Automated single species identification in camera trap images: architecture choice, training strategies, and the interpretation of performance metrics

Yannick Burkard[1], Emanuele Francazi[1], Edward Lavender[1], Tina Dubach[2], Sabrina Wehrli[2], Jakob Brodersen[2], Michele Volpi[3], Marco Baity-Jesi[1], and Helen Moor[1]

[1]Eawag, Department Systems Analysis, Integrated Assessment and Modelling, Überlandstrasse 133, CH-8600 Dübendorf
[2]Eawag, Department Fish Ecology & Evolution, Seestrasse 79, CH-6047 Kastanienbaum
[3]Swiss Data Science Center, ETH Zurich, Andreasstrasse 5, CH-8050 Zürich

Corresponding author:
Yannick Burkard[1]

Email address: y.burkard25@gmail.com

ABSTRACT

Automated species detection in camera trap images with deep learning techniques has become common in ecological monitoring. Camera trap image data sets are a challenging task, because of modest data set size, high class imbalance owing to low prevalence of the species of interest, and image backgrounds that vary within and between cameras. Strategies to tackle these difficulties can be adopted at the data handling and pre-processing stage, in the choice of model architecture, and during model training. We here report on insights regarding these strategies from a case study that aimed to detect a large wading bird (grey heron, *Ardea cinerea*) in images from different camera traps. Model performance improved with data splitting according to a non-random strategy, higher resolution images, and standard minority oversampling with data augmentation in color space. An object detection architecture (YOLOv5x6) performed better than an image classification architecture (MobileNetV2), while using fewer computing resources. Transfer learning through initial weights derived from models pre-trained on similar data was beneficial, but fine-tuning models on the data set at hand remained important. Finally, we highlight the dependence of predictive performance on class imbalance, and the assumption that the prevalence in the test set is representative of intended application sets. We discuss different performance metrics, emphasizing the importance of reporting the complete set of basic metrics along with the test set prevalence, and illustrate the use of metrics in downstream ecological analyses.

INTRODUCTION

A core task in ecology is to gather information on species occurrences to understand their distribution and interactions with the environment. This information is crucial for both basic ecological research and applied conservation efforts. With technological advancements, camera traps have become a popular, non-intrusive method for monitoring wildlife (Burton et al., 2015). These devices allow researchers to capture images at regular intervals or when triggered by motion, generating large data sets over time. Such data sets can provide valuable insights into species occurrence, behavior, and habitat use. Manual processing of large data sets is extremely time consuming, and automated image classification provides significant benefits (Tuia et al., 2022).

Deep learning techniques, particularly convolutional neural networks (CNNs), offer a promising solution to automate species identification in camera trap images (Borowiec et al., 2022). CNNs are usually trained on a (sub)set of data that was labeled manually, i.e., images are screened by a human for presence of the species of interest and labeled accordingly. This labeled image data set is split into training data with which to train the CNN, validation data to evaluate and optimize model performance during training, and held out test data to quantify the performance of the final trained model. With sufficient training, CNNs can extract meaningful patterns from images, learning to classify objects with impressive generalization capabilities.

Two common approaches are *image classification* and *object detection*. Classification assigns a label (e.g., species or empty) to an entire image, while object detection also localizes the species (object) within the image, marks it with a bounding box, and classifies the object. There is now a wide variety of models publicly available, each containing a set of weights that are either randomly initialized or have been (pre-)trained on an existing data set (Vélez et al., 2023). Popular examples of classifiers are MobileNets (Howard et al., 2017), DenseNet (Huang et al., 2018), or EfficientNet (Tan and Le, 2019). MobileNets and their successor MobileNetV2 (Sandler et al., 2018) were designed for mobile applications and are efficient and small compared to other architectures. MobileNetV2 has been pre-trained on the ImageNet data set (Deng et al., 2009a), comprising millions of images labeled according to categories; however, these images are not typical of camera trap images. Object detection architectures commonly used are YOLO (Redmon et al., 2016), R-CNN (Girshick et al., 2014), or RetinaNet (Lin et al., 2017). A model based on the YOLOv5 architecture is MegaDetector, now popular for camera trap image analysis in ecology (e.g., Vélez et al. 2023). MegaDetector is trained on a large global data set of camera trap images to detect animals, people and vehicles (Beery et al., 2019).

Despite the success of these methods, several challenges remain in applying deep learning to ecological data. Schneider et al. (2020) identified three critical factors that affect the performance of automated species recognition in camera trap images. First, the size of the data set available for training or fine-tuning CNNs is often modest. In single species detection there are two classes: the species of interest is present in an image (positive) or not (empty). Schneider et al. (2020) recommend at least 1000 images per class to achieve high recall (true positive rate). For rare species, this may be difficult to achieve. Second, camera trap images frequently suffer from *class imbalance*, as many frames capture empty scenes and only a small percentage of images contain animals, especially in the case of rare species or when cameras continuously capture images at regular intervals. Differences in this imbalance between cameras pose additional difficulty to model training. Third, the ability of CNNs to generalize to new settings, e.g., to new species or across different camera locations, also remains a significant challenge. Given that the background remains relatively static in one camera location (apart from daily and seasonal changes in weather and vegetation), applying the model to data from new locations may not be successful. Different data distributions in the training and the test set may constitute a *domain shift*, where the model is asked to predict in a distributional range not encountered during training. Strategies to overcome these challenges can be adopted during input data handling and pre-processing, as well as during model training (Chen et al. (2024)).

Input data handling decisions include the following. First, appropriate *data splitting* into training, validation, and test sets is crucial to avoid data leakage, which occurs if information from the test or validation set is unintentionally used in the training procedure, leading to inflated estimates of model performance. This has to be considered in light of the ecological question of interest. Second, for camera trap images, *image resolution* is important. Images are typically downsized to a lower resolution in order to minimize memory requirements. Downsizing images constitutes an information loss, however, that can reduce model performance (Thambawita et al., 2021). Third, class imbalance can be addressed with resampling and data augmentation techniques, often in combination. Class imbalance in camera trap image data for single species detection usually involves few positives (the minority class) and many

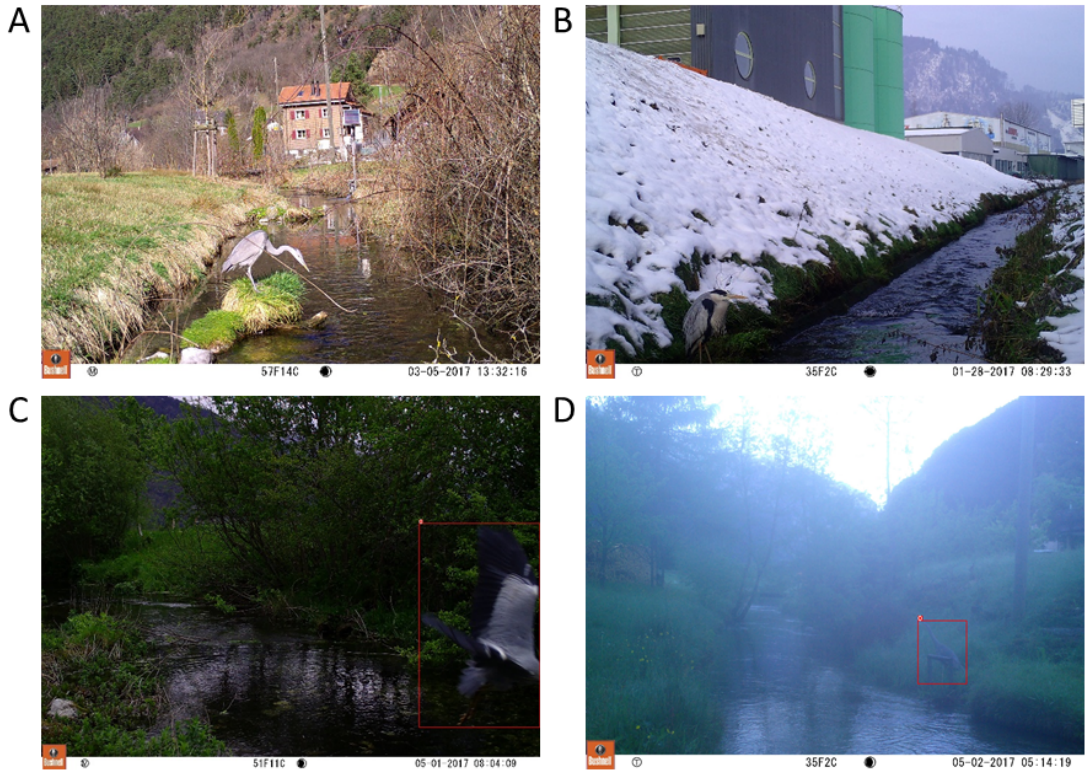

Figure 1. Image examples from different camera traps, showing the difference in backgrounds, seasons, and lighting conditions. Herons are increasingly difficult to detect in panels A to D. Panels C and D show bounding boxes around herons, as used in the labeling of images to train the object detection algorithm.

more negatives (the majority class). *Resampling* can balance the number of positives and negatives (empty frames) through undersampling, where a random subset of the negatives are used (Gomez Villa et al., 2017), or through oversampling, where copies of the positives are generated to achieve better class balance (Zualkernan et al., 2022). To avoid overfitting to specific, duplicated images, oversampling is usually combined with *data augmentation*, where copies of positives are distorted, e.g., in color space (Tabak et al., 2019; Whytock et al., 2021; Ferreira et al., 2020).

During *model training*, a number of decisions are made based on model performance on a validation set. The first decision is the choice of model architecture, which depends on the task at hand but is also constrained by skill, available memory and computing power. A strategy to deal with small data sets is *transfer learning*, where weights (CNN parameter values) are imported from a model that was trained on a different, larger data set (Willi et al., 2019; Norouzzadeh et al., 2018). Weights can then be kept for all or some layers (by freezing layers; Yang et al. 2024) or updated through training on the data set at hand. *Hyperparameter tuning* includes decisions about the learning rate, batch size, optimization- and model-specific parameters, or the cut-off (threshold) value that translates a score to a label.

Ultimately, ecologists are interested in the application of camera trap monitoring in the context of an ecological question such as species occurrence probability in different habitats. While machine learning platforms that automate image recognition have been reviewed repeatedly in the ecological literature (Christin et al., 2019; Borowiec et al., 2022; Vélez et al., 2023), less attention has been paid to the interpretation of classifier performance metrics and their usage in downstream ecological analyses (but see Rhinehart et al. (2022) for an alternative approach).

Common recommendations for using machine learning models are to evaluate and report recall, accuracy, precision, or the F1 score to judge a model's performance on a test set (e.g., Norouzzadeh et al. 2018; Christin et al. 2019; Vélez et al. 2023). The specific meaning of each metric, and the sufficient set of metrics that should be reported to enable a full assessment of model performance, is often ignored. Subtle differences in metric interpretation, well understood in, e.g., the medical sciences in the context of diagnostic tests (Trevethan, 2017), are less frequently highlighted in ecological applications of classifiers. In particular, the dependence of selected metrics on the proportion of positive images (the 'prevalence' or 'class imbalance rate') in test versus application datasets is underappreciated. Low prevalence (one form of high class imbalance) is common in monitoring, especially of rare species, and strongly affects a model's predictive performance metrics, with important corollaries for their interpretation. Lastly, little attention has been given to how these metrics can be used to account for model uncertainty in downstream ecological analyses of species detections in images. A proliferation of different terms used in different fields for the same metrics further complicates understanding (Kapoor and Narayanan, 2023). An overview of the key differences between various performance metrics and their interpretation in the context of ecological modeling is lacking.

Here, we report on a case study of a single species recognition task in camera trap images to discuss data handling and model training strategies, compare architectures and provide guidance on the interpretation of performance metrics. The image data set is from a camera trap study designed to monitor the presence of grey heron (*Ardea cinerea*) at small streams in Switzerland. The ecological motivation was to understand whether there is a difference between streams in the probability of heron presence, and what ecological variables drive these differences. Typical for ecological monitoring data, the data set has high class imbalance (few positives, i.e., images with heron present), which furthermore differs between camera locations. Seasonal differences in light and background conditions render species recognition in these images a difficult task (Fig. 1).

We first conducted an ablation study to investigate data handling strategies, especially data splitting and preprocessing, resampling and data augmentation strategies. Using the best settings determined in the ablation study, we then compared the performances of a classification network (a fine-tuned MobileNetV2 model) and an object detection network (the pre-trained MegaDetector model, and a fine-tuned YOLOv5x6 model) in two scenarios: with training data from the single camera (site) with the most positives, and with training data from all cameras (sites), which differ in heron prevalence. We compared the generalization capacity of the models trained on the single camera by their performance on out-of-sample tests sets from different cameras.

Finally, we discuss different performance metrics, their interpretation and their usage in downstream ecological analyses of heron occurrence probability.

METHODS

Data

The entire labeled data set consists of 415406 images taken by camera traps in 23 different locations between Jan 27, 2017 and July 17, 2017. Camera locations are distributed along 6 different streams, and identified according to stream (GBU, KBU, NEN, PSU, SBU, SGN) and camera number (e.g., GBU3). Every camera set contains between 10k and 22k images, with a median value of 18992. The camera model was a Bushnell Trophy Cam HD Essential E2. Images were taken at regular 15 minute intervals or when triggered by motion. Night-time images were taken in infrared mode, but they were excluded from this study because there were insufficient positives, and because they constitute a separate classification challenge since it is not trivial to convert infrared and RGB images to a comparable color mode.

The day-time data set used consists of 251479 images, with only 3177 (1.3%) of frames containing herons. In addition, the distribution of positives across different cameras is uneven. The camera with the most positives, SBU4, contained 1545 (12.0%) images with heron present, and three cameras (GBU1-3) had no positives. Average prevalence (proportion of positives) across cameras was 1.2%. Further details are in Appendix A.

Each image was initially labeled by a human as positive when containing at least one heron (heron, h), and negative when no heron was found (empty, e). These labels were considered the ground truth for the classification task. For training the object detection algorithm, bounding boxes were set around herons. To set bounding boxes we first applied the MegaDetector algorithm (Hernandez et al., 2024) to all positives and manually checked results; this set correct boxes for about half of herons. The remaining box labels were set manually with the *Roboflow* tool (Dwyer et al., 2024).

The data set presents challenges typical for camera trap studies. Weather and light conditions occasionally affected image quality: fog led to wet cameras and blurry images, and varying sun angle gave strong or weak contrast. The heron's position could make it difficult to spot, when it was distant from the camera, occluded by vegetation, or when only a small part of it was visible within the image plane. Occasionally, flying herons appeared blurry due to their fast motion (Fig. 1).

Ablation study

To determine optimal data handling and pre-processing strategies, we first conducted an ablation study by training MobileNetV2 models, with initial weights derived from ImageNet data (Deng et al., 2009b), on the single camera data set (SBU4) and comparing their performance through the F1-score. We considered data splitting, image resolution, resampling strategies and data augmentation (i.e., techniques for increasing the diversity of the data set without actually collecting new data), as well as transfer learning (i.e., using weights derived from other data).

The best strategies and settings determined during the ablation study were then used for training MobileNetV2 and the object detection architecture YOLOv5x6 to compare their performance in two scenarios. The YOLOv5x6 architecture for some aspects facilitated different some pre-processing strategies (e.g., higher image resolution); these choices were not part of the ablation study, which was conducted using MobileNetV2, but are also described in this section.

Data splitting The train-validation-test set splitting was implemented on the full data set in two ways. Note that these splitting procedures were applied to both day- and night-time images, before night-time infrared images were excluded.

Split 1 was chronological, with the 85% earliest images allocated to the training and validation data, and the latest 15% to the test data. We divided the initial portion, again chronologically, into 85% training and 15% validation data, ensuring the training data contained the earliest images. This method intends to minimize temporal data leakage, which could occur in a random split that places consecutive, nearly identical images into different data sets.

Split 2 was seasonal. Here, we allocated the 7.5% earliest (January) and 7.5% latest (July) images to the test set. Of the remaining data, we again assigned the 7.5% first and 7.5% last images to the validation set, while remaining images constituted the training set. This was motivated by the fact that the model's application is the classification of images from all seasons in following years (not be the continuation of a time series). We used only the earliest and latest images as the test set to minimize data leakage from boundary effects (similar consecutive images across training and test set boundaries).

Image resolution During the ablation study with MobileNetV2, we first downscaled the full images to a resolution of 448×448 (direct resizing), and then tested the effects on performance when increasing the input resolution to 896×896. Even higher resolutions increased computational load for MobileNetV2 substantially.

For the object detection task, images were resized to the YOLOv5x6 default resolution 1280×1280 via letterboxing (i.e., aspect ratio maintenance with additional padding). Despite information loss, this technique is preferable for object detection performance.

For both classification and object detection, the pixels of resized images were generated via bilinear interpolation.

Background removal We considered background removal as an additional image manipulation strategy to reduce the effect of complex backgrounds. Leveraging the time series of regularly taken photographs, pixel values over previous n images were averaged and subtracted from each image, such that objects that newly appeared in an image became more visible relative to the background. This did not improve performance notably, and was not further pursued. Details in Appendix B.

Resampling We tested different resampling techniques to address class imbalance. We first randomly undersampled negatives to achieve a 1:1 ratio between both classes. Undersampling was applied to the training and validation data separately, allowing us to decrease training times and explore multiple configurations. While having the advantage of using a smaller training data set, undersampling is often a suboptimal strategy (Loffredo et al., 2024). We therefore proceeded to training with oversampling, which allowed us to use the entire data set and can improve both training time (Francazi et al., 2023) and overall performance (Loffredo et al., 2024). We oversampled the training data with two techniques: the synthetic minority over-sampling technique (SMOTE) (Chawla et al., 2002) and standard oversampling.

SMOTE rebalances positives and negatives by producing new artificial minority (positive) instances through linear interpolation. These artificial images were generated from the preprocessed images, not from raw images.

Standard oversampling rebalances the two classes by sufficiently resampling the positives such that positives and negatives are seen equally often during training. Copies of the positive set are combined with a random subsample of the minority class to achieve a balanced data set. We ensured variability among the copied heron images through random augmentation techniques during training, since using exact copies of the training images would lead to detrimental image memorization and overfitting.

In trainings using images from all cameras, we additionally tested a novel logarithmic oversampling technique, with the goal to even out differences in class imbalance across different camera subsets. Performance did not improve noticeably, and results are therefore not shown. Details are in Appendix C.

Data Augmentation For training MobileNetV2, we used data augmentation in color space by randomly altering the saturation, brightness and contrast values to simulate light changes that occur across different times of the day. Affine transformations like rotation, scaling and translations were discarded due to the possibility of herons disappearing from the frame or becoming significantly smaller and hence undetectable.

For training YOLOv5x6, data augmentation was achieved with random transformations in hue-saturation-value (hsv) space. We also applied the YOLO-specific feature of mosaic augmentation with the aim of decreasing background dependence. This technique consists in combining three random images into a single image as a mosaic, where each subsample is located in a quadrant. Image resizing and color augmentations were then applied to the entire mosaic.

Transfer learning MobileNetV2 was initialized with weights pre-trained on the ImageNet data set. We started with freezing all but the last layer (i.e., keeping pre-trained weights in frozen layers) and adjusting the final layer for binary classification. Then we unfroze all layers and trained the full model, using pre-trained weights as initial values. Subsequent trainings were performed with all layers unfrozen.

For YOLOv5x6, initial weights of the MegaDetector model (v5) were used, which is a YOLOv5x6 architecture trained on camera trap data to distinguish animals, humans, and vehicles (Hernandez et al., 2024). Throughout the training procedures, we only fine-tuned weights from the last 11 layers, corresponding to the detection and classification tasks, while keeping the first 12 backbone layers frozen (responsible for feature extraction), thus maximizing training efficiency and reducing the potential for overfitting.

Architecture comparison

After the initial ablation study we proceeded with training the classification network MobileNetV2 and the object detection network YOLOv5x6 with the optimal settings determined by the ablation study and standard oversampling, while evaluating the F1-score of the model on undersampled validation data after every training epoch. An epoch is defined as one pass through the full dataset (for oversampling, this includes all generated copies). This method gives an effective estimate of model performance via metric curves as the training progresses, allowing the user to monitor parameter convergence. After determining the optimal training configurations and number of epochs for each setting, we retrained models on the merged training and validation sets, and report final results on held out test sets.

Classification The selected classifier, MobileNetV2 (Sandler et al., 2018), contains 55 layers and a total of 3.4M parameters. This model is more compact and parameter-efficient than other classification networks, such as the larger ResNet, DenseNet, or EfficientNet architectures, and more recently Vision Transformers (Dosovitskiy et al., 2021). Although containing fewer trainable parameters, the inclusion of linear bottlenecks and depthwise separable convolution make it achieve excellent performance, reaching 91% top-5 accuracy on the ImageNet data set (Sandler et al., 2018).

We chose MobileNetV2 because the task consisted of a binary classification rather than the detection of multiple species. Deeper models are more prone to overfitting and not likely to provide an improvement in accuracy at the cost of additional compute power.

We used the optimizer AdamW (Loshchilov and Hutter, 2019) while keeping default moment parameters, learning rate 10^{-5}, batch size 32, and default dropout probability 0.2. No benefit was evident from introducing weight decay. The confidence threshold was set to the commonly used value of 0.5.

Object Detection The object detection model used was YOLOv5x6 (Ultralytics, 2021) with initial MegaDetector weights (version 5), which are derived from fine-tuning a YOLOv5x6 architecture on global camera trap data (Hernandez et al., 2024). YOLOv5x6 contains a total of 33 layers with 14M parameters. While there are more complex models with a two-stage pipeline (e.g., Cascade and Faster RCNNs) as well as more recent versions of YOLO available, we chose version 5 because it allowed us to use the pre-trained MegaDetector weights.

We refer to the pre-trained, zero-shot model without further fine-tuning as the MegaDetector model, and provide results from this network alongside results from YOLOv5x6 trained on our data set (with MegaDetector weights as initial values). While the confidence threshold was optimized for the trained models, we fixed this value to the default 0.2 for the MegaDetector model.

To fine-tune the MegaDetector weights, we made partial use of the YOLOv5 framework (Ultralytics, 2021). Many of the training configurations were kept to their default values, including a stochastic gradient descent (SGD) optimizer (Ruder, 2017) with momentum 0.937, as well as initial and final learning rates of 10^{-2} and 10^{-4}, respectively. The batch size was set to 32 and the input resolution was fixed to 1280×1280.

The confidence threshold during the object detection task was varied after every epoch to maximize the detection F1-score on the validation set, as implemented by default in the YOLOv5 framework. The model with maximum fitness is saved as the best model, and we used the corresponding confidence threshold for the final test metrics. The fitness F is defined as $F = 0.1\text{mAP@}0.5 + 0.9\text{mAP@}0.5{:}0.95$, where mAP@0.5 is the mean average precision (mAP) at intersection over union (IoU) 0.5, and mAP@0.5:0.95 is the mAP averaged over IoU values from 0.5 to 0.95 with step size 0.95 (Ultralytics, 2021).

Scenarios We trained models on two data scenarios: a *single-camera* data set from the camera with most positives (SBU4) and the data set from *all cameras* combined (all cams). Models fine-tuned on the single-camera set were evaluated against held out test sets from the same camera and from other cameras (non-SBU4). Performance on non-SBU4 data was quantified to evaluate their generalization capabilities to different cameras, with different backgrounds and heron prevalence. Models trained on all cameras were evaluated against a held out test set from across all cameras, as well as camera-specific subsets. Alongside the models trained on our data, we also tested the zero-shot capabilities of the pre-trained MegaDetector model.

For each test set and model, we report metrics derived from the confusion matrix, which compares the classification returned by the model to the ground truth label of an image (Box 1). In the case of object detection models, an image was classified as positive when at least one object (heron) was detected in the image.

Specifically, we report the True Positive Rate (TPR), the True Negative Rate (TNR), the balanced accuracy, as well as the prevalence of positives in the test set and the Positive Predictive Value (PPV), the Negative Predictive Value (NPV) and the F1 score (definitions in Box 1). Key metrics were calculated for test sets corresponding to training sets (SBU4 and all cameras) as well as subsets of these to facilitate comparison.

Performance metrics

We provide definitions and a description of key performance metrics and their meaning in (Box 1). We clarify which metrics can be used in a downstream ecological analysis to account for the error rates of the chosen species detection model (with the example of a simple state space model). We highlight the impact of rarity, i.e., low prevalence of the species of interest (the positive class), on the predictive capacity of a chosen model, as well as avenues to improve model performance or predictive capacity.

Box 1. Performance metrics There is a fundamental yet often underappreciated distinction between performance metrics commonly reported for classifiers. It has to do with the reference set to which the metric applies and, consequently, its independence (for **classifier performance**) or dependence (for **predictive performance**) on the context, specifically the prevalence (proportion of positives) in the data set under study (in other words, the class imbalance).

Performance metrics are calculated based on the correspondence of ground truth and classification by a model in a test set, as reported in the confusion matrix, i.e., the number of true positives (TP), false positives (FP), false negatives (FN) and true negatives (TN):

		Ground truth x		
		Positive (h)	Negative (e)	
Classification (label) y	Positive (+)	TP	FP	→ PPV
	Negative (-)	FN	TN	→ NPV
		→ TPR	→ TNR	

Classifier performance Metrics describing classifier performance pertain to the ability of the model to make a correct guess $y = \{+,-\}$ given the truth $x = \{h,e\}$, that is they pertain to $\Pr(y|x)$. These metrics are the true positive rate **TPR** (also called **recall** or **sensitivity**), which estimates $\Pr(+|h)$, and the true negative rate **TNR** (or **specificity**), which estimates $\Pr(-|e)$. Based on the confusion matrix, TPR and TNR are calculated, conditional on the truth, as $TPR = \frac{TP}{TP+FN}$ and $TNR = \frac{TN}{TN+FP}$, respectively. **Accuracy** is often used as a summary measure of how well the classifier correctly identifies the truth; it is calculated as $ACC = \frac{TP+TN}{TP+FP+TN+FN}$, i.e., the proportion of all classifications that are correct. Accuracy is misleading when the data set is imbalanced; e.g., when the vast majority is negative, a baseline model classifying all instances as negative would achieve a high TNR and high accuracy. In such cases, **balanced accuracy**, calculated as the mean of TPR and TNR, gives a comparatively more objective assessment of model performance.

Predictive performance Metrics describing the predictive capacity of an algorithm in the context of a model's application (prediction in context) quantify the probability of predicting the truth x correctly, given a particular guess y, that is they pertain to $\Pr(x|y)$. These metrics are the positive predictive value **PPV** (or **precision**), which estimates $\Pr(h|+)$, and the negative predictive value **NPV**, which estimates $\Pr(e|-)$. Based on the confusion matrix, they are calculated, conditional on the label, as $PPV = \frac{TP}{TP+FP}$ and $NPV = \frac{TN}{TN+FN}$, but their validity is dependent on the prevalence $\Pr(x = h)$ of the test set used. These metrics are not intrinsic to the model but also depend on the context, specifically the prevalence, of the data set to which the models are applied. This is evident from an alternative calculation: using Bayes' theorem, predictive values can be derived from TPR, TNR and prevalence. In this formulation, the **PPV** is calculated as

$$P(h|+) = \frac{\Pr(+|h)\Pr(h)}{\Pr(+|h)\Pr(h) + (1-\Pr(-|e))\Pr(e)} \approx \frac{TPR \cdot \Pr(h)}{TPR \cdot \Pr(h) + (1-TNR)\cdot(1-\Pr(h))} \quad (1)$$

Note that $\Pr(+|e) = 1 - \Pr(-|e)$. The NPV can be calculated analogously. The estimate of predictive performance derived from the confusion matrix depends critically on the assumption that the prevalence is representative. PPV and NPV estimated from performance on a test set are only valid for the test set used or other data sets with the exact same prevalence as in that test set. Any violation of this assumption will lead to false estimates of the predictive uncertainty of the model. Especially in the case of very low prevalences, small differences can affect predictive metrics substantially (Fig. 5).

Finally, the **F1-score** is calculated as the harmonic mean of TPR (recall) and PPV (precision; as estimated from the test set), and thereby gives information on both the classification and the predictive performance, with the corollary that it is also sensitive to the prevalence, i.e., class imbalance.

To meaningfully assess the performance and predictive capacity of a classifier, all four metrics (TPR, NPR, PPV and NPV), as well as the prevalence of positives in the test set, should be reported (Trevethan, 2017).

RESULTS

Ablation study

Training the full MobileNetV2 model (pre-trained weights used as initial values), increased model performance substantially compared to fine-tuning only the last layer (Table 1; Fig. 2). Splitting the data into training, validation, and test sets according to the seasonal strategy resulted in better model performance than the chronological split. Higher input image resolution (896x896 pixels instead of 448x448) was also beneficial. Regarding oversampling strategies, SMOTE lead to a more unstable and marginally lower F1 curve (Fig. 2) than standard oversampling. Generating synthetic samples corresponds to a longer and computationally more expensive process than producing copies of minority data. Overall, standard oversampling in combination with random data augmentation in color space resulted in the best performance among the evaluated resampling and augmentation strategy combinations (Table 1).

Table 1. Ablation study results showing the maximum F1 score achieved with MobileNetV2 trained on SBU4 data given different data splitting strategies, input image resolution, resampling methods, data augmentation, and degrees of transfer learning (freezing initial weights in all but the last layer, or training the full model). Oversampling refers to standard oversampling.

Data split	Resolution	Resampling	Augmentation	Layers trained	Max. F1
Chronological	448x448	undersampling	none	last layer	0.34
Chronological	448x448	undersampling	none	full model	0.77
Seasonal	448x448	undersampling	none	full model	0.87
Seasonal	896x896	undersampling	none	full model	0.91
Seasonal	896x896	undersampling	color	full model	0.93
Seasonal	896x896	SMOTE	none	full model	0.92
Seasonal	896x896	oversampling	color	full model	0.94

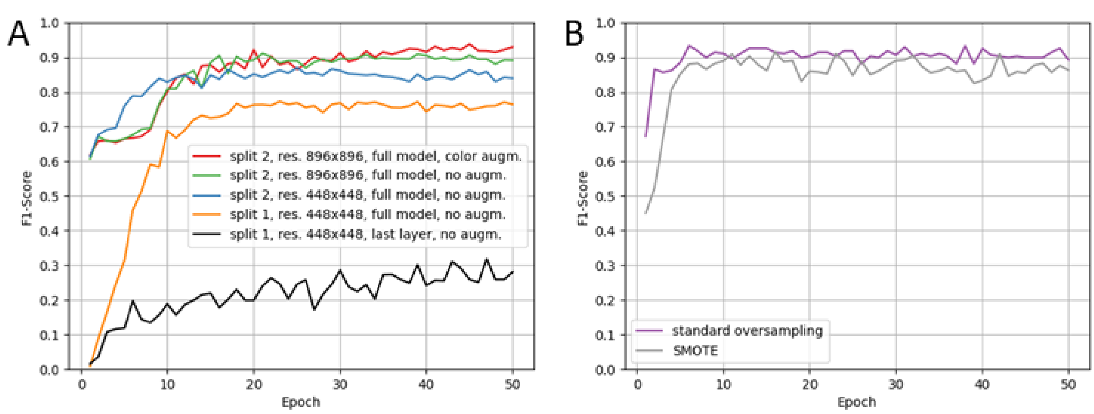

Figure 2. Validation F1-score curves for the ablation study using MobileNetV2, A) training on undersampled SBU4 data with different configurations, and B) training with SMOTE and standard oversampling on SBU4 data.

Single camera scenario

Both models converged after 15 epochs during training and evaluation on the validation set. The optimal confidence threshold for the object detection network was 0.07. Results are given for both architectures fine-tuned on the joint training and validation sets for 15 epochs, and evaluated on the held out test sets from the SBU4 camera and from non-SBU4 cameras. The non-SBU4 test set assesses generalization capabilities to out-of-sample data. Final F1-score training curves are in Appendix D.

For the in-sample (SBU4) test set, the trained object detection algorithm YOLOv5x6 performed best, achieving the highest TPR (recall) and highest balanced accuracy, as well as the highest PPV (precision) and F1 score (Fig. 3, Table 2). Second was the trained classifier MobileNetV2, and last the zero-shot, pre-trained object detection model MegaDetector. Note though that in spite of high balanced accuracy (0.94) and a high TPR (recall) of 0.90, the best performing model YOLOv5x6 still only achieved a PPV (precision) of 0.71 (Fig. 3). That is, 29% of images classified as containing a heron (labeled +), were actually false positives.

Out-of-sample performance for images from cameras not seen during training (non-SBU4) was low for all models. In terms of balanced accuracy and TPR (recall), the zero-shot MegaDetector model performed better than the models trained on SBU4 data (Fig. 3, Table 2). PPV (precision), however, was < 0.1 for all models. In terms of PPV (precision) and F1-score, the trained YOLOv5x6 model performed slightly better than MegaDetector. Note that the prevalence in the non-SBU4 set was six times lower than in SBU4 (Table 2).

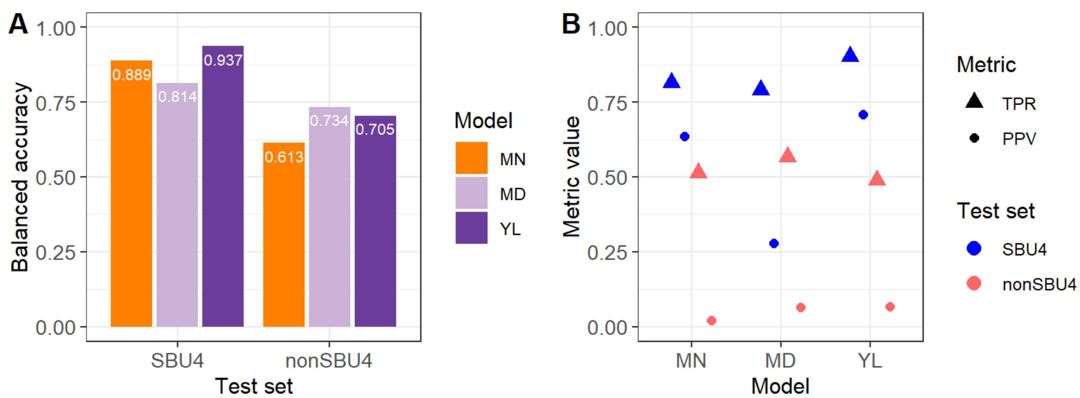

Figure 3. Performance of the classifier MobileNetV2 (MN) and the object detection model YOLOv5x6 (YL) trained on data from the single camera with most positives, for held-out test data from SBU4 and new data from other cameras (non-SBU4), as quantified by balanced accuracy (A), and TPR (recall) and PPV (precision) (B). Also shown are zero-shot results for the pre-trained MegaDetector model (MD).

All cameras scenario

Both models converged already after one full epoch; the numerous positive copies generated with standard oversampling likely accelerated the pattern recognition and learning processes.Extending the trainings to more epochs decreased validation metrics, indicating that networks started overfitting. A confidence threshold of 0.27 yielded maximum YOLOv5x6 fitness. Final training F1-score curves are in Appendix D.

Within-sample performance of all models trained on images from all cameras was worse than for the single camera scenario.

Evaluated on the full test set across all cameras, the trained object detection algorithm YOLOv5x6 performed best, with highest, albeit modest, values of TPR (recall), PPV (precision), and F1-score (Table 2). However, PPV (precision) was again very low also for the best model: YOLOv5x6 achieved a precision of only 0.33, in spite of comparatively high TPR (recall)

Table 2. Performance metrics for the classifier MobileNetV2 (MN) and the object detection model YOLOv5x6 (YL) trained on the single camera with the highest prevalence (SBU4) and on all cameras (all cams), along with results for the pre-trained MegaDetector model (MD), when evaluated against test sets from SBU4, non-SBU4 cameras (¬SBU4), and all cameras. Note that for models trained on SBU4, non-SBU4 constitutes an out-of-sample test set; for models trained on all cameras, SBU4 and non-SBU4 are subsets of the all camera test set. p(h) is the prevalence (proportion) of positives (heron) in the respective test sets. TPR (recall), TNR (specificity) and balanced accuracy (BA, the mean of TPR and TNR) do not depend on prevalence; they are intrinsic to the models. PPV (precision), NPV and F1-score (the harmonic mean of TPR and PPV) depend on prevalence; they are here calculated given the prevalence of the respective test set.

Model	Train	Test	TPR (recall)	TNR	BA	p(h)	PPV (precision)	NPV	F1
MN	SBU4	SBU4	0.815	0.963	0.889	0.073	0.635	0.985	0.714
MD	SBU4	SBU4	0.790	0.838	0.814	0.073	0.278	0.981	0.411
YL	SBU4	SBU4	0.903	0.971	0.937	0.073	0.709	0.992	0.794
MN	SBU4	¬SBU4	0.513	0.714	0.614	0.012	0.021	0.992	0.040
MD	SBU4	¬SBU4	0.568	0.901	0.734	0.012	0.064	0.994	0.115
YL	SBU4	¬SBU4	0.490	0.919	0.704	0.012	0.067	0.993	0.118
MN	all	all	0.626	0.942	0.784	0.015	0.141	0.993	0.230
MD	all	all	0.622	0.898	0.760	0.015	0.084	0.994	0.148
YL	all	all	0.803	0.975	0.889	0.015	0.328	0.997	0.466
MN	all	SBU4	0.927	0.401	0.664	0.073	0.109	0.986	0.195
YL	all	SBU4	0.895	0.942	0.919	0.073	0.547	0.991	0.679
MN	all	¬SBU4	0.529	0.969	0.749	0.012	0.170	0.994	0.257
YL	all	¬SBU4	0.773	0.977	0.875	0.012	0.285	0.997	0.416

of 0.80 and balanced accuracy of 0.89. That is, 67% of images classified as positives were false positives; in other words, the predictive certainty for heron presence was only 33%.

Both models trained on data from all cameras had higher performance metric values than the zero-shot MegaDetector model, although in terms of TPR (recall) MegaDetector got close to the trained classifier MobileNetV2 (Fig. 4).

Models fine-tuned on all cameras performed worse on the SBU4 test subset but better on the non-SBU4 test subset compared to models trained on SBU4 data only (Table 2). Note that the prevalence across all cameras is nearly five times lower than in SBU4.

Performances varied considerably across cameras (Fig. 4). The balanced accuracy of YOLOv5x6 differed most from MobileNetV2 for cameras with the highest (e.g., SBU4, NEN1) and lowest prevalences (e.g., SBU2).

In summary, the trained object detection model YOLOv5x6 performed best, especially for the intended use case of applying a model to images from all cameras. For all cameras but SBU4, training YOLOv5x6 on images from all cameras was better than training this model on images from the single camera with most positives (SBU4) and applying it by generalizing to all cameras. However, the achieved PPV (precision) of the best model remained low.

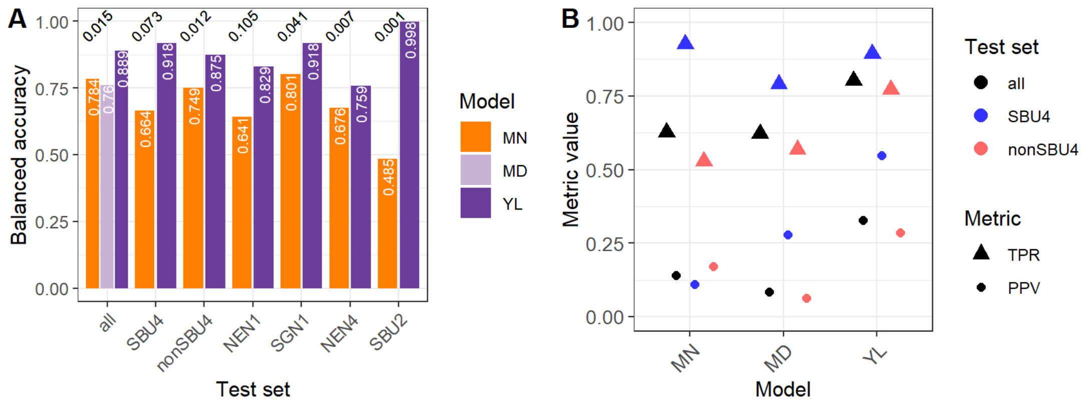

Figure 4. Performance of models trained on data from all cameras. A) Balanced accuracy of the classifier MobileNetV2 (MN), the object detection model YOLOv5x6 (YL) and the zero-shot MegaDectector (MD) on test sets across all cameras (all), SBU4, non-SBU4, and four individual cameras, two with relatively high prevalence in the test set (NEN1, SGN1) and two with relatively low prevalence in the test set (NEN4, SBU2). The value above the bars (black, angled) shows the prevalence (proportion of positives) in each respective test set. B) TPR (recall) and PPV (precision) of these models on test sets across all cameras (all), SBU4, and non-SBU4.

DISCUSSION

Using deep learning to automate single species recognition in a typical camera trap image data set, we report on insights regarding data pre-processing strategies and the results of a comparison between a classification and an object detection architecture. We use this case study to discuss the interpretation of performance metrics and to highlight the frequently neglected effect of class imbalance (low prevalence of the positive class) on predictive performance.

Data handling and pre-processing Appropriate splitting of data into training, validation and test sets is important to avoid data leakage and overestimation of model performance (Kapoor and Narayanan, 2023). The intended use of the model has to be considered, however, which in our case was to classify new images across all seasons and cameras. The second, seasonal splitting method improved model performance for this task, since the training and validation sets were more similar in terms of weather and vegetation patterns, as influenced by seasonality. We stress that few studies have implemented non-random data splitting, and recommend consideration of the intended model application to improve performance under this specific task.

Image resolution used during model training is often low and determined by default values used in open-source models (often 256×256, as in, e.g., Norouzzadeh et al. 2018; Tabak et al. 2019). Humans tend to underestimate the required resolution because they are better than CNNs in detecting animals at low image resolution (Leorna and Brinkman, 2022). The performance improvement of MobileNetV2 with higher resolution demonstrates that this variable should be considered an additional hyperparameter in wildlife recognition, especially in cases where animals can be expected to cover only a small number of pixels. Higher resolution input data requires more memory, however, and trades off with the size of the architecture itself. With resolution 896×896 we nearly saturated the available GPU memory capacity, making it impractical to fully train larger CNNs while maintaining the same resolution. A systematic further exploration of these trade-offs would be useful.

Transfer learning is an option to avoid model training altogether, as in the application of the pre-trained MegaDetector model, or to facilitate model training by fine-tuning some or all weights derived from a model trained on similar data. MobileNetV2 benefited from unfreezing and fine-tuning all model layers, which implies that the ImageNet weights are not well-suited for camera trap data and that complete tuning is preferable over training only the last layer. Similarly, Yang et al. (2024) observed a substantial CNN classifier improvement from increasing the number of tunable layers when using camera trap data. While the zero-shot MegaDetector model, pre-trained on camera trap images, overall performed poorly, it still achieved TPR values comparable to the fine-tuned MobileNetV2, which points to the importance of using transfer learning with models pre-trained on comparable tasks (Christin et al., 2019).

Resampling in combination with data augmentation offers avenues to overcome high class imbalance during model training (Klasen et al., 2022). Standard oversampling with augmentations in color space improved model performance the most, suggesting that these transformations were able to effectively simulate light changes across images.

Object detection outperformed classification The trained object detection model YOLOv5x6 performed overall better than the classification model MobileNetV2, for both in-sample test data and out-of-sample generalization. Additionally, YOLOv5x6 required approximately five times fewer computational resources than MobileNetV2 when trained on the same data set.

Object detection may be overall more suitable to camera trap images, where backgrounds vary depending on weather and seasons as well as between cameras. This may be intrinsic to the architecture, which focuses the attention of the model on an identified object. Or it may stem from architecture-specific features such as the ability to use images of higher resolution with the YOLO-framework, or the YOLO-specific feature of mosaic augmentation. Initial weights obtained from pre-training on other camera trap images may have contributed to better fine-tuning of this model, pointing to the relevance of transfer learning (Willi et al., 2019). Potentially also the automated optimization of the confidence threshold via model fitness could lead to higher performance than achieved by the classification algorithms, where the threshold needs to be set.

Taken together, these features may explain the more accentuated difference in performance between the two architectures in the all cameras scenario. Classification performance values for MobileNetV2 were considerably lower than for YOLOv5x6 in this scenario, and only slightly above the ones obtained with MegaDetector. The marked difference in the single camera SBU2 stems from the fact that there was a single positive image in this camera set, which MobileNetv2 failed to detect (resulting in TPR and PPV values of zero).

Also when generalizing to test sets unseen during training (non-SBU4 in the single camera scenario), object detection performed better than classification, achieving higher F1 scores and predictive performance. Irrespective of this comparatively higher generalization capacity of the object detection model, performance on new data remained poor for both trained networks. When considering the intended use case of classifying images from all cameras, training models on images from all cameras was overall better than training models on the single camera with most positives. This was in spite of strong differences in class imbalance between cameras.

Poor performance on images from new camera traps has also been observed by Beery et al. (2018), who found more than a 90% increase in error rates after evaluating a self-trained Inception-V3 model on samples from new locations. Additionally, using the same training and test sets, Beery et al. (2018) found a mean average precision decrease from 77% to 71% when evaluating a fine-tuned Faster R-CNN model on images with new backgrounds. Tabak et al. (2019) trained a ResNet-18 classifier on extensive camera trap data from 5 US locations, and observed a 18% drop in accuracy after evaluating this network on a set from Canada.

Most past work has focused either on classification or on object detection. Both methods are occasionally applied sequentially (Norouzzadeh et al., 2021; Schneider et al., 2023), by cropping

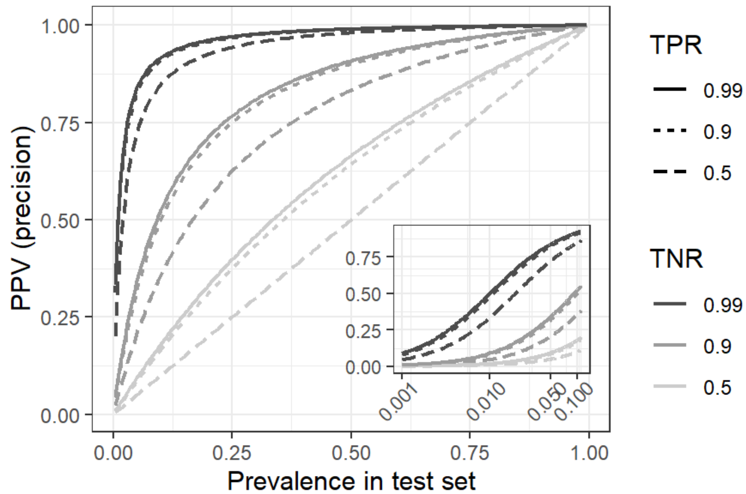

Figure 5. Effect of prevalence (proportion of positives in the test set) on PPV (precision) for different combinations of TPR (recall) and TNR (specificity). Also at high values of TPR and TNR, precision is highly sensitive to low values of prevalence. The inset shows PPV at small values of prevalence (logarithmic scale).

out animal frames with a pre-trained network and fine-tuning a classifier to recognize species in the cropped images. Few studies have independently fine-tuned and compared classification with object detection architectures on the same camera trap data set. Örn (2021) reported superior performance of YOLOv3 and MegaDetector over a DenseNet201 architecture. Beery et al. (2018) observed better generalization capabilities of the object detection architecture Faster R-CNN against the classification model InceptionV3. Object detection algorithms may thus generally be the better choice for camera trap image classification tasks than classification networks.

High performance metrics do not imply low uncertainty Relationships between individual performance metrics are not consistent or predictable. It is therefore essential to report all four basic metrics (the performance metrics TPR and TNR, as well as the predictive metrics PPV and NPV) together with the prevalence in the test set based on which they were calculated to allow researchers to assess model performance comprehensively (Trevethan, 2017).

High TPR (recall) and TNR, as well as balanced accuracy, do not necessarily indicate high predictive certainty. Predictive values (PPV and NPV) are critically dependent on prevalence. This has important consequences that are often overlooked in the discussion of classifier performance. First, low prevalence (high class imbalance) necessarily leads to low positive predictive values (see Box 1, Fig. 5). In our results this could partly explain the low PPV (precision) values observed for the non-SBU4 and all camera sets (Fig. 4), which have a prevalence of less than 2%. Second, the estimated values for the predictive performance metrics PPV (or precision) and NPV are only valid for data sets with the exact same prevalence as observed in the test set. The test set is hence assumed to be representative of the wider population and any potential application case.

Third, under high class imbalance, small differences in prevalence (between, e.g., a test set and an unseen application set) have large effects on predictive certainty (Fig. 5). Consider a simplified example, where TPR and TNR are both high at 0.9. Decreasing prevalence in a low class imbalance scenario by 10% from 0.5 to 0.4, decreases the resulting PPV from 0.9 to 0.86. Decreasing prevalence in a high class imbalance scenario by only 1% from 0.02 to 0.01, halves the resulting (low) PPV from 0.16 to 0.08.

Box 2. Using metrics in downstream ecological analyses Ecological field data with observation uncertainty is nowadays typically analyzed with state-space models such as the occupancy model (MacKenzie et al. 2003), where the observation process is modeled separately and conditional on the latent ecological process of interest.

Let x_{it} denote the latent true presence of the species in an image taken at time t at site i (defined here by the camera viewshed), and y_{it} the image label (detection/non-detection) obtained by automated detection of the species in the image by a classification algorithm. A simple occupancy, or state-space, model, spelled out in hierarchical submodels, could then be formulated as follows.

The *observation process* here corresponds to the classifier's detection/non-detection of the species in an image, conditional on the species' actual presence in the image (and hence the site). Note that an additional level could be included that makes the probability of presence in an image conditional on the separate probability of presence at a site, $\Pr(x_{it}|z_{it})$; we omit this here for the sake of brevity and simply note that the interpretation of $\Pr(x)$ here combines the probability of being both present in a camera viewshed at the time of a snapshot *and* visible in an image. The false negative and false positive error rates of detection, in the context of classifier uncertainty, then correspond directly to $\Pr(-|h) \approx (1-TPR)$ and $\Pr(+|e) \approx (1-TNR)$ (*cf.* Box 1). The detection model can make direct use of the estimators of the true positive and true negative detection probabilities, that is, the classifier performance metrics $TPR \approx \Pr(+|h)$ and $TNR \approx \Pr(-|e)$:

$$\Pr(y_{it}|x_{it}) \sim \text{Bernoulli}(x_{it}\Pr(+|h) + (1-x_{it})\Pr(-|e)) \tag{2}$$

These classifier performance metrics are independent of the prevalence of the class of interest (Box 1), and are therefore valid estimators of the true positive and true negative rate also in applications to new data sets, which might differ in prevalence from the test set. Avenues to improve the values of these metrics include model ensembling, model averaging, or other, similar ways of combining different algorithms trained on the same set of training data (e.g., Jurek et al. 2013; Kuncheva et al. 2000).

The *ecological process model*, estimating the probability of heron presence in an image and hence site i at time t, would typically be formulated as:

$$\Pr(x_{it}) \sim \text{Bernoulli}\left(\text{logit}^{-1}(\alpha + \sum^{l} \beta_l Z_{lit})\right) \tag{3}$$

where Z_l are l environmental covariates hypothesized to influence, through a logit-linear relationship, the presence of heron in a site and image. Covariates could be, e.g., the availability of food resources (fish abundance in the stream), the distance to the nearest heron colony, human disturbance, or time of day. Alongside $\Pr(x)$, these effects β_l are often a target of inference for the ecologist.

Since $\Pr(x)$ here corresponds to an updated (posterior) estimate of our prior knowledge of prevalence (as based on the proportion of positives in the test data set), the inclusion of any additional covariate data that can inform this probability helps reduce the predictive uncertainty in the posterior. In other words, meaningful explanatory variables, based on domain knowledge of hypothesized ecological preferences of the target species, could help overcome the low predictive values that the classifier alone is likely to exhibit in the case of low prevalence.

A second avenue for improving the estimate of $\Pr(x)$ would be the inclusion and formal integration of a second (or more), independent observation data set(s) of the same sites in a 'double-observer approach' (Nakashima et al., 2022). Additional complementary data could be obtained by adding more cameras that view the same site from different angles, or by including human observations of the same sites.

Improving predictive values is, however, routinely done in ecological analyses, through the inclusion of environmental covariates that inform the posterior estimate of $\Pr(X=h)$ (see Box 2). Nonetheless, the limitations of predictive performance of the classifier alone under high class imbalance should be kept in mind when using machine learning techniques for the automated detection of species using camera trap images.

CONCLUSIONS

Object detection algorithms may be preferable over classification for species detection tasks in wildlife camera trap data. Model performance was significantly improved by fine-tuning the full model on the data set at hand, while likely benefiting from initial weights derived from a model pre-trained on a similar data set (i.e., camera trap images).

Pre-processing and model training decisions affect performance. Non-random data splitting that took the intended application into account, and the use of high resolution images were beneficial. To tackle the high class imbalance typical of camera trap data sets, standard minority oversampling with data augmentation in color space yielded the best results.

We emphasize the importance of reporting all four basic performance metrics (TPR, TNR, PPV, and NPV) as well as the prevalence in the test set to enable meaningful interpretation of model performance. We highlight that predictive performance (PPV, NPV) can be expected to be low when prevalence is low, in spite of high classifier performance (TPR, TNR) and balanced accuracy values. This sensitivity of predictive performance to low prevalence (class imbalance) is rarely considered but can be expected to be common in camera trap projects designed to monitor rare species.

This problem can be partly mediated by the addition of ecologically informative covariates in a downstream analysis, as is usually done in ecological applications. Practitioners should nonetheless be aware that high performance metrics of the classifier (TPR, TNR) do not in and of themselves imply low posterior uncertainty in applications.

ACKNOWLEDGMENTS

We thank Catherine Graham, Luca Pegoraro, and Fiona Hefti for prior work on the data set and friendly support; Andreas Scheidegger for insightful discussions; and the SIAM department at Eawag for financial support and the collaborative spirit across disciplines. This work was supported by the Swiss National Foundation, SNF grant # 196902.

REFERENCES

Beery, S., Morris, D., and Yang, S. (2019). Efficient pipeline for camera trap image review. *arXiv preprint arXiv:1907.06772*.

Beery, S., Van Horn, G., and Perona, P. (2018). Recognition in terra incognita. In *European Conference on Computer Vision (ECCV)*, Munich, Germany. Caltech.

Borowiec, M. L., Dikow, R. B., Frandsen, P. B., McKeeken, A., Valentini, G., and White, A. E. (2022). Deep learning as a tool for ecology and evolution. *Methods in Ecology and Evolution*, 2022(April):1640–1660.

Burton, A. C., Neilson, E., Moreira, D., Ladle, A., Steenweg, R., Fisher, J. T., Bayne, E., and Boutin, S. (2015). Wildlife camera trapping: A review and recommendations for linking surveys to ecological processes. *Journal of Applied Ecology*, 52(3):675–685.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357.

Chen, C., Kyathanahally, S., Reyes, M., Merkli, S., Merz, E., Francazi, E., Hoege, M., Pomati, F., and Baity-Jesi, M. (2024). Producing plankton classifiers that are robust to dataset shift. *arXiv preprint arXiv:2401.14256*.

Christin, S., Hervet, É., and Lecomte, N. (2019). Applications for deep learning in ecology. *Methods in Ecology and Evolution*, 10(10):1632–1644.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009a). ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009b). Imagenet: A large-scale hierarchical image database. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255.

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. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*.

Dwyer, B., Nelson, J., Hansen, T., et al. (2024). Roboflow (version 1.0) [software]. Available from https://roboflow.com.

Ferreira, A. C., Silva, L. R., Renna, F., Brandl, H. B., Renoult, J. P., Farine, D. R., Covas, R., and Doutrelant, C. (2020). Deep learning-based methods for individual recognition in small birds. *Methods in Ecology and Evolution*, 11:1072–1085.

Francazi, E., Baity-Jesi, M., and Lucchi, A. (2023). Characterizing the effect of class imbalance on the learning dynamics.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. *arXiv preprint arXiv:1311.2524*.

Gomez Villa, A., Salazar, A., and Vargas, F. (2017). Towards automatic wild animal monitoring: Identification of animal species in camera-trap images using very deep convolutional neural networks. *Ecological Informatics*, 41:24–32.

Hernandez, A., Miao, Z., Vargas, L., Dodhia, R., Arbelaez, P., and Lavista Ferres, J. M. (2024). Pytorch-wildlife: A collaborative deep learning framework for conservation. *arXiv preprint arXiv:2405.12930*.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. (2018). Densely connected convolutional networks. *arXiv preprint arXiv:1608.06993*.

Jurek, A., Bi, Y., Wu, S., and Nugent, C. (2013). A survey of commonly used ensemble-based classification techniques. *The Knowledge Engineering Review*, 29(5):551–581.

Kapoor, S. and Narayanan, A. (2023). Leakage and the reproducibility crisis in machine-learning-based science. *Patterns*, 4:100804.

Klasen, M., Ahrens, D., Eberle, J., and Steinhage, V. (2022). Image-based automated species identification: can virtual data augmentation Overcome problems of insufficient sampling ? *Systematic Biology*, 71(2):320–333.

Kuncheva, L. I., Whitaker, C. J., Shipp, C. A., and Duin, R. P. (2000). Is independence good for combining classifiers? *Proceedings - International Conference on Pattern Recognition*, 15(2):168–171.

Leorna, S. and Brinkman, T. (2022). Human vs. machine: Detecting wildlife in camera trap images. *Ecological Informatics*, 72:101876.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. (2017). Focal loss for dense object detection. *arXiv preprint arXiv:1708.02002*.

Loffredo, E., Pastore, M., Cocco, S., and Monasson, R. (2024). Restoring balance: principled under/oversampling of data for optimal classification. *International Conference of Machine Learning (ICML)*.

Loshchilov, I. and Hutter, F. (2019). Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*.

MacKenzie, D. I., Nichols, J. D., Hines, J. E., Knutson, M. G., Franklin, B., and Franklin,

A. B. (2003). Estimating site occupancy, colonization, and local extinction when a species is detected imperfectly. *Ecology*, 84(8):2200–2207.

Nakashima, Y., Hongo, S., Mizuno, K., Yajima, G., and Dzefck, Z. C. (2022). Double-observer approach with camera traps can correct imperfect detection and improve the accuracy of density estimation of unmarked animal populations. *Scientific Reports*, 12(1):1–10.

Norouzzadeh, M. S., Morris, D., Beery, S., Joshi, N., Jojic, N., and Clune, J. (2021). A deep active learning system for species identification and counting in camera trap images. *Methods in Ecology and Evolution*, 12:150–161.

Norouzzadeh, M. S., Nguyen, A., Kosmala, M., Swanson, A., Palmer, M. S., Packer, C., and Clune, J. (2018). Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning. *Proceedings of the National Academy of Sciences*, 115(25):E5716–E5725.

Örn, F. (2021). Computer vision for camera trap footage: Comparing classification with object detection. Master's thesis, Uppsala University, Uppsala. UPTEC F 21037, Civilingenjörsprogrammet i teknisk fysik, Teknisk-naturvetenskapliga fakulteten.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. *arXiv preprint arXiv:1506.02640*.

Rhinehart, T. A., Turek, D., and Kitzes, J. (2022). A continuous-score occupancy model that incorporates uncertain machine learning output from autonomous biodiversity surveys. *Methods in Ecology and Evolution*, 2022(July 2021):1778–1789.

Ruder, S. (2017). An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. *arXiv preprint arXiv:1801.04381*.

Schneider, D., Lindner, K., Vogelbacher, M., Bellafkir, H., Mühling, M., Farwig, N., and Freisleben, B. (2023). Recognizing european mammals and birds in camera trap images using convolutional neural networks. In *Workshop on Camera Traps, AI and Ecology*, Marburg, Germany. Department of Mathematics & Computer Science, University of Marburg.

Schneider, S., Greenberg, S., Taylor, G. W., and Kremer, S. C. (2020). Three critical factors affecting automated image species recognition performance for camera traps. *Ecology and Evolution*, 10(7):3503–3517.

Tabak, M. A., Norouzzadeh, M. S., Wolfson, D. W., Sweeney, S. J., Vercauteren, K. C., Snow, N. P., Halseth, J. M., Di Salvo, P. A., Lewis, J. S., White, M. D., Teton, B., Beasley, J. C., Schlichting, P. E., Boughton, R. K., Wight, B., Newkirk, E. S., Ivan, J. S., Odell, E. A., Brook, R. K., Lukacs, P. M., Moeller, A. K., Mandeville, E. G., Clune, J., and Miller, R. S. (2019). Machine learning to classify animal species in camera trap images: Applications in ecology. *Methods in Ecology and Evolution*, 10:585–590.

Tan, M. and Le, Q. V. (2019). Efficientnet: Rethinking model scaling for convolutional neural networks. *arXiv preprint arXiv:1905.11946*.

Thambawita, V., Strümke, I., Hicks, S. A., Halvorsen, P., Parasa, S., and Riegler, M. A. (2021). Impact of image resolution on deep learning performance in endoscopy image classification: An experimental study using a large dataset of endoscopic images. *Diagnostics*, 11.

Trevethan, R. (2017). Sensitivity, specificity, and predictive values: foundations, pliabilities, and pitfalls in research and practice. *Frontiers in Public Health*, 5(November):1–7.

Tuia, D., Kellenberger, B., Beery, S., Costelloe, B. R., Zuffi, S., Risse, B., Mathis, A., Mathis, M. W., van Langevelde, F., Burghardt, T., Kays, R., Klinck, H., Wikelski, M., Couzin, I. D., van Horn, G., Crofoot, M. C., Stewart, C. V., and Berger-Wolf, T. (2022). Perspectives in machine learning for wildlife conservation. *Nature Communications*, 13:792.

Ultralytics (2021). YOLOv5: A state-of-the-art real-time object detection system. `https:`

//docs.ultralytics.com.

Vélez, J., McShea, W., Shamon, H., Castiblanco-Camacho, P. J., Tabak, M. A., Chalmers, C., Fergus, P., and Fieberg, J. (2023). An evaluation of platforms for processing camera-trap data using artificial intelligence. *Methods in Ecology and Evolution*, 14(2):459–477.

Whytock, R. C., Swiezewski, J., Zwerts, J. A., Bara-Slupski, T., Pambo, A. F. K., Rogala, M., Bahaa-el din, L., Boekee, K., Brittain, S., Cardoso, A. W., et al. (2021). Robust ecological analysis of camera trap data labelled by a machine learning model. In *Proceedings of the 20th International Conference on Computer Information Systems and Industrial Management Applications (CISIM)*, pages 1080–1092, Elk, Poland. Springer.

Willi, M., Pitman, R. T., Cardoso, A. W., Locke, C., Swanson, A., Boyer, A., Veldthuis, M., and Fortson, L. (2019). Identifying animal species in camera trap images using deep learning and citizen science. *Methods in Ecology and Evolution*, 10:80–91.

Yang, D.-Q., Meng, D.-Y., Li, H.-X., Li, M.-T., Jiang, H.-L., Tan, K., Huang, Z.-P., Li, N., Wu, R.-H., Li, X.-W., Chen, B.-H., Zhang, M., Ren, G.-P., and Xiao, W. (2024). A systematic study on transfer learning: Automatically identifying empty camera trap images using deep convolutional neural networks. *Ecological Informatics*, 71:101034.

Zualkernan, I., Dhou, S., Judas, J., Sajun, A. R., Gomez, B. R., and Hussain, L. A. (2022). An iot system using deep learning to classify camera trap images on the edge. *Computers*, 11(13).

A DATA SET

In this section we provide more detailed information on the heron data set used in this project. We excluded night-time images from this study, meaning that all statistics given here correspond to day-time samples. Since the original splitting procedures were implemented on the entire data set, there were some deviations from the splitting ratios given in the Methods section. Table 3 displays the total and per-camera sample size along with the number of positives. The same quantities are shown in Tables 4-9 for training, validation and test sets according to the first, chronological, and second, seasonal, splitting method.

camera	GBU1	GBU2	GBU3	GBU4	KBU1	KBU2	KBU3	KBU4
# samples	12276	12274	14349	11382	12485	11215	13543	7172
# positives	0	0	0	3	0	16	4	5
camera	NEN1	NEN2	NEN3	NEN4	PSU1	PSU2	PSU3	SBU1
#samples	10032	12226	12261	10224	6222	8526	8833	10118
#positives	252	128	163	53	13	10	30	40
camera	SBU2	SBU3	SBU4	SGN1	SGN2	SGN3	SGN4	Total
# samples	11618	10952	12899	11334	7849	11341	12348	251479
# positives	44	181	1545	183	97	223	187	3177

Table 3. Total and per-camera number of samples (# samples) and positives (# positives) for the entire data set

A.1 Split 1: chronological

camera	GBU1	GBU2	GBU3	GBU4	KBU1	KBU2	KBU3	KBU4
# samples	8922	12274	12274	8255	7977	7474	12605	3296
# positives	0	0	0	3	0	15	4	5
camera	NEN1	NEN2	NEN3	NEN4	PSU1	PSU2	PSU3	SBU1
#samples	6415	8587	6737	6161	2857	5843	5764	7120
#positives	66	127	155	38	3	8	5	39
camera	SBU2	SBU3	SBU4	SGN1	SGN2	SGN3	SGN4	Total
# samples	6342	6219	7327	8229	6711	7695	7577	171230
# positives	36	150	1144	183	87	145	119	2332

Table 4. Total and per-camera number of samples (# samples) and positives (# positives) for the training set according to the first, chronological splitting method

camera	GBU1	GBU2	GBU3	GBU4	KBU1	KBU2	KBU3	KBU4
# samples	1665	0	1999	1743	1658	1514	406	1426
# positives	0	0	0	0	0	0	0	0
camera	NEN1	NEN2	NEN3	NEN4	PSU1	PSU2	PSU3	SBU1
# samples	1489	2271	2087	1217	1398	1307	1291	1319
# positives	2	0	7	11	0	2	6	0
camera	SBU2	SBU3	SBU4	SGN1	SGN2	SGN3	SGN4	Total
# samples	2384	2371	2928	1559	1138	1576	3108	37854
# positives	1	24	252	0	10	41	42	398

Table 5. Total and per-camera number of samples (# samples) and positives (# positives) for the validation set according to the first, chronological splitting method

camera	GBU1	GBU2	GBU3	GBU4	KBU1	KBU2	KBU3	KBU4
# samples	1689	0	1507	1384	2850	2227	532	2450
# positives	0	0	0	0	0	1	0	0
camera	NEN1	NEN2	NEN3	NEN4	PSU1	PSU2	PSU3	SBU1
# samples	2128	1368	3437	2846	1967	1376	1778	1679
# positives	184	1	1	4	10	0	19	1
camera	SBU2	SBU3	SBU4	SGN1	SGN2	SGN3	SGN4	Total
# samples	2892	2362	2644	1546	0	2070	1663	42395
# positives	7	7	149	0	0	37	26	447

Table 6. Total and per-camera number of samples (# samples) and positives (# positives) for the test set according to the first, chronological splitting method

A.2 Split 2: seasonal

camera	GBU1	GBU2	GBU3	GBU4	KBU1	KBU2	KBU3	KBU4
# samples	9652	9433	11008	9035	8153	7927	11292	4635
# positives	0	0	0	1	0	15	2	5
camera	NEN1	NEN2	NEN3	NEN4	PSU1	PSU2	PSU3	SBU1
# samples	7045	10028	8111	6630	4107	6463	6395	7517
# positives	36	92	102	37	3	10	12	9
camera	SBU2	SBU3	SBU4	SGN1	SGN2	SGN3	SGN4	Total
# samples	8121	7805	9508	8666	6735	8015	9753	186034
# positives	37	158	1263	111	83	181	146	2303

Table 7. Total and per-camera number of samples (# samples) and positives (# positives) for the training set according to the second, seasonal splitting method

camera	GBU1	GBU2	GBU3	GBU4	KBU1	KBU2	KBU3	KBU4
# samples	629	1822	903	638	2134	1460	922	1059
# positives	0	0	0	0	0	0	2	0

camera	NEN1	NEN2	NEN3	NEN4	PSU1	PSU2	PSU3	SBU1
# samples	1415	1498	2086	1967	833	874	1223	1336
# positives	51	10	28	5	3	0	11	11

camera	SBU2	SBU3	SBU4	SGN1	SGN2	SGN3	SGN4	Total
# samples	2016	1658	1694	1437	532	1706	1446	31288
# positives	6	9	158	21	4	24	23	366

Table 8. Total and per-camera number of samples (# samples) and positives (# positives) for the validation set according to the second, seasonal splitting method

camera	GBU1	GBU2	GBU3	GBU4	KBU1	KBU2	KBU3	KBU4
# samples	1995	1019	2438	1709	2198	1828	1329	1478
# positives	0	0	0	2	0	1	0	0

camera	NEN1	NEN2	NEN3	NEN4	PSU1	PSU2	PSU3	SBU1
# samples	1572	700	2064	1627	1282	1189	1215	1265
# positives	165	26	33	11	7	0	7	20

camera	SBU2	SBU3	SBU4	SGN1	SGN2	SGN3	SGN4	Total
# samples	1481	1489	1697	1231	582	1620	1149	34157
# positives	1	14	124	51	10	18	18	508

Table 9. Total and per-camera number of samples (# samples) and positives (# positives) for the test set according to the second, seasonal splitting method

B BACKGROUND REMOVAL

In an effort to increase heron visibility – especially where the bird is obstructed by environmental conditions –, we applied a background removal technique on input pixels. Such method consists in taking the average *ith* pixel value $\overline{p}_i^n$ of the last n images relative, and subtract it from the *ith* pixel value p_i of the current image, followed by a renormalization operation to ensure the new pixel value p'_i remains in interval $[0, 255]$:

$$p'_i = \frac{p_i - \overline{p}_i^n + 255}{2} \tag{4}$$

We tested this technique on the undersampled SBU4 set for $n = 2, 4, 8, 16, 32$. An example for $n = 4$ is provided in Figure 6. As seen in Figure 7, background removal actually hinders performance, with validation F1-score curves slightly lower than original images. A reason for this could similar heron positioning across image sequences, making the birds less salient or vanish completely after pixel subtraction. Another explanation might be the presence of background noise due to light variations across different times of the day. Additional research is suggested to thoroughly investigate the origin of performance decrease after background removal.

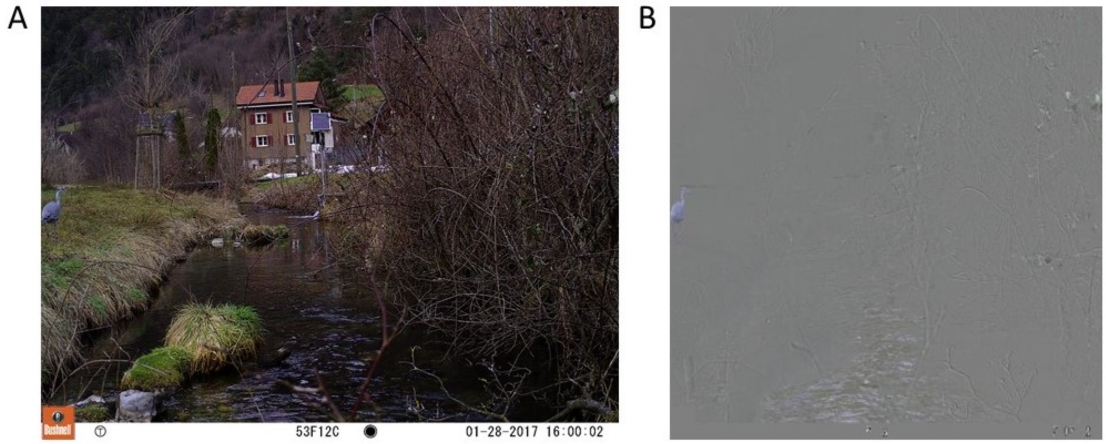

Figure 6. Example of image before (original, A) and after resizing and background removal using the previous 4 images (B).

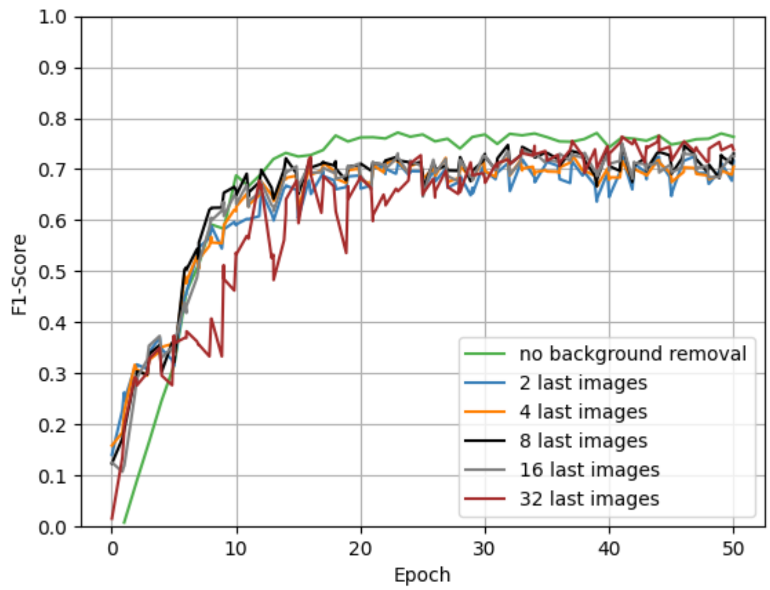

Figure 7. F1-score validation curves using background removal

C LOGARITHMIC OVERSAMPLING

As seen in appendix A, the distribution of positives across distinct cameras is quite uneven, with a high concentration of heron images in SBU4. In order to achieve a more even distribution and decrease potential camera background bias, we applied a novel logarithmic oversampling technique which moderately increases the number of positives from non-majority cameras. Given the number of positive samples n_i from camera i, we increased the number of instances with oversampling to

$$n_i' = n_i \frac{\log\left(1 + \frac{n_{\text{SBU4}}}{n_i}\right)}{\log(2)} \tag{5}$$

This procedure was applied individually to the 10 cameras with most herons after SBU4, while positives from the remaining cameras were rearranged into a single group (‘regrouped cams’) and oversampled collectively. The rebalancing of positives across different cameras was then followed by standard oversampling. A logarithmic oversampling prevents the generation of excessive copies from minority cameras that would be obtained with a standard oversampling technique and may potentially lead to pattern memorization. The combined application of logarithmic and standard oversampling techniques on data from all cameras is referred to as log oversampling, and is implemented in the training procedures using all camera sets. As seen in Figures 8 and 9, the application of log oversampling has little impact on model performance for individual camera subsets. The only noticeable change occurs for MobileNetV2 on the SBU2 set, which can be explained by low heron incidence.

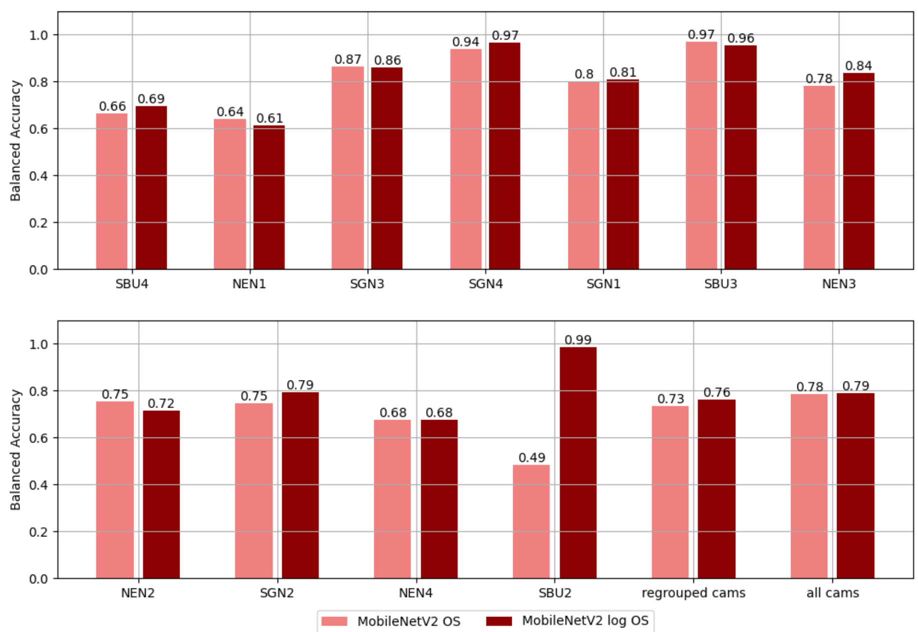

Figure 8. Balanced accuracy of MobileNetV2 trained on all cameras with oversampling (OS) and logarithmic oversampling (log OS)

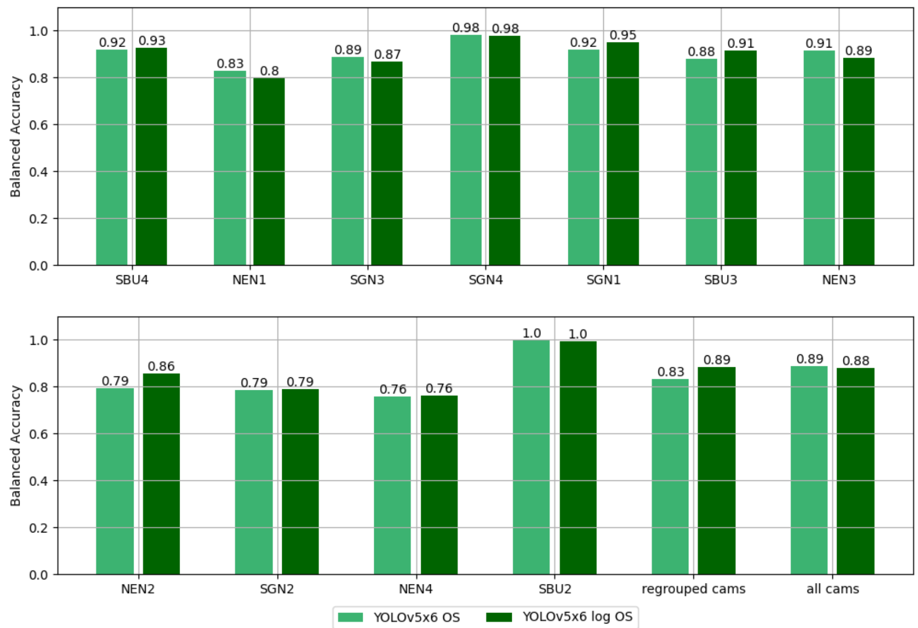

Figure 9. Balanced accuracy of YOLOv5x6 trained on all cameras with oversampling (OS) and logarithmic oversampling (log OS)

D COMPLETE RESULTS

The training F1-score curves for both MobileNetV2 and YOLOv5x6 trained on both the SBU4 and full data sets are displayed in Figure 10. The confusion matrix elements for the single- and multi-camera scenarios are shown in Tables 10 and 11, respectively.

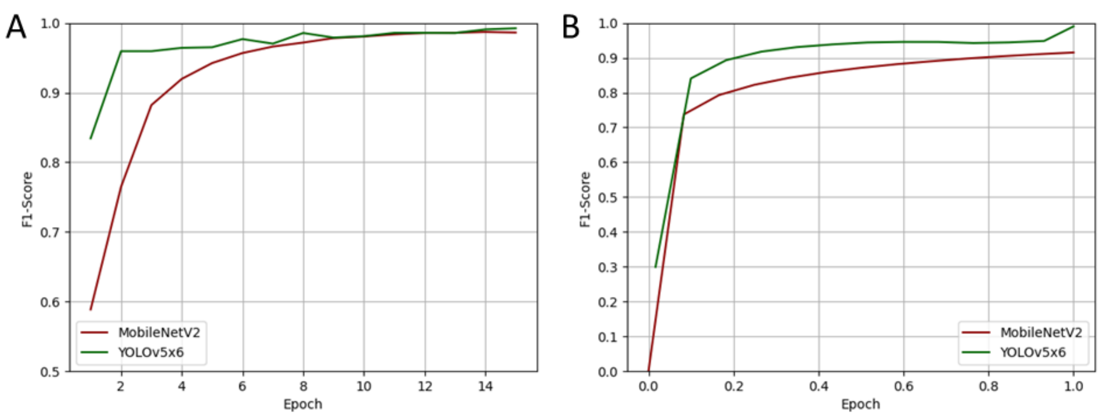

Figure 10. Final training curves. A) F1-score curve of models trained on the SBU4 set. B) F1-score curve of models trained on all cameras.

model	evaluated on	TN	FP	FN	TP
MobileNetV2	SBU4	1515	58	23	101
	non-SBU4	22906	9170	187	197
YOLOv5x6	SBU4	1527	46	12	112
	non-SBU4	29467	2609	196	188
MegaDetector	SBU4	1318	255	26	98
	non-SBU4	28890	3186	166	218

Table 10. Confusion matrix elements obtained with models trained on the SBU4 set and untrained MegaDetector weights when evaluated on SBU4 and non-SBU4 sets. Matrix elements are true negatives (TN), false positives (FP), false negatives (FN) and true positives (TP).

model	evaluated on	TN	FP	FN	TP
MobileNetV2	SBU4	631	942	9	115
	NEN1	1283	124	104	61
	SGN3	1528	74	4	14
	SGN4	1060	71	1	17
	SGN1	1057	123	15	36
	SBU3	1393	82	0	14
	NEN3	1938	93	13	20
	NEN2	654	20	12	14
	SGN2	567	5	5	5
	NEN4	1597	19	7	4
	SBU2	1436	44	1	0
	regrouped cams	18570	338	19	18
	all cams	31714	1935	190	318
YOLOv5x6	SBU4	1481	92	13	111
	NEN1	1291	116	43	122
	SGN3	1509	93	3	15
	SGN4	1091	40	0	18
	SGN1	1033	147	2	49
	SBU3	1436	39	3	11
	NEN3	1993	38	5	28
	NEN2	655	19	10	16
	SGN2	557	15	4	6
	NEN4	1573	43	5	6
	SBU2	1472	8	0	1
	regrouped cams	18722	186	12	25
	all cams	32813	836	100	408
MegaDetector	all cams	30206	3443	192	316

Table 11. Confusion matrix elements obtained with models trained on the full data set and untrained MegaDetector weights when evaluated on various camera subsets. Matrix elements are true negatives (TN), false positives (FP), false negatives (FN) and true positives (TP).