779–788.

[27] RUDIN, C. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence 1*, 5 (2019), 206–215.

[28] RYO, M., ANGELOV, B., MAMMOLA, S., KASS, J. M., BENITO, B. M., AND HARTIG, F. Explainable artificial intelligence enhances the ecological interpretability of black-box species distribution models. *Ecography 44*, 2 (2021), 199–205.

[29] SCHNEIDER, S., TAYLOR, G. W., LINQUIST, S., AND KREMER, S. C. Past, present and future approaches using computer vision for animal re-identification from camera trap data. *Methods in Ecology and Evolution 10*, 4 (2019), 461–470.

[30] SELVARAJU, R. R., COGSWELL, M., DAS, A., VEDANTAM, R., PARIKH, D., AND BATRA, D. Grad-CAM: Visual explanations from deep networks via gradient-based localization. In *2017 IEEE International Conference on Computer Vision (ICCV)* (2017), pp. 618–626.

[31] SHNEIDERMAN, B. Human-centered artificial intelligence: Reliable, safe & trustworthy. *International Journal of Human–Computer Interaction 36*, 6 (2020), 495–504.

[32] SRIVASTAVA, N., HINTON, G., KRIZHEVSKY, A., SUTSKEVER, I., AND SALAKHUTDINOV, R. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research 15*, 1 (2014), 1929–1958.

[33] STEINER, A., KOLESNIKOV, A., ZHAI, X., WIGHTMAN, R., USZKOREIT, J., AND BEYER, L. How to train your ViT? Data, augmentation, and regularization in vision transformers. *Transactions on Machine Learning Research* (2022).

[34] SUNDARARAJAN, M., TALY, A., AND YAN, Q. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning* (2017), vol. 70 of *Proceedings of Machine Learning Research*, pp. 3319–3328.

[35] TABAK, M. A., NOROUZZADEH, M. S., WOLFSON, D. W., ET AL. Machine learning to classify animal species in camera trap images: Applications in ecology. *Methods in Ecology and Evolution 10*, 4 (2019), 585–590.

[36] TAN, M., AND LE, Q. V. EfficientNet: Rethinking model scaling for convolutional neural networks. In *Proceedings of the 36th International Conference on Machine Learning* (2019), vol. 97 of *Proceedings of Machine Learning Research*, pp. 6105–6114.

[37] TESTI, M., BALLABIO, M., FRONTONI, E., IANNELLO, G., MOCCIA, S., SODA, P., AND VESSIO, G. MLOps: A taxonomy and a methodology. *IEEE Access 10* (2022), 63606–63618.

[38] THABTAH, F., HAMMOUD, S., KAMALOV, F., AND GONSALVES, A. Data imbalance in classification: Experimental evaluation. *Information Sciences 513* (2020), 429–441.

[39] TOMPKINS, D. M. The research strategy for a "Predator Free" New Zealand. In *Proceedings of the Vertebrate Pest Conference* (2018), vol. 28.

[40] TUIA, D., KELLENBERGER, B., BEERY, S., ET AL. Perspectives in machine learning for wildlife conservation. *Nature Communications 13*, 1 (2022), 792.

[41] VILONE, G., AND LONGO, L. Classification of explainable artificial intelligence methods through their output formats. *Machine Learning and Knowledge Extraction 3*, 3 (2021), 615–661.

[42] WAZIR, S., KASHYAP, G. S., AND SAXENA, P. MLOps: A review. *arXiv preprint arXiv:2308.10908* (2023).

[43] ZAHARIA, M., CHEN, A., DAVIDSON, A., GHODSI, A., HONG, S. A., KONWINSKI, A., MURCHING, S., NYKODYM, T., OGILVIE, P., PARKHE, M., ET AL. Accelerating the machine learning lifecycle with MLflow. *IEEE Data Engineering Bulletin 41*, 4 (2018), 39–45.

[44] ZERILLI, J., BHATT, U., AND WELLER, A. How transparency modulates trust in artificial intelligence. *Patterns 3*, 4 (2022), 100455.

[45] ZHAO, X., WANG, L., ZHANG, Y., HAN, X., DEVECI, M., AND PARMAR, M. A review of convolutional neural networks in computer vision. *Artificial Intelligence Review 57*, 4 (2024), 99.

[46] ZWERTS, J. A., STEPHENSON, P. J., MAISELS, F., ROWCLIFFE, M., ASTARAS, C., JANSEN, P. A., VAN DER WAARDE, J., STERCK, L. E. H. M., VERWEIJ, P. A., BRUCE, T., BRITTAIN, S., AND VAN KUIJK, M. Methods for wildlife monitoring in tropical forests: Comparing human observations, camera traps, and passive acoustic sensors. *Conservation Science and Practice 3*, 12 (2021), e568.